aws

Guida per gli sviluppatori

AWS X-Ray

AWS X-Ray: Guida per gli sviluppatori

Table of Contents

Che cos'è AWS X-Ray? ... 1
Nozioni di base ... 3
Concetti ... 4
Segmenti ... 4
Sottosegmenti ... 5
Grafico del servizio ... 9
Tracce ... 10
Campionamento ... 11
Intestazione di tracciamento ... 12
Espressioni filtro ... 13
Gruppi ... 14
Annotazioni e metadati ... 14
Errori, malfunzionamenti ed eccezioni ... 15
Console X-Ray ... 16
Mappa di tracciamento ... 17
Visualizzazione della mappa di tracciamento ... 17
Filtraggio della mappa di traccia per gruppo ... 22
Traccia la legenda e le opzioni della mappa ... 23
Tracce ... 24
Visualizzazione dei tracciamenti ... 24
Esplorazione della sequenza temporale di tracciamento ... 29
Visualizzazione dei dettagli del segmento ... 31
Visualizzazione dei dettagli del sottosegmento ... 31
Espressioni filtro ... 33
Filtraggio dei dettagli delle espressioni ... 33
Utilizzo delle espressioni filtro con i gruppi ... 35
Sintassi delle espressioni filtro ... 35
Parole chiave booleane ... 36
Parole chiave numeriche ... 37
Parole chiave stringa ... 38
Parole chiave complesse ... 40
funzione id ... 44
Tracciamento tra account ... 45
Configura l'osservabilità tra account ... 45

Visualizzazione delle tracce tra account ... 46
Tracciamento di applicazioni basate sugli eventi ... 49
Visualizza le tracce collegate nella mappa di tracciamento ... 50
Visualizza i dettagli della traccia collegata ... 51
Seleziona una singola traccia all'interno di un insieme di tracce collegate ... 52
Istogrammi ... 53
Latenza ... 53
Interpretazione dei dettagli del servizio ... 54
Informazioni dettagliate ... 56
Abilita gli approfondimenti nella console X-Ray ... 57
Abilita le notifiche di approfondimenti ... 59
Panoramica di Insight ... 60
Esamina lo stato di avanzamento di un'analisi ... 63
Analisi ... 65
Caratteristiche della console ... 65
Distribuzione del tempo di risposta ... 68
Attività delle serie temporali ... 69
Esempi di flussi di lavoro ... 69
Osservare gli errori sul grafico di servizio ... 69
Identificare i picchi del tempo di risposta ... 70
Visualizzare tutti le tracce contrassegnate con un codice di stato ... 70
Visualizzare tutti gli elementi in un sottogruppo e associati a un utente ... 71
Confrontare due set di tracce con criteri diversi ... 71
Identificare un tracciamento rilevante e visualizzare i dettagli ... 72
Gruppi ... 72
Creazione di un gruppo ... 74
Applica un gruppo ... 76
Modifica un gruppo ... 77
Clonare un gruppo ... 79
Eliminazione di un gruppo ... 80
Visualizza le metriche di gruppo in Amazon CloudWatch ... 81
Campionamento ... 82
Configurazione delle regole di campionamento di ... 82
Personalizzazione delle regole di campionamento ... 83
Opzioni delle regole di campionamento ... 84
Esempi di regole di campionamento ... 86

Configurazione del servizio per l'utilizzo delle regole di campionamento 87
Visualizzazione dei risultati del campionamento 87
Passaggi successivi 88
Deep linking tra console 88
Tracce 89
Espressioni filtro 89
Intervallo temporale 90
Regione 90
Combinazioni 91
Demone X-Ray 92
Download del daemon 92
Verifica della firma dell'archivio del daemon 94
Esecuzione del daemon 95
Dare al demone il permesso di inviare dati a X-Ray 95
Registri dei daemon X-Ray 96
Configurazione 97
Variabili di ambiente supportate 97
Utilizzo delle opzioni della riga di comando 98
Utilizzo di un file di configurazione 99
Esecuzione del daemon in locale 101
Esegui il daemon X-Ray su Linux 101
Esegui X-Ray daemon in un contenitore Docker 101
Esegui X-Ray daemon su Windows 103
Esegui il daemon su OS X 104
Su Elastic Beanstalk 104
Utilizzo dell'integrazione Elastic Beanstalk X-Ray per eseguire il daemon X-Ray 105
Scaricamento ed esecuzione del daemon X-Ray in modalità manuale (Avanzata) 106
su Amazon EC2 108
Su Amazon ECS ECS ECS EC 110
Utilizzo dell'immagine Docker ufficiale 110
Creare e compilare un'immagine Docker 111
Configura le opzioni della riga di comando nella console Amazon ECS 114
Strumentazione della vostra applicazione 115
Strumentate la vostra applicazione con Distro per AWS OpenTelemetry 116
AWS X-Ray Strumentazione dell'applicazione con gli SDK 117
Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray 118

Strumentazione con Go 119
AWSDistro perOpenTelemetryVai 119
SDK X-Ray per Go 120
Strumentazione con Java 136
AWSDistro perOpenTelemetryGiava 137
X-Ray SDK per Java 137
Strumentazione con Node.js 194
AWSDistro perOpenTelemetry JavaScript 195
X-Ray SDK per Node.js 195
Strumentazione con Python 221
AWS Distro per OpenTelemetry Python 221
X-Ray SDK per Python 221
Strumentazione con .NET 254
AWS Distro per OpenTelemetry .NET 254
X-Ray SDK per.NET 255
Strumentare con Ruby 281
AWSDistro perOpenTelemetryRubino 281
X-Ray SDK per Ruby 282
Integrazione con Servizi AWS 301
AWS Distro per OpenTelemetry 303
AWS Distro per OpenTelemetry 303
API Gateway 304
App Mesh 306
App Runner 309
AWS AppSync 309
CloudTrail 309
Eventi di gestione dei raggi X in CloudTrail 311
Eventi relativi ai dati a raggi X in CloudTrail 312
Esempi di eventi X-Ray 313
CloudWatch 316
CloudWatch RUM 317
CloudWatch Synthetics 318
AWS Config 327
Creazione di un trigger di una funzione Lambda 328
Creazione di una regola personalizzata per X-Ray in AWS Config 329
Risultati di esempio 330

Notifiche Amazon SNS ... 331
Amazon EC2 ... 331
Elastic Beanstalk ... 331
Sistema di bilanciamento del carico elastico ... 332
EventBridge ... 332
Visualizzazione della sorgente e delle destinazioni sulla mappa dei servizi X-Ray ... 333
Propaga il contesto di traccia alle destinazioni degli eventi ... 333
Lambda ... 339
Amazon SNS ... 341
Configurazione del tracciamento attivo di Amazon SNS ... 341
Visualizza le tracce degli editori e degli abbonati di Amazon SNS nella console X-Ray ... 343
Step Functions ... 345
Amazon SQS ... 346
Invio dell'intestazione di traccia HTTP ... 348
Recupero dell'intestazione di traccia e recupero del contesto di traccia ... 348
Amazon S3 ... 349
Configurazione delle notifiche degli eventi di Amazon S3 ... 350
Creazione di risorse radiografiche conCloudFormation ... 352
X-Ray eAWS CloudFormationmodelli ... 352
Ulteriori informazioni su AWS CloudFormation ... 352
Assegnazione di tag ... 353
Limitazioni applicate ai tag ... 354
Gestione dei tag nella console ... 355
Aggiunta di tag a un nuovo gruppo (console) ... 355
Aggiunta di tag a una nuova regola di campionamento (console) ... 356
Modificare o eliminare tag per un gruppo (console) ... 356
Modificare o eliminare tag per una regola di campionamento (console) ... 357
Gestione dei tag nellaAWS CLI ... 357
Aggiunta di tag a un nuovo gruppo di X-Ray o a una regola di campionamento (CLI) ... 358
Aggiunta di tag a una risorsa esistente (CLI) ... 360
Elenca i tag su una risorsa (CLI) ... 360
Elimina i tag su una risorsa (CLI) ... 361
Controlla l'accesso alle risorse X-Ray in base ai tag ... 361
Applicazione di esempio ... 363
Tutorial Scorekeep ... 365
Prerequisiti ... 366

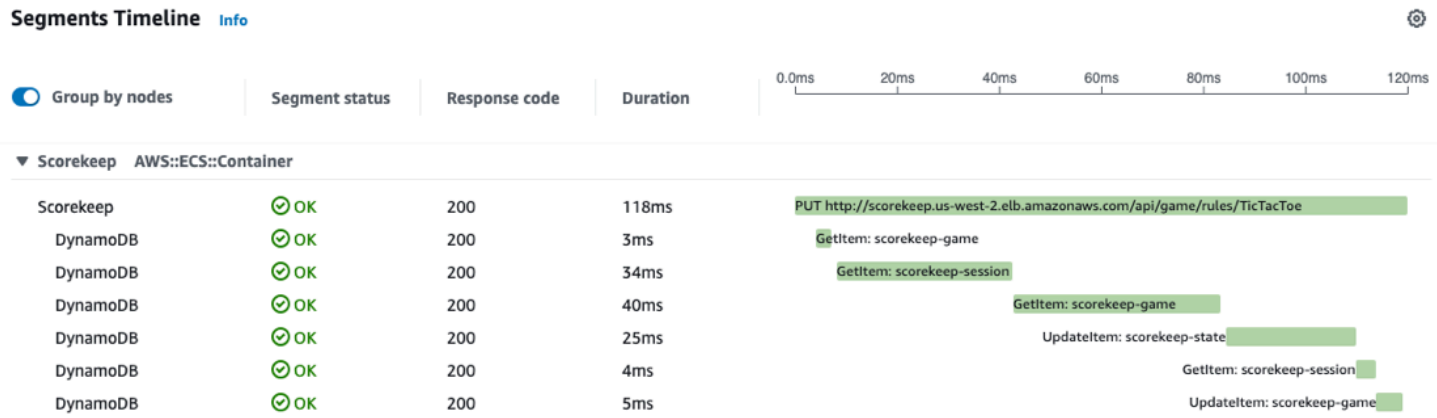
Installa l'applicazione Scorekeep utilizzando CloudFormation ... 367
Generare i dati di tracciamento ... 368
Visualizza la mappa di tracciamento nel AWS Management Console ... 369
Configurazione delle notifiche Amazon SNS ... 377
Esplorare l'applicazione di esempio ... 379
Opzionale: policy con privilegio minimo ... 384
Eliminazione ... 386
Passaggi successivi ... 387
AWS Client SDK ... 388
Sottosegmenti personalizzati ... 388
Annotazioni e metadati ... 389
Client HTTP ... 390
Client SQL ... 391
AWS Lambda funzioni ... 393
Nome casuale ... 394
Worker ... 396
Analisi del codice di avvio ... 398
Analisi degli script ... 400
Strumentazione dei client web ... 402
Thread worker ... 406
Risoluzione dei problemi ... 408
Mappa di tracciamento a raggi X e pagine con i dettagli della traccia ... 408
Non vedo tutti i miei registri CloudWatch ... 408
Non vedo tutti i miei allarmi sulla mappa di tracciamento a raggi X ... 409
Non vedo alcune AWS risorse sulla mappa di tracciamento ... 409
Ci sono troppi nodi sulla mappa di traccia ... 410
X-Ray SDK per Java ... 410
X-Ray SDK per Node.js ... 410
Il demone X-Ray ... 411
Sicurezza ... 412
... 412
Protezione dei dati ... 412
Gestione dell'identità e degli accessi ... 415
Destinatari ... 415
Autenticazione con identità ... 416
Gestione dell'accesso con policy ... 419

Come AWS X-Ray funziona con IAM ... 422
Esempi di policy basate su identità ... 430
Risoluzione dei problemi ... 443
Registrazione e monitoraggio ... 446
Convalida della conformità ... 447
Resilienza ... 448
Sicurezza dell'infrastruttura ... 448
Endpoint VPC ... 449
Creazione di un endpoint VPC per X-Ray ... 449
Controllo dell'accesso all'endpoint X-Ray VPC ... 451
Regioni supportate ... 452
API X-Ray ... 454
Tutorial ... 455
Prerequisiti ... 455
Generare i dati di tracciamento ... 455
Usa l'API X-Ray ... 456
Pulizia ... 459
Invio dei dati ... 459
Generazione degli ID di tracciamento ... 461
Usando PutTraceSegments ... 462
Invio di documenti di segmento al demone X-Ray ... 463
Ottenimento dei dati ... 464
Recupero del grafico del servizio ... 464
Recupero del grafico del servizio dal gruppo ... 471
Recupero dei tracciamenti ... 471
Recupero e definizione dell’analisi delle cause principali ... 476
Configurazione ... 479
Impostazioni di crittografia ... 479
Regole di campionamento ... 480
Gruppi ... 485
Campionamento ... 486
Documenti di un segmento ... 490
Campi del segmento ... 491
Sottosegmenti ... 494
Dati sulle richieste HTTP ... 499
Annotazioni ... 502

Metadati 503
AWS dati relativi alle risorse 504
Errori ed eccezioni 507
Query SQL 508
Cronologia dei documenti 510
..... dxix

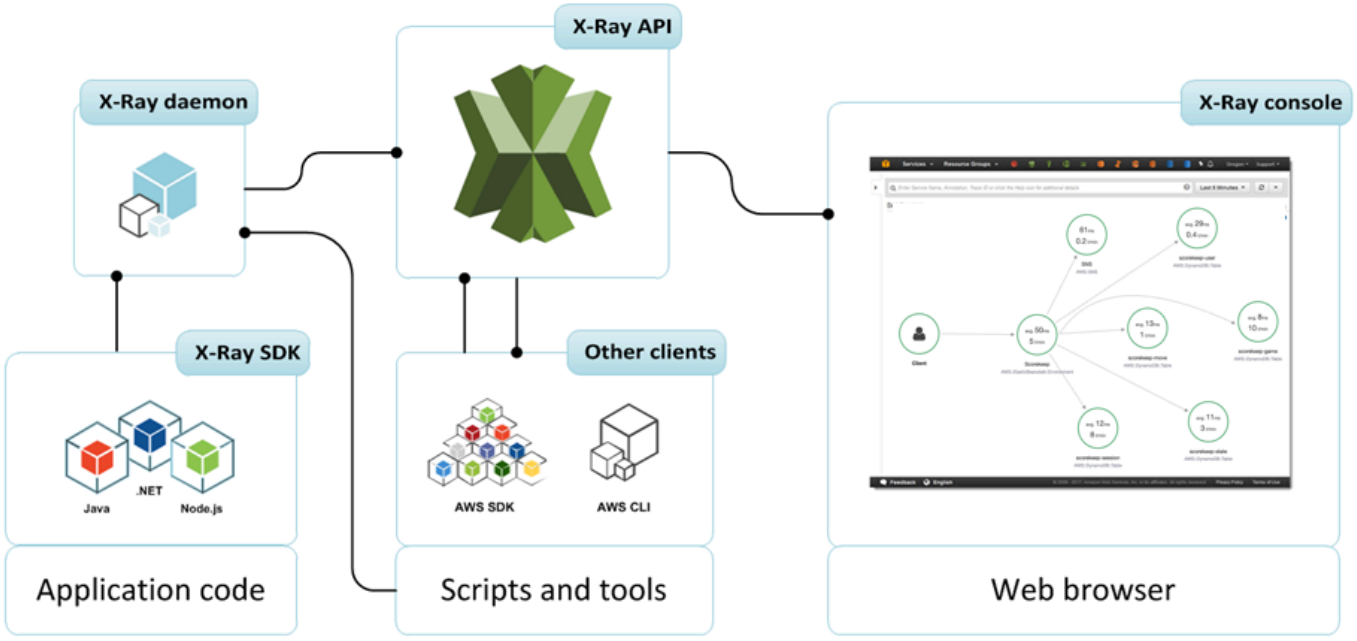
Che cos'è AWS X-Ray?

AWS X-Ray è un servizio che raccoglie dati sulle richieste servite dall'applicazione e fornisce strumenti che è possibile utilizzare per visualizzare, filtrare e acquisire informazioni su tali dati per identificare problemi e opportunità di ottimizzazione. Per ogni richiesta tracciata all'applicazione, è possibile visualizzare informazioni dettagliate non solo sulla richiesta e sulla risposta, ma anche sulle chiamate effettuate dall'applicazione a AWS risorse downstream, microservizi, database e API Web.



AWS X-Ray riceve tracce dall'applicazione, oltre agli usi dell' Servizi AWS applicazione che sono già integrati con X-Ray. La strumentazione dell'applicazione comporta l'invio di dati di traccia per le richieste in entrata e in uscita e altri eventi all'interno dell'applicazione, insieme ai metadati relativi a ciascuna richiesta. Molti scenari di analisi richiedono solo modifiche alla configurazione. Ad esempio, potete indirizzare tutte le richieste HTTP in entrata e le chiamate downstream in base a quelle effettuate dall'applicazione Java. Servizi AWS Esistono diversi SDK, agenti e strumenti che possono essere utilizzati per strumentare l'applicazione per il tracciamento X-Ray. Per ulteriori informazioni, consulta Strumentazione dell'applicazione.

Servizi AWS che sono integrati con X-Ray possono aggiungere intestazioni di tracciamento alle richieste in arrivo, inviare dati di traccia a X-Ray o eseguire il demone X-Ray. Ad esempio, AWS Lambda può inviare dati di traccia sulle richieste alle funzioni Lambda ed eseguire il demone X-Ray sui worker per semplificare l'utilizzo dell'SDK X-Ray.



Invece di inviare i dati di traccia direttamente a X-Ray, ogni client SDK invia i documenti del segmento JSON a un processo daemon che ascolta il traffico UDP. Il demone X-Ray memorizza i segmenti in una coda e li carica su X-Ray in batch. Il daemon è disponibile per Linux, Windows e macOS ed è incluso nelle piattaforme e. AWS Elastic Beanstalk AWS Lambda

X-Ray utilizza i dati di traccia provenienti dalle AWS risorse che alimentano le applicazioni cloud per generare una mappa di tracciamento dettagliata. La mappa di tracciamento mostra il client, il servizio front-end e i servizi di backend richiamati dal servizio front-end per elaborare le richieste e conservare i dati. Utilizza la mappa di tracciamento per identificare strozzature, picchi di latenza e altri problemi da risolvere per migliorare le prestazioni delle applicazioni.

Guida introduttiva a X-Ray

Per iniziare con AWS X-Ray:

- Avvia un'applicazione di esempio già dotata di strumenti per generare dati di traccia. In pochi minuti, puoi avviare l'app di esempio, generare traffico, inviare segmenti a X-Ray e visualizzare una mappa di tracciamento e le tracce in. AWS Management Console
- Scopri come strumentare la tua applicazione, ad esempio utilizzando gli X-Ray SDK o AWS Distro per inviare dati di traccia OpenTelemetry a X-Ray.
- Esplora altre Servizi AWSfunzionalità integrate con X-Ray, tra cui il campionamento e l'aggiunta di intestazioni alle richieste in arrivo, l'esecuzione del demone X-Ray e l'invio automatico dei dati di traccia a X-Ray.
- Utilizza l'API X-Ray, che fornisce l'accesso a tutte le funzionalità X-Ray tramite l' AWS SDK o direttamente tramite HTTPS AWS Command Line Interface.

AWS X-Ray concetti

AWS X-Ray riceve i dati dai servizi sotto forma di segmenti. X-Ray raggruppa quindi i segmenti che hanno una richiesta comune in tracce. X-Ray elabora le tracce per generare un grafico di servizio che fornisce una rappresentazione visiva dell'applicazione.

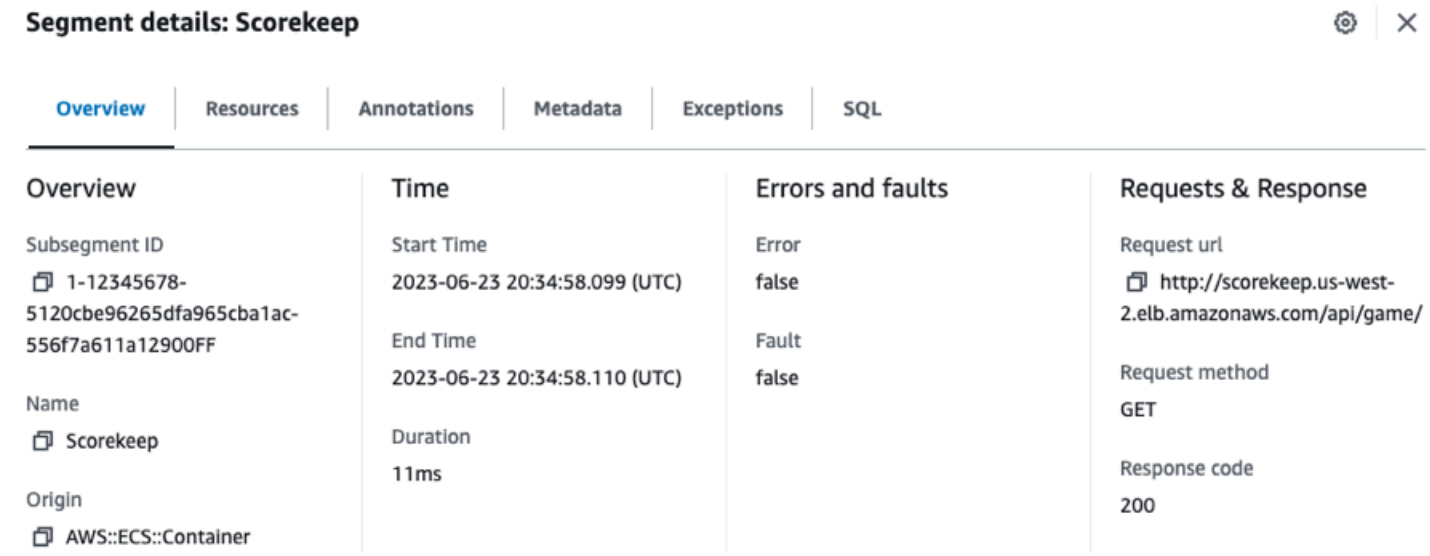
Concetti

- Segmenti
- Sottosegmenti
- Grafico del servizio
- Tracce
- Campionamento
- Intestazione di tracciamento
- Espressioni filtro
- Gruppi
- Annotazioni e metadati
- Errori, malfunzionamenti ed eccezioni

Segmenti

Le risorse di elaborazione che eseguono la logica dell'applicazione inviano i dati sulle loro attività sotto forma di segmenti. Un segmento contiene il nome della risorsa, i dettagli relativi alla richiesta e i dettagli sull'attività eseguita. Ad esempio, quando una richiesta HTTP raggiunge l'applicazione, il segmento può memorizzare i seguenti dati:

- L'host: nome host, alias o indirizzo IP
- La richiesta: metodo, indirizzo del client, percorso, agente utente
- La risposta: stato, contenuto
- Il lavoro svolto: orari di inizio e fine, sottosegmenti
- Problemi che si verificano: errori, errori ed eccezioni, inclusa l'acquisizione automatica di pile di eccezioni.



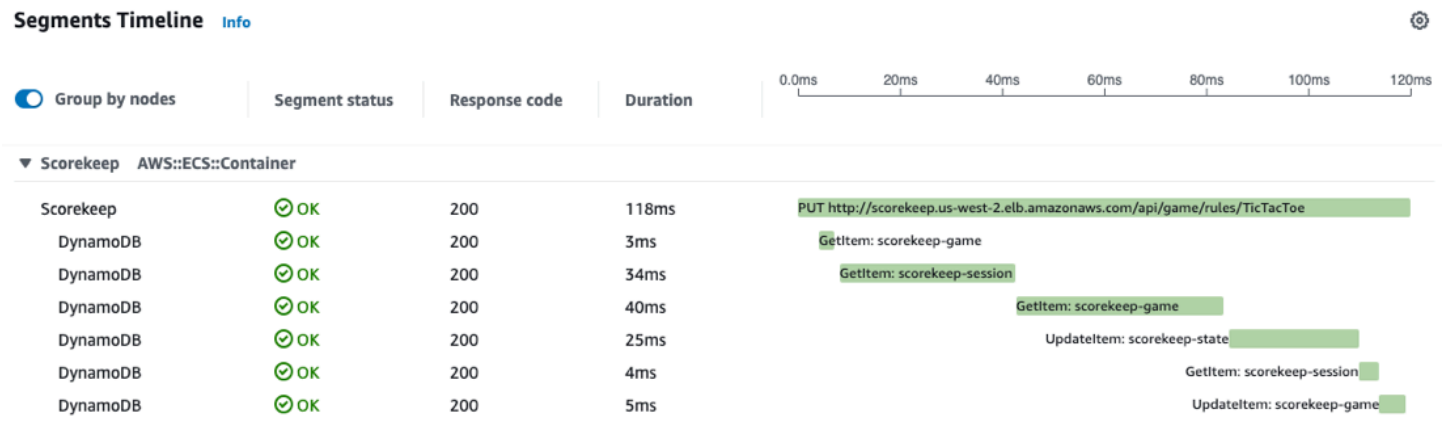
L'SDK X-Ray raccoglie informazioni dalle intestazioni di richiesta e risposta, dal codice dell'applicazione e dai metadati sulle risorse su cui viene AWS eseguita. Scegliete i dati da raccogliere modificando la configurazione o il codice dell'applicazione per strumentare le richieste in entrata, le richieste downstream e i client SDK. AWS

> **Richieste inoltrate**
>
> Se un sistema di bilanciamento del carico o un altro intermediario inoltra una richiesta all'applicazione, X-Ray prende l'IP del client dall'`X-Forwarded-For`intestazione della richiesta anziché dall'IP di origine nel pacchetto IP. L'IP del client registrato per una richiesta inoltrata può essere falsificato, quindi non dovrebbe essere considerato attendibile.

È possibile utilizzare X-Ray SDK per registrare informazioni aggiuntive come annotazioni e metadati. Per ulteriori informazioni sulla struttura e le informazioni memorizzate nei segmenti e nei sottosegmenti, consulta AWS X-Ray segmentare i documenti. I documenti dei segmenti possono avere dimensioni fino a 64 kB.

Sottosegmenti

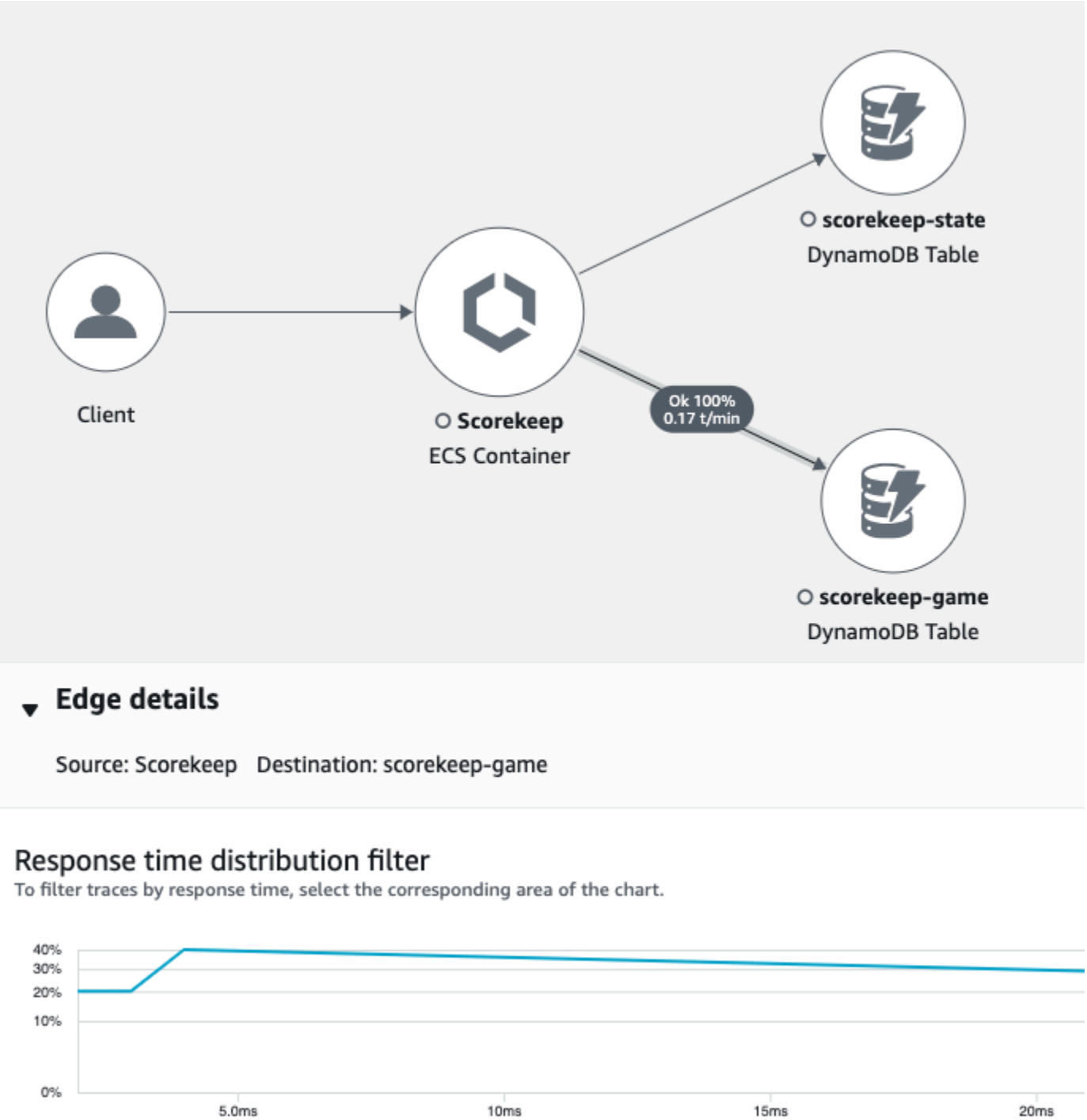
Un segmento può suddividere i dati sull'attività eseguita in sottosegmenti. I sottosegmenti offrono una maggiore granularità delle informazioni temporali e più dettagli sulle chiamate a valle invocate dall'applicazione per soddisfare la richiesta originale. Un sottosegmento può contenere dettagli aggiuntivi su una chiamata a un' Servizio AWS API HTTP esterna o a un database SQL. Puoi anche definire sottosegmenti arbitrari per analizzare specifiche funzioni o righe di codice nell'applicazione.



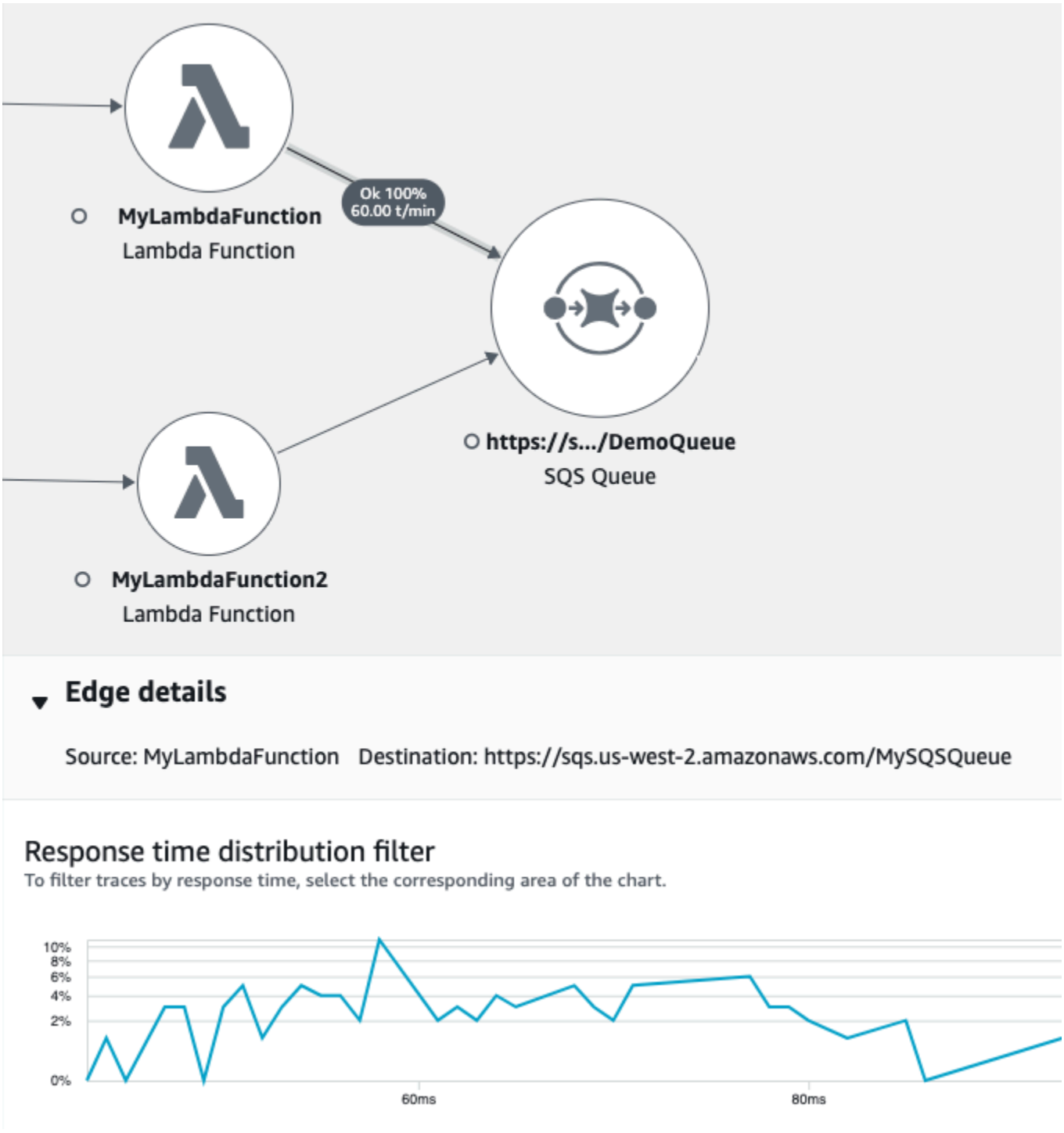
Per i servizi che non inviano segmenti propri, come Amazon DynamoDB, X-Ray utilizza sottosegmenti per generare segmenti dedotti e nodi downstream sulla mappa di traccia. Questo ti consente di visualizzare tutte le dipendenze a valle, anche se non supportano il tracciamento o sono risorse esterne.

I sottosegmenti rappresentano la vista di una chiamata a valle da parte dell'applicazione che opera come client. Se anche il servizio a valle è analizzato, il segmento che questo invia sostituisce il segmento dedotto generato dal segmento del client a monte. Il nodo sul grafo del servizio utilizza sempre le informazioni del segmento del servizio, se disponibili, mentre il collegamento tra due nodi utilizza le informazioni del sottosegmento del servizio a monte.

Ad esempio, quando si chiama DynamoDB con un client SDK AWS strumentato, l'SDK X-Ray registra un sottosegmento per quella chiamata. DynamoDB non invia un segmento, quindi il segmento dedotto nella traccia, il nodo DynamoDB sul grafico del servizio e l'edge tra il servizio e DynamoDB contengono tutti informazioni dal sottosegmento.



Quando chiami un altro servizio analizzato utilizzando con un'applicazione analizzata, il servizio a valle invia proprio segmento per memorizzare la sua vista della stessa chiamata che il servizio a monte ha memorizzato in un sottosegmento. Nel grafo del servizio, entrambi i nodi del servizio contengono le informazioni su tempi ed errori provenienti dai segmenti di tali servizi, mentre il collegamento tra essi è basato sulle informazioni provenienti dal sottosegmento del servizio a monte.



Entrambi i punti di vista sono utili, poiché il servizio a valle memorizza esattamente l'ora di inizio e fine dell'elaborazione della richiesta, mentre il servizio a monte memorizza la latenza totale che include il tempo che la richiesta ha consumato nel trasferimento tra i due servizi.

Grafico del servizio

X-Ray utilizza i dati inviati dall'applicazione per generare un grafico di servizio. Ogni AWS risorsa che invia dati a X-Ray appare come un servizio nel grafico. I collegamenti connettono i servizi che collaborano per elaborare le richieste. I collegamenti connettono i client alla tua applicazione e la tua applicazione ai servizi e alle risorse a valle utilizzate.

> **Nomi dei servizi**
>
> I segmenti `name` devono corrispondere al nome di dominio o al nome logico del servizio che genera il segmento. Tuttavia, questo non viene applicato. Qualsiasi applicazione autorizzata `PutTraceSegments`può inviare segmenti con qualsiasi nome.

Un grafo del servizio è un documento in formato JSON che contiene le informazioni sui servizi e sulle risorse che costituiscono la tua applicazione. La console X-Ray utilizza il grafico del servizio per generare una visualizzazione o una mappa dei servizi.

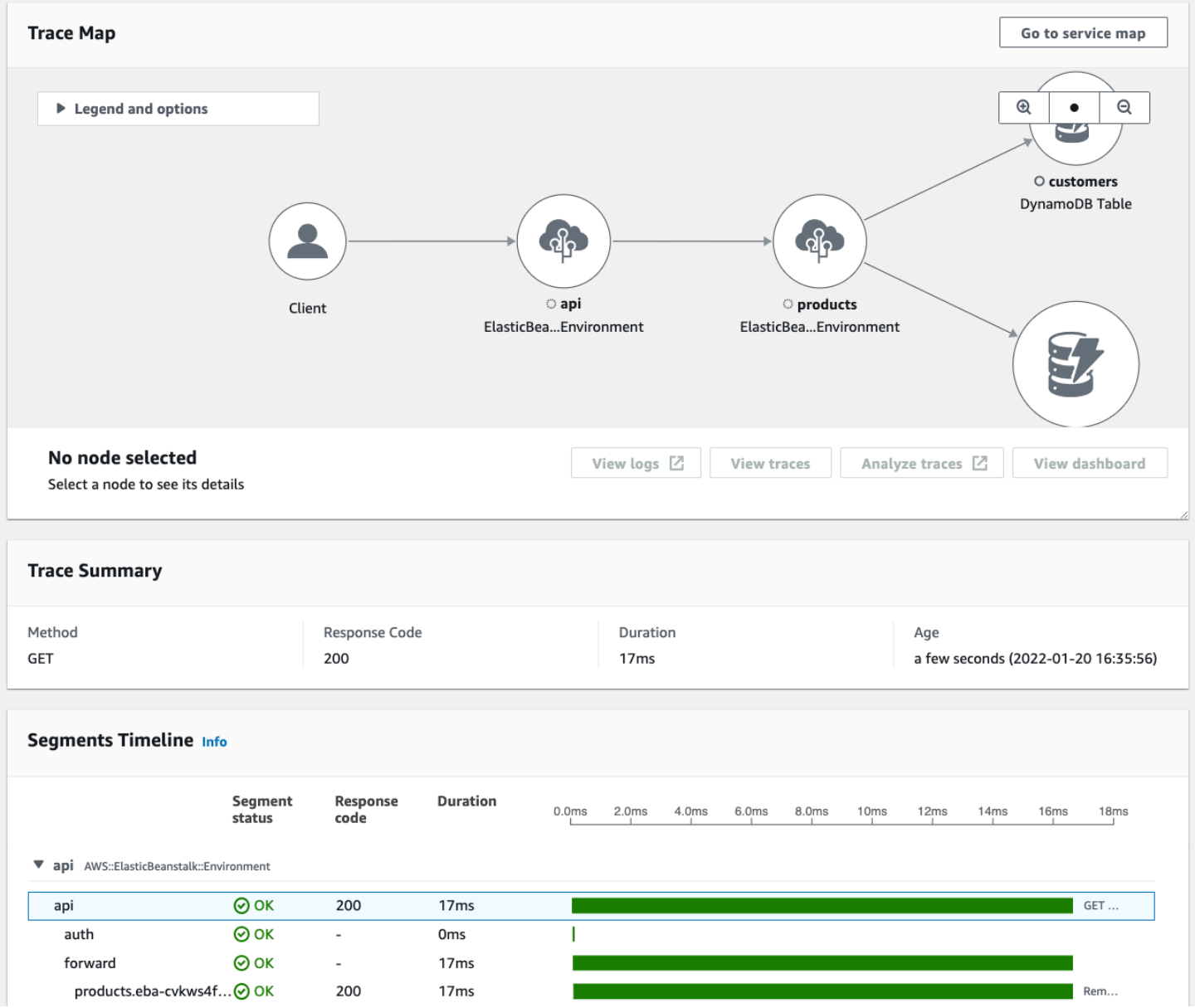
Per un'applicazione distribuita, X-Ray combina i nodi di tutti i servizi che elaborano le richieste con lo stesso ID di traccia in un unico grafico di servizio. Il primo servizio a cui accede la richiesta aggiunge un'intestazione di tracciamento che viene propagata tra il front-end e servizi che vengono invocati.

Ad esempio, Scorekeep esegue un'API web che invoca un microservizio (una funzione AWS Lambda) per generare un nome casuale utilizzando una libreria di Node.js. L'X-Ray SDK for Java genera l'ID di traccia e lo include nelle chiamate a Lambda. Lambda invia i dati di tracciamento e trasmette l'ID di traccia alla funzione. L'X-Ray SDK per Node.js utilizza anche l'ID di traccia per inviare dati. Di conseguenza, i nodi per l'API, il servizio Lambda e la funzione Lambda appaiono tutti come nodi separati, ma connessi, sulla mappa di traccia.

I dati del grafico dei servizi vengono conservati per 30 giorni.

Tracce

Un ID di tracciamento traccia il percorso di una richiesta attraverso l'applicazione. Un tracciamento raccoglie tutti i segmenti generati da una singola richiesta. Questa richiesta è in genere una richiesta HTTP GET o POST che attraversa un sistema di bilanciamento del carico, raggiunge il codice dell'applicazione e genera chiamate downstream verso altri AWS servizi o API Web esterne. Il primo servizio supportato con cui la richiesta HTTP interagisce aggiunge un'intestazione contenente un ID di tracciamento alla richiesta e la propaga a valle per tracciare latenza, completamento e altri dati relativi alla richiesta.



Consulta AWS X-Ray i prezzi per informazioni sulla fatturazione delle tracce X-Ray. I dati di tracciamento vengono conservati per 30 giorni.

Campionamento

Per garantire un tracciamento efficiente e fornire un campione rappresentativo delle richieste servite dall'applicazione, l'SDK X-Ray applica un algoritmo di campionamento per determinare quali richieste vengono tracciate. Per impostazione predefinita, l'SDK X-Ray registra la prima richiesta ogni secondo e il cinque percento di eventuali richieste aggiuntive.

Per evitare di incorrere in costi del servizio durante l'acquisizione delle nozioni di base, il tasso di campionamento di default è conservativo. È possibile configurare X-Ray per modificare la regola di campionamento predefinita e configurare regole aggiuntive che applicano il campionamento in base alle proprietà del servizio o della richiesta.

Ad esempio, potresti voler disabilitare il campionamento e tracciare tutte le richieste di chiamate che modificano lo stato o gestiscono utenti o transazioni. In caso di elevati volumi di chiamate di sola lettura, come il polling in background, i controlli dello stato della connessione o la manutenzione della connessione, puoi campionare con una bassa percentuale e continuare comunque a disporre di un numero sufficiente di dati per rilevare eventuali problemi.

Per ulteriori informazioni, consulta Configurazione delle regole di campionamento di .

Intestazione di tracciamento

Tutte le richieste sono tracciate, fino a un valore minimo configurabile. Dopo aver raggiunto tale minimo, viene tracciata solo una percentuale delle richieste per evitare costi non necessari. La selezione per il campionamento e l'ID di tracciamento vengono aggiunti alle richieste HTTP nelle intestazioni di tracciamento denominate `X-Amzn-Trace-Id`. Il primo servizio integrato a raggi X raggiunto dalla richiesta aggiunge un'intestazione di tracciamento, che viene letta dall'SDK X-Ray e inclusa nella risposta.

Example Intestazione di tracciamento con ID di tracciamento root e selezione per il campionamento

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

> **Sicurezza dell'intestazione di tracciamento**
>
> Un'intestazione di tracciamento può provenire da X-Ray SDK Servizio AWS, an o dalla richiesta del client. La tua applicazione può rimuovere l'intestazione `X-Amzn-Trace-Id` dalle richieste in entrata per evitare problemi causati dall'aggiunta alle richieste di ID di tracciamento e selezione per il campionamento da parte degli utenti.

Se la richiesta proviene da un'applicazione analizzata, l'intestazione di tracciamento può anche contenere un ID segmento padre. Ad esempio, se l'applicazione richiama un'API Web HTTP downstream con un client HTTP strumentato, X-Ray SDK aggiunge l'ID del segmento per la richiesta originale all'intestazione di tracciamento della richiesta downstream. Un'applicazione analizzata che elabora la richiesta a valle può memorizzare l'ID del segmento padre per collegare le due richieste.

Example Intestazione di tracciamento con ID di tracciamento root, ID del segmento padre e selezione per il campionamento

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

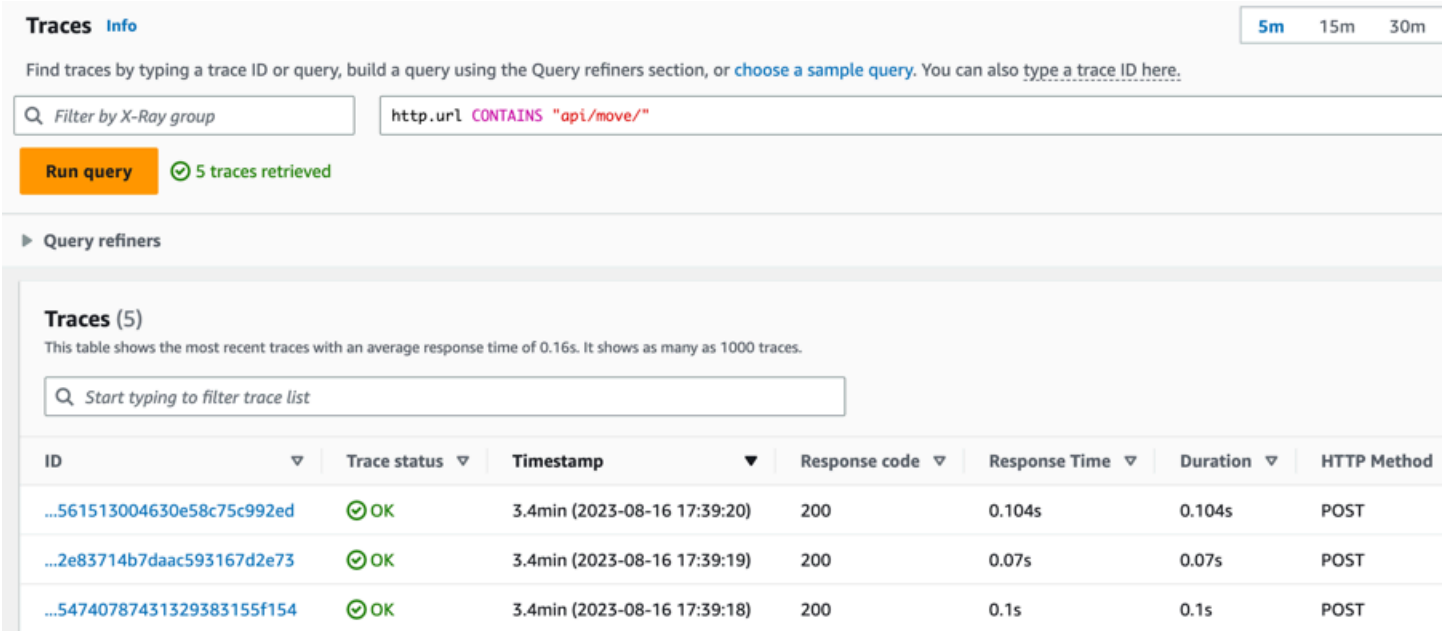
`Lineage`può essere aggiunto all'intestazione della traccia da Lambda e altri Servizi AWS come parte dei loro meccanismi di elaborazione e non deve essere utilizzato direttamente.

Example Intestazione di tracciamento con Lineage

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

Espressioni filtro

Anche utilizzando il campionamento, un'applicazione complessa genera una notevole quantità di dati. La AWS X-Ray console fornisce una easy-to-navigate visualizzazione del grafico del servizio. Mostra le informazioni di stato e prestazionali che ti permettono di identificare i problemi e le opportunità di ottimizzazione della tua applicazione. In caso di tracciamento avanzato, puoi approfondire l'analisi dei tracciamenti delle singole richieste, oppure utilizzare espressioni filtro per individuare tracciamenti correlati a specifici percorsi o utenti.



Gruppi

Estendendo le espressioni dei filtri, X-Ray supporta anche la funzione di gruppo. Con un'espressione di filtro, è possibile definire i criteri in base ai quali accettare le tracce nel gruppo.

Puoi chiamare il gruppo per nome o tramite Amazon Resource Name (ARN) per generare il proprio grafico dei servizi, i riepiloghi delle tracce e i parametri Amazon. CloudWatch Una volta creato un gruppo, le tracce in entrata vengono confrontate con l'espressione del filtro del gruppo mentre vengono archiviate nel servizio X-Ray. Le metriche relative al numero di tracce che corrispondono a ciascun criterio vengono pubblicate ogni minuto. CloudWatch

L'aggiornamento dell'espressione di filtro di un gruppo non modifica i dati già registrati. L'aggiornamento si applica solo per le successive tracce. Ciò può comportare un grafo unito delle espressioni nuove e vecchie. Per evitare questo problema, elimina il gruppo corrente e creane uno nuovo.

> **Note**
>
> I gruppi sono fatturati per il numero di tracce recuperate corrispondenti all'espressione di filtro. Per ulteriori informazioni, consultare Prezzi di AWS X-Ray.

Per ulteriori informazioni sui gruppi, consultaConfigurazione dei gruppi.

Annotazioni e metadati

Quando si strumenta l'applicazione, l'X-Ray SDK registra le informazioni sulle richieste in entrata e in uscita, sulle AWS risorse utilizzate e sull'applicazione stessa. Puoi aggiungere al documento di segmento altre informazioni come annotazioni e metadati. Le annotazioni e i metadati vengono aggregati a livello di traccia e possono essere aggiunti a qualsiasi segmento o sottosegmento.

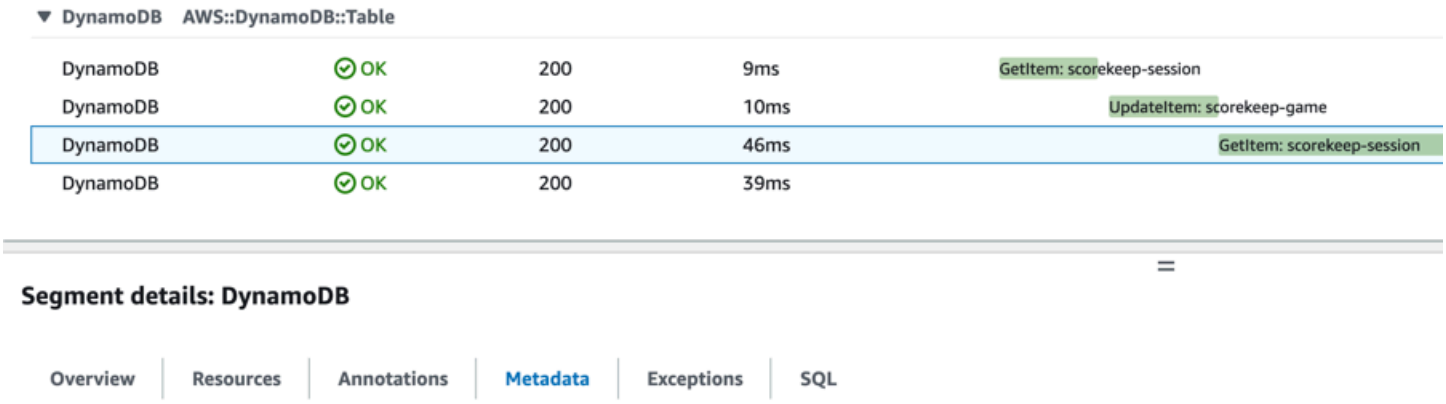
Le annotazioni sono semplici coppie chiave-valore indicizzate per l'uso con espressioni filtro. Utilizzale per registrare i dati che desideri utilizzare per raggruppare le tracce nella console oppure per chiamare l'API `GetTraceSummaries`.

X-Ray indica fino a 50 annotazioni per traccia.

I metadati sono coppie chiave-valore che possono assumere valori di qualsiasi tipo, inclusi oggetti ed elenchi, ma che non vengono indicizzate. Utilizza i metadati per registrare dati aggiuntivi che desideri

memorizzare all'interno del tracciamento ma non ti servono per effettuare ricerche all'interno dei tracciamenti.

È possibile visualizzare annotazioni e metadati nella finestra dei dettagli del segmento o del sottosegmento, all'interno della pagina dei dettagli di Trace nella console. CloudWatch



Errori, malfunzionamenti ed eccezioni

X-Ray tiene traccia degli errori che si verificano nel codice dell'applicazione e degli errori restituiti dai servizi downstream. Gli errori sono classificati come segue.

- **`Error`**— Errori del client (errori della serie 400)
- **`Fault`**— Guasti del server (errori della serie 500)
- **`Throttle`**— Errori di limitazione (429 troppe richieste)

Quando si verifica un'eccezione mentre l'applicazione sta servendo una richiesta strumentata, l'SDK X-Ray registra i dettagli sull'eccezione, inclusa la traccia dello stack, se disponibile. È possibile visualizzare le eccezioni nei dettagli dei segmenti nella console X-Ray.

AWS X-Ray consolle

Usa la AWS X-Ray console per visualizzare una mappa dei servizi e delle tracce associate per le richieste servite dalle tue applicazioni e per configurare gruppi e regole di campionamento che influiscono sul modo in cui le tracce vengono inviate a X-Ray.

> **Note**
>
> CloudWatch ora include Application Signals, che può scoprire e monitorare i servizi applicativi, i client, i canali Synthetics e le dipendenze dei servizi. Utilizza Application Signals per visualizzare un elenco o una mappa visiva dei tuoi servizi, visualizzare i parametri di integrità in base agli obiettivi del livello di servizio (SLO) e approfondire le tracce X-Ray correlate per una risoluzione dei problemi più dettagliata.
> La mappa e la mappa del servizio X-Ray sono state combinate nella CloudWatch ServiceLens mappa di tracciamento X-Ray all'interno della console Amazon. CloudWatch Apri la CloudWatch console e scegli Trace Map in X-Ray Traces dal riquadro di navigazione a sinistra.

La pagina principale della console X-Ray è la mappa di traccia, che è una rappresentazione visiva del grafico del servizio JSON generato da X-Ray a partire dai dati di traccia generati dalle applicazioni. La mappa è composta da nodi di servizio per ogni applicazione che elabora richieste nel tuo account, nodi client a monte che rappresentano le origini delle richieste e nodi di servizio a valle che rappresentano i servizi e le risorse web utilizzate da un'applicazione durante l'elaborazione della richiesta. Sono disponibili pagine aggiuntive per visualizzare le tracce e i dettagli delle tracce e configurare gruppi e regole di campionamento.

Alla scoperta della console X-Ray

- Utilizzo della mappa di tracciamento a raggi X
- Visualizzazione delle tracce e dei dettagli delle tracce
- Utilizzo delle espressioni di filtro
- Tracciamento tra account
- Tracciamento di applicazioni basate sugli eventi
- Utilizzo degli istogrammi di latenza
- Utilizzo di X-Ray Insights

- Interazione con la console di analisi
- Configurazione dei gruppi
- Configurazione delle regole di campionamento di
- Deep linking tra console

Utilizzo della mappa di tracciamento a raggi X

Visualizza la mappa di tracciamento X-Ray per identificare i servizi in cui si verificano errori, le connessioni con latenza elevata o le tracce delle richieste che non hanno avuto esito positivo.

> **Note**
>
> CloudWatch ora include Application Signals, in grado di rilevare e monitorare i servizi applicativi, i client, i synthetics canary e le dipendenze dei servizi. Utilizza Application Signals per visualizzare un elenco o una mappa visiva dei tuoi servizi, visualizzare i parametri di integrità in base agli obiettivi del livello di servizio (SLO) e approfondire le tracce X-Ray correlate per una risoluzione dei problemi più dettagliata.
> La mappa e la mappa del servizio X-Ray vengono combinate nella CloudWatch ServiceLens mappa di tracciamento X-Ray all'interno della console Amazon. CloudWatch Apri la CloudWatch console e scegli Trace Map in X-Ray Traces dal riquadro di navigazione a sinistra.

Visualizzazione della mappa di tracciamento

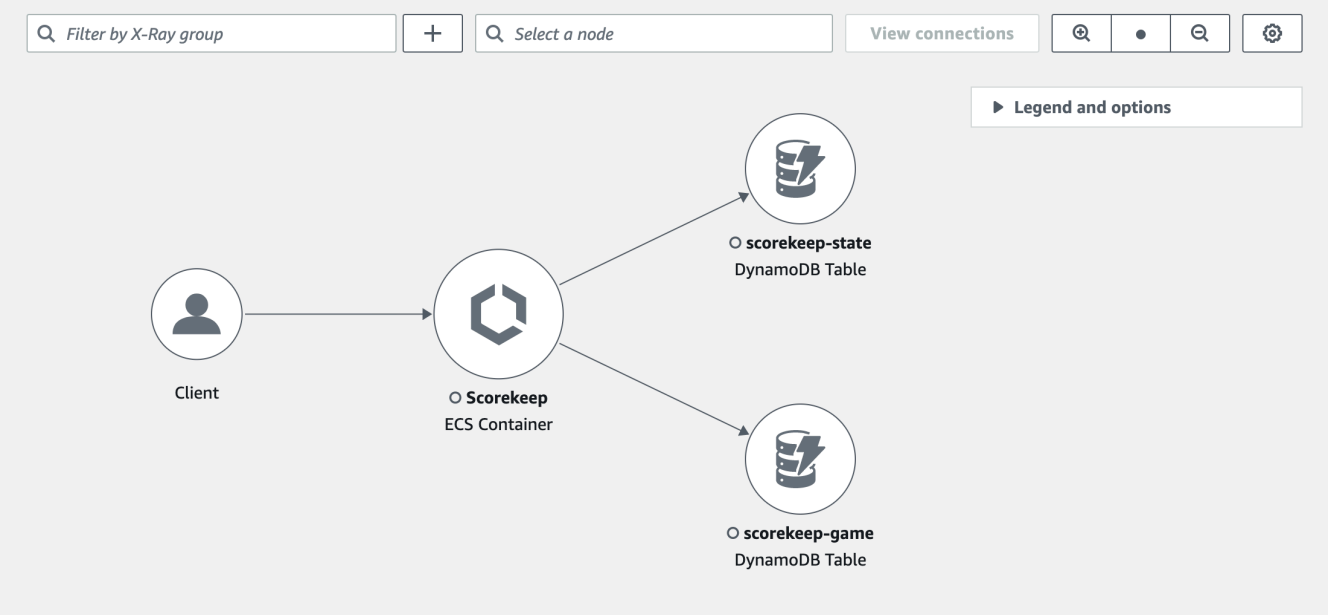
La mappa di traccia è una rappresentazione visiva dei dati di traccia generati dalle tue applicazioni. La mappa mostra i nodi di servizio che soddisfano le richieste, i nodi client upstream che rappresentano le origini delle richieste e i nodi di servizio downstream che rappresentano i servizi e le risorse Web utilizzati da un'applicazione durante l'elaborazione di una richiesta.

La mappa di traccia mostra una vista connessa delle tracce tra applicazioni basate su eventi che utilizzano Amazon SQS e Lambda. Per ulteriori informazioni, consulta la sezione di tracciamento delle applicazioni basate sugli eventi. La mappa di tracciamento supporta anche il tracciamento tra account, visualizzando i nodi di più account in un'unica mappa.

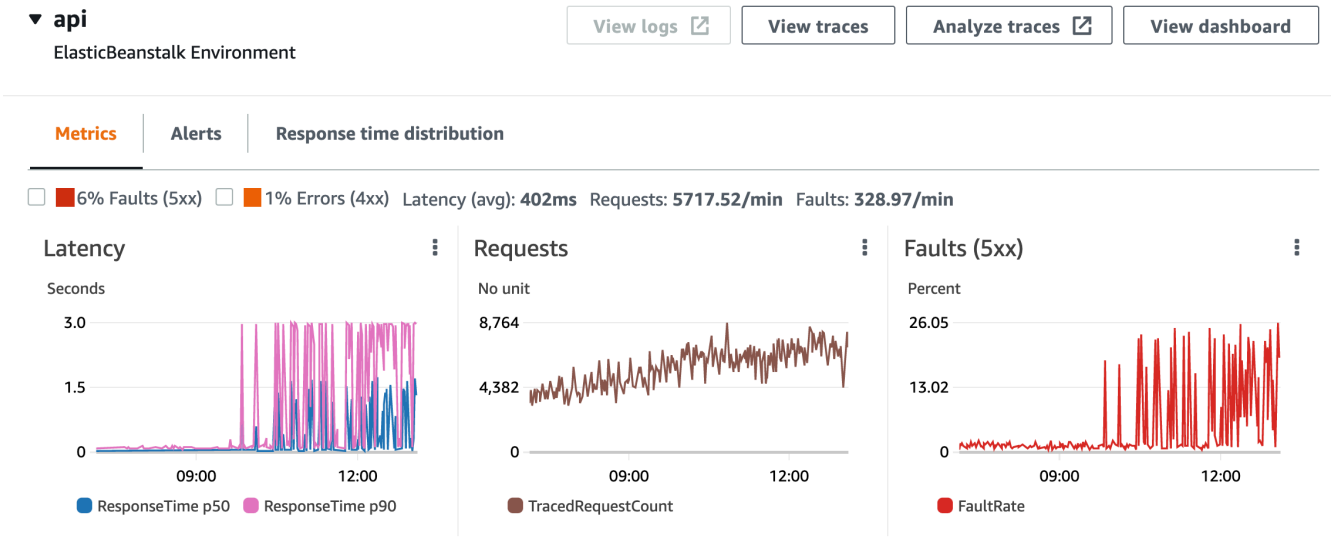
CloudWatch console

Per visualizzare la mappa di tracciamento nella console CloudWatch

1. Apri la CloudWatch console. Scegli Trace Map nella sezione X-Ray Traces nel riquadro di navigazione a sinistra.



2. Scegliere un nodo di servizio per visualizzare le richieste per tale nodo o un collegamento tra due nodi per visualizzare le richieste che sono passate attraverso tale connessione.
3. Sotto la mappa di traccia vengono visualizzate informazioni aggiuntive, incluse le schede per le metriche, gli avvisi e la distribuzione dei tempi di risposta. Nella scheda Metriche, seleziona un intervallo all'interno di ciascun grafico per visualizzare maggiori dettagli oppure scegli le opzioni Guasti o Errori per filtrare le tracce. Nella scheda Distribuzione del tempo di risposta, seleziona un intervallo all'interno del grafico per filtrare le tracce in base al tempo di risposta.



4. Visualizza le tracce scegliendo Visualizza le tracce oppure, se è stato applicato un filtro, scegli Visualizza le tracce filtrate.
5. Scegli Visualizza registri per visualizzare i CloudWatch log associati al nodo selezionato. Non tutti i nodi della mappa di traccia supportano la visualizzazione dei log. Per ulteriori informazioni, consulta CloudWatch i log di risoluzione dei problemi.

La mappa di tracciamento indica i problemi all'interno di ciascun nodo delineandoli con colori:

- Rosso per i malfunzionamenti del server (errori serie 500)
- Giallo per gli errori client (errori serie 400)
- Viola per gli errori di superamento dei limiti (429 Too Many Requests)

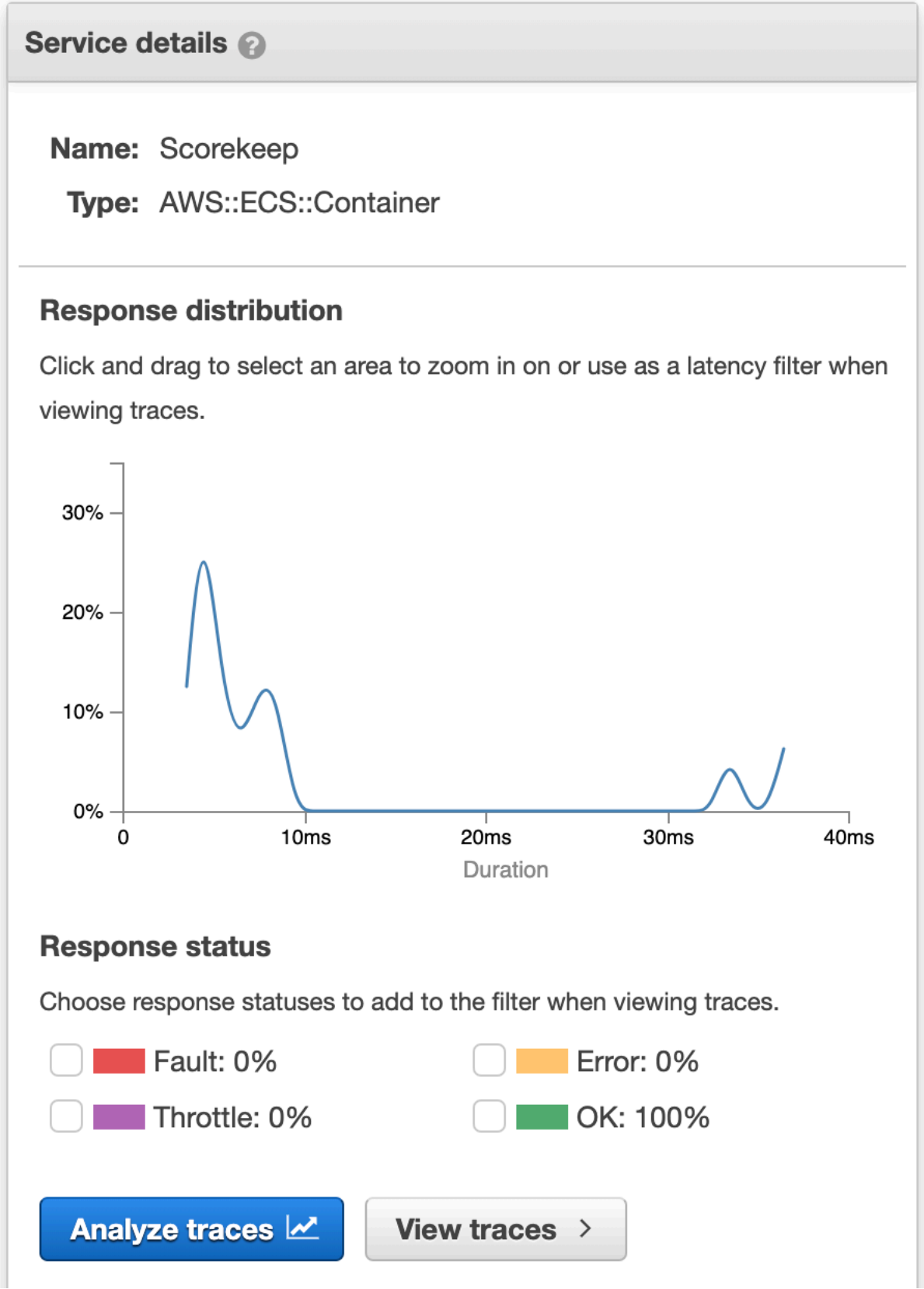
Se la mappa di tracciamento è grande, usa i controlli sullo schermo o il mouse per ingrandire e rimpicciolire e spostare la mappa.

X-Ray console

Per visualizzare la mappa dei servizi

1. Apri la console X-Ray. La mappa dei servizi viene visualizzata per impostazione predefinita. Puoi anche scegliere Service Map dal riquadro di navigazione a sinistra.

2. Scegliere un nodo di servizio per visualizzare le richieste per tale nodo o un collegamento tra due nodi per visualizzare le richieste che sono passate attraverso tale connessione.
3. Utilizza l'istogramma di distribuzione delle risposte per filtrare le tracce in base alla durata e seleziona i codici di stato per i quali desideri visualizzare le tracce. Quindi scegliere View traces (Visualizza tracciamenti) per aprire l'elenco dei tracciamenti con l'espressione filtro applicata.


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

La mappa del servizio indica lo stato di salute di ogni nodo colorandolo in base al rapporto delle chiamate concluse con successo rispetto a errori e malfunzionamenti:

- Verde per le chiamate concluse con successo
- Rosso per i malfunzionamenti del server (errori serie 500)
- Giallo per gli errori client (errori serie 400)
- Viola per gli errori di superamento dei limiti (429 Too Many Requests)

Se la mappa dei servizi è ampia, usa i controlli sullo schermo o il mouse per ingrandire e rimpicciolire e spostare la mappa.

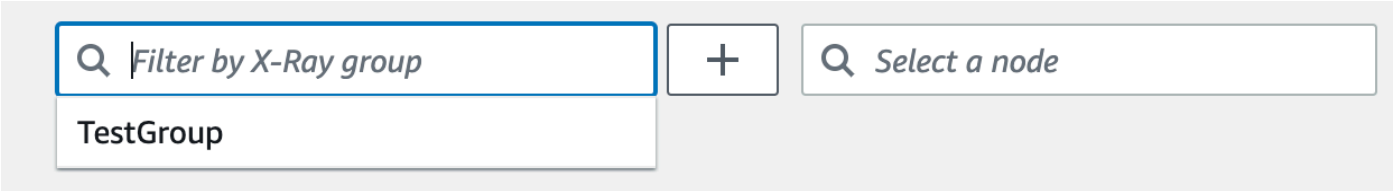
> **Note**
>
> La mappa di tracciamento a raggi X può visualizzare fino a 10.000 nodi. In rari scenari in cui il numero totale di nodi di servizio supera questo limite, è possibile che venga visualizzato un errore e non sia possibile visualizzare una mappa di traccia completa nella console.

Filtraggio della mappa di traccia per gruppo

Utilizzando un'espressione di filtro, è possibile definire criteri in base ai quali includere le tracce all'interno di un gruppo. Utilizzate i seguenti passaggi per visualizzare quindi quel gruppo specifico nella mappa di traccia.

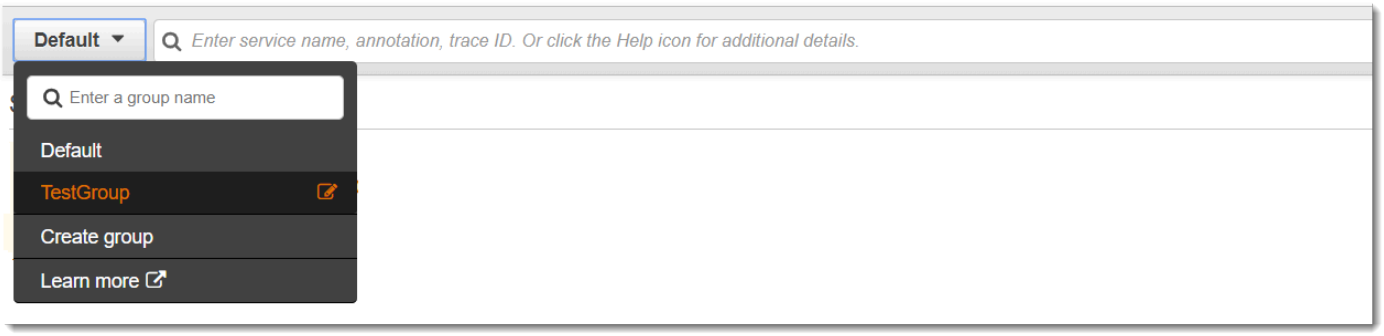
CloudWatch console

Scegli il nome di un gruppo dal filtro di gruppo in alto a sinistra della mappa di traccia.



X-Ray console

Scegliere un nome di gruppo dal menu a discesa a sinistra della barra di ricerca.



La mappa dei servizi verrà ora filtrata per visualizzare le tracce che corrispondono all'espressione del filtro del gruppo selezionato.

Traccia la legenda e le opzioni della mappa

La mappa di tracciamento include una legenda e diverse opzioni per personalizzare la visualizzazione della mappa.

CloudWatch console

Scegli il menu a discesa Legenda e opzioni in alto a destra della mappa. Scegli cosa visualizzare all'interno dei nodi, tra cui:

- Le metriche mostrano il tempo di risposta medio e il numero di tracce inviate al minuto nell'intervallo di tempo scelto.
- Nodes visualizza l'icona del servizio all'interno di ogni nodo.

Scegli impostazioni aggiuntive della mappa dal riquadro Preferenze, a cui puoi accedere tramite l'icona a forma di ingranaggio in alto a destra della mappa. Queste impostazioni includono la selezione della metrica utilizzata per determinare la dimensione di ciascun nodo e quali canarini devono essere visualizzati sulla mappa.

X-Ray console

Visualizza la legenda della mappa dei servizi scegliendo il link Legenda della mappa in alto a destra della mappa. Le opzioni della mappa dei servizi possono essere scelte in basso a destra nella mappa di traccia, tra cui:

- Service Icons alterna la visualizzazione all'interno di ciascun nodo, visualizzando l'icona del servizio o il tempo di risposta medio e il numero di tracce inviate al minuto durante l'intervallo di tempo scelto.
- Dimensionamento dei nodi: Nessuno imposta tutti i nodi alla stessa dimensione.
- Dimensionamento dei nodi: Health ridimensiona i nodi in base al numero di richieste interessate, inclusi errori, guasti o richieste limitate.
- Dimensionamento dei nodi: il traffico ridimensiona i nodi in base al numero totale di richieste.

Visualizzazione delle tracce e dei dettagli delle tracce

Usa la pagina Traces nella console X-Ray per trovare le tracce tramite URL, codice di risposta o altri dati dal riepilogo della traccia. Dopo aver selezionato una traccia dall'elenco delle tracce, la pagina dei dettagli della traccia visualizza una mappa dei nodi di servizio associati alla traccia selezionata e una sequenza temporale dei segmenti di traccia.

Visualizzazione dei tracciamenti

CloudWatch console

Per visualizzare le tracce nella console CloudWatch

1. Accedere AWS Management Console e aprire la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Nel riquadro di navigazione a sinistra, scegli Tracce a raggi X, quindi scegli Tracce. Puoi filtrare per gruppo o inserire un'espressione di filtro. Questo filtra le tracce visualizzate nella sezione Tracce nella parte inferiore della pagina.

   In alternativa, è possibile utilizzare la mappa dei servizi per accedere a un nodo di servizio specifico e quindi visualizzare le tracce. Si apre la pagina Traces con una query già applicata.
3. Perfeziona la tua query nella sezione Query refiners. Per filtrare le tracce in base a un attributo comune, scegli un'opzione dalla freccia rivolta verso il basso accanto a Refine query per. Le opzioni sono le seguenti:

   - Nodo: filtra le tracce per nodo di servizio.
   - ARN della risorsa: filtra le tracce in base a una risorsa associata a una traccia. Esempi di queste risorse includono un'istanza Amazon Elastic Compute Cloud (Amazon EC2), AWS Lambda una funzione o una tabella. Amazon DynamoDB

- Utente: filtra le tracce con un ID utente.
- Messaggio relativo alla causa principale dell'errore: filtra le tracce in base alla causa principale dell'errore.
- URL: filtra le tracce in base a un percorso URL utilizzato dall'applicazione.
- Codice di stato HTTP: filtra le tracce in base al codice di stato HTTP restituito dall'applicazione. È possibile specificare un codice di risposta personalizzato o selezionare una delle seguenti opzioni:
  - `200`— La richiesta è andata a buon fine.
  - `401`— Nella richiesta mancavano credenziali di autenticazione valide.
  - `403`— La richiesta non aveva autorizzazioni valide.
  - `404`— Il server non è riuscito a trovare la risorsa richiesta.
  - `500`— Il server ha rilevato una condizione imprevista e ha generato un errore interno.

Scegliete una o più voci, quindi scegliete Aggiungi alla query per aggiungerle all'espressione di filtro nella parte superiore della pagina.

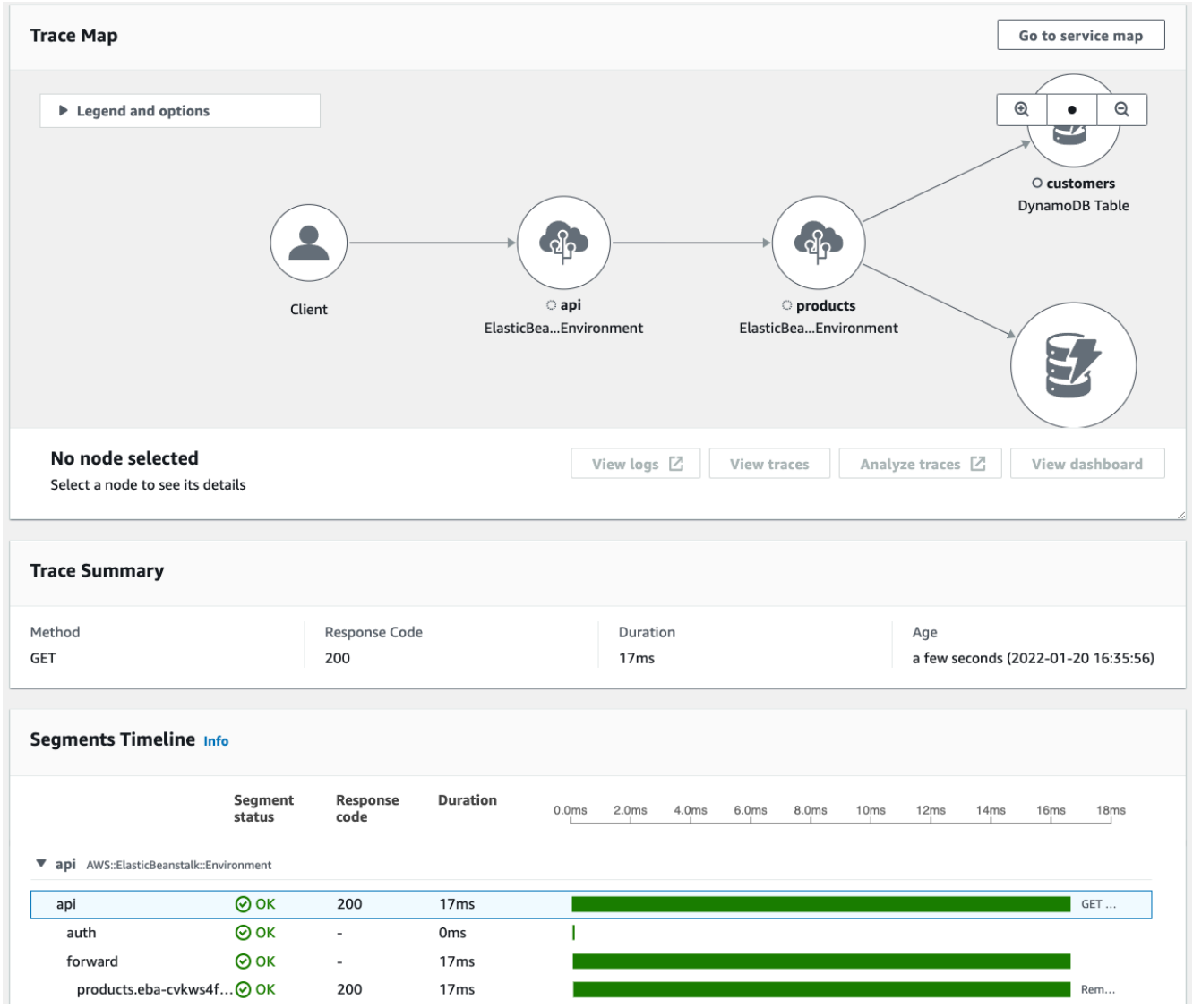
4. Per trovare una singola traccia, inserisci un ID di traccia direttamente nel campo della query. È possibile utilizzare il formato X-Ray o il formato World Wide Web Consortium (W3C). Ad esempio, una traccia creata utilizzando AWS Distro for è in formato W3C. OpenTelemetry

   > **Note**
   >
   > Quando si interrogano tracce create con un ID di traccia in formato W3C, la console visualizza la traccia corrispondente in formato X-Ray. Per esempio, se si esegue una query per `4efaaf4d1e8720b39541901950019ee5` in formato W3C, la console visualizza l'equivalente X-Ray:. `1-4efaaf4d-1e8720b39541901950019ee5`

5. Scegli Esegui query in qualsiasi momento per visualizzare un elenco di tracce corrispondenti nella sezione Tracce nella parte inferiore della pagina.
6. Per visualizzare la pagina dei dettagli di traccia per una singola traccia, seleziona un ID di traccia dall'elenco.

   L'immagine seguente mostra una mappa di Trace contenente i nodi di servizio associati alla traccia e gli spigoli tra i nodi che rappresentano il percorso seguito dai segmenti che compongono la traccia. Un riepilogo di Trace segue la Trace Map. Il riepilogo contiene informazioni su un'`GET`operazione di esempio, il relativo codice di risposta, la durata

dell'esecuzione della traccia e l'età della richiesta. La cronologia dei segmenti segue il riepilogo della traccia che mostra la durata dei segmenti e dei sottosegmenti di traccia.



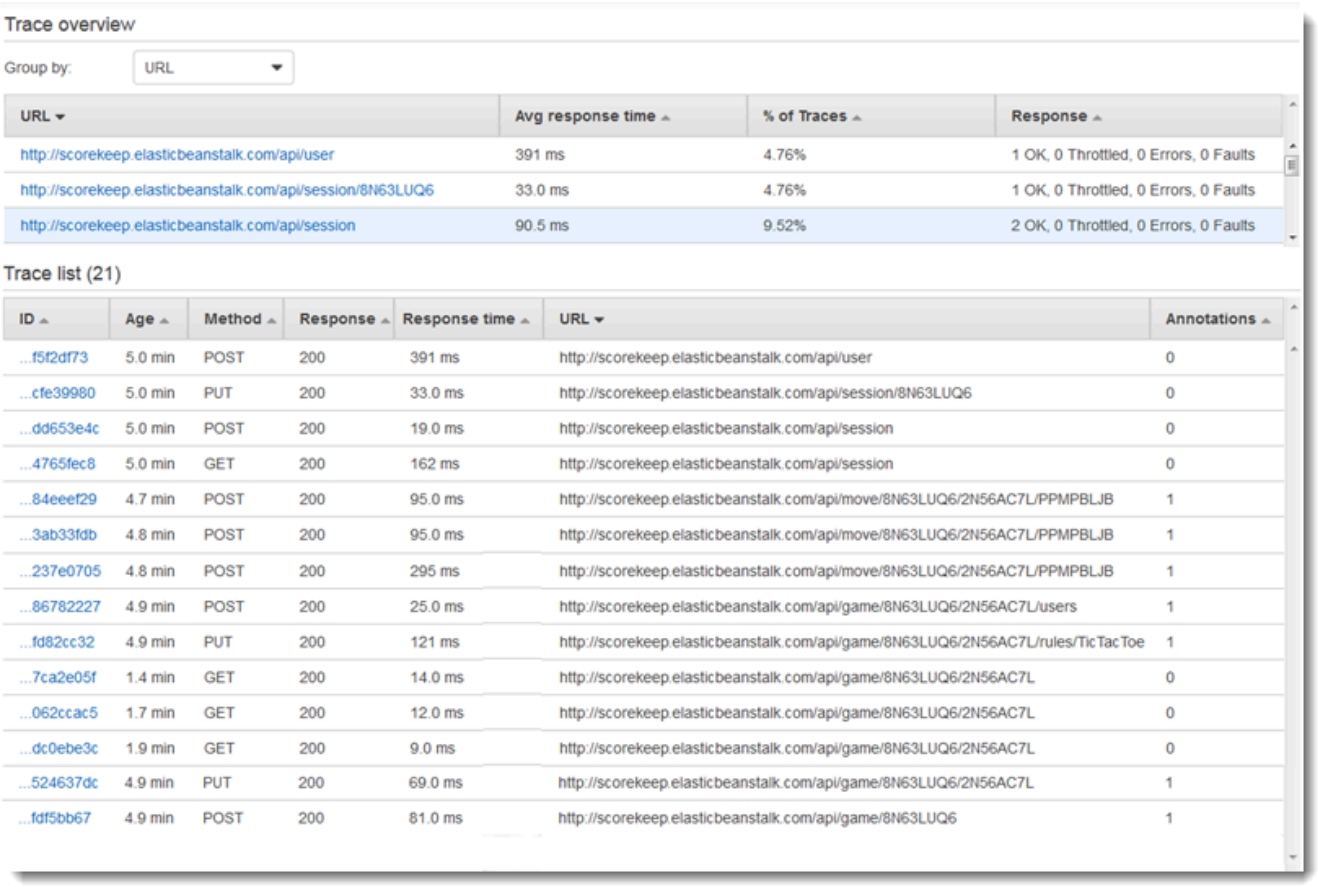
Se disponi di un'applicazione basata sugli eventi che utilizza Amazon SQS e Lambda, puoi visualizzare una vista connessa delle tracce per ogni richiesta nella mappa di Trace. Nella mappa, le tracce dei produttori di messaggi sono collegate alle tracce dei AWS Lambda consumatori e vengono visualizzate come un bordo tratteggiato. Per ulteriori informazioni sulle applicazioni basate sugli eventi, consulta. Tracciamento di applicazioni basate sugli eventi

Le pagine Traces e Trace details supportano anche il tracciamento tra account, che può elencare le tracce di più account nell'elenco delle tracce e all'interno di un'unica mappa di traccia.

X-Ray console

Per visualizzare le tracce nella console X-Ray

1. Apri la pagina Traces nella console X-Ray. Il pannello di panoramica di Trace mostra un elenco di tracce raggruppate in base a funzionalità comuni, tra cui Error root cause, resourceARN e. InstanceId
2. Per selezionare una funzionalità comune per visualizzare un insieme raggruppato di tracce, espandi la freccia rivolta verso il basso accanto a Raggruppa per. L'illustrazione seguente mostra una panoramica delle tracce raggruppate per URL per e un elenco delle tracce associate. AWS X-Ray applicazione di esempio



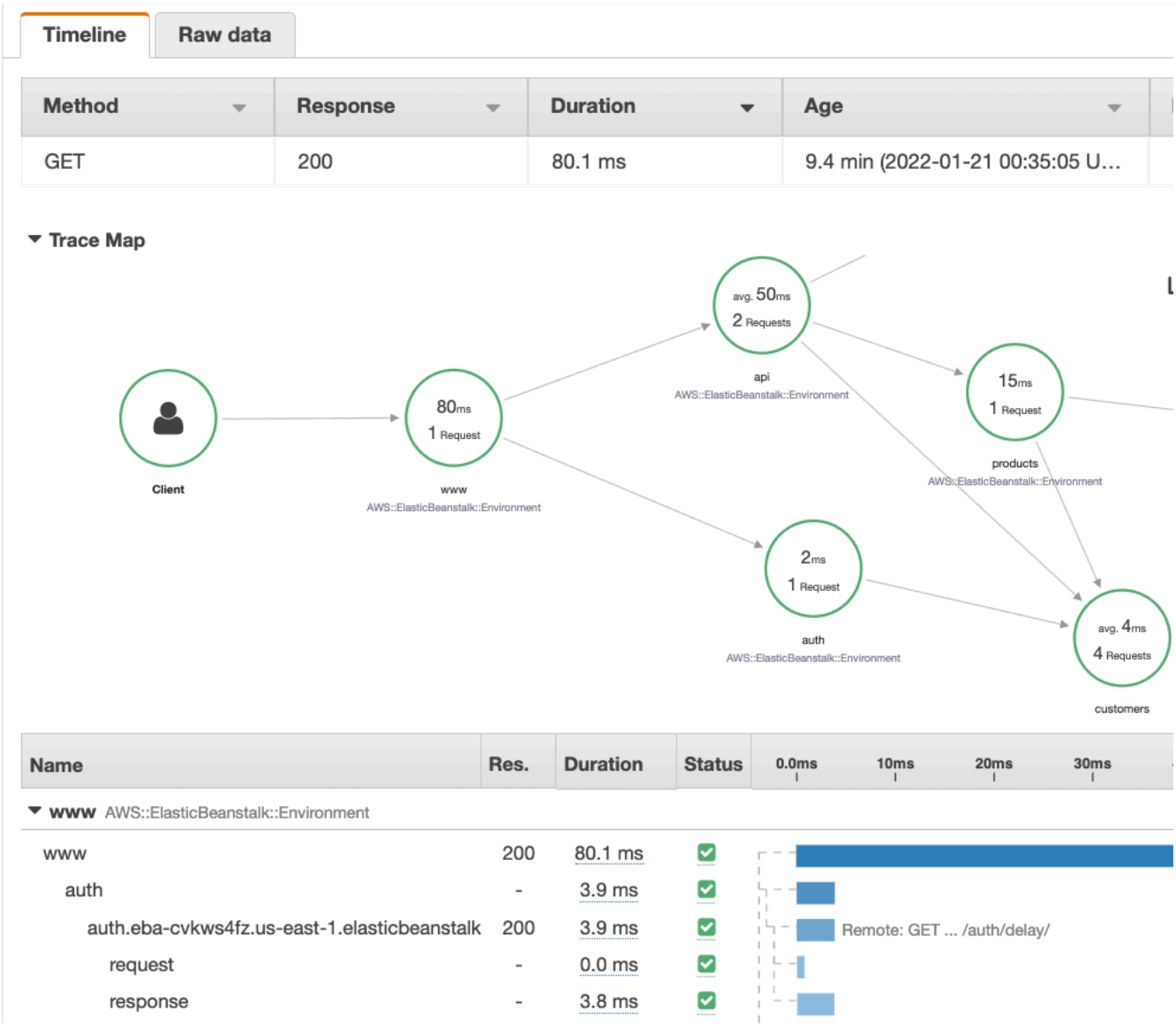
3. Scegli l'ID di una traccia per visualizzarla nell'elenco Trace. Puoi anche scegliere Mappa dei servizi nel riquadro di navigazione per visualizzare le tracce per un nodo di servizio specifico. Quindi puoi visualizzare le tracce associate a quel nodo.

   La scheda Cronologia mostra il flusso di richieste per la traccia e include quanto segue:

   - Una mappa del percorso per ogni segmento della traccia.

- Quanto tempo ha impiegato il segmento a raggiungere un nodo nella mappa di traccia.
- Quante richieste sono state fatte al nodo nella mappa di traccia.

L'illustrazione seguente mostra un esempio di Trace Map associata a una GET richiesta effettuata a un'applicazione di esempio. Le frecce mostrano il percorso seguito da ogni segmento per completare la richiesta. I nodi di servizio mostrano il numero di richieste effettuate durante la GET richiesta.



Per ulteriori informazioni sulla scheda Cronologia, consultate la seguente sezione Esplorazione della cronologia di traccia.

La scheda Dati grezzi mostra le informazioni sulla traccia e sui segmenti e sottosegmenti che la compongono, in formato. JSON Queste informazioni possono includere quanto segue:

- Timestamp
- ID univoci
- Risorse associate al segmento o al sottosegmento
- La fonte o l'origine del segmento o del sottosegmento
- Informazioni aggiuntive sulla richiesta all'applicazione, ad esempio la risposta a una richiesta HTTP

Esplorazione della sequenza temporale di tracciamento

La sezione Cronologia mostra una gerarchia di segmenti e sottosegmenti accanto a una barra orizzontale che corrisponde al tempo impiegato per completare le attività. La prima voce nell'elenco è il segmento, che rappresenta tutti i dati registrati tramite il servizio per una singola richiesta. I sottosegmenti sono rientrati ed elencati dopo il segmento. Le colonne contengono informazioni su ogni segmento.

CloudWatch console

Nella CloudWatch console, la cronologia dei segmenti fornisce le seguenti informazioni:

- La prima colonna: elenca i segmenti e i sottosegmenti nella traccia selezionata.
- La colonna Stato del segmento: elenca il risultato dello stato di ogni segmento e sottosegmento.
- La colonna Codice di risposta: elenca un codice di stato della risposta HTTP a una richiesta del browser effettuata dal segmento o dal sottosegmento, se disponibile.
- La colonna Durata: elenca la durata del segmento o del sottosegmento.
- La colonna Hosted in: elenca lo spazio dei nomi o l'ambiente in cui viene eseguito il segmento o il sottosegmento, se applicabile. Per ulteriori informazioni, consulta https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/AppSignals-StandardMetrics.html#AppSignals-StandardMetrics-Dimensions.
- L'ultima colonna: mostra le barre orizzontali che corrispondono alla durata di esecuzione del segmento o del sottosegmento, in relazione agli altri segmenti o sottosegmenti della timeline.

Per raggruppare l'elenco di segmenti e sottosegmenti per nodo di servizio, attiva Raggruppa per nodi.

X-Ray console

Nella pagina dei dettagli di traccia, scegli la scheda Cronologia per visualizzare la sequenza temporale di ogni segmento e sottosegmento che compone una traccia.

Nella console X-Ray, la Timeline fornisce le seguenti informazioni:

- La colonna Nome: elenca i nomi dei segmenti e dei sottosegmenti nella traccia.
- La colonna Res.: elenca un codice di stato della risposta HTTP a una richiesta del browser effettuata dal segmento o dal sottosegmento, se disponibile.
- La colonna Durata: elenca per quanto tempo è stato eseguito il segmento o il sottosegmento.
- La colonna Stato: elenca il risultato dello stato del segmento o del sottosegmento.
- L'ultima colonna: mostra le barre orizzontali che corrispondono alla durata di esecuzione del segmento o del sottosegmento, in relazione agli altri segmenti o sottosegmenti della timeline.

Per visualizzare i dati di traccia non elaborati utilizzati dalla console per generare la timeline, scegli la scheda Dati grezzi. I dati grezzi mostrano le informazioni sulla traccia e sui segmenti e sottosegmenti che la compongono nel formato. `JSON` Queste informazioni possono includere quanto segue:

- Timestamp
- ID univoci
- Risorse associate al segmento o al sottosegmento
- La fonte o l'origine del segmento o del sottosegmento
- Informazioni aggiuntive sulla richiesta all'applicazione, ad esempio la risposta a una richiesta HTTP.

Quando si utilizza un AWS SDK strumentato o un SQL client per effettuare chiamate a risorse esterneHTTP, l'SDK X-Ray registra automaticamente i sottosegmenti. Puoi anche usare X-Ray SDK per registrare sottosegmenti personalizzati per qualsiasi funzione o blocco di codice. I sottosegmenti aggiuntivi che vengono registrati mentre un sottosegmento personalizzato è aperto diventano figli del sottosegmento personalizzato.

Visualizzazione dei dettagli del segmento

Dalla timeline di traccia, scegliete il nome di un segmento per visualizzarne i dettagli.

Il pannello dei dettagli del segmento mostra le schede Panoramica, Risorse, Annotazioni, Metadati, Eccezioni e SQL. Si applica quanto segue:

- La scheda Overview (Panoramica) mostra le informazioni su richiesta e risposta. Le informazioni includono il nome, l'ora di inizio, l'ora di fine, la durata, l'URL della richiesta, l'operazione della richiesta, il codice di risposta alla richiesta ed eventuali errori e guasti.
- La scheda Risorse per un segmento mostra le informazioni dell'SDK X-Ray e sulle AWS risorse che eseguono l'applicazione. Utilizza i plug-in Amazon EC2 o Amazon ECS per l'SDK X-Ray per registrare informazioni sulle risorse specifiche del servizio. AWS Elastic BeanstalkPer ulteriori informazioni sui plugin, consulta la sezione Plugin di servizio in. Configurazione dell'X-Ray SDK per Java
- Le schede rimanenti mostrano le annotazioni, i metadati e le eccezioni registrate per il segmento. Le eccezioni vengono acquisite automaticamente quando vengono generate da una richiesta strumentata. Le annotazioni e i metadati contengono informazioni aggiuntive registrate utilizzando le operazioni fornite da X-Ray SDK. Per aggiungere annotazioni o metadati ai tuoi segmenti, usa l'SDK X-Ray. Per ulteriori informazioni, consulta il link specifico per la lingua elencato in Strumentazione dell'applicazione con SDK in. AWS X-Ray Strumentazione della tua applicazione per AWS X-Ray

Visualizzazione dei dettagli del sottosegmento

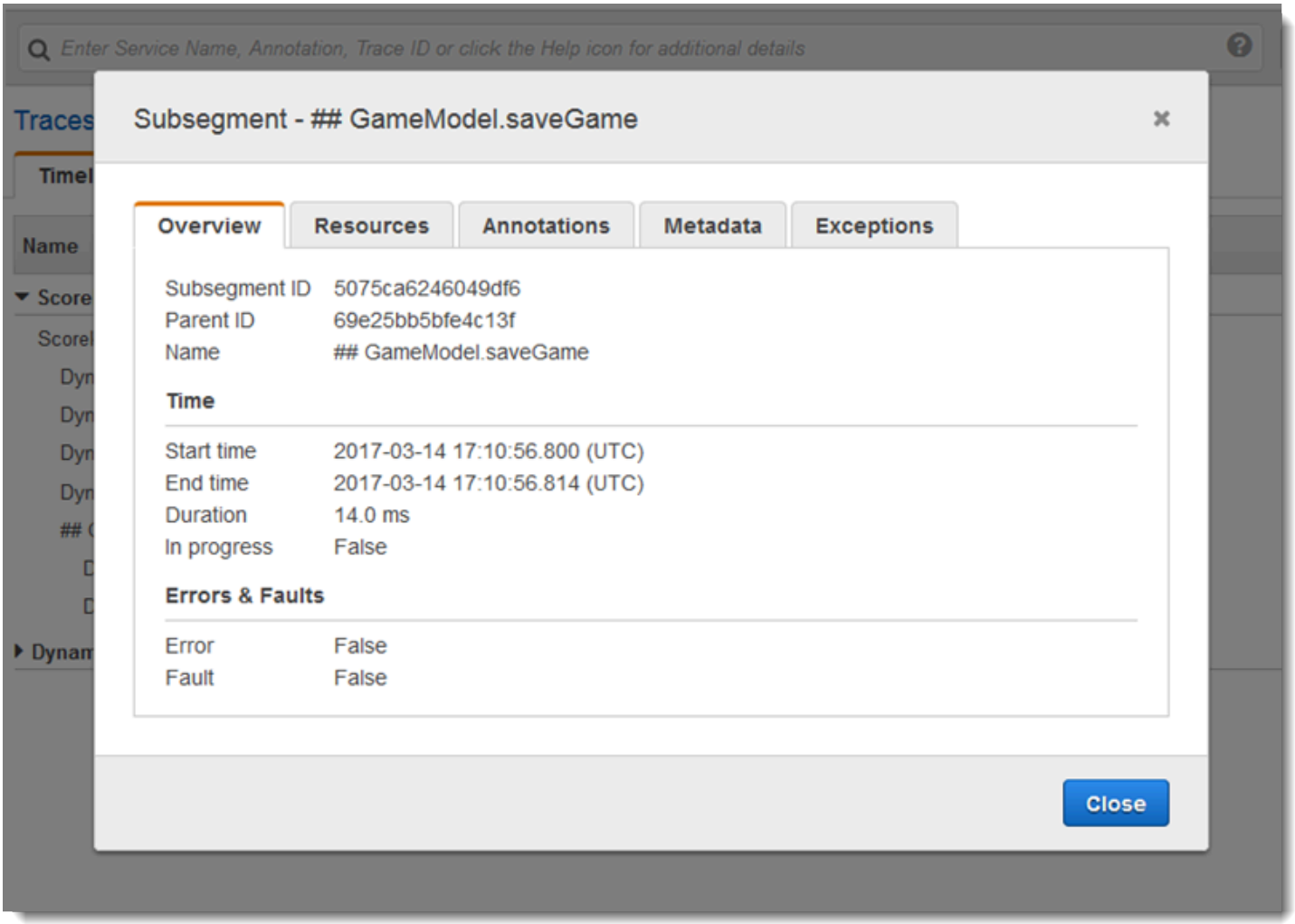
Dalla timeline di tracciamento, scegli il nome di un sottosegmento per visualizzarne i dettagli:

- La scheda Panoramica contiene informazioni sulla richiesta e sulla risposta. Ciò include il nome, l'ora di inizio, l'ora di fine, la durata, la richiestaURL, l'operazione della richiesta, il codice di risposta alla richiesta ed eventuali errori e guasti. Per i sottosegmenti generati dai client analizzati, la scheda Overview (Panoramica) contiene informazioni sulla richiesta e la risposta dal punto di vista dell'applicazione.
- La scheda Risorse per un sottosegmento mostra i dettagli sulle AWS risorse utilizzate per eseguire il sottosegmento. Ad esempio, la scheda delle risorse può includere una AWS Lambda funzione ARN, informazioni su una tabella DynamoDB, qualsiasi operazione chiamata e l'ID della richiesta.
- Le schede rimanenti mostrano le annotazioni, i metadati e le eccezioni registrate sul sottosegmento. Le eccezioni vengono acquisite automaticamente quando vengono generate

da una richiesta strumentata. Le annotazioni e i metadati contengono informazioni aggiuntive registrate utilizzando le operazioni fornite da X-Ray SDK. Usa l'SDK X-Ray per aggiungere annotazioni o metadati ai tuoi segmenti. Per ulteriori informazioni, consulta il link specifico per la lingua elencato in Strumentazione dell'applicazione con SDK in. AWS X-Ray [Strumentazione della tua applicazione per AWS X-Ray]

Nel caso di sottosegmenti personalizzati, la scheda Overview (Panoramica) mostra il nome del sottosegmento, che è possibile impostare per specificare l'area del codice o della funzione che memorizza. Per ulteriori informazioni, consulta il link specifico per la lingua riportato in Strumentazione dell'applicazione con SDK in. AWS X-Ray [Generazione di sottosegmenti personalizzati con X-Ray SDK for Java]

L'immagine seguente mostra la scheda Panoramica per un sottosegmento personalizzato. La panoramica contiene l'ID del sottosegmento, l'ID principale, il nome, l'ora di inizio e di fine, la durata, lo stato e gli errori o gli errori.



La scheda Metadati per un sottosegmento personalizzato contiene informazioni in JSON formato sulle risorse utilizzate da quel sottosegmento.

Utilizzo delle espressioni di filtro

Utilizza le espressioni di filtro per visualizzare una o più tracce per una richiesta, un servizio, una connessione tra due servizi (un edge) o richieste che soddisfano una condizione. X-Ray fornisce un linguaggio di espressione di filtro per filtrare richieste, servizi e bordi in base ai dati nelle intestazioni delle richieste, allo stato della risposta e ai campi indicizzati sui segmenti originali.

Quando scegli un periodo di tempo per le tracce da visualizzare nella console X-Ray, potresti ottenere più risultati di quelli che la console può visualizzare. Nell'angolo in alto a destra, la console mostra il numero di tracciamenti individuati e se sono disponibili ulteriori tracciamenti. È possibile utilizzare un'espressione di filtro per restringere i risultati alle sole tracce che si desidera trovare.

Argomenti

- Filtraggio dei dettagli delle espressioni
- Utilizzo delle espressioni filtro con i gruppi
- Sintassi delle espressioni filtro
- Parole chiave booleane
- Parole chiave numeriche
- Parole chiave stringa
- Parole chiave complesse
- funzione id

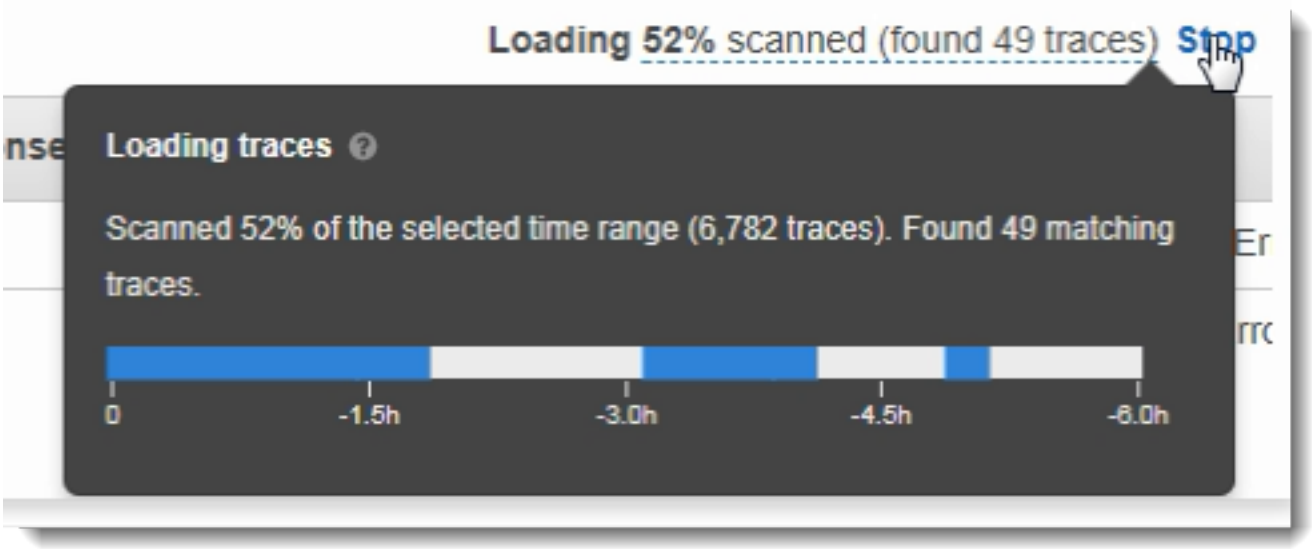
Filtraggio dei dettagli delle espressioni

Quando si sceglie un nodo nella mappa di traccia, la console crea un'espressione di filtro in base al nome di servizio del nodo e ai tipi di errore presenti in base alla selezione effettuata. Per individuare tracciamenti che mostrano problemi di prestazioni o che riguardano richieste specifiche, puoi modificare l'espressione generata dalla console oppure crearne una tua. Se aggiungi annotazioni con X-Ray SDK, puoi anche filtrare in base alla presenza di una chiave di annotazione o al valore di una chiave.

> **Note**
>
> Se scegli un intervallo di tempo relativo nella mappa di traccia e scegli un nodo, la console converte l'intervallo di tempo in un'ora di inizio e di fine assolute. Per garantire che i tracciamenti relativi al nodo vengano visualizzati tra risultati di ricerca, e per evitare la scansione dei periodi di tempo in cui il nodo non era attivo, l'intervallo di tempo include solo i periodi durante i quali il nodo ha inviato dei tracciamenti. Se desideri eseguire una ricerca rispetto all'ora corrente, puoi ripristinare l'intervallo di tempo relativo nella pagina dei tracciamenti ed eseguire nuovamente la scansione.

Se vengono ancora restituiti più risultati di quanti la console possa visualizzare, la console mostra quanti tracciamenti soddisfano il filtro e di quanti è stata eseguita la scansione. La percentuale visualizzata è la percentuale dell'intervallo di tempo selezionato di cui è stata eseguita la scansione. Restringi ulteriormente l'espressione filtro, oppure scegli un intervallo di tempo più breve, in modo da visualizzare tra i risultati tutti i tracciamenti corrispondenti al filtro.

Per disporre prima dei risultati più recenti, la console avvia la scansione partendo dal termine dell'intervallo di tempo e procedendo in senso cronologico inverso. In caso di un numero elevato di tracciamenti, la console suddivide l'intervallo di tempo in blocchi ed esegue le scansioni in parallelo. La barra di avanzamento mostra le porzioni dell'intervallo di tempo relativamente alle quali è stata eseguita la scansione.



Utilizzo delle espressioni filtro con i gruppi

I gruppi costituiscono una raccolta di tracciamenti definiti da un'espressione di filtro. Puoi utilizzare i gruppi per generare grafici di servizio aggiuntivi e fornire CloudWatch metriche Amazon.

I gruppi sono identificati dal nome o da un Amazon Resource Name (ARN) e contengono un'espressione di filtro. Il servizio confronta i tracciamenti in entrata con l'espressione e le memorizza di conseguenza.

È possibile creare e modificare i gruppi utilizzando il menu a discesa a sinistra della barra di ricerca delle espressioni filtro.

> **Note**
>
> Se il servizio rileva un errore nel qualificare un gruppo, tale gruppo non viene più incluso nell'elaborazione dei tracciamenti in entrata e viene registrata un parametro di errore.

Per ulteriori informazioni sui gruppi, consulta. Configurazione dei gruppi

Sintassi delle espressioni filtro

Le espressioni filtro possono contenere una parola chiave, un operatore unario o binario e un valore per il confronto.

```
keyword operator value
```

Sono disponibili diversi operatori per diversi tipi di parola chiave. Ad esempio, `responsetime` è una parola chiave numerica e può essere confrontata con operatori correlati a numeri.

Example — richieste in cui il tempo di risposta è stato superiore a 5 secondi

```
responsetime > 5
```

Puoi combinare più espressioni in un'espressione composta mediante gli operatori `AND` o `OR`.

Example — richieste la cui durata totale era di 5—8 secondi

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

Le parole chiave e gli operatori semplici individuano problemi solo a livello di tracciamento. Se si verifica un errore a valle, ma è gestito direttamente dall'applicazione e non viene restituito all'utente, una ricerca per `error` non lo individuerà.

Per individuare tracciamenti che includano problemi con chiamate a valle, puoi utilizzare le parole chiave complesse `service()` e `edge()`. Queste parole chiave ti permettono di applicare un'espressione filtro a tutti i nodi a valle, a un singolo nodo a valle o ad un collegamento tra due nodi. Per una maggiore granularità, puoi filtrare servizi ed edge per tipo con la funzione id().

Parole chiave booleane

I valori delle parole chiave booleane sono true o false. Utilizza queste parole chiave per individuare i tracciamenti che hanno generato errori.

Parole chiave booleane

- ok— Il codice di stato della risposta era 2XX Success.
- `error`— Il codice di stato della risposta era 4XX Client Error.
- `throttle`— Il codice di stato della risposta era 429 Too Many Requests.
- `fault`— Il codice di stato della risposta era 5XX Server Error.
- `partial`— La richiesta ha segmenti incompleti.
- `inferred`— La richiesta ha segmenti dedotti.
- `first`— L'elemento è il primo di un elenco enumerato.
- `last`— L'elemento è l'ultimo di un elenco enumerato.
- `remote`— L'entità della causa principale è remota.
- `root`— Il servizio è il punto di ingresso o il segmento principale di una traccia.

Gli operatori booleani individuano i segmenti in cui la chiave specificata è `true` o `false`.

Operatori booleani

- nessuno: l'espressione è vera se la parola chiave è vera.
- `!`— L'espressione è vera se la parola chiave è falsa.
- =, `!=` — Confronta il valore della parola chiave con la stringa `true` or`false`. Questi operatori agiscono allo stesso modo degli altri operatori, ma sono più espliciti.

Example — lo stato della risposta è 2XX OK

```
ok
```

Example — lo stato della risposta non è 2XX OK

```
!ok
```

Example — lo stato della risposta non è 2XX OK

```
ok = false
```

Example — l'ultima traccia di errore enumerata ha il nome di errore «deserializza»

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

Example — richieste con segmenti remoti in cui la copertura è maggiore di 0,7 e il nome del servizio è «trace»

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

Example — richieste con segmenti dedotti in cui il tipo di servizio è «aws:DynamoDB»

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

Example — richieste che hanno come radice un segmento con il nome «data-plane»

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

Parole chiave numeriche

Utilizzare le parole chiave numeriche per ricercare le richieste con un tempo di risposta, una durata o un codice di stato della risposta specifici.

Parole chiave numeriche

- `responsetime`— Tempo impiegato dal server per inviare una risposta.
- `duration`— Durata totale della richiesta, incluse tutte le chiamate downstream.

- `http.status`— Codice di stato della risposta.
- `index`— Posizione di un elemento in un elenco enumerato.
- `coverage`— Percentuale decimale del tempo di risposta dell'entità rispetto al tempo di risposta del segmento principale. Applicabile solo per le entità causa principale del tempo di risposta.

Operatori numerici

Le parole chiave numeriche utilizzano gli operatori di uguaglianza e di confronto standard.

- =, `!` = — La parola chiave è uguale o diversa da un valore numerico.
- <,<=,>, >= — La parola chiave è minore o maggiore di un valore numerico.

Example — lo stato della risposta non è 200 OK

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

Example — richiesta in cui la durata totale era di 5—8 secondi

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

Example — richieste completate con successo in meno di 3 secondi, comprese tutte le chiamate a valle

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

Example — entità di elenco enumerato con un indice maggiore di 5

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

Example — richieste in cui l'ultima entità con copertura è superiore a 0,8

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

Parole chiave stringa

Le parole chiave stringa permettono di trovare i tracciamenti con testo specifico all'interno delle intestazioni della richiesta o con specifici ID utente.

Parole chiave stringa

- `http.url`— URL della richiesta.
- `http.method`— Metodo di richiesta.
- `http.useragent`— Richiedere la stringa dell'agente utente.
- `http.clientip`— Indirizzo IP del richiedente.
- `user`— Valore del campo utente su qualsiasi segmento della traccia.
- `name`— Il nome di un servizio o di un'eccezione.
- `type`— Tipo di servizio.
- `message`— Messaggio di eccezione.
- `availabilityzone`— Valore del campo availabilityzone su qualsiasi segmento della traccia.
- `instance.id`— Valore del campo ID dell'istanza su qualsiasi segmento della traccia.
- `resource.arn`— Valore del campo ARN della risorsa su qualsiasi segmento della traccia.

Gli operatori di stringa selezionano valori che sono uguali a, o contengono, un testo specifico. I valori devono essere sempre specificati tra virgolette.

Operatori di stringa

- `=`, `!=` — La parola chiave è uguale o diversa da un valore numerico.
- `CONTAINS`— La parola chiave contiene una stringa specifica.
- `BEGINSWITH`, `ENDSWITH` — La parola chiave inizia o termina con una stringa specifica.

Example — filtro http.url

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

Per verificare se un campo esiste all'interno di un tracciamento, indipendentemente dal suo valore, verificare se contiene la stringa vuota.

Example — filtro utente

Trova tutti i tracciamenti contenenti ID degli utenti.

```
user CONTAINS ""
```

Example — seleziona le tracce con una causa principale dell'errore che include un servizio denominato «Auth»

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

Example — seleziona le tracce con una root cause del tempo di risposta il cui ultimo servizio ha un tipo di DynamoDB

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

Example — seleziona le tracce con una causa principale di errore la cui ultima eccezione è il messaggio «accesso negato per account_id: 1234567890"

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

Parole chiave complesse

Le parole chiave complesse permettono di selezionare le richieste in base a nome del servizio, nome edge o valore dell'annotazione. Per servizi e collegamenti, puoi specificare un ulteriore espressione filtro che si applica al servizio o al collegamento. Per le annotazioni, puoi filtrare in base al valore di un'annotazione con una specifica chiave utilizzando operatori booleani, numerici o stringa.

Parole chiave complesse

- `annotation.`*`key– Valore di un'annotazione con chiave di campo.`* Il valore di un'annotazione può essere un valore booleano, numerico o stringa, perciò puoi usare un operatore di confronto di uno qualsiasi di questi tipi. È possibile utilizzare questa parola chiave in combinazione con la `edge` parola chiave `service` o.
- `edge(`*`source`*`, `*`destination`*`) {`*`filter`*`}`— Connessione tra *`origine`* e *`destinazione`* dei servizi. Le parentesi graffe opzionali possono contenere un'espressione filtro che si applica ai segmenti della connessione.
- `group.`*`name`* `/ group.`*`arn`*— Il valore dell'espressione di filtro di un gruppo, a cui fa riferimento il nome del gruppo o l'ARN del gruppo.
- `json`— Oggetto della causa principale JSON. Vedi Acquisizione di dati da AWS X-Ray per i passaggi per creare entità JSON a livello di codice.

- `service(name) {filter}` — *Servizio con nome e nome.* Le parentesi graffe opzionali possono contenere un'espressione filtro che si applica ai segmenti creati dal servizio.

Utilizza la parola chiave service per trovare le tracce delle richieste che raggiungono un determinato nodo sulla mappa di tracciamento.

Gli operatori di parole chiave complesse trovano i segmenti in cui è stata impostata o non è stata impostata la chiave specificata.

Operatori di parole chiave complessi

- nessuno: l'espressione è vera se la parola chiave è impostata. Se la parola chiave è di tipo booleano, restituirà il valore booleano.
- `!` — L'espressione è vera se la parola chiave non è impostata. Se la parola chiave è di tipo booleano, restituirà il valore booleano.
- `=`, `!=` — Confronta il valore della parola chiave.
- `edge(source, destination) {filter}` — Connessione tra *origine* e *destinazione* dei servizi. Le parentesi graffe opzionali possono contenere un'espressione filtro che si applica ai segmenti della connessione.
- `annotation.key` — Valore di un'annotazione con *chiave* di campo. Il valore di un'annotazione può essere un valore booleano, numerico o stringa, perciò puoi usare un operatore di confronto di uno qualsiasi di questi tipi. È possibile utilizzare questa parola chiave in combinazione con la `edge` parola chiave `service` o.
- `json` — Oggetto della causa principale JSON. Vedi Acquisizione di dati da AWS X-Ray per i passaggi per creare entità JSON a livello di codice.

Utilizza la parola chiave service per trovare le tracce delle richieste che raggiungono un determinato nodo sulla mappa di tracciamento.

Example — Filtro di servizio

Richieste che includevano una chiamata a `api.example.com` e che si sono concluse con un malfunzionamento (errore serie 500).

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

Puoi escludere il nome del servizio per applicare un'espressione filtro a tutti i nodi della tua mappa del servizio.

Example — filtro di servizio

Richieste che hanno causato un errore in qualsiasi punto della mappa di tracciamento.

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

La parola chiave edge applica un'espressione filtro a una connessione tra due nodi.

Example — filtro perimetrale

Richiesta in cui il servizio `api.example.com` ha eseguito una chiamata verso `backend.example.com` che si è conclusa con un errore.

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

Puoi inoltre utilizzare l'operatore `!` con le parole chiave service ed edge per escludere un servizio o un collegamento dai risultati di un'altra espressione filtro.

Example — filtro di servizio e richiesta

Richiesta in cui l'URL inizia con `http://api.example.com/` e contiene `/v2/` ma non raggiunge un servizio denominato `api.example.com`.

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

Example — filtro del servizio e del tempo di risposta

Trova le tracce dove `http url` è impostato e il tempo di risposta è superiore a 2 secondi.

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

Per le annotazioni, puoi richiamare tutte le tracce in cui `annotation.`*`key`* è impostato o utilizzare gli operatori di confronto che corrispondono al tipo di valore.

Example — annotazione con valore di stringa

Richieste con un'annotazione denominata `gameid` con valore stringa `"817DL6V0"`.

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

Example — l'annotazione è impostata

Richieste con un'annotazione denominata `age` set.

```
annotation.age
```

Example — l'annotazione non è impostata

Richieste senza un'annotazione denominata `age` set.

```
!annotation.age
```

Example — annotazione con valore numerico

Richieste con annotazione age con valore numerico maggiore di 29.

```
annotation.age > 29
```

Example — annotazione in combinazione con service o edge

```
service { annotation.request_id = "917DL6V0" }
```

```
edge { source.annotation.request_id = "916DL6V0" }
```

```
edge { destination.annotation.request_id = "918DL6V0" }
```

Example — gruppo con utente

Richiede in cui le tracce soddisfano il filtro di `high_response_time` gruppo (ad esempio`responseTime > 3`) e l'utente si chiama Alice.

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

Example — JSON con entità root cause

Le richieste con entità causa principale corrispondenti.

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

funzione id

Quando indichi un nome di servizio per le parole chiave `service` o `edge`, ottieni risultati per tutti i nodi con tale nome. Per una maggiore precisione del filtraggio, puoi utilizzare la funzione `id` per specificare un tipo di servizio in aggiunta ad un nome per distinguere tra nodi con lo stesso nome.

Utilizza la `account.id` funzione per specificare un particolare account per il servizio, quando visualizzi le tracce di più account in un account di monitoraggio.

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

Puoi utilizzare la funzione `id` al posto di un nome di servizio nei filtri service ed edge.

```
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 }
```

Ad esempio, AWS Lambda le funzioni generano due nodi nella mappa di traccia, uno per la chiamata della funzione e uno per il servizio Lambda. I due nodi hanno lo stesso nome ma appartengono a tipi diversi. Un filtro sul servizio standard troverà i tracciamenti relativi a entrambi.

Example — filtro di servizio

Richieste che includono un errore su qualsiasi servizio denominato `random-name`.

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

Utilizza la funzione `id` per restringere la ricerca a errori sulla funzione stessa, escludendo gli errori del servizio.

Example — filtro di servizio con funzione id

Richieste che includono un errore su un servizio denominato `random-name` di tipo `AWS::Lambda::Function`.

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

Per ricercare i nodi in base al tipo, puoi anche omettere completamente il nome.

Example — filtro di servizio con funzione id e tipo di servizio

Richieste che includono un errore su un servizio di tipo `AWS::Lambda::Function`.

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

Per cercare i nodi per un particolare Account AWS, specifica un ID account.

Example — filtro di servizio con funzione id e ID account

Richieste che includono un servizio all'interno di un ID account specifico`AWS::Lambda::Function`.

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

Tracciamento tra account

AWS X-Ray supporta l'osservabilità tra account, consentendoti di monitorare e risolvere i problemi delle applicazioni che si estendono su più account all'interno di uno stesso. Regione AWS Puoi cercare, visualizzare e analizzare senza interruzioni le tue metriche, i log e le tracce in tutti gli account collegati come se stessi operando in un unico account. Ciò fornisce una visione completa delle richieste che viaggiano su più account. È possibile visualizzare le tracce tra account nella mappa di tracciamento X-Ray e le pagine delle tracce all'interno della CloudWatch console.

I dati di osservabilità condivisi possono includere uno dei seguenti tipi di telemetria:

- Metriche in Amazon CloudWatch
- Gruppi di log in Amazon CloudWatch Logs
- Tracce in AWS X-Ray
- Applicazioni in Amazon CloudWatch Application Insights

Configura l'osservabilità tra account

Per attivare l'osservabilità tra più account, configura uno o più account di AWS monitoraggio e collegali a più account di origine. Un account di monitoraggio è una centrale in Account AWS grado

di visualizzare e interagire con i dati di osservabilità generati dagli account di origine. Un account di origine è un individuo Account AWS che genera dati di osservabilità per le risorse che contiene.

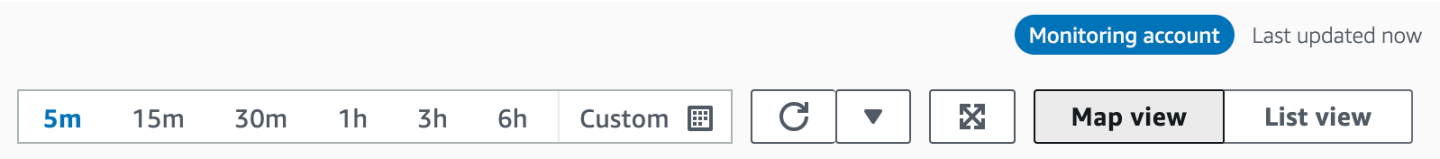
Gli account di origine condividono i dati di osservabilità con gli account di monitoraggio. Le tracce vengono copiate da ciascun account di origine su un massimo di cinque account di monitoraggio. Le copie delle tracce dagli account di origine al primo account di monitoraggio sono gratuite. Le copie delle tracce inviate ad account di monitoraggio aggiuntivi vengono addebitate a ciascun account di origine, in base alla tariffa standard. Per ulteriori informazioni, consulta AWS X-Ray i prezzi e i CloudWatch prezzi di Amazon.

Per creare collegamenti tra gli account di monitoraggio e gli account di origine, utilizza la CloudWatch console o i nuovi comandi di Observability Access Manager nell'API AWS CLI and. Per maggiori informazioni, consulta la sezione Osservabilità su più account di CloudWatch .

> **Note**
>
> Le tracce a raggi X vengono fatturate nel Account AWS luogo in cui vengono ricevute. Se una richiesta campionata riguarda più di un servizio Account AWS, ogni account registra una traccia separata e tutte le tracce condividono lo stesso ID di traccia. Per ulteriori informazioni sui prezzi osservabili su più account, consulta i prezzi e AWS X-Ray i prezzi di Amazon CloudWatch .

Visualizzazione delle tracce tra account

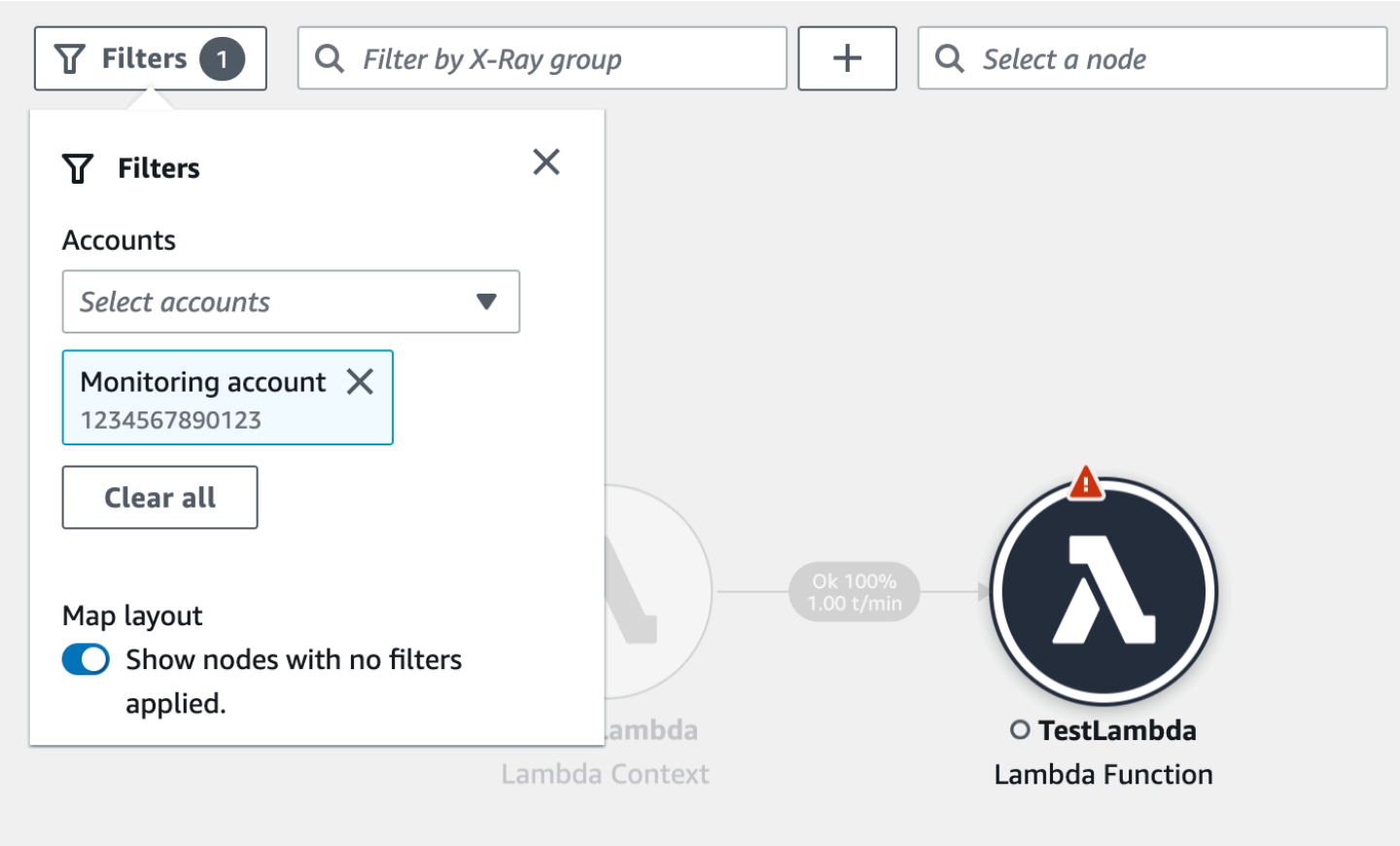
Le tracce tra account vengono visualizzate nell'account di monitoraggio. Ogni account di origine visualizza solo le tracce locali per quell'account specifico. Le sezioni seguenti presuppongono che tu abbia effettuato l'accesso all'account di monitoraggio e che tu abbia aperto la CloudWatch console Amazon. Sia sulla mappa di tracciamento che sulla pagina delle tracce, nell'angolo in alto a destra viene visualizzato un badge dell'account di monitoraggio.



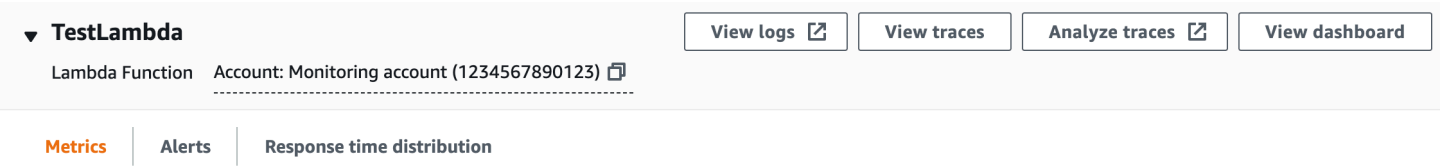
Mappa di tracciamento

Nella CloudWatch console, scegli Trace Map in X-Ray Traces dal riquadro di navigazione a sinistra. Per impostazione predefinita, la mappa di tracciamento mostra i nodi per tutti gli account di origine

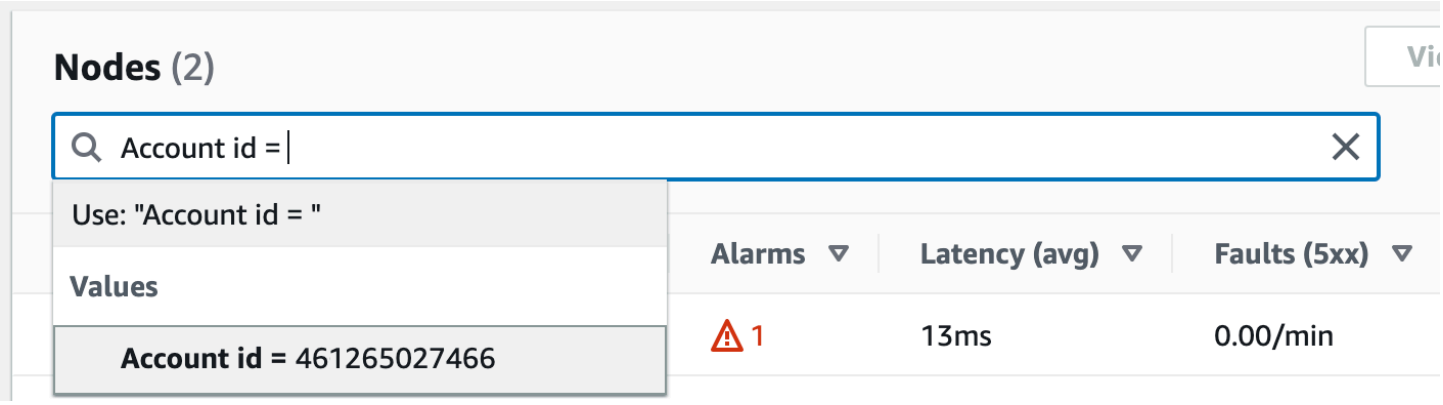
che inviano le tracce all'account di monitoraggio e i nodi per l'account di monitoraggio stesso. Sulla mappa di traccia, scegli Filtri in alto a sinistra per filtrare la mappa di traccia utilizzando il menu a discesa Account. Dopo aver applicato un filtro di account, i nodi di servizio degli account che non corrispondono al filtro corrente vengono visualizzati in grigio.



Quando si sceglie un nodo di servizio, il riquadro dei dettagli del nodo include l'ID e l'etichetta dell'account del servizio.



Nell'angolo in alto a destra della mappa di traccia, scegli Visualizzazione elenco per visualizzare un elenco di nodi di servizio. L'elenco dei nodi di servizio include i servizi dell'account di monitoraggio e tutti gli account di origine configurati. Filtra l'elenco dei nodi in base all'etichetta dell'account o all'ID dell'account scegliendoli dal filtro Nodi.

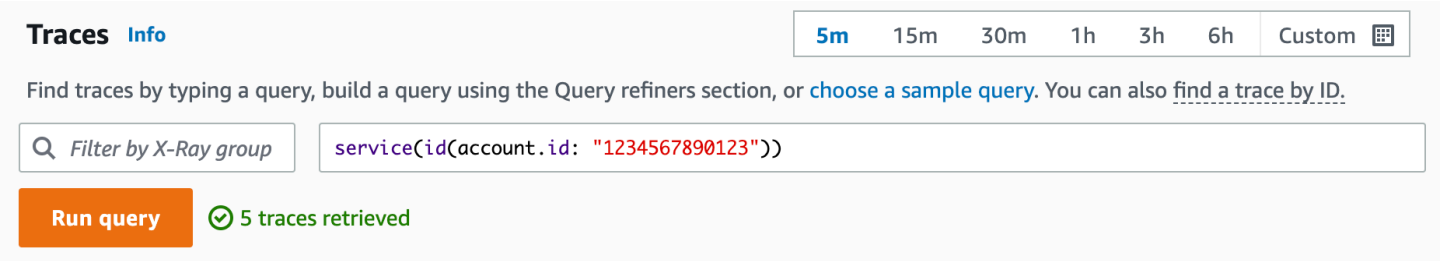


Tracce

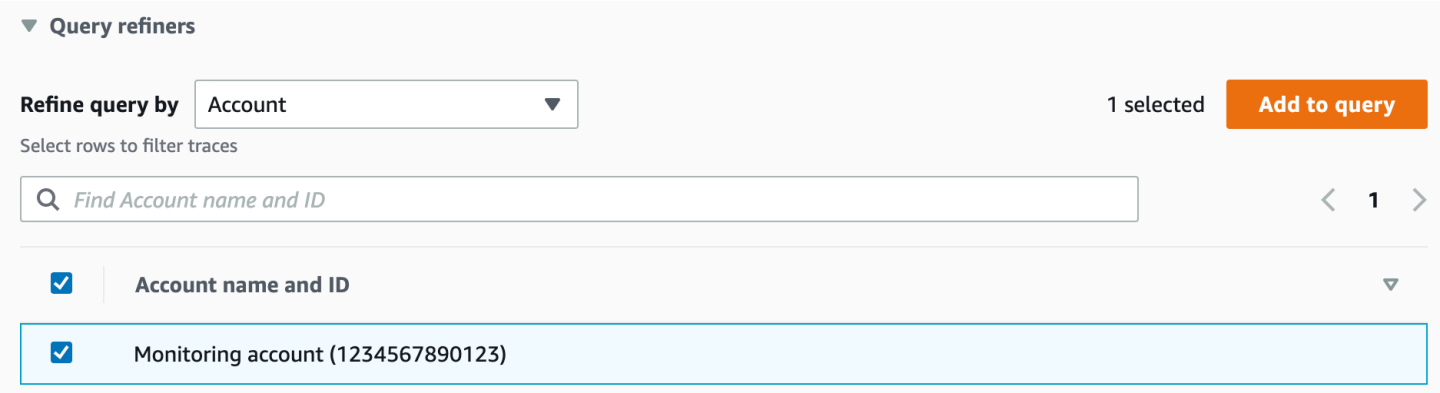
Visualizza i dettagli di tracciamento per le tracce che si estendono su più account aprendo la CloudWatch console dall'account di monitoraggio e selezionando Tracce in Tracce a raggi X nel riquadro di navigazione a sinistra. È inoltre possibile aprire questa pagina scegliendo un nodo nella Mappa di tracciamento a raggi X, quindi scegliendo Visualizza tracce dal riquadro dei dettagli del nodo.

La pagina Traces supporta l'interrogazione per ID account. Per iniziare, inserisci una query che includa uno o più ID account. Di seguito sono riportate le query di esempio per le tracce che sono passate attraverso l'ID dell'account X o Y:

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



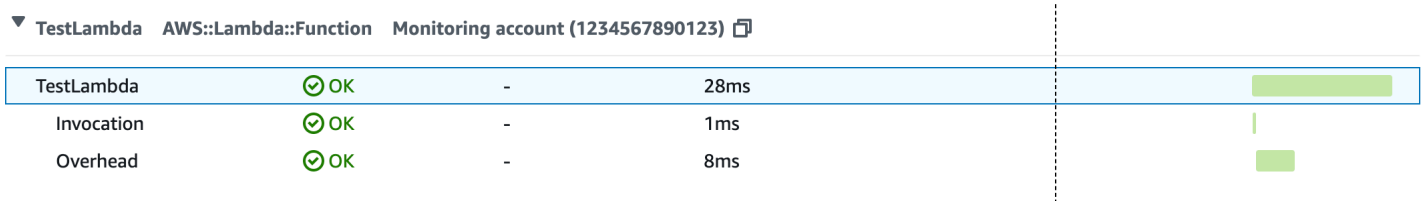
Perfeziona la tua richiesta per account. Seleziona uno o più account dall'elenco e scegli Aggiungi alla query.



Dettagli di tracciamento

Visualizza i dettagli di una traccia selezionandola dall'elenco Tracce nella parte inferiore della pagina Tracce. Vengono visualizzati i dettagli di Trace, inclusa una mappa dei dettagli di traccia con i nodi di servizio di tutti gli account attraversati dalla traccia. Scegli un nodo di servizio specifico per visualizzare l'account corrispondente.

La sezione Cronologia dei segmenti mostra i dettagli dell'account per ogni segmento della sequenza temporale.



Tracciamento di applicazioni basate sugli eventi

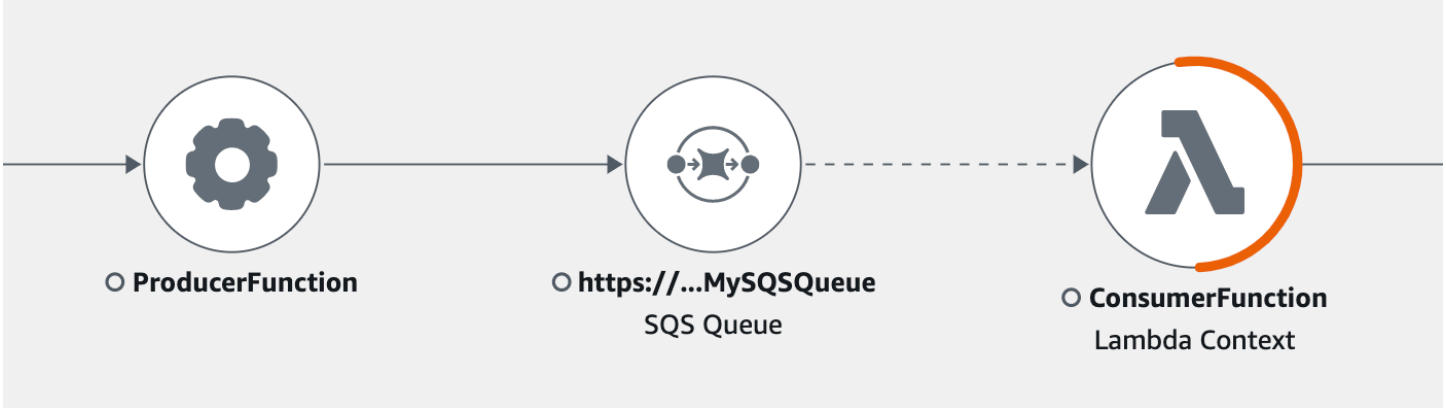
AWS X-Ray supporta il tracciamento di applicazioni basate su eventi utilizzando Amazon SQS e. AWS Lambda Usa la CloudWatch console per visualizzare una vista connessa di ogni richiesta mentre viene messa in coda con Amazon SQS ed elaborata da una o più funzioni Lambda. Le tracce dei produttori di messaggi upstream vengono automaticamente collegate alle tracce dei nodi consumer Lambda a valle, creando una end-to-end visualizzazione dell'applicazione.

> **Note**
>
> Ogni segmento di traccia può essere collegato a un massimo di 20 tracce, mentre una traccia può includere un massimo di 100 link. In alcuni scenari, il collegamento di tracce aggiuntive può comportare il superamento della dimensione massima del documento di

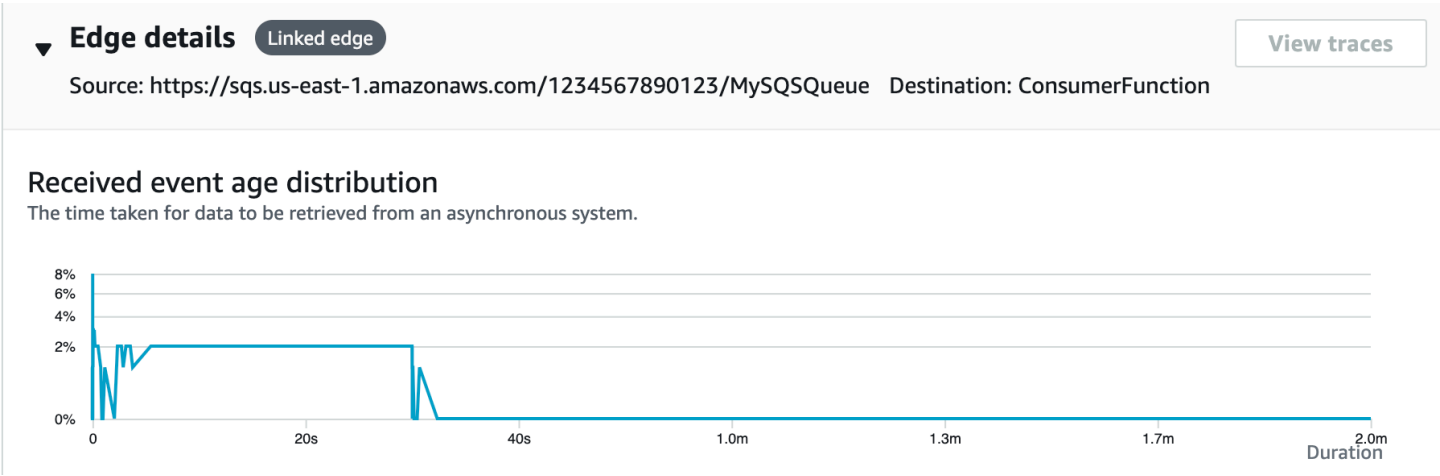
traccia, causando una traccia potenzialmente incompleta. Ciò può accadere, ad esempio, quando una funzione Lambda con tracciamento abilitato invia molti messaggi SQS a una coda in un'unica chiamata. Se si verifica questo problema, è disponibile una soluzione di mitigazione che utilizza gli X-Ray SDK. Per ulteriori informazioni, consulta X-Ray SDK per Java, Node.js,Python, Go o.NET.

Visualizza le tracce collegate nella mappa di tracciamento

Utilizza la pagina Trace Map all'interno della CloudWatchconsole per visualizzare una mappa di traccia con le tracce dei produttori di messaggi collegate alle tracce dei consumatori Lambda. Questi collegamenti vengono visualizzati con un bordo tratteggiato che collega il nodo Amazon SQS e i nodi consumer Lambda downstream.



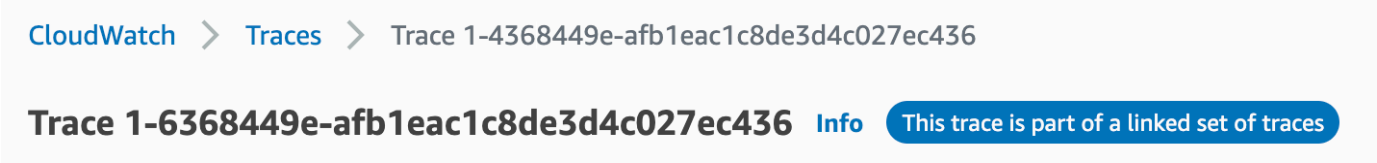
Seleziona un bordo tratteggiato per visualizzare un istogramma dell'età degli eventi ricevuti, che mappa la diffusione dell'età dell'evento quando viene ricevuto dai consumatori. L'età viene calcolata ogni volta che viene ricevuto un evento.



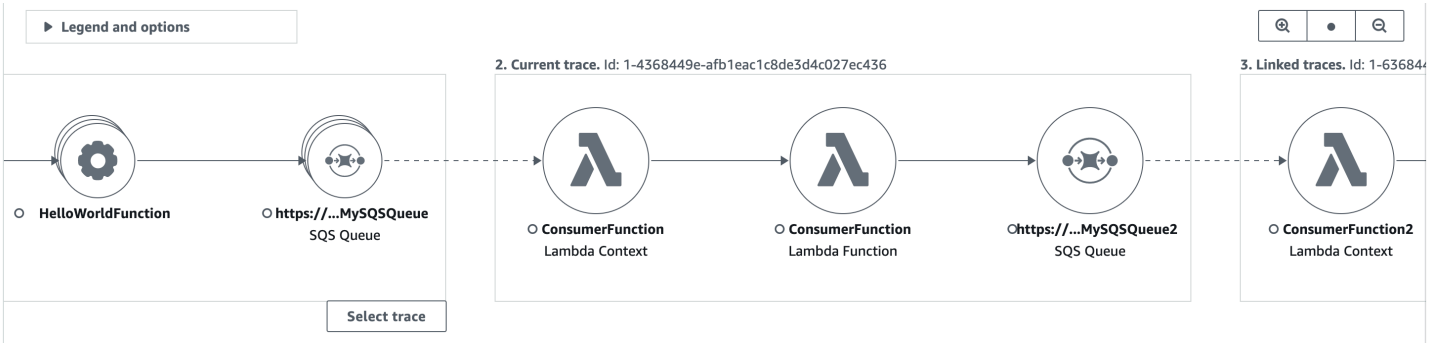
Visualizza i dettagli della traccia collegata

Visualizza i dettagli di tracciamento inviati da un produttore di messaggi, da una coda Amazon SQS o da un consumatore Lambda:

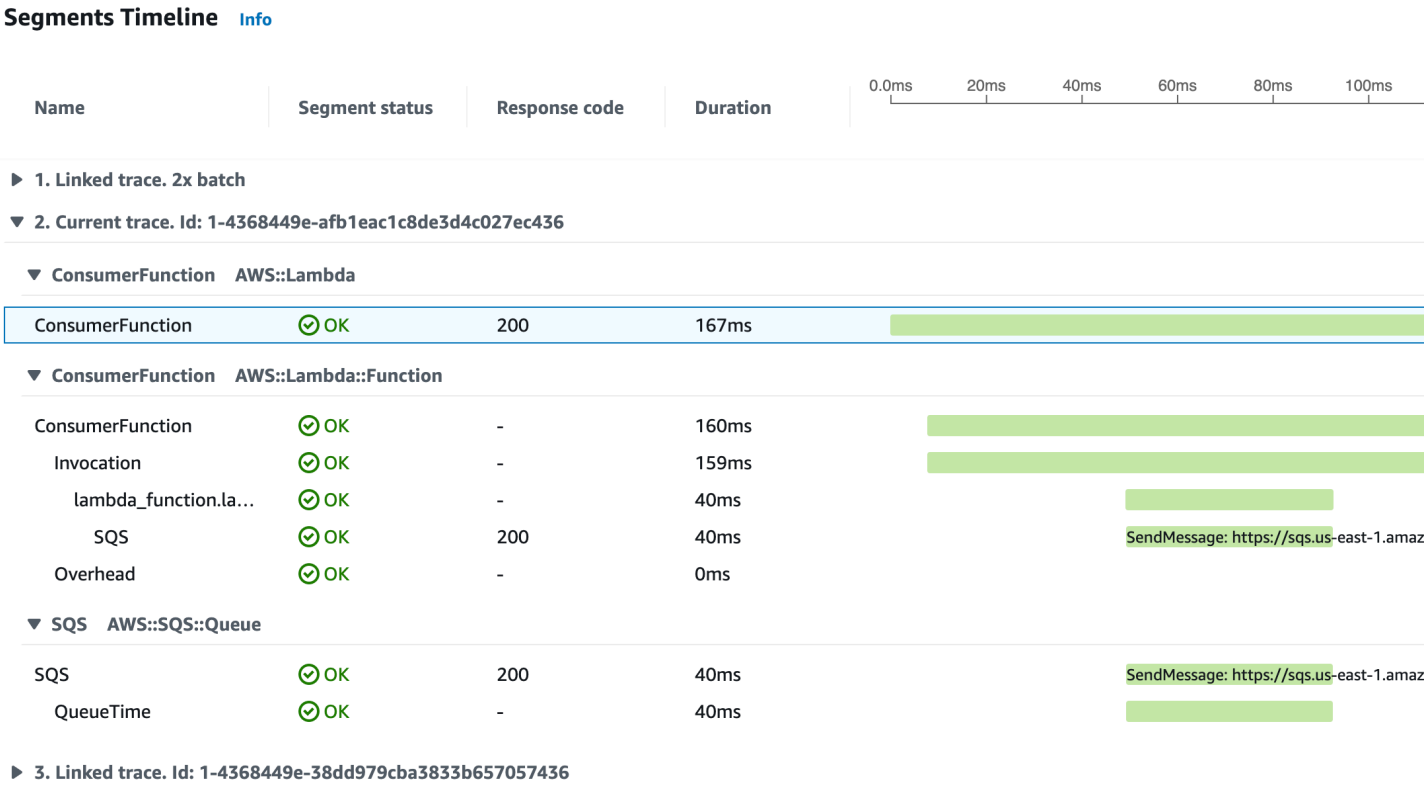
1. Usa la Trace Map per selezionare un produttore di messaggi, un nodo consumer Amazon SQS o Lambda.
2. Scegli Visualizza tracce dal riquadro dei dettagli del nodo per visualizzare un elenco di tracce. Puoi anche accedere direttamente alla pagina Traces all'interno della CloudWatch console.
3. Scegli una traccia specifica dall'elenco per aprire la pagina dei dettagli della traccia. La pagina dei dettagli della traccia visualizza un messaggio quando la traccia selezionata fa parte di un set di tracce collegato.



La mappa dei dettagli della traccia mostra la traccia corrente, insieme alle tracce collegate a monte e a valle, ognuna delle quali è contenuta in un riquadro che indica i limiti di ciascuna traccia. Se la traccia attualmente selezionata è collegata a più tracce a monte o a valle, i nodi all'interno delle tracce collegate a monte o a valle vengono impilati e viene visualizzato il pulsante Seleziona traccia.



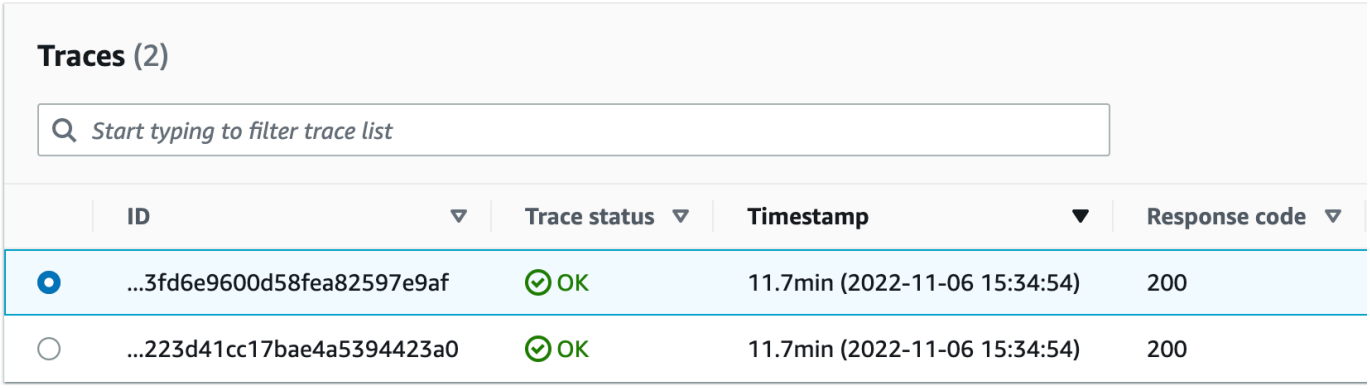
Sotto la mappa dei dettagli della traccia, viene visualizzata una sequenza temporale dei segmenti di traccia, incluse le tracce collegate a monte e a valle. Se sono presenti più tracce collegate a monte o a valle, i relativi dettagli del segmento non possono essere visualizzati. Per visualizzare i dettagli del segmento per una singola traccia all'interno di un insieme di tracce collegate, selezionate una singola traccia come descritto di seguito.



Seleziona una singola traccia all'interno di un insieme di tracce collegate

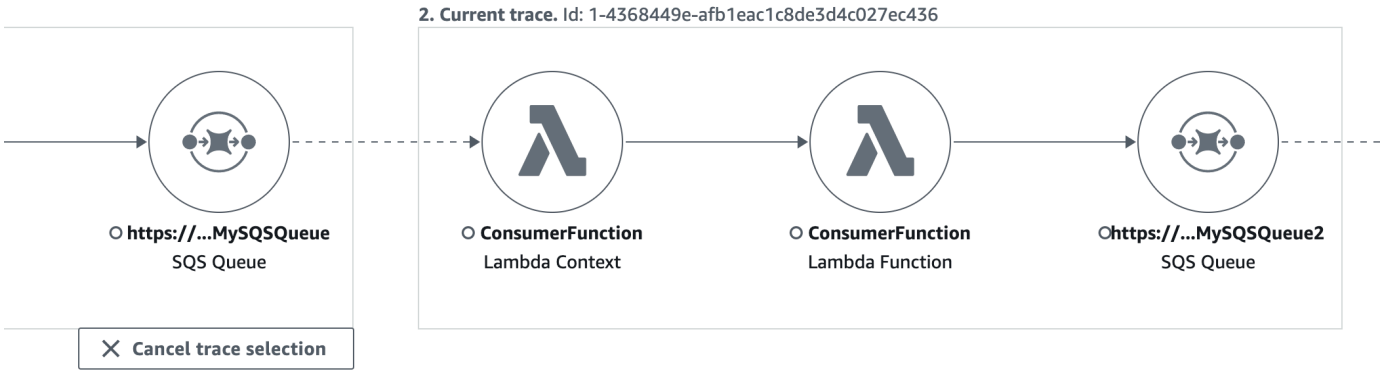
Filtra un set di tracce collegato a una singola traccia per visualizzare i dettagli del segmento nella timeline.

1. Scegli Seleziona traccia sotto le tracce collegate sulla mappa dei dettagli della traccia. Viene visualizzato un elenco di tracce.



2. Seleziona il pulsante di opzione accanto a una traccia per visualizzarla all'interno della mappa dei dettagli della traccia.

3. Scegliete Annulla la selezione della traccia per visualizzare l'intero set di tracce collegate.



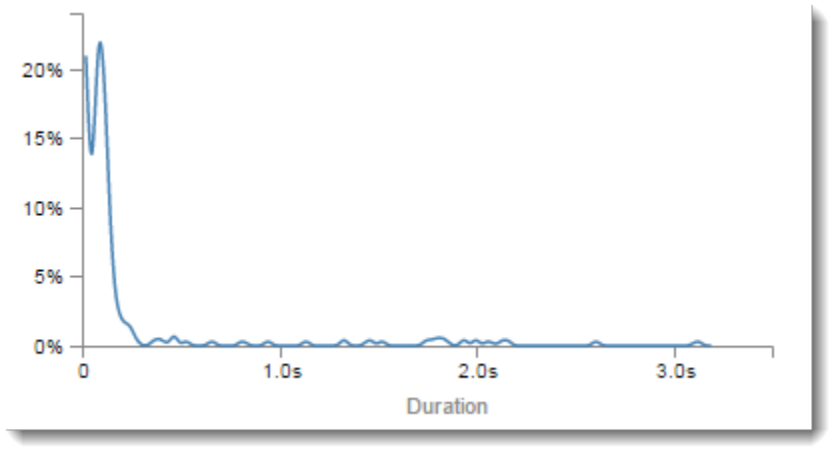
Utilizzo degli istogrammi di latenza

Quando si seleziona un nodo o un bordo su una mappa di AWS X-Ray traccia, la console X-Ray mostra un istogramma di distribuzione della latenza.

Latenza

La latenza è la quantità di tempo compresa tra il momento in cui una richiesta inizia e quello in cui viene completata. Un istogramma mostra una distribuzione di latenze. Mostra la durata sull'asse delle X e la percentuale di richieste che corrispondono a ciascuna durata sull'asse delle Y.

Questo istogramma mostra un servizio che completa la maggior parte delle richieste in meno di 300 ms. Una piccola percentuale di richieste richiedere fino a 2 secondi e alcuni valori erratici richiedono più tempo.



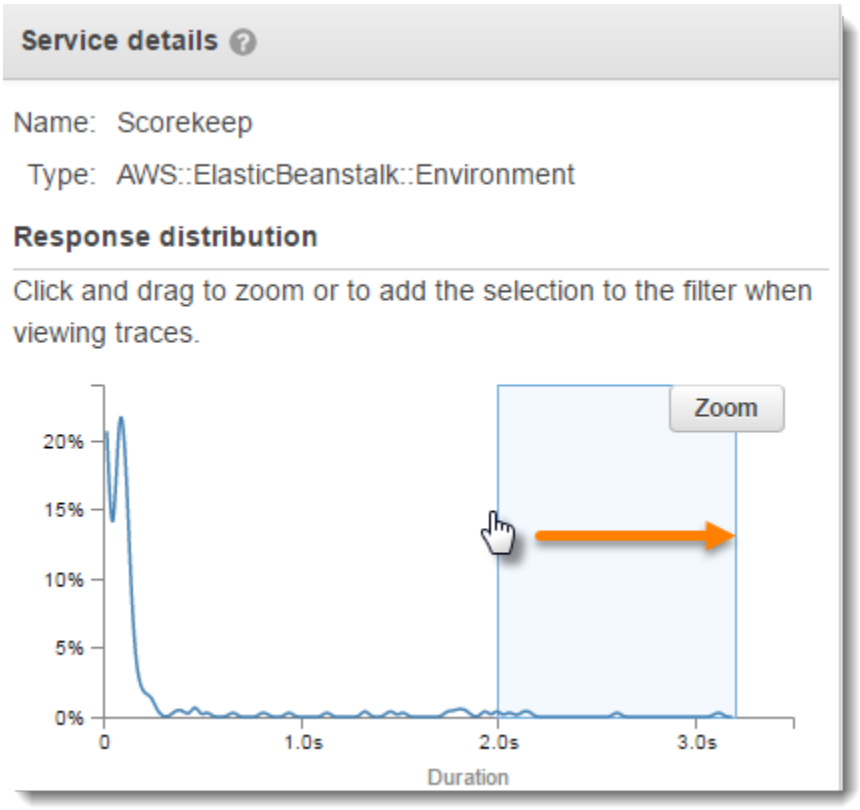
Interpretazione dei dettagli del servizio

Gli istogrammi del servizio e del collegamento forniscono una rappresentazione visiva della latenza dal punto di vista di un servizio o di un richiedente.

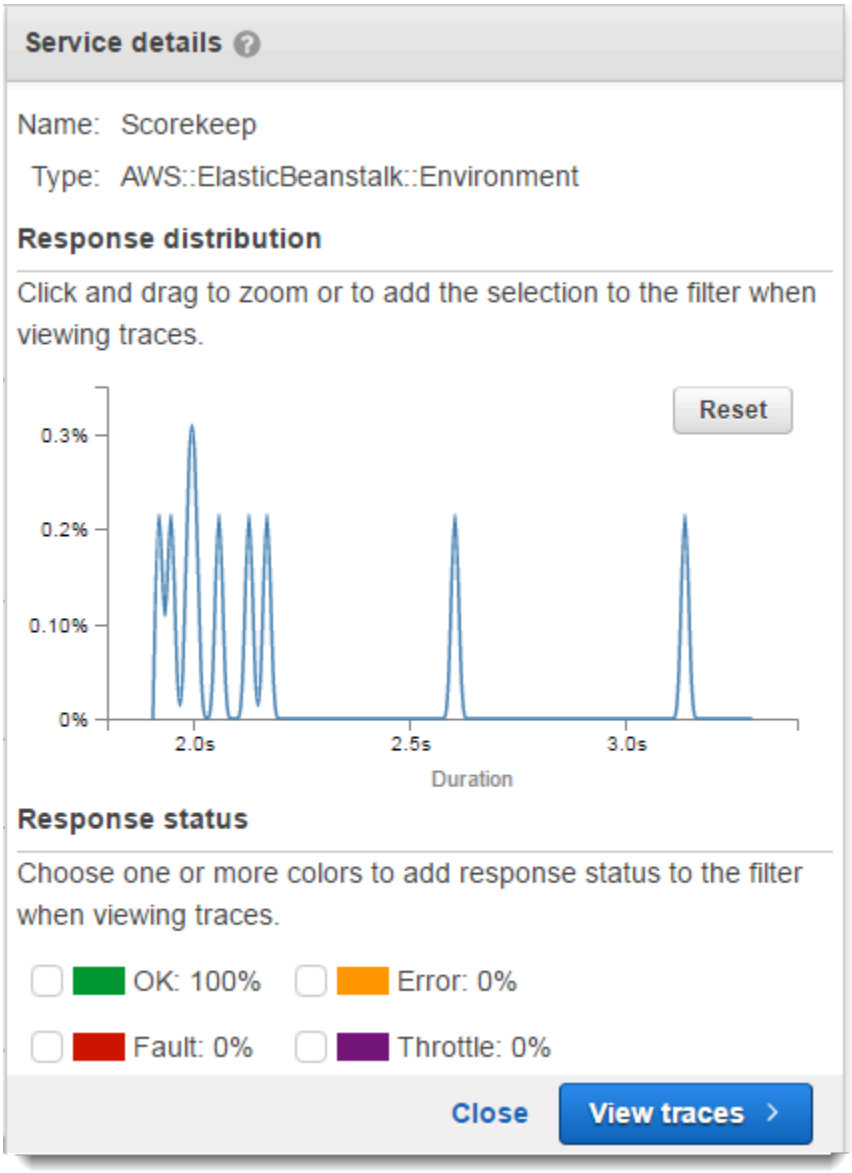
- Scegliete un nodo di servizio facendo clic sul cerchio. X-Ray mostra un istogramma per le richieste servite dal servizio. Le latenze sono quelle registrate dal servizio e non includono la latenza di rete tra il servizio e il richiedente.
- Scegliete uno spigolo facendo clic sulla linea o sulla punta della freccia dello spigolo tra due servizi. X-Ray mostra un istogramma per le richieste del richiedente che sono state servite dal servizio downstream. Le latenze sono quelle registrate dal richiedente e includono la latenza di rete tra i due servizi.

Per interpretare il pannello dell'istogramma dei Service details (Dettagli del servizio), puoi cercare i valori che differiscono maggiormente dalla gran parte dei valori nell'istogramma. Questi valori erratici possono essere visualizzati come picchi o punte nell'istogramma, ed è possibile visualizzare i tracciamenti relativi ad una specifica area per analizzare il problema.

Per visualizzare i tracciamenti filtrati per la latenza, seleziona un intervallo sull'istogramma. Fai clic sul punto in cui desideri avviare la selezione e trascina da sinistra verso destra per evidenziare un intervallo di latenze da includere nel filtro dei tracciamenti.



Dopo aver selezionato un intervallo, puoi scegliere Zoom per visualizzare solo quella parte di istogramma e restringere la selezione.



Una volta impostata l'area di interesse, scegli View traces (Visualizza tracciamenti).

Utilizzo di X-Ray Insights

AWS X-Ray analizza continuamente i dati di traccia presenti nel tuo account per identificare problemi emergenti nelle tue applicazioni. Quando le percentuali di errore superano l'intervallo previsto, crea una panoramica che registra il problema e ne monitora l'impatto fino alla sua risoluzione. Con Insights, puoi:

- Identifica l'area dell'applicazione in cui si verificano i problemi, la causa principale del problema e l'impatto associato. L'analisi dell'impatto fornita da Insights consente di determinare la gravità e la priorità di un problema.
- Ricevi notifiche man mano che il problema cambia nel tempo. Le notifiche Insights possono essere integrate con la tua soluzione di monitoraggio e avviso utilizzando Amazon EventBridge. Questa integrazione consente di inviare e-mail o avvisi automatici in base alla gravità del problema.

La console X-Ray identifica i nodi con incidenti in corso nella mappa di tracciamento. Per visualizzare un riepilogo delle informazioni, scegli il nodo interessato. Puoi anche visualizzare e filtrare gli approfondimenti scegliendo Insights dal riquadro di navigazione a sinistra.

X-Ray crea informazioni dettagliate quando rileva un'anomalia in uno o più nodi della mappa dei servizi. Il servizio utilizza modelli statistici per prevedere i tassi di errore previsti dei servizi dell'applicazione. Nell'esempio precedente, l'anomalia è un aumento dei guasti da. AWS Elastic Beanstalk Il server Elastic Beanstalk ha registrato diversi timeout delle chiamate API, che hanno causato un'anomalia nei nodi downstream.

Abilita gli approfondimenti nella console X-Ray

Insights deve essere abilitato per ogni gruppo con cui desideri utilizzare le funzionalità di Insights. Puoi abilitare gli approfondimenti dalla pagina Gruppi.

1. Apri la console X-Ray.
2. Seleziona un gruppo esistente o creane uno nuovo scegliendo Crea gruppo, quindi seleziona Abilita Insights. Per ulteriori informazioni sulla configurazione dei gruppi nella console X-Ray, vedere. Configurazione dei gruppi
3. Nel riquadro di navigazione a sinistra, scegli Insights, quindi scegli un approfondimento da visualizzare.

Note

X-Ray utilizza GetInsightSummaries, GetInsight GetInsightEvents, e le operazioni GetInsightImpactGraph API per recuperare i dati dagli approfondimenti. Per visualizzare gli approfondimenti, utilizza la policy gestita da AWSXrayReadOnlyAccess IAM o aggiungi la seguente policy personalizzata al tuo ruolo IAM:

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

Per ulteriori informazioni, consulta Come AWS X-Ray funziona con IAM.

Abilita le notifiche di approfondimenti

Con le notifiche di approfondimenti, viene creata una notifica per ogni evento di approfondimento, ad esempio quando un approfondimento viene creato, cambia in modo significativo o viene chiuso. I clienti possono ricevere queste notifiche tramite Amazon EventBridge Events e utilizzare regole condizionali per intraprendere azioni come la notifica SNS, la chiamata Lambda, la pubblicazione di messaggi in una coda SQS o uno qualsiasi dei target supportati. EventBridge Le notifiche Insights vengono emesse con la massima diligenza possibile, ma non sono garantite. Per ulteriori informazioni sugli obiettivi, consulta Amazon EventBridge Targets.

Puoi abilitare le notifiche di Insights per qualsiasi gruppo abilitato a Insights dalla pagina Gruppi.

Per abilitare le notifiche per un gruppo X-Ray

1. Apri la console X-Ray.
2. Seleziona un gruppo esistente o creane uno nuovo scegliendo Crea gruppo, assicurati che sia selezionato Enable Insights, quindi seleziona Abilita notifiche. Per ulteriori informazioni sulla configurazione dei gruppi nella console X-Ray, vedere. Configurazione dei gruppi

Per configurare le regole EventBridge condizionali di Amazon

1. Apri la EventBridge console Amazon.
2. Vai a Regole nella barra di navigazione a sinistra e scegli Crea regola.
3. Fornisci un nome e una descrizione per la regola.
4. Scegliete Modello di evento, quindi scegliete Modello personalizzato. Fornisci uno schema contenente `"source": [ "aws.xray" ]` e`"detail-type": [ "AWS X-Ray Insight Update" ]`. Di seguito sono riportati alcuni esempi di modelli possibili.
   - Schema di eventi per abbinare tutti gli eventi in arrivo da X-Ray Insights:

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

- Schema di eventi che corrisponda a uno specifico **`state`** e: **`category`**

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

5. Seleziona e configura i target che desideri richiamare quando un evento corrisponde a questa regola.
6. (Facoltativo) Fornisci tag per identificare e selezionare più facilmente questa regola.
7. Scegli Crea.

> **Note**
>
> Le notifiche X-Ray Insights inviano eventi ad Amazon EventBridge, che attualmente non supporta le chiavi gestite dai clienti. Per ulteriori informazioni, consulta Protezione dei dati in AWS X-Ray.

Panoramica di Insight

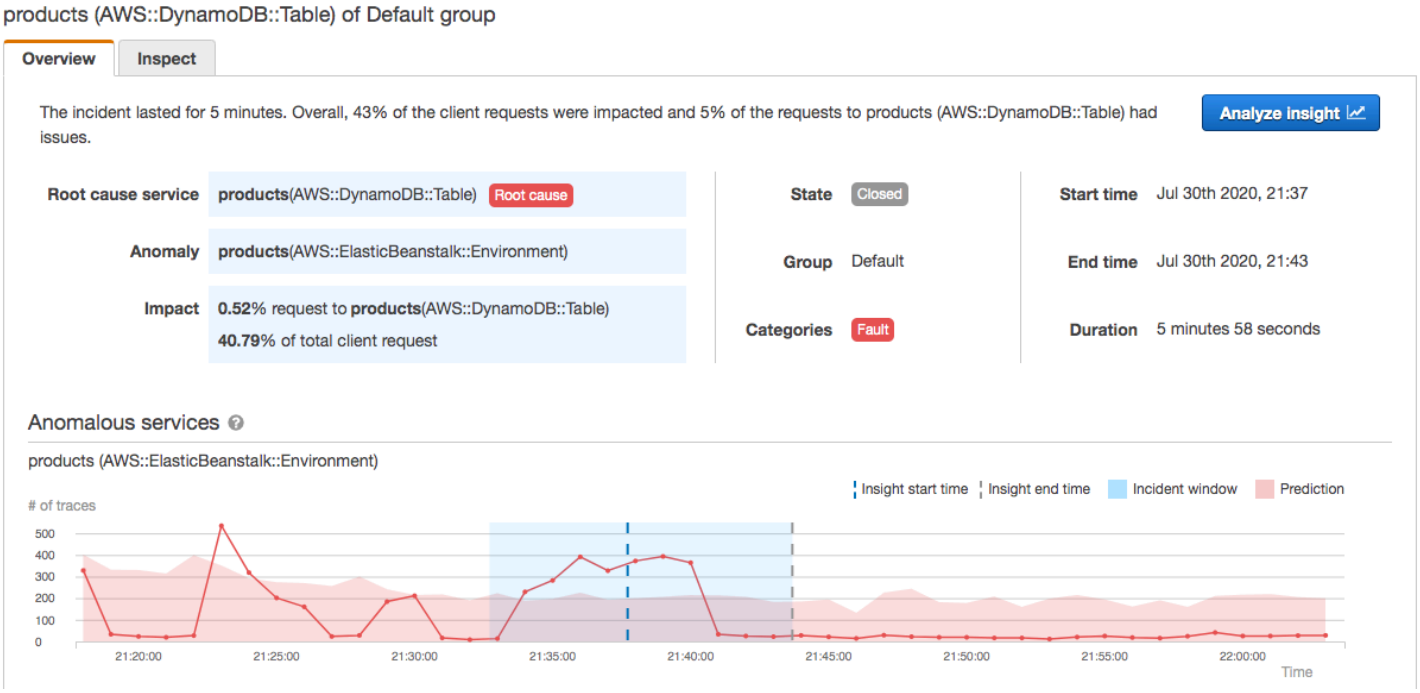
La pagina di panoramica per un approfondimento tenta di rispondere a tre domande chiave:

- Qual è il problema di fondo?
- Qual è la causa principale?
- Qual è l'impatto?

La sezione Servizi anomali mostra una sequenza temporale per ogni servizio che illustra la variazione delle percentuali di guasto durante l'incidente. La sequenza temporale mostra il numero di tracce con guasti sovrapposti su una banda continua che indica il numero previsto di guasti in base alla quantità di traffico registrata. La durata dell'analisi viene visualizzata nella finestra Incidente. La

finestra incidente inizia quando X-Ray osserva che la metrica diventa anomala e persiste mentre l'analisi è attiva.

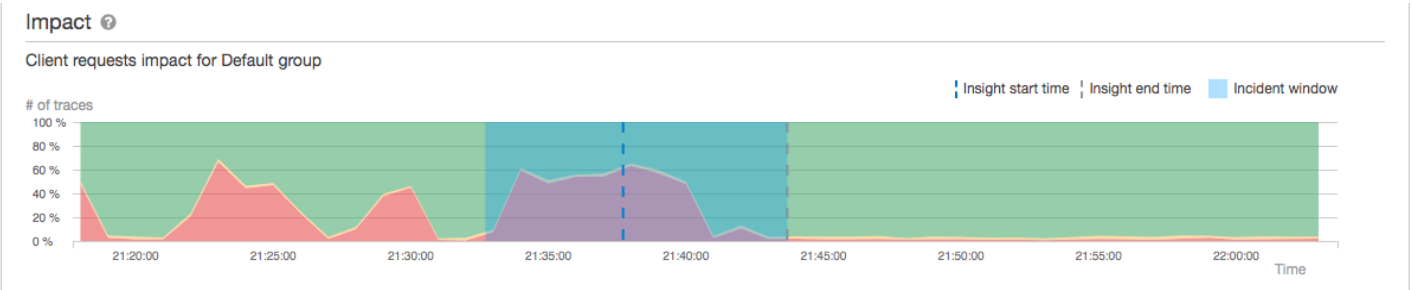
L'esempio seguente mostra un aumento dei guasti che hanno causato un incidente:



La sezione Root Cause mostra una mappa di traccia incentrata sul servizio della causa principale e sul percorso interessato. È possibile nascondere i nodi non interessati selezionando l'icona a forma di occhio in alto a destra nella mappa della causa principale. Il servizio root cause è il nodo più a valle in cui X-Ray ha identificato un'anomalia. Può rappresentare un servizio strumentato dall'utente o un servizio esterno richiamato dal servizio con un client dotato di strumentazione. Ad esempio, se chiami Amazon DynamoDB con un client SDK AWS strumentato, un aumento degli errori causati da DynamoDB si traduce in un'analisi approfondita con DynamoDB come causa principale.

Per approfondire la causa principale, seleziona Visualizza i dettagli della causa principale nel grafico della causa principale. Puoi utilizzare la pagina Analytics per esaminare la causa principale e i messaggi correlati. Per ulteriori informazioni, consulta Interazione con la console di analisi .

I guasti che continuano a monte della mappa possono avere un impatto su più nodi e causare anomalie multiple. Se un errore viene passato all'utente che ha effettuato la richiesta, il risultato è un errore del client. Si tratta di un errore nel nodo principale della mappa di traccia. Il grafico Impact fornisce una cronologia dell'esperienza del cliente per l'intero gruppo. Questa esperienza viene calcolata in base alle percentuali dei seguenti stati: Fault, Error, Throttle e Okay.



Questo esempio mostra un aumento delle tracce con un guasto al nodo principale durante il periodo di un incidente. Gli incidenti nei servizi downstream non sempre corrispondono a un aumento degli errori dei client.

Scegliendo Analyze insight si apre la console di X-Ray Analytics in una finestra in cui è possibile approfondire l'insieme di tracce che causano l'analisi. Per ulteriori informazioni, consulta Interazione con la console di analisi .

Comprendere l'impatto

AWS X-Ray misura l'impatto causato da un problema in corso nell'ambito della generazione di approfondimenti e notifiche. L'impatto viene misurato in due modi:

- Impatto sul gruppo X-Ray
- Impatto sul servizio root cause

Questo impatto è determinato dalla percentuale di richieste non riuscite o che causano un errore entro un determinato periodo di tempo. Questa analisi dell'impatto consente di determinare la gravità e la priorità del problema in base allo scenario specifico. Questo impatto è disponibile come parte dell'esperienza da console, oltre alle notifiche di approfondimenti.

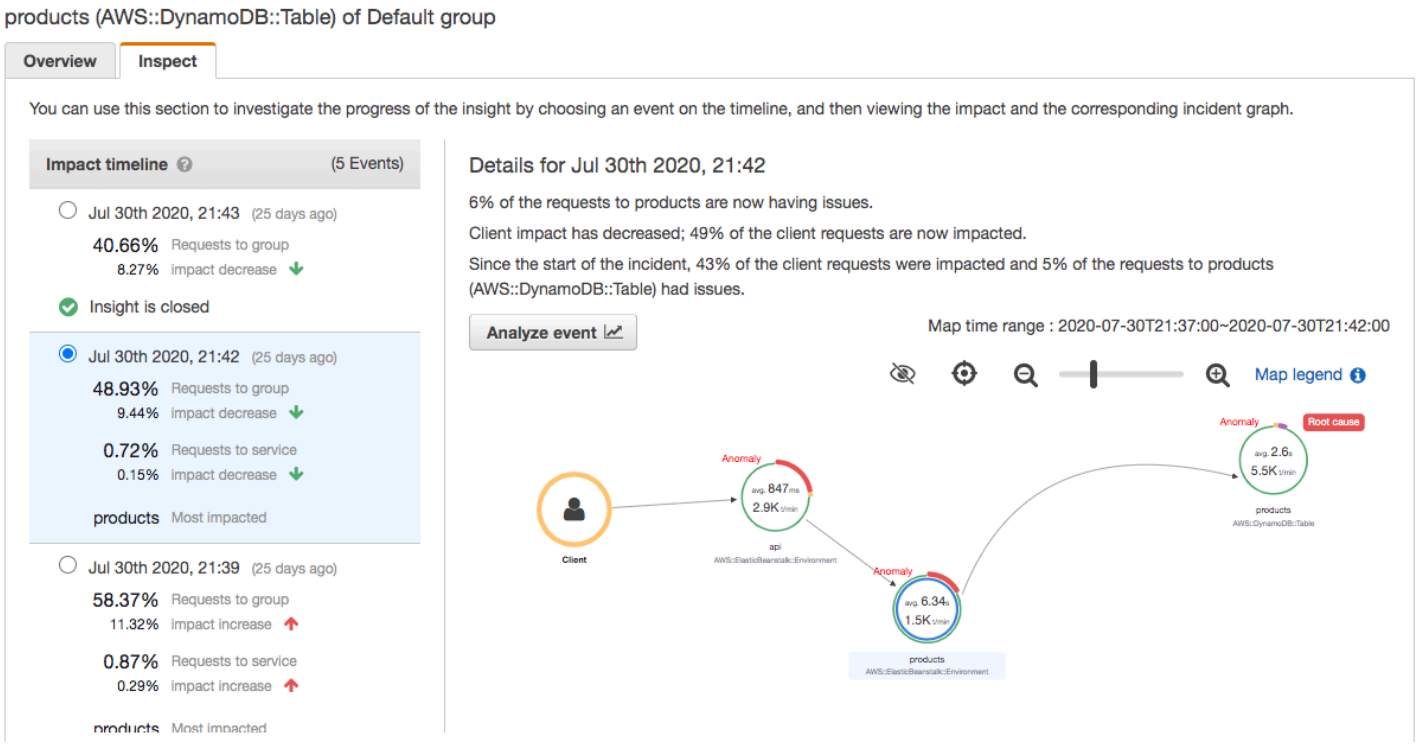
Deduplicazione

AWS X-Ray insights deduplica i problemi su più microservizi. Utilizza il rilevamento delle anomalie per determinare il servizio che è la causa principale di un problema, determina se altri servizi correlati presentano un comportamento anomalo dovuto alla stessa causa principale e registra il risultato come un'unica analisi.

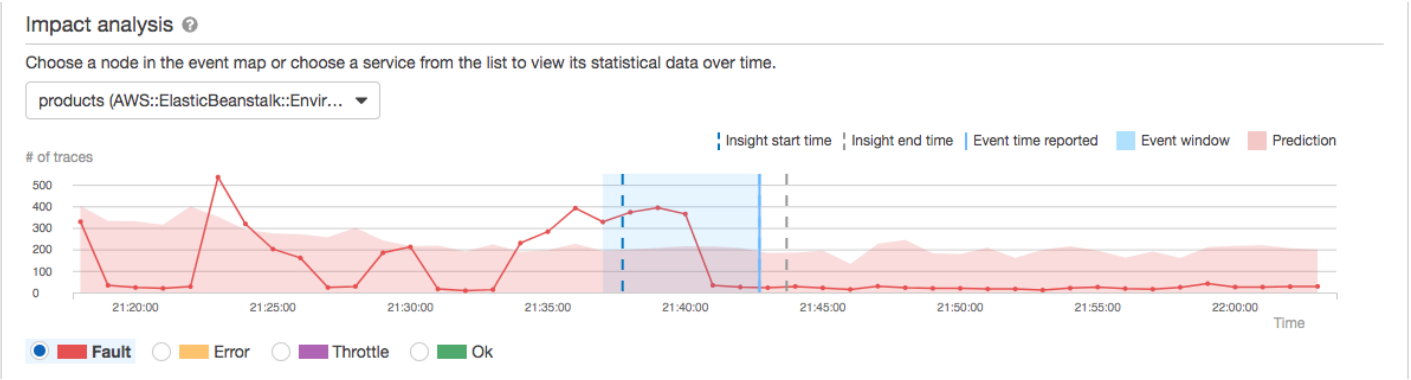
Esamina lo stato di avanzamento di un'analisi

X-Ray rivaluta periodicamente gli approfondimenti fino a quando non vengono risolti e registra ogni modifica intermedia importante come notifica, che può essere inviata come evento Amazon. EventBridge Ciò consente di creare processi e flussi di lavoro per determinare in che modo il problema è cambiato nel tempo e di intraprendere le azioni appropriate, come l'invio di un'e-mail o l'integrazione con un sistema di avviso utilizzando. EventBridge

È possibile esaminare gli eventi relativi agli incidenti nella cronologia degli impatti nella pagina Inspect. Per impostazione predefinita, la cronologia mostra il servizio più interessato fino a quando non si sceglie un servizio diverso.



Per visualizzare una mappa di tracciamento e i grafici di un evento, sceglili dalla timeline dell'impatto. La mappa di tracciamento mostra i servizi dell'applicazione interessati dall'incidente. Nella sezione Analisi dell'impatto, i grafici mostrano le tempistiche dei guasti per il nodo selezionato e per i client del gruppo.



Per dare un'occhiata più approfondita alle tracce coinvolte in un incidente, scegli Analizza evento nella pagina Inspect. Puoi utilizzare la pagina Analytics per rifinire l'elenco delle tracce e identificare gli utenti interessati. Per ulteriori informazioni, consulta Interazione con la console di analisi .

Interazione con la console di analisi

La console AWS X-Ray Analytics è uno strumento interattivo per interpretare i dati di traccia per comprendere rapidamente le prestazioni dell'applicazione e dei servizi sottostanti. La console consente di esplorare, analizzare e visualizzare i tracciamenti tramite tempo di risposta interattivo e grafici di serie temporali.

Quando si effettuano selezioni nella console di analisi, la console crea dei filtri per riflettere il sottoinsieme selezionato di tutti i tracciamenti. È possibile definire il set di dati attivo con filtri sempre più granulari facendo clic sui grafici e i pannelli di parametri e campi associati con il set di tracciamenti corrente.

Argomenti

- Caratteristiche della console
- Distribuzione del tempo di risposta
- Attività delle serie temporali
- Esempi di flussi di lavoro
- Osservare gli errori sul grafico di servizio
- Identificare i picchi del tempo di risposta
- Visualizzare tutti le tracce contrassegnate con un codice di stato
- Visualizzare tutti gli elementi in un sottogruppo e associati a un utente
- Confrontare due set di tracce con criteri diversi
- Identificare un tracciamento rilevante e visualizzare i dettagli

Caratteristiche della console

La console X-Ray Analytics utilizza le seguenti funzionalità chiave per raggruppare, filtrare, confrontare e quantificare i dati di traccia.

Funzionalità

Funzionalità	Descrizione
Groups (Gruppi)	Il gruppo selezionato inizialmente è `Default`. Per modificare il gruppo recuperato, seleziona

Funzionalità	Descrizione
	re un gruppo diverso dal menu a destra della barra di ricerca dell'espressione di filtro principale. Per ulteriori informazioni sui gruppi, consulta Utilizzo delle espressioni filtro con i gruppi.
Retrieved traces (Tracciamenti recuperati)	Per impostazione predefinita, la console di analisi genera grafici in base a tutti i tracciamenti nel gruppo selezionato. I tracciamenti recuperati rappresentano tutti i tracciamenti nel set di lavoro. Puoi trovare il conteggio dei tracciamenti in questo riquadro. Le espressioni di filtro applicate alla barra di ricerca principale definiscono e aggiornano i tracciamenti recuperati.
Show in charts/Hide from charts (Mostra in grafici/Nascondi da grafici)	Un elemento di attivazione/disattivazione per confrontare il gruppo attivo rispetto ai tracciamenti recuperati. Per confrontare i dati relativi al gruppo rispetto a tutti i filtri attivi, scegliere Show in charts (Mostra in grafici). Per rimuovere questa visualizzazione dai grafici, scegliere Hide from charts (Nascondi da grafici).
Filtered trace set A (Set di tracciamenti filtrato A)	Attraverso le interazioni con i grafici e le tabelle, applica i filtri per creare i criteri per il set di tracce filtrato A. Man mano che i filtri vengono applicati, il numero di tracce applicabili e la percentuale di tracce recuperate rispetto al totale vengono calcolati all'interno di questo riquadro. I filtri sono inseriti come tag all'interno del riquadro Filtered trace set A (Set di tracce filtrate A e possono anche essere rimossi dal riquadro.

Funzionalità	Descrizione
Refine (Definisci)	Questa funzione aggiorna il set di tracce recuperate in base ai filtri applicati al set di tracce A. Il perfezionamento del set di tracce recuperate aggiorna il working set di tutte le tracce recuperate in base ai filtri per il set di tracce A. Il working set di tracce recuperate è un sottoinsieme campionato di tutte le tracce nel gruppo.
Filtered trace set B (Set di tracciamenti filtrato B)	Una volta creato, il set di tracce filtrato B è una copia del set di tracce filtrato A. Per confrontare i due set di tracce, effettuate nuove selezioni di filtri da applicare al set di tracce B, mentre il set di tracce A rimane fisso. Quando i filtri vengono applicati, il numero di tracciamenti applicabi li e la percentuale di tracciamenti dal totale recuperati sono calcolati all'interno di questo riquadro. I filtri sono inseriti come tag all'interno del riquadro Filtered trace set B (Set di tracce filtrate B) e possono anche essere rimossi dal riquadro.
Response Time Root Cause Entity Paths (Percorsi di entità causa principale del tempo di risposta)	Una tabella dei percorsi delle entità registrate. X-Ray determina quale percorso della traccia è la causa più probabile del tempo di risposta. Il formato indica una gerarchia di entità incontrate che termina in una causa principale del tempo di risposta. Utilizza queste righe per filtrare gli errori ricorrenti del tempo di risposta. Per ulteriori informazioni sulla personalizzazione di un filtro di causa principale e su come ottenere i dati tramite l'API, consultare l'argomento relativo al recupero e definizione dell’analisi delle cause principali.

Funzionalità	Descrizione
Delta (Δ)	Una colonna che viene aggiunta alle tabelle di parametri quando sia il set di tracce A che il set di tracce B sono attivi. La colonna del Delta calcola la differenza nella percentuale di tracce tra il set di tracce A e il set di tracce B.

Distribuzione del tempo di risposta

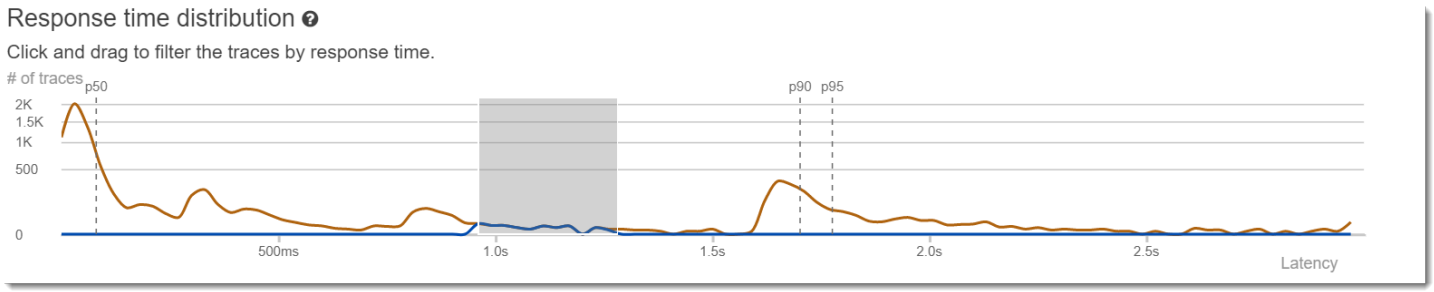
La console X-Ray Analytics genera due grafici principali per aiutarti a visualizzare le tracce: la distribuzione del tempo di risposta e l'attività delle serie temporali. In questa sezione e nella seguente sono forniti esempi di ognuno e le informazioni di base per leggere i grafici.

Di seguito sono elencati i colori associati al grafico della linea temporale di risposta (il grafico delle serie temporali impiega la stessa combinazione di colori):

- Tutte le tracce del gruppo: grigie
- Tracce recuperate: arancione
- Set di tracce filtrate A — verde
- Set di tracce filtrate B — blu

Example — Distribuzione del tempo di risposta

La distribuzione del tempo di risposta è un grafico che mostra il numero di tracciamenti con un determinato tempo di risposta. Fare clic e trascinare per effettuare selezioni all'interno della distribuzione del tempo di risposta. Questa operazione seleziona e crea un filtro sul set di tracciamenti di lavoro denominato `responseTime` per tutti i tracciamenti all'interno di un determinato tempo di risposta.

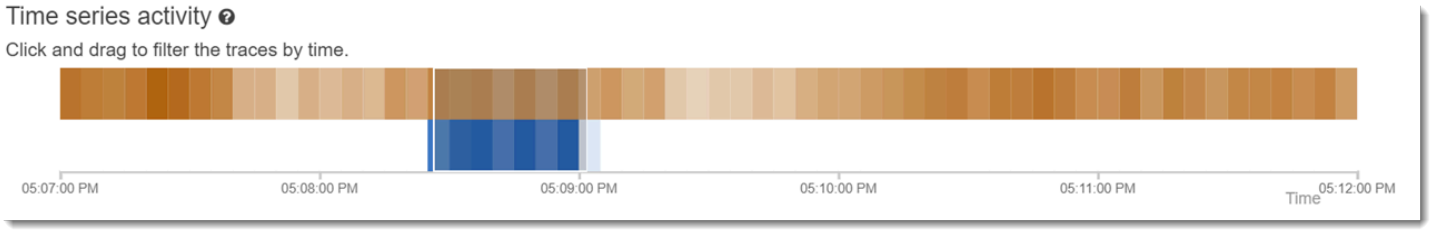


Attività delle serie temporali

Il grafico dell'attività di serie temporali mostra il numero di tracciamenti in un determinato intervallo di tempo. Gli indicatori di colore rispecchiano i colore del grafico a linee della distribuzione del tempo di risposta. Più è scuro e pieno il blocco di colore all'interno della serie dell'attività, più sono i tracciamenti rappresentati in quel determinato momento.

Example — Attività delle serie temporali

Fare clic e trascinare per effettuare selezioni all'interno del grafico di attività delle serie temporali. Questa operazione seleziona e crea un filtro denominato `timerange` sul set di tracciamenti di lavoro per tutti i tracciamenti all'interno di uno specifico intervallo di tempo.



Esempi di flussi di lavoro

Gli esempi seguenti mostrano casi d'uso comuni per la console X-Ray Analytics. Ogni esempio illustra una funzione chiave dell'esperienza della console. Come gruppo, gli esempi seguono un flusso di lavoro di risoluzione dei problemi di base. La procedura illustra come individuare i nodi non integri e quindi come interagire con la console di Analytics per generare automaticamente query comparative. Una volta ristretto l'ambito attraverso le query, sarà possibile visualizzare finalmente i dettagli delle tracce di interesse per determinare cosa sta danneggiando lo stato del servizio.

Osservare gli errori sul grafico di servizio

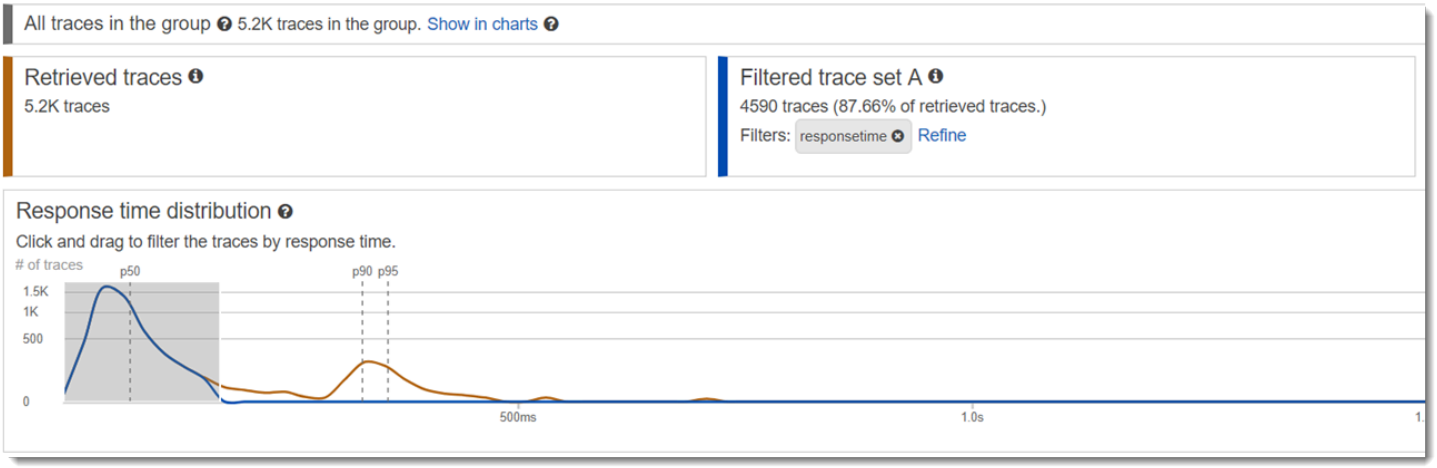
La mappa di traccia indica lo stato di salute di ogni nodo colorandolo in base al rapporto tra chiamate riuscite ed errori e guasti. Quando viene visualizzata una percentuale di rosso sul nodo, è segnalato un errore. Usa la console X-Ray Analytics per analizzarlo.

Per ulteriori informazioni su come leggere la mappa di traccia, vedere Visualizzazione della mappa di traccia.

Identificare i picchi del tempo di risposta

Utilizzando la distribuzione del tempo di risposta, è possibile osservare i picchi nel tempo di risposta. Selezionando questo picco nel tempo di risposta, le tabelle sotto i grafici si aggiorneranno per esporre tutti i parametri associati, come i codici di stato.

Quando si fa clic e si trascina, X-Ray seleziona e crea un filtro. Viene mostrato in un'ombreggiatura grigia sulla parte superiore delle righe del grafico. È ora possibile trascinare l'ombreggiatura a sinistra e destra lungo la distribuzione per aggiornare la selezione e il filtro.



Visualizzare tutti le tracce contrassegnate con un codice di stato

È possibile esaminare i dettagli dei tracciamenti all'interno del picco selezionato utilizzando le tabelle di parametri sotto ai grafici. Facendo clic su una riga nella tabella HTTP STATUS CODE, viene creato automaticamente un filtro sul set di dati di lavoro. Ad esempio, è possibile visualizzare tutti i tracciamenti del codice di stato 500. In questo modo viene creato un tag di filtro nel riquadro del set di tracciamenti denominato `http.status`.

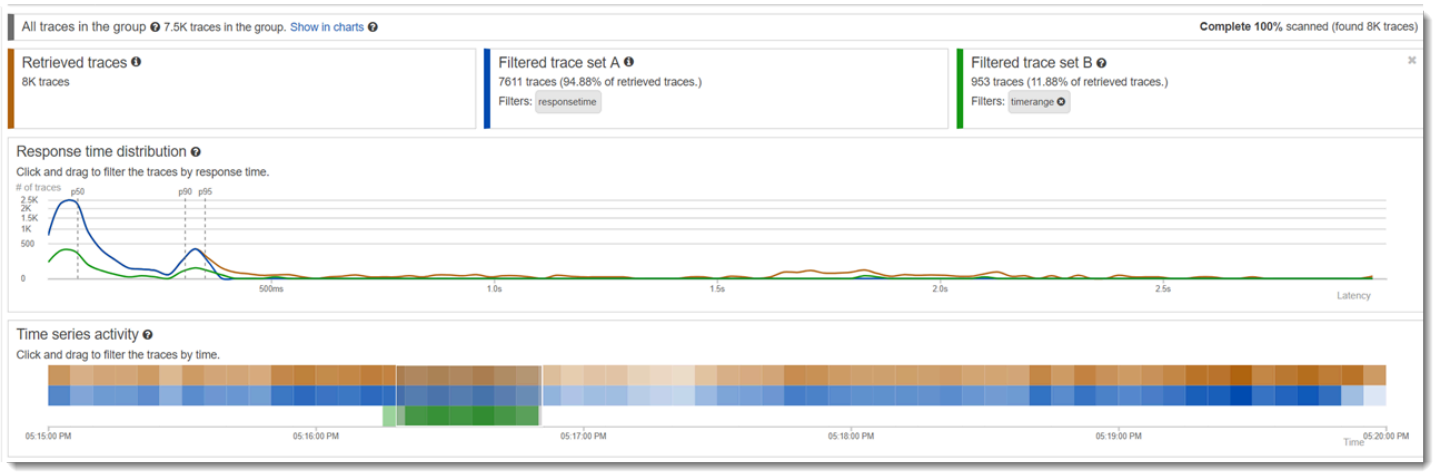
Visualizzare tutti gli elementi in un sottogruppo e associati a un utente

Analizzare il set di errori in base a utente, URL, causa principale del tempo di risposta o altri attributi predefiniti. Ad esempio, per filtrare ulteriormente il set di tracciamenti con un codice di stato 500, selezionare una riga dalla tabella USERS. Questo crea due tag di filtro nel riquadro del set di tracciamenti: `http.status`, come indicato in precedenza, e `user`.

Confrontare due set di tracce con criteri diversi

Confrontare i diversi utenti e le relative richieste POST per trovare altre discrepanze e correlazioni. Applicare il primo set di filtri. Questi sono definiti da una linea blu nella distribuzione del tempo di risposta. Quindi selezionare Compare (Confronta). Inizialmente, questo crea una copia dei filtri nel set di tracciamenti A.

Per procedere, definire un nuovo set di filtri da applicare al set di tracciamenti B. Questo secondo set è rappresentato da una linea verde. L'esempio seguente mostra le diverse linee in base allo schema di colori blu e verde.



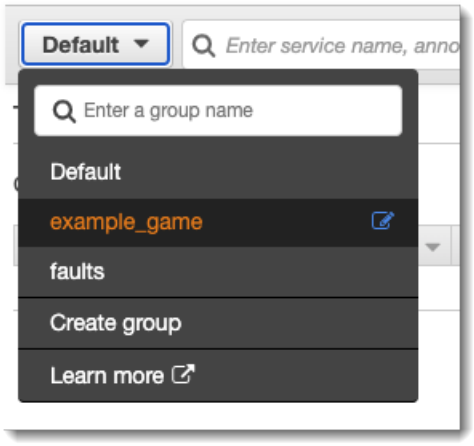
Identificare un tracciamento rilevante e visualizzare i dettagli

Quando si delimita l'ambito usando i filtri della console, l'elenco dei tracciamenti sotto le tabelle di parametri diventa più significativo. La tabella dell'elenco delle tracce unisce le informazioni su URL, USER e STATUS CODE in un'unica visualizzazione. Per informazioni più approfondite, selezionare una riga di questa tabella per aprire la pagina dei dettagli del tracciamento e visualizzare la sequenza temporale e i dati non elaborati.

Configurazione dei gruppi

I gruppi costituiscono una raccolta di tracciamenti definiti da un'espressione di filtro. Puoi utilizzare i gruppi per generare grafici di servizio aggiuntivi e fornire CloudWatch metriche Amazon. Puoi utilizzare la AWS X-Ray console o l'API X-Ray per creare e gestire gruppi per i tuoi servizi. Questo argomento descrive come creare e gestire gruppi utilizzando la console X-Ray. Per informazioni su come gestire i gruppi utilizzando l'API X-Ray, vedere. Gruppi

È possibile creare gruppi di tracce per mappe di traccia, tracce o analisi. Quando crei un gruppo, il gruppo diventa disponibile come filtro nel menu a discesa del gruppo su tutte e tre le pagine: Trace Map, Traces e Analytics.



I gruppi sono identificati dal nome o da un Amazon Resource Name (ARN) e contengono un'espressione di filtro. Il servizio confronta i tracciamenti in entrata con l'espressione e le memorizza di conseguenza. Per ulteriori informazioni su come creare un'espressione di filtro, vedereUtilizzo delle espressioni di filtro.

L'aggiornamento dell'espressione di filtro di un gruppo non modifica i dati già registrati. L'aggiornamento si applica solo per le successive tracce. Ciò può comportare un grafo unito delle espressioni nuove e vecchie. Per evitare ciò, eliminate un gruppo corrente e createne uno nuovo.

> **Note**
>
> I gruppi sono fatturati per il numero di tracce recuperate corrispondenti all'espressione di filtro. Per ulteriori informazioni, consultare Prezzi di AWS X-Ray.

Argomenti

- Creazione di un gruppo
- Applica un gruppo
- Modifica un gruppo
- Clonare un gruppo
- Eliminazione di un gruppo
- Visualizza le metriche di gruppo in Amazon CloudWatch

Creazione di un gruppo

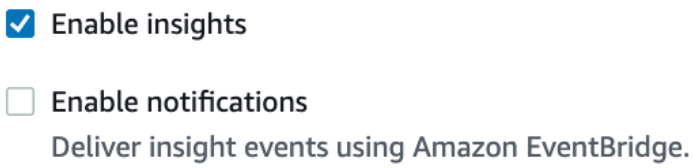
> **Note**
>
> Ora puoi configurare i gruppi X-Ray direttamente dalla console Amazon CloudWatch . È inoltre possibile continuare a utilizzare la console X-Ray.

CloudWatch console

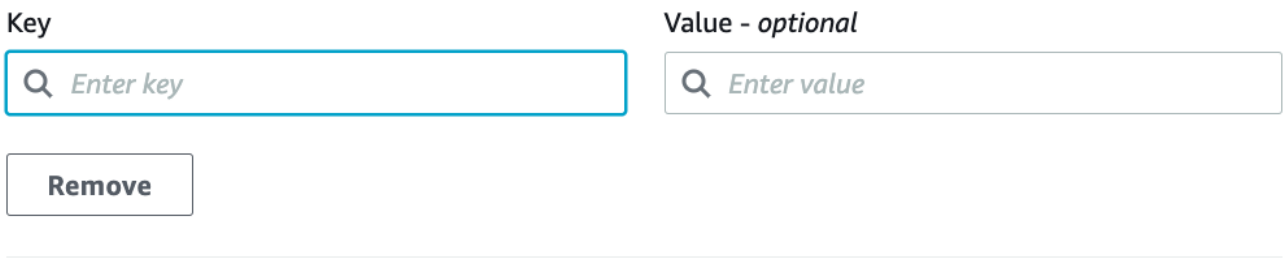
1. Accedi AWS Management Console e apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Scegli Impostazioni nel riquadro di navigazione a sinistra.
3. Scegli Visualizza impostazioni in Gruppi all'interno della sezione Tracce X-Ray.
4. Scegli Crea gruppo sopra l'elenco dei gruppi.
5. Nella pagina Crea gruppo, inserisci un nome per il gruppo. Il nome di un gruppo può avere un massimo di 32 caratteri e contenere caratteri alfanumerici e trattini. I nomi dei gruppi distinguono tra maiuscole e minuscole
6. Immettere un'espressione di filtro. Per ulteriori informazioni su come creare un'espressione di filtro, vedereUtilizzo delle espressioni di filtro. Nell'esempio seguente, il gruppo filtra le tracce di errore provenienti dal servizio `api.example.com` e le richieste al servizio in cui il tempo di risposta è stato maggiore o uguale a cinque secondi.

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

7. In Insights, abilita o disabilita l'accesso a Insights per il gruppo. Per ulteriori informazioni sulle informazioni dettagliate, consulta Utilizzo di X-Ray Insights.

8. In Tag, scegli Aggiungi nuovo tag per inserire una chiave di tag e, facoltativamente, un valore per il tag. Continua ad aggiungere altri tag, se lo desideri. Le chiavi dei tag devono essere uniche. Per eliminare un tag, scegli Rimuovi sotto ogni tag. Per ulteriori informazioni sui tag, consulta Tagging delle regole e dei gruppi di campionamento di X-Ray.



9. Seleziona Crea gruppo.

X-Ray console

1. Accedere AWS Management Console e aprire la console X-Ray all'indirizzo https://console.aws.amazon.com/xray/home.
2. Apri la pagina Crea gruppo dalla pagina Gruppi nel riquadro di navigazione a sinistra o dal menu di gruppo in una delle seguenti pagine: Trace Map, Traces e Analytics.
3. Nella pagina Crea gruppo, inserisci un nome per il gruppo. Il nome di un gruppo può avere un massimo di 32 caratteri e contenere caratteri alfanumerici e trattini. I nomi dei gruppi distinguono tra maiuscole e minuscole
4. Immettere un'espressione di filtro. Per ulteriori informazioni su come creare un'espressione di filtro, vedereUtilizzo delle espressioni di filtro. Nell'esempio seguente, il gruppo filtra le tracce di errore provenienti dal servizio `api.example.com` e le richieste al servizio in cui il tempo di risposta è stato maggiore o uguale a cinque secondi.

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

5. In Insights, abilita o disabilita l'accesso a Insights per il gruppo. Per ulteriori informazioni sulle informazioni dettagliate, consulta Utilizzo di X-Ray Insights.

6. In Tag, inserisci una chiave di tag e, facoltativamente, un valore del tag. Quando aggiungi un tag, viene visualizzata una nuova riga in cui inserire un altro tag. Le chiavi dei tag devono essere uniche. Per eliminare un tag, scegli X alla fine della riga del tag. Per ulteriori informazioni sui tag, consulta Tagging delle regole e dei gruppi di campionamento di X-Ray.

7. Seleziona Crea gruppo.

Applica un gruppo

CloudWatch console

1. Accedi AWS Management Console e apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Apri una delle seguenti pagine dal riquadro di navigazione sotto Tracce X-Ray:
   - Mappa di tracciamento
   - Tracce
3. Immettete il nome di un gruppo nel filtro di gruppo Filtra per X-Ray. I dati mostrati nella pagina vengono modificati in base all'espressione di filtro impostata nel gruppo.

X-Ray console

1. Accedere AWS Management Console e aprire la console X-Ray all'indirizzo https://console.aws.amazon.com/xray/home.
2. Apri una delle seguenti pagine dal pannello di navigazione:
   - Mappa di tracciamento
   - Tracce
   - Analisi
3. Nel menu del gruppo, scegli il gruppo in cui hai creatothe section called “Creazione di un gruppo”. I dati mostrati nella pagina vengono modificati in base all'espressione di filtro impostata nel gruppo.

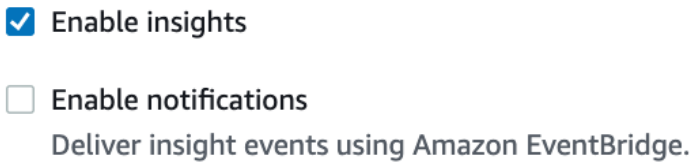
Modifica un gruppo

CloudWatch console

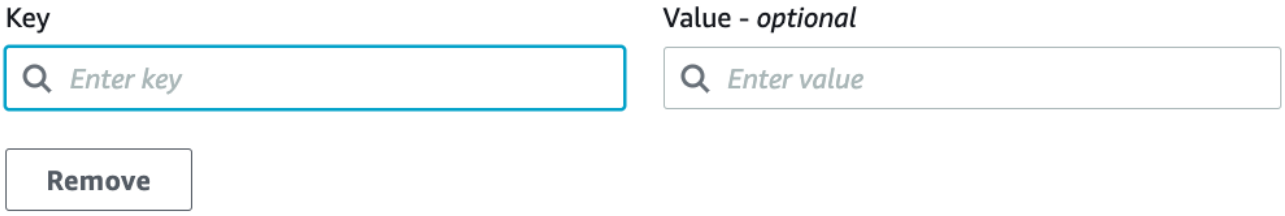
1. Accedi AWS Management Console e apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Scegli Impostazioni nel riquadro di navigazione a sinistra.
3. Scegli Visualizza impostazioni in Gruppi all'interno della sezione Tracce X-Ray.
4. Scegliete un gruppo dalla sezione Gruppi, quindi scegliete Modifica.
5. Sebbene non sia possibile rinominare un gruppo, è possibile aggiornare l'espressione del filtro. Per ulteriori informazioni su come creare un'espressione di filtro, consultaUtilizzo delle espressioni di filtro. Nell'esempio seguente, il gruppo filtra le tracce di errore provenienti dal servizio`api.example.com`, in cui è contenuto l'indirizzo URL della richiesta`example/game`, e il tempo di risposta per le richieste è stato maggiore o uguale a cinque secondi.

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

6. In Insights, abilita o disabilita l'accesso a Insights per il gruppo. Per ulteriori informazioni sulle informazioni dettagliate, consulta Utilizzo di X-Ray Insights.

7. In Tag, scegli Aggiungi nuovo tag per inserire una chiave di tag e, facoltativamente, un valore per il tag. Continua ad aggiungere altri tag, se lo desideri. Le chiavi dei tag devono essere uniche. Per eliminare un tag, scegli Rimuovi sotto ogni tag. Per ulteriori informazioni sui tag, consulta Tagging delle regole e dei gruppi di campionamento di X-Ray.

8. Quando hai finito di aggiornare il gruppo, scegli Aggiorna gruppo.

X-Ray console

1. Accedere AWS Management Console e aprire la console X-Ray all'indirizzo https://console.aws.amazon.com/xray/home.
2. Effettua una delle seguenti operazioni per aprire la pagina Modifica gruppo.

   a. Nella pagina Gruppi, scegli il nome di un gruppo per modificarlo.

   b. Nel menu del gruppo in una delle pagine seguenti, posiziona il puntatore su un gruppo, quindi scegli Modifica.

   - Traccia la mappa
   - Tracce
   - Analisi
3. Sebbene non sia possibile rinominare un gruppo, è possibile aggiornare l'espressione del filtro. Per ulteriori informazioni su come creare un'espressione di filtro, consultaUtilizzo delle espressioni di filtro. Nell'esempio seguente, il gruppo filtra le tracce di errore provenienti dal servizio`api.example.com`, in cui è contenuto l'indirizzo URL della richiesta`example/game`, e il tempo di risposta per le richieste è stato maggiore o uguale a cinque secondi.

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

4. In Insights, abilita o disabilita gli approfondimenti e le notifiche di approfondimenti per il gruppo. Per ulteriori informazioni sulle informazioni dettagliate, consulta Utilizzo di X-Ray Insights.

5. In Tag, modifica le chiavi e i valori dei tag. Le chiavi dei tag devono essere uniche. I valori dei tag sono facoltativi; puoi eliminarli, se lo desideri. Per eliminare un tag, scegli X alla fine della riga del tag. Per ulteriori informazioni sui tag, consulta Tagging delle regole e dei gruppi di campionamento di X-Ray.

6. Quando hai finito di aggiornare il gruppo, scegli Aggiorna gruppo.

Clonare un gruppo

La clonazione di un gruppo crea un nuovo gruppo con l'espressione di filtro e i tag di un gruppo esistente. Quando clonate un gruppo, il nuovo gruppo ha lo stesso nome del gruppo da cui è stato clonato, con l'`-clone`aggiunta del nome.

CloudWatch console

1. Accedi AWS Management Console e apri la console all'indirizzo https://console.aws.amazon.com/cloudwatch/. CloudWatch
2. Scegli Impostazioni nel riquadro di navigazione a sinistra.
3. Scegli Visualizza impostazioni in Gruppi all'interno della sezione Tracce X-Ray.
4. Scegli un gruppo dalla sezione Gruppi, quindi scegli Clona.
5. Nella pagina Crea gruppo, il nome del gruppo è *`-clonegroup-name`*. Facoltativamente, inserisci un nuovo nome per il gruppo. Il nome di un gruppo può avere un massimo di 32 caratteri e contenere caratteri alfanumerici e trattini. I nomi dei gruppi distinguono tra maiuscole e minuscole
6. È possibile mantenere l'espressione di filtro dal gruppo esistente o, facoltativamente, immettere una nuova espressione di filtro. Per ulteriori informazioni su come creare un'espressione di filtro, vedereUtilizzo delle espressioni di filtro. Nell'esempio seguente, il gruppo filtra le tracce di errore provenienti dal servizio `api.example.com` e le richieste al servizio in cui il tempo di risposta è stato maggiore o uguale a cinque secondi.

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

7. In Tag, modificate le chiavi e i valori dei tag, se necessario. Le chiavi dei tag devono essere uniche. I valori dei tag sono facoltativi; puoi eliminarli se lo desideri. Per eliminare un tag, scegli X alla fine della riga del tag. Per ulteriori informazioni sui tag, consulta Tagging delle regole e dei gruppi di campionamento di X-Ray.
8. Seleziona Crea gruppo.

X-Ray console

1. Accedere AWS Management Console e aprire la console X-Ray all'indirizzo https://console.aws.amazon.com/xray/home.
2. Apri la pagina Gruppi dal riquadro di navigazione a sinistra e scegli il nome del gruppo che desideri clonare.
3. Scegli il gruppo Clone dal menu Azioni.
4. Nella pagina Crea gruppo, il nome del gruppo è *-clonegroup-name*. Facoltativamente, inserisci un nuovo nome per il gruppo. Il nome di un gruppo può avere un massimo di 32 caratteri e contenere caratteri alfanumerici e trattini. I nomi dei gruppi distinguono tra maiuscole e minuscole
5. È possibile mantenere l'espressione di filtro dal gruppo esistente o, facoltativamente, immettere una nuova espressione di filtro. Per ulteriori informazioni su come creare un'espressione di filtro, vedereUtilizzo delle espressioni di filtro. Nell'esempio seguente, il gruppo filtra le tracce di errore provenienti dal servizio `api.example.com` e le richieste al servizio in cui il tempo di risposta è stato maggiore o uguale a cinque secondi.

   ```
   service("api.example.com") { fault = true OR responsetime >= 5 }
   ```
6. In Tag, modificate le chiavi e i valori dei tag, se necessario. Le chiavi dei tag devono essere uniche. I valori dei tag sono facoltativi; puoi eliminarli se lo desideri. Per eliminare un tag, scegli X alla fine della riga del tag. Per ulteriori informazioni sui tag, consulta Tagging delle regole e dei gruppi di campionamento di X-Ray.
7. Seleziona Crea gruppo.

Eliminazione di un gruppo

Segui i passaggi descritti in questa sezione per eliminare un gruppo. Non puoi eliminare il gruppo predefinito.

CloudWatch console

1. Accedi AWS Management Console e apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Scegli Impostazioni nel riquadro di navigazione a sinistra.
3. Scegli Visualizza impostazioni in Gruppi all'interno della sezione Tracce X-Ray.

4. Scegliete un gruppo dalla sezione Gruppi, quindi scegliete Elimina.
5. Quando ti viene richiesto di confermare, scegli Elimina.

X-Ray console

1. Accedere AWS Management Console e aprire la console X-Ray all'indirizzo https://console.aws.amazon.com/xray/home.
2. Apri la pagina Gruppi dal riquadro di navigazione a sinistra e scegli il nome del gruppo che desideri eliminare.
3. Nel menu Azioni, scegli Elimina gruppo.
4. Quando ti viene richiesto di confermare, scegli Elimina.

Visualizza le metriche di gruppo in Amazon CloudWatch

Dopo la creazione di un gruppo, le tracce in entrata vengono confrontate con l'espressione del filtro del gruppo mentre vengono archiviate nel servizio X-Ray. Le metriche relative al numero di tracce che soddisfano ogni criterio vengono pubblicate su Amazon CloudWatch ogni minuto. Scegliendo Visualizza metrica nella pagina Modifica gruppo si apre la CloudWatch console alla pagina Metrica. Per ulteriori informazioni su come utilizzare i CloudWatch parametri, consulta Using Amazon CloudWatch Metrics nella Amazon CloudWatch User Guide.

CloudWatch console

1. Accedi AWS Management Console e apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Scegli Impostazioni nel riquadro di navigazione a sinistra.
3. Scegli Visualizza impostazioni in Gruppi all'interno della sezione Tracce X-Ray.
4. Scegliete un gruppo dalla sezione Gruppi, quindi scegliete Modifica.
5. Nella pagina Modifica gruppo, scegli Visualizza metrica.

   La pagina Metriche della CloudWatch console si apre in una nuova scheda.

X-Ray console

1. Accedere AWS Management Console e aprire la console X-Ray all'indirizzo https://console.aws.amazon.com/xray/home.

2. Apri la pagina Gruppi dal riquadro di navigazione a sinistra, quindi scegli il nome del gruppo di cui desideri visualizzare le metriche.
3. Nella pagina Modifica gruppo, scegli Visualizza metrica.

   La pagina Metriche della CloudWatch console si apre in una nuova scheda.

Configurazione delle regole di campionamento di

Puoi utilizzare la AWS X-Ray console per configurare le regole di campionamento per i tuoi servizi. X-Ray SDK e Servizi AWS che supportano il tracciamento attivo con configurazione di campionamento utilizzano regole di campionamento per determinare quali richieste registrare.

Argomenti

- Configurazione delle regole di campionamento di
- Personalizzazione delle regole di campionamento
- Opzioni delle regole di campionamento
- Esempi di regole di campionamento
- Configurazione del servizio per l'utilizzo delle regole di campionamento
- Visualizzazione dei risultati del campionamento
- Passaggi successivi

Configurazione delle regole di campionamento di

Puoi configurare il campionamento per i seguenti casi d'uso:

- API Gateway Entrypoint: API Gateway supporta il campionamento e il tracciamento attivo. Per abilitare il tracciamento attivo su una fase API, consultare Supporto di tracciamento attivo di Amazon API Gateway per AWS X-Ray.
- AWS AppSync— AWS AppSync supporta il campionamento e il tracciamento attivo. Per abilitare il tracciamento attivo sulle AWS AppSync richieste, vedere Tracciamento con AWS X-Ray.
- Instrument X-Ray SDK su piattaforme di elaborazione: quando si utilizzano piattaforme di elaborazione come Amazon EC2, Amazon ECS AWS Elastic Beanstalk o, il campionamento è supportato se l'applicazione è stata strumentata con l'SDK X-Ray più recente.

Personalizzazione delle regole di campionamento

Personalizzando le regole di campionamento, puoi controllare la quantità di dati che registri. È inoltre possibile modificare il comportamento di campionamento senza modificare o ridistribuire il codice. Le regole di campionamento indicano all'X-Ray SDK quante richieste registrare per una serie di criteri. Per impostazione predefinita, l'SDK X-Ray registra la prima richiesta ogni secondo e il cinque percento di eventuali richieste aggiuntive. Una richiesta al secondo è la riserva. In questo modo viene registrata almeno una traccia al secondo, purché il servizio soddisfi le richieste. Cinque percento è la percentuale di richieste aggiuntive oltre la dimensione della riserva che vengono campionate.

Puoi configurare l'SDK X-Ray per leggere le regole di campionamento da un documento JSON incluso nel codice. Tuttavia, quando esegui più istanze del servizio, ogni istanza esegue il campionamento in modo indipendente. Ciò fa sì che la percentuale globale di richieste campionate aumenti perché tutte le riserve delle istanze sono cumulate insieme in modo efficace. Inoltre, per aggiornare le regole di campionamento locali, è necessario ridistribuire il codice.

Definendo le regole di campionamento nella console X-Ray e configurando l'SDK per leggere le regole dal servizio X-Ray, puoi evitare entrambi questi problemi. Il servizio gestisce la riserva per ciascuna regola e assegna delle quote ad ogni istanza del servizio per distribuire la riserva in modo uniforme, in base al numero di istanze in esecuzione. Il limite del serbatoio viene calcolato in base alle regole impostate. Poiché le regole sono configurate nel servizio, è possibile gestirle senza effettuare implementazioni aggiuntive.

> **Note**
>
> X-Ray utilizza un approccio ottimale nell'applicazione delle regole di campionamento e, in alcuni casi, la frequenza di campionamento effettiva potrebbe non corrispondere esattamente alle regole di campionamento configurate. Tuttavia, nel tempo il numero di richieste campionate dovrebbe avvicinarsi alla percentuale configurata.

Ora puoi configurare le regole di campionamento a raggi X direttamente dalla console Amazon. CloudWatch È inoltre possibile continuare a utilizzare la console X-Ray.

CloudWatch console

Per configurare le regole di campionamento nella console CloudWatch

1. Accedere AWS Management Console e aprire la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Scegli Impostazioni nel riquadro di navigazione a sinistra.
3. Scegliete Visualizza impostazioni in Regole di campionamento all'interno della sezione Tracce X-Ray.
4. Per creare una regola, scegliere Create sampling rule (Crea regola di campionamento).

   Per modificare una regola, scegli una regola e scegli Modifica per modificarla.

   Per eliminare una regola, scegli una regola e scegli Elimina per eliminarla.

X-Ray console

Per configurare le regole di campionamento nella console X-Ray

1. Apri la console X-Ray.
2. Scegli Sampling nel riquadro di navigazione a sinistra.
3. Per creare una regola, scegliere Create sampling rule (Crea regola di campionamento).

   Per modificare una regola, scegliere il nome di una regola.

   Per eliminare una regola, sceglierne una e utilizzare il menu Actions (Operazioni) per eliminarla.

Opzioni delle regole di campionamento

Per ogni regola sono disponibili le seguenti opzioni. I valori di stringa possono utilizzare caratteri jolly per corrispondere a un solo carattere (?) o a zero o a più caratteri (*).

Opzioni delle regole di campionamento

- Nome regola (stringa): un nome univoco per la regola.

- Priorità (numero intero compreso tra 1 e 9999): la priorità della regola di campionamento. I servizi valutano le regole in ordine crescente di priorità e prendono una decisione sul campionamento in base alla prima regola corrispondente.
- Reservoir (numero intero non negativo): un numero fisso di richieste corrispondenti allo strumento al secondo, prima dell'applicazione del tasso fisso. Il reservoir non viene utilizzato direttamente dai servizi, ma si applica a tutti i servizi che utilizzano la regola nel loro complesso.
- Tasso (numero intero compreso tra 0 e 100): la percentuale di richieste corrispondenti allo strumento, dopo l'esaurimento del serbatoio. Quando configuri una regola di campionamento nella console, scegli una percentuale compresa tra 0 e 100. Quando configuri una regola di campionamento in un SDK client utilizzando un documento JSON, fornisci un valore percentuale compreso tra 0 e 1.
- Nome del servizio (stringa): il nome del servizio strumentato, così come appare nella mappa di traccia.
  - X-Ray SDK: il nome del servizio configurato sul registratore.
  - Amazon API Gateway —*`api-name`*/*`stage`*.
- Tipo di servizio (stringa): il tipo di servizio, così come appare nella mappa di traccia. Per X-Ray SDK, imposta il tipo di servizio applicando il plug-in appropriato:
  - `AWS::ElasticBeanstalk::Environment`— Un AWS Elastic Beanstalk ambiente (plugin).
  - `AWS::EC2::Instance`— Un'istanza Amazon EC2 (plug-in).
  - `AWS::ECS::Container`— Un contenitore Amazon ECS (plug-in).
  - `AWS::APIGateway::Stage`— Una fase di Amazon API Gateway.
  - `AWS::AppSync::GraphQLAPI` — Una richiesta AWS AppSync API.
- Host (stringa): il nome host dall'intestazione dell'host HTTP.
- Metodo HTTP (stringa) — Il metodo della richiesta HTTP.
- Percorso URL (stringa) — Il percorso URL della richiesta.
  - X-Ray SDK: la parte del percorso dell'URL della richiesta HTTP.
- ARN della risorsa (stringa) — L'ARN della AWS risorsa che esegue il servizio.
  - X-Ray SDK: non supportato. L'SDK può solo utilizzare le regole con Resource ARN (ARN risorse) impostato su *.
  - Amazon API Gateway: l'ARN di fase.
- (Facoltativo) Attributi (chiave e valore): attributi del segmento noti al momento della decisione di campionamento.

- X-Ray SDK: non supportato. L'SDK ignora le regole che specificano degli attributi.
- Amazon API Gateway: intestazioni della richiesta HTTP originale.

Esempi di regole di campionamento

Example — Regola predefinita senza riserva e con una tariffa bassa

Puoi modificare la riserva e la percentuale della regola di default. La regola di default si applica alle richieste che non soddisfano nessun'altra regola.

- Serbatoio: **`0`**
- Frequenza: **`5`** (**`0.05`**se configurata utilizzando un documento JSON)

Example — Regola di debug per tracciare tutte le richieste relative a un percorso problematico

Un regola ad alta priorità applicata temporaneamente per il debug.

- Nome della regola: **`DEBUG - history updates`**
- Priorità: **`1`**
- Serbatoio: **`1`**
- Frequenza: **`100`** (**`1`**se configurata utilizzando un documento JSON)
- Nome del servizio: **`Scorekeep`**
- Tipo di servizio: **`*`**
- Ospite: **`*`**
- Metodo HTTP: **`PUT`**
- Percorso URL: **`/history/*`**
- ARN della risorsa: **`*`**

Example — Tariffa minima più elevata per i POST

- Nome della regola: **`POST minimum`**
- Priorità: **`100`**
- Serbatoio: **`10`**
- Frequenza: **`10`** (**`.1`**se configurata utilizzando un documento JSON)

- Nome del servizio: *****
- Tipo di servizio: *****
- Ospite: *****
- Metodo HTTP: **`POST`**
- Percorso URL: *****
- ARN della risorsa: *****

Configurazione del servizio per l'utilizzo delle regole di campionamento

L'SDK X-Ray richiede una configurazione aggiuntiva per utilizzare le regole di campionamento configurate nella console. Consulta la sezione dedicata alla configurazione per il tuo linguaggio per scoprire i dettagli sulla configurazione di una strategia di campionamento:

- Java: Regole di campionamento
- Vai: Regole di campionamento
- Node.js: Regole di campionamento
- Python: Regole di campionamento
- Rubino: Regole di campionamento
- .NET: Regole di campionamento

Per API Gateway, vedereSupporto di tracciamento attivo di Amazon API Gateway per AWS X-Ray.

Visualizzazione dei risultati del campionamento

La pagina di campionamento della console X-Ray mostra informazioni dettagliate su come i servizi utilizzano ciascuna regola di campionamento.

La colonna Trend (Tendenza) mostra come la regola è stata utilizzata negli ultimi minuti. Ogni colonna mostra le statistiche relative a una finestra di 10 secondi.

Statistiche di campionamento

- Regola di corrispondenza totale: il numero di richieste corrispondenti a questa regola. Questo numero non include le richieste che avrebbero potuto soddisfare questa regola, ma che hanno soddisfatto prima una regola ad alta priorità.
- Totale campionato: il numero di richieste registrate.

- Campionato con tasso fisso: il numero di richieste campionate applicando la tariffa fissa della regola.
- Campionato con limite di riserva: il numero di richieste campionate utilizzando una quota assegnata da X-Ray.
- Prese in prestito dal serbatoio: il numero di richieste prese in prestito dal serbatoio. La prima volta che un servizio abbina una richiesta a una regola, X-Ray non gli ha ancora assegnato una quota. Tuttavia, se il serbatoio è almeno 1, il servizio prende in prestito una traccia al secondo fino a quando X-Ray non assegna una quota.

Per ulteriori informazioni sulle statistiche di campionamento e su come i servizi utilizzano le regole di campionamento, consulta Utilizzo delle regole di campionamento con l'API X-Ray.

Passaggi successivi

È possibile utilizzare l'API X-Ray per gestire le regole di campionamento. Con l'API puoi creare e aggiornare le regole in modo programmatico in base a una pianificazione o in risposta ad allarmi o notifiche. Consulta Configurazione delle impostazioni di campionamento e di crittografia tramite l'API AWS X-Ray per le istruzioni di utilizzo e ulteriori esempi di regole.

X-Ray SDK utilizza Servizi AWS anche l'API X-Ray per leggere le regole di campionamento, riportare i risultati del campionamento e ottenere obiettivi di campionamento. I servizi devono tenere traccia della frequenza con cui applicano ogni regola, valutare le regole in base alla priorità e prendere in prestito dal serbatoio quando una richiesta corrisponde a una regola per la quale X-Ray non ha ancora assegnato una quota al servizio. Per ulteriori informazioni su come un servizio utilizza l'API per la campionatura, consulta Utilizzo delle regole di campionamento con l'API X-Ray.

Quando l'SDK X-Ray chiama le API di campionamento, utilizza il demone X-Ray come proxy. Se la porta TCP 2000 è già utilizzata, puoi configurare il daemon per funzionare da proxy su un'altra porta. Per informazioni dettagliate, vedi Configurazione del demone AWS X-Ray.

Deep linking tra console

Puoi utilizzare percorsi e query per creare collegamenti diretti a tracce specifiche o visualizzazioni filtrate delle tracce e della mappa di traccia.

Pagine della console

- Pagina di benvenuto — xray/home#/welcome

- Guida introduttiva — xray/home#/getting-started
- Mappa delle tracce — xray/home#/service-map
- Tracce — xray/home#/trace

Tracce

Puoi generare link per la sequenza temporale, i dati non elaborati e le visualizzazioni sulla mappa di specifici tracciamenti.

Traccia la cronologia — `xray/home#/traces/`*`trace-id`*

Dati di tracciamento grezzi — `xray/home#/traces/`*`trace-id`*`/raw`

Example — dati di traccia non elaborati

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

Espressioni filtro

Collegamento a un elenco filtrato di tracciamenti.

Visualizzazione filtrata delle tracce — `xray/home#/traces?filter=`*`filter-expression`*

Example — espressione di filtro

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

Example — espressione di filtro (URL codificato)

```
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
```

Per ulteriori informazioni sulle espressioni filtro, consulta Utilizzo delle espressioni di filtro.

Intervallo temporale

Specifica la durata di un intervallo di tempo o le ore di inizio e fine in formato ISO8601. Gli intervalli di tempo sono in UTC e possono durare fino a 6 ore.

Periodo di tempo — `xray/home#/page?timeRange=range-in-minutes`

Example — mappa di tracciamento dell'ultima ora

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

Ora di inizio e fine — `xray/home#/page?timeRange=start~end`

Example — intervallo di tempo preciso in secondi

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

Example — intervallo di tempo preciso in minuti

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

Regione

Specificate un link Regione AWS alle pagine di quella regione. Se non si specifica una regione, la console reindirizza l'utente all'ultima regione visitata.

Regione: `xray/home?region=region#/page`

Example — mappa di tracciamento negli Stati Uniti occidentali (Oregon) (us-west-2)

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

Quando includete una regione con altri parametri di interrogazione, la query Region precede l'hash e le query specifiche per i raggi X vanno dopo il nome della pagina.

Example — tracciate la mappa dell'ultima ora negli Stati Uniti occidentali (Oregon) (us-west-2)

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

Combinazioni

Example — tracce recenti con un filtro di durata

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

Output

- Pagina — Tracce
- Intervallo di tempo: ultimi 15 minuti
- Filtro: durata >= 5 E durata <= 8

AWS X-Ray demone

> **Note**
>
> Ora puoi utilizzare l' CloudWatch agente per raccogliere parametri, log e tracce dalle istanze Amazon EC2 e dai server locali. CloudWatch la versione dell'agente 1.300025.0 e successive può raccogliere tracce dagli SDK dei nostri client OpenTelemetryX-Ray e inviarle a X-Ray. L'utilizzo dell' CloudWatch agente al posto del demone AWS Distro for OpenTelemetry (ADOT) Collector o X-Ray per raccogliere le tracce può aiutare a ridurre il numero di agenti da gestire. Per ulteriori informazioni, consultate l'argomento relativo all'CloudWatch agente nella Guida per l'utente CloudWatch .

Il AWS X-Ray daemon è un'applicazione software che ascolta il traffico sulla porta UDP 2000, raccoglie i dati grezzi dei segmenti e li inoltra all'API. AWS X-Ray Il daemon funziona insieme agli AWS X-Ray SDK e deve essere in esecuzione in modo che i dati inviati dagli SDK possano raggiungere il servizio X-Ray. Il demone X-Ray è un progetto open source. Puoi seguire il progetto e inviare problemi e pull request su: github.com/aws/ GitHub aws-xray-daemon

Su AWS Lambda e AWS Elastic Beanstalk, usa l'integrazione di questi servizi con X-Ray per eseguire il demone. Lambda esegue il demone automaticamente ogni volta che viene richiamata una funzione per una richiesta campionata. Su Elastic Beanstalk, `XRayEnabled` usa l'opzione di configurazione per eseguire il demone sulle istanze del tuo ambiente. Per ulteriori informazioni, consulta la pagina

Per eseguire il demone X-Ray localmente, in locale o in altro modo, è necessario scaricarlo Servizi AWS, eseguirlo e quindi autorizzarlo a caricare i documenti dei segmenti su X-Ray.

Download del daemon

Puoi scaricare il daemon da Amazon S3, Amazon ECR o Docker Hub, quindi eseguirlo localmente o installarlo su un'istanza Amazon EC2 all'avvio.

Amazon S3

Installatori ed eseguibili dei daemon X-Ray

- Linux (eseguibile) — (sig) **`aws-xray-daemon-linux-3.x.zip`**

- Linux (programma di installazione RPM) — aws-xray-daemon-3.x.rpm
- Linux (programma di installazione DEB) — aws-xray-daemon-3.x.deb
- Linux (ARM64, eseguibile) — **aws-xray-daemon-linux-arm64-3.x.zip**(sig)
- Linux (ARM64, programma di installazione RPM) — aws-xray-daemon-arm64-3.x.rpm
- Linux (ARM64, programma di installazione DEB) — aws-xray-daemon-arm64-3.x.deb
- OS X (eseguibile) — **aws-xray-daemon-macos-3.x.zip**(sig)
- Windows (eseguibile) — aws-xray-daemon-windows-process-3.x.zip(sig)
- Windows (servizio) — aws-xray-daemon-windows-service-3.x.zip(sig)

Questi collegamenti rimandano sempre all'ultima versione 3.x del demone. Per scaricare una determinata versione, sostituire `3.x` con il numero di versione. Ad esempio, `2.1.0`.

Le risorse X-Ray vengono replicate su bucket in ogni regione supportata. Per usare il bucket più vicino a te o alle tue risorse AWS , sostituisci la regione nei precedenti collegamenti con il nome della tua regione.

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

Amazon ECR

A partire dalla versione 3.2.0, il daemon è disponibile su Amazon ECR. Prima di estrarre un'immagine, devi autenticare il tuo client docker nel registro pubblico Amazon ECR.

Estrai il tag della versione 3.x rilasciata più recente eseguendo il seguente comando:

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

Le versioni precedenti o alpha possono essere scaricate sostituendole `3.x` con `alpha` o con un numero di versione specifico. Non è consigliabile utilizzare un'immagine daemon con un tag alfa in un ambiente di produzione.

Docker Hub

Il demone può essere trovato su Docker Hub. Per scaricare l'ultima versione 3.x rilasciata, esegui il seguente comando:

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

Le versioni precedenti del demone possono essere rilasciate sostituendole `3.x` con la versione desiderata.

Verifica della firma dell'archivio del daemon

I file di firma GPG per le risorse del daemon compresse in archivi ZIP sono inclusi. La chiave pubblica si trova qui: [aws-xray.gpg].

Puoi utilizzare la chiave pubblica per verificare che l'archivio ZIP del daemon sia originale e non modificato. Prima di tutto, importa la chiave pubblica con GnuPG.

Per importare la chiave pubblica

1. Scarica la chiave pubblica.

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

2. Importa la chiave pubblica nel tuo keyring.

```
$ 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
```

Utilizza la chiave importata per verificare la firma del dell'archivio ZIP del daemon.

Per verificare la firma di un archivio

1. Scaricare l'archivio e il file della firma.

```
$ 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. Eseguire `gpg --verify` per verificare la firma.

```
$ 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
```

Prendi nota dell'avviso sulla trust. Una chiave è considerata attendibile solo se è stata firmata da te o da un utente attendibile. Questo non significa che la firma non sia valida, ma soltanto che la chiave pubblica non è stata verificata.

Esecuzione del daemon

Esegui il daemon localmente dalla riga di comando. Utilizza l'opzione `-o` per eseguirlo in modalità locale e `-n` per impostare la regione.

```
~/Downloads$ ./xray -o -n us-east-2
```

Per ulteriori istruzioni specifiche per la piattaforma, consulta le seguenti sezioni:

- Linux (locale) — Esegui il daemon X-Ray su Linux
- Windows (locale) — Esegui X-Ray daemon su Windows
- Elastic Beanstalk — Esecuzione del daemon X-RayAWS Elastic Beanstalk
- Amazon EC2 — Esecuzione del daemon X-Ray su Amazon EC2
- Amazon ECS — Esecuzione del daemon X-Ray su Amazon ECECS ECS ECS ECS EC

Puoi personalizzare ulteriormente il comportamento del daemon utilizzando le opzioni della riga di comando o un file di configurazione. Per informazioni dettagliate, vedi Configurazione del demone AWS X-Ray.

Dare al demone il permesso di inviare dati a X-Ray

Il demone X-Ray utilizza l' AWS SDK per caricare i dati di traccia su X-Ray e per farlo necessita AWS di credenziali con autorizzazione.

Su Amazon EC2, il daemon utilizza automaticamente il ruolo del profilo dell'istanza. Per informazioni sulle credenziali richieste per eseguire il demone localmente, consulta Esecuzione dell'applicazione localmente.

Se specifichi le credenziali in più di una posizione (file delle credenziali, profilo dell'istanza oppure variabili di ambiente), le credenziali utilizzate sono determinate dalla catena del provider dell'SDK. Per ulteriori informazioni sulla fornitura di credenziali all'SDK, consulta Specificing Credentials nella SDK for AWS Go Developer Guide.

Il ruolo o l'utente IAM a cui appartengono le credenziali del daemon devono avere l'autorizzazione di scrivere i dati sul servizio per tuo conto.

- Per utilizzare il daemon su Amazon EC2, crea un nuovo ruolo del profilo di istanza o aggiungi la policy gestita a una esistente.
- Per utilizzare il daemon su Elastic Beanstalk, aggiungi la policy gestita al ruolo del profilo di istanza predefinito di Elastic Beanstalk.
- Per eseguire il daemon localmente, consulta Esecuzione dell'applicazione localmente.

Per ulteriori informazioni, consulta Gestione delle identità e degli accessi per AWS X-Ray.

Registri dei daemon X-Ray

Il daemon genera informazioni sulla sua configurazione corrente e sui segmenti che invia a 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)
```

Per impostazione predefinita, il daemon invia i log in output su STDOUT. Se esegui il daemon in background, utilizza l'opzione della riga di comando `--log-file` o un file di configurazione per impostare il percorso del file di log. Puoi anche impostare il livello di tracciamento e disabilitare la rotazione dei log. Per istruzioni, consulta Configurazione del demone AWS X-Ray.

Su Elastic Beanstalk, la piattaforma imposta la posizione dei log del demone. Per informazioni dettagliate, vedi Esecuzione del daemon X-RayAWS Elastic Beanstalk.

Configurazione del demone AWS X-Ray

È possibile utilizzare le opzioni della riga di comando o un file di configurazione per personalizzare il comportamento del demone X-Ray. La maggior parte delle opzioni sono disponibili utilizzando entrambi i metodi, ma alcune sono disponibili solo per i file di configurazione e alcune solo utilizzando la riga di comando.

Per iniziare, l'unica opzione che devi conoscere è `-n` or`--region`, che usi per impostare la regione che il demone usa per inviare i dati di traccia a X-Ray.

```
~/xray-daemon$ ./xray -n us-east-2
```

Se esegui il daemon localmente, ovvero non su Amazon EC2, puoi aggiungere l'opzione di saltare il controllo delle credenziali `-o` del profilo, ad esempio, in modo che il daemon sia pronto più rapidamente.

```
~/xray-daemon$ ./xray -o -n us-east-2
```

Il resto delle opzioni della riga di comando consentono di configurare le opzioni di log, ascoltare su una porta diversa, limitare la quantità di memoria che il daemon può utilizzare o assumere un ruolo per l'invio dei dati di tracciamento a un altro account.

È possibile passare un file di configurazione al demone per accedere a opzioni di configurazione avanzate e fare cose come limitare il numero di chiamate simultanee a X-Ray, disabilitare la rotazione dei log e inviare traffico a un proxy.

Sections

- Variabili di ambiente supportate
- Utilizzo delle opzioni della riga di comando
- Utilizzo di un file di configurazione

Variabili di ambiente supportate

Il demone X-Ray supporta le seguenti variabili di ambiente:

- `AWS_REGION`— specifica l'endpoint Regione AWSdel servizio X-Ray.

- `HTTPS_PROXY`— Speciifica un indirizzo proxy per il demone tramite il quale caricare i segmenti. Può trattarsi dei nomi di dominio DNS o degli indirizzi IP e dei numeri di porta utilizzati dai server proxy.

Utilizzo delle opzioni della riga di comando

Passa queste opzioni al daemon quando lo esegui in locale o tramite un script dati utente.

Opzioni della riga di comando

- `-b`, `--bind` — Ascolta i documenti dei segmenti su una porta UDP diversa.

```
--bind "127.0.0.1:3000"
```

  Predefinito:`2000`.

- `-t`, `--bind-tcp` — Ascolta le chiamate al servizio X-Ray su una porta TCP diversa.

```
-bind-tcp "127.0.0.1:3000"
```

  Predefinito:. `2000`

- `-c`, `--config` — Carica un file di configurazione dal percorso specificato.

```
--config "/home/ec2-user/xray-daemon.yaml"
```

- `-f`, `--log-file` — Registri di output nel percorso del file specificato.

```
--log-file "/var/log/xray-daemon.log"
```

- `-l`, `--log-level` — Livello di log, dal più dettagliato al meno dettagliato: dev, debug, info, warn, error, prod.

```
--log-level warn
```

  Predefinito: `prod`

- `-m`, `--buffer-memory` — Modifica la quantità di memoria in MB che i buffer possono utilizzare (minimo 3).

```
--buffer-memory 50
```

Impostazione predefinita: 1% della memoria disponibile.

- `-o`, `--local-mode` — Non controllare i metadati delle istanze EC2.
- `-r`, `--role-arn` — Assumi il ruolo IAM specificato per caricare i segmenti su un altro account.

```
--role-arn "arn:aws:iam::123456789012:role/xray-cross-account"
```

- `-a`, `--resource-arn` — Amazon Resource Name (ARN) della AWS risorsa che esegue il daemon.
- `-p`, `--proxy-address` — Carica i segmenti tramite un proxy. AWS X-Ray È necessario specificare il protocollo del server proxy.

```
--proxy-address "http://192.0.2.0:3000"
```

- `-n`, `--region` — Invia segmenti al servizio X-Ray in una regione specifica.
- `-v`, `--version` — Mostra la versione del AWS X-Ray demone.
- `-h`, `--help` — Mostra la schermata di aiuto.

Utilizzo di un file di configurazione

Per configurare il daemon puoi anche utilizzare un file in formato YAML. Trasferisci il file di configurazione al daemon utilizzando l'opzione `-c`.

```
~$ ./xray -c ~/xray-daemon.yaml
```

Opzioni file di configurazione

- `TotalBufferSizeMB`— Dimensione massima del buffer in MB (minimo 3). Scegli 0 per utilizzare l'1% della memoria dell'host.
- `Concurrency`— Numero massimo di chiamate simultanee per caricare documenti AWS X-Ray di segmento.
- `Region`— Invia segmenti al AWS X-Ray servizio in una regione specifica.
- `Socket`— Configura l'associazione del demone.
  - `UDPAddress`— Cambia la porta su cui il demone ascolta.

  - `TCPAddress`— Ascolta le chiamate al servizio X-Ray su una porta TCP diversa.
- `Logging`— Configura il comportamento di registrazione.
  - `LogRotation`— Impostato per `false` disabilitare la rotazione dei log.
  - `LogLevel`— Modifica il livello di registro, dal più dettagliato al meno:`dev`,`debug`, `info` o`prod`, `warnerror`,`prod`. L'impostazione predefinita è`prod`, che è equivalente a`info`.
  - `LogPath`— Registri di output nel percorso del file specificato.
- `LocalMode`— Impostato per `true` ignorare il controllo dei metadati delle istanze EC2.
- `ResourceARN`— Amazon Resource Name (ARN) della AWS risorsa che esegue il daemon.
- `RoleARN`— Assumi il ruolo IAM specificato per caricare i segmenti su un altro account.
- `ProxyAddress`— Carica i segmenti AWS X-Ray tramite un proxy.
- `Endpoint`— Modificare l'endpoint del servizio X-Ray a cui il demone invia i documenti del segmento.
- `NoVerifySSL`— Disabilita la verifica del certificato TLS.
- `Version`— Versione del formato del file di configurazione del demone. La versione del formato del file è un campo obbligatorio.

Example Xray-daemon.yaml

Questo file di configurazione modifica la porta di ascolto del daemon a 3000, disattiva i controlli dei metadati dell'istanza, imposta un ruolo da utilizzare per il caricamento dei segmenti e modifica regione e opzioni di logging.

```
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
```

Esegui il daemon X-Ray

Puoi eseguire il daemon AWS X-Ray in locale su Linux, MacOS, Windows o in un container Docker. Esegui il daemon per inoltrare i dati di traccia a X-Ray durante lo sviluppo e il test dell'applicazione strumentata. Scarica ed estrai il daemon utilizzando le istruzioni contenute in questa pagina.

Quando viene eseguito localmente, il demone può leggere le credenziali da un file di credenzialiAWS SDK (`.aws/credentials`nella directory utente) o da variabili di ambiente. Per ulteriori informazioni, consulta Dare al demone il permesso di inviare dati a X-Ray.

Il daemon resta in attesa dei dati UDP sulla porta 2000. Puoi modificare la porta e altre opzioni utilizzando un file di configurazione e le opzioni della riga di comando. Per ulteriori informazioni, consulta Configurazione del demone AWS X-Ray.

Esegui il daemon X-Ray su Linux

Puoi avviare il file eseguibile del daemon dalla riga di comando. Utilizza l'opzione `-o` per eseguirlo in modalità locale e `-n` per impostare la regione.

```
~/xray-daemon$ ./xray -o -n us-east-2
```

Per eseguire il daemon in background, utilizza &.

```
~/xray-daemon$ ./xray -o -n us-east-2 &
```

Termina un processo daemon in esecuzione in background `pkill`.

```
~$ pkill xray
```

Esegui X-Ray daemon in un contenitore Docker

Per eseguire il daemon in locale in un container Docker, salva il testo seguente in un file denominato `Dockerfile`. Scarica l'immagine di esempio completa su Amazon ECR. Per ulteriori informazioni, consulta il download del demone.

Example Docker

```
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
```

Crea l'immagine del container `docker build`.

```
~/xray-daemon$ docker build -t xray-daemon .
```

Esegui l'immagine in un container con `docker run`.

```
~/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
```

Questo comando utilizza le seguenti opzioni:

- `--attach STDOUT`— Visualizza l'output del demone nel terminale.
- `-v ~/.aws/:/root/.aws/:ro`— Concedi al contenitore l'accesso in sola lettura alla `.aws` directory per consentirgli di leggere le tue credenzialiAWS SDK.
- `AWS_REGION=`*`us-east-2`*— Imposta la variabile di`AWS_REGION` ambiente per dire al demone quale regione usare.
- `--net=host`— Collegare il contenitore alla`host` rete. I containers sulla rete dell'host possono comunicare tra loro senza pubblicare le porte.
- `-p 2000:2000/udp`— Associa la porta UDP 2000 della tua macchina alla stessa porta del contenitore. Questo non è necessaria per la comunicazione tra container nella stessa rete, ma consente di inviare segmenti al daemon dalla riga di comando o da un'applicazione non in esecuzione su Docker.
- `--name xray-daemon`— Assegna un nome al contenitore`xray-daemon` invece di generare un nome casuale.

- `-o`(dopo il nome dell'immagine) — Aggiunge l'`-o`opzione al punto di ingresso che esegue il demone all'interno del contenitore. Questa opzione indica al demone di funzionare in modalità locale per evitare che tenti di leggere i metadati dell'istanza Amazon EC2.

Per arrestare il daemon, utilizza `docker stop`. Se effettui modifiche al file `Dockerfile` e crei una nuova immagine, devi cancellare il container esistente prima di poterne creare un altro con lo stesso nome. Per eliminare il container, utilizza `docker rm`.

```
$ docker stop xray-daemon
$ docker rm xray-daemon
```

Esegui X-Ray daemon su Windows

Puoi avviare il file eseguibile del daemon dalla riga di comando. Utilizza l'opzione `-o` per eseguirlo in modalità locale e `-n` per impostare la regione.

```
> .\xray_windows.exe -o -n us-east-2
```

Usa uno PowerShell script per creare ed eseguire un servizio per il demone.

Example PowerShell sceneggiatura - 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
```

Esegui il daemon su OS X

Puoi avviare il file eseguibile del daemon dalla riga di comando. Utilizza l'opzione `-o` per eseguirlo in modalità locale e `-n` per impostare la regione.

```
~/xray-daemon$ ./xray_mac -o -n us-east-2
```

Per eseguire il daemon in background, utilizza &.

```
~/xray-daemon$ ./xray_mac -o -n us-east-2 &
```

Utilizza nohup per evitare che il daemon venga terminato alla chiusura del terminale.

```
~/xray-daemon$ nohup ./xray_mac &
```

Esecuzione del daemon X-RayAWS Elastic Beanstalk

Per inoltrare i dati di traccia dall'applicazione aAWS X-Ray, puoi eseguire il daemon X-Ray sulle istanze Amazon EC2 dell'ambiente Elastic Beanstalk. Per un elenco delle piattaforme supportate, consultaConfigurazione diAWS X-RayDebugnellaAWS Elastic BeanstalkGuida per gli sviluppatori.

> **Note**
>
> Il daemon utilizza il profilo dell'istanza dell'ambiente per acquisire le autorizzazioni. Per le istruzioni sull'aggiunta di autorizzazioni al profilo dell'istanza Elastic Beanstalk, consultaDare al demone il permesso di inviare dati a X-Ray.

Le piattaforme Elastic Beanstalk forniscono un'opzione di configurazione che è puoi impostare per eseguire il daemon automaticamente. È possibile abilitare il daemon in un file di configurazione nel tuo codice sorgente o scegliendo un'opzione nella console Elastic Beanstalk. Quando abiliti l'opzione di configurazione, il daemon viene installato sull'istanza e viene eseguito come servizio.

La versione inclusa nelle piattaforme Elastic Beanstalk potrebbe non essere la versione più recente. Consulta la sezione Piattaforme Supportate per scoprire la versione del daemon che è disponibile per la configurazione della tua piattaforma.

Elastic Beanstalk non fornisce il daemon X-Ray sulla piattaforma Multicontainer Docker (Amazon ECS).

Utilizzo dell'integrazione Elastic Beanstalk X-Ray per eseguire il daemon X-Ray

Utilizza la console per attivare l'integrazione X-Ray oppure configurala nel codice sorgente dell'applicazione con un file di configurazione.

Per abilitare il daemon X-Ray nella console Elastic Beanstalk

1. Apertura dellaConsole Elastic Beanstalk.
2. Passare alla .Console di gestioneper l'ambiente.
3. Scegliere Configuration (Configurazione).
4. Scegliere Software Settings (Impostazioni software)
5. Alla voce X-Ray Daemon (Daemon X-Ray), scegliere Enabled (Abilitato).
6. Seleziona Apply (Applica).

Per rendere la configurazione portabile tra più ambienti puoi includere un file di configurazione nel codice sorgente.

Example .ebextensions/xray-daemon.config

```
option_settings:
  aws:elasticbeanstalk:xray:
    XRayEnabled: true
```

Elastic Beanstalk passa un file di configurazione al daemon e invia i log in una posizione standard.

Sulle piattaforme Windows Server

- File di configurazione–`C:\Program Files\Amazon\XRay\cfg.yaml`
- Log–`c:\Program Files\Amazon\XRay\logs\xray-service.log`

Sulle piattaforme Linux

- File di configurazione–`/etc/amazon/xray/cfg.yaml`
- Log–`/var/log/xray/xray.log`

Elastic Beanstalk fornisce strumenti per estrarre i log delle istanze dalAWS Management Consoleo riga di comando. Puoi stabilire che Elastic Beanstalk includa i log del daemon X-Ray di aggiungere un'attività con un file di configurazione.

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
```

Consulta .Visualizzazione dei log dalle istanze Amazon EC2 dell'ambiente Elastic BeanstalknellaAWS Elastic BeanstalkGuida per gli sviluppatoriper ulteriori informazioni.

Scaricamento ed esecuzione del daemon X-Ray in modalità manuale (Avanzata)

Se il daemon X-Ray non è disponibile per la configurazione della tua piattaforma, puoi scaricarlo da Amazon S3 ed eseguirlo con un file di configurazione.

Utilizza un file di configurazione Elastic Beanstalk per scaricare ed eseguire il daemon.

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
```

Questi esempi aggiungono anche il file di log del daemon all'attività di Elastic Beanstalk tail dei log di, in modo che sia incluso nella richiesta di log tramite la console o l'interfaccia a riga di comando di Elastic Beanstalk Command Line Interface (EB CLI).

Esecuzione del daemon X-Ray su Amazon EC2

Puoi eseguire il daemon X-Ray sui seguenti sistemi operativi di Amazon EC2:

- Amazon Linux
- Ubuntu

- Windows Server (2012 R2 e versioni successive)

Utilizza un profilo dell'istanza per concedere al daemon le autorizzazioni necessarie per caricare i dati di tracciamento su X-Ray. Per ulteriori informazioni, consultare Dare al demone il permesso di inviare dati a X-Ray.

Utilizza uno script dati utente per eseguire automaticamente il daemon all'avvio dell'istanza.

Example Script di dati utente - 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 Script dati utente - 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>
```

Esecuzione del daemon X-Ray su Amazon ECECS ECS ECS ECS EC

In Amazon ECS, crea un'immagine Docker che esegue il demone X-Ray, caricala in un repository di immagini Docker e quindi distribuiscila nel tuo cluster Amazon ECS. Puoi utilizzare la mappatura delle porte e la modalità di impostazione della rete nel file di definizione dell'attività per consentire alla tua applicazione di comunicare con il container del daemon.

Utilizzo dell'immagine Docker ufficiale

X-Ray fornisce un'immagine del contenitore Docker su Amazon ECR che puoi distribuire insieme alla tua applicazione. Per ulteriori informazioni, consulta il download del demone.

Example Definizione di attività

```
{
  "name": "xray-daemon",
  "image": "amazon/aws-xray-daemon",
  "cpu": 32,
  "memoryReservation": 256,
  "portMappings" : [
      {
          "hostPort": 0,
          "containerPort": 2000,
          "protocol": "udp"
      }
   ]
}
```

Creare e compilare un'immagine Docker

Per la configurazione personalizzata, potrebbe essere necessario definire la propria immagine Docker.

Aggiungi policy gestite al daemon ha l'autorizzazione per caricare i dati non elaborati su X-Ray. Per ulteriori informazioni, consulta Dare al demone il permesso di inviare dati a X-Ray.

Utilizza uno dei seguenti Dockerfiles per creare un'immagine che esegue il daemon.

Example Dockerfile — Amazon Linux Linux Linux Linux Linux Linux Linux 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**
>
> I contrassegni `-t` e `-b` sono necessari per specificare un indirizzo di binding per ascoltare il loopback di un ambiente multi-container.

Example Dockerfile — Ubuntu ECS — Ubuntu ECS

Per le derivazioni di Debian, è inoltre necessario installare i certificati della Certificate Autohority (CA) per evitare problemi durante il download del programma di installazione.

```
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
```

Nella tua definizione di attività, la configurazione dipende dalla modalità di rete utilizzata. La modalità di rete bridge è l'impostazione predefinita e può essere utilizzato nel VPC di default. In una rete bridge, impostate la variabile diAWS_XRAY_DAEMON_ADDRESS ambiente per indicare all'SDK X-Ray a quale porta contenitore fare riferimento e impostate la porta host. In una rete bridge, pubblica la porta UDP 2000 e crea un collegamento dal container della tua applicazione al container del daemon.

Example Definizione di attività

```
    {
      "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"
```

```
      ]
    }
```

Se esegui il cluster nella sottorete privata di un VPC, puoi utilizzare la modalità di rete `awsvpc` per collegare ai tuoi container un'interfaccia di rete elastica (ENI). In questo modo puoi evitare di utilizzare i collegamenti. Ometti la porta dell'host nel mappaggio delle porte, il collegamento e la variabile di ambiente `AWS_XRAY_DAEMON_ADDRESS`.

Example Definizione dell'attività del VPC

```
{
    "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
                }
            ]
        }
    ]
```

```
}
```

Configura le opzioni della riga di comando nella console Amazon ECS

Le opzioni della riga di comando sovrascrivono qualsiasi valore in conflitto nel file di configurazione dell'immagine. Le opzioni della riga di comando sono in genere utilizzate per i test locali, ma possono anche essere usate per comodità durante l'impostazione delle variabili di ambiente o per controllare il processo di avvio.

Aggiungendo le opzioni della riga di comando, aggiorni il Docker `CMD` che viene passato al contenitore. Per ulteriori informazioni, consulta la pagina di riferimento per l'esecuzione del Docker.

Come impostare un'opzione della riga di comando

1. Apri la console Amazon ECS ECS ECS classica console ECS ECS ECS ECS classica https://console.aws.amazon.com/ecs/
2. Dalla barra di navigazione, scegli la regione in cui si trova la definizione di attività.
3. Nel pannello di navigazione, scegli Task Definitions (Definizioni di attività).
4. Nella pagina Task Definitions (Definizioni di attività), seleziona la casella a sinistra della definizione di attività da modificare e scegli Create new revision (Crea nuova revisione).
5. Nella pagina Create new revision of Task Definition (Crea nuova revisione della definizione di attività) selezionare il contenitore.
6. Nella sezione ENVIRONMENT (AMBIENTE) aggiungere l'elenco separato da virgole di opzioni della riga di comando al campo Command (Comando).
7. Scegli Update (Aggiorna).
8. Verifica le informazioni e scegli Create (Crea).

Nell'esempio seguente viene illustrato come scrivere un'opzione della riga di comando separata da virgole per l'opzione `RoleARN`. L'`RoleARN`opzione assume il ruolo IAM specificato per caricare i segmenti su un altro account.

Example

```
--role-arn, arn:aws:iam::123456789012:role/xray-cross-account
```

Per ulteriori informazioni sulle opzioni della riga di comando disponibili in X-Ray, vedere Configurazione delAWS X-Ray demone.

Strumentazione della tua applicazione per AWS X-Ray

La strumentazione dell'applicazione implica l'invio di dati di traccia per le richieste in entrata e in uscita e altri eventi all'interno dell'applicazione, insieme ai metadati relativi a ciascuna richiesta. Esistono diverse opzioni di strumentazione tra cui scegliere o combinare, in base alle proprie esigenze particolari:

- Strumentazione automatica: strumentazione della tua applicazione senza modifiche al codice, in genere tramite modifiche alla configurazione, aggiunta di un agente di strumentazione automatica o altri meccanismi.
- Strumentazione della libreria: apporta modifiche minime al codice dell'applicazione per aggiungere strumentazione predefinita destinata a librerie o framework specifici, come SDK, client Apache HTTP o client SQL. AWS
- Strumentazione manuale: aggiungi il codice di strumentazione all'applicazione in ogni posizione in cui desideri inviare le informazioni di traccia.

Esistono diversi SDK, agenti e strumenti che possono essere utilizzati per strumentare l'applicazione per il tracciamento X-Ray.

Argomenti

- Strumentate la vostra applicazione con Distro per AWS OpenTelemetry
- AWS X-Ray Strumentazione dell'applicazione con gli SDK
- Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray
- Strumentate la vostra applicazione con Go
- Strumentate la vostra applicazione con Java
- Strumentate la vostra applicazione con Node.js
- Strumentate la vostra applicazione con Python
- Strumentate la vostra applicazione con .NET
- Strumentazione della tua applicazione con Ruby

Strumentate la vostra applicazione con Distro per AWS OpenTelemetry

The AWS Distro for OpenTelemetry (ADOT) è una AWS distribuzione basata sul progetto Cloud Native Computing Foundation (CNCF). OpenTelemetry OpenTelemetry fornisce un unico set di API, librerie e agenti open source per raccogliere tracce e metriche distribuite. Questo toolkit è una distribuzione di OpenTelemetry componenti upstream tra cui SDK, agenti di strumentazione automatica e raccoglitori testati, ottimizzati, protetti e supportati da. AWS

Con ADOT, gli ingegneri possono strumentare le loro applicazioni una sola volta e inviare metriche e tracce correlate a più soluzioni di AWS monitoraggio tra cui Amazon CloudWatch AWS X-Ray e Amazon Service. OpenSearch

L'utilizzo di X-Ray con ADOT richiede due componenti: un OpenTelemetry SDK abilitato per l'uso con X-Ray e AWS Distro for OpenTelemetry Collector abilitato per l'uso con X-Ray. Per ulteriori informazioni sull'utilizzo di Distro for OpenTelemetry with AWS X-Ray e altro, AWS consulta Distro for Documentation. Servizi AWSAWS OpenTelemetry

Per ulteriori informazioni sul supporto e l'utilizzo della lingua, vedere AWS Observability on. GitHub

> **Note**
>
> Ora puoi utilizzare l' CloudWatch agente per raccogliere parametri, log e tracce dalle istanze Amazon EC2 e dai server locali. CloudWatch la versione dell'agente 1.300025.0 e successive può raccogliere tracce dagli SDK dei nostri client OpenTelemetryX-Ray e inviarle a X-Ray. L'utilizzo dell' CloudWatch agente al posto del demone AWS Distro for OpenTelemetry (ADOT) Collector o X-Ray per raccogliere le tracce può aiutare a ridurre il numero di agenti da gestire. Per ulteriori informazioni, consultate l'argomento relativo all'CloudWatch agente nella Guida per l'utente CloudWatch .

ADOT include quanto segue:

- AWS Distro for Go OpenTelemetry
- AWS Distro per Java OpenTelemetry
- AWS Distro per OpenTelemetry JavaScript
- AWS Distro per Python OpenTelemetry

- AWS Distro per.NET OpenTelemetry

ADOT attualmente include il supporto per la strumentazione automatica per Java e Python. Inoltre, ADOT consente la strumentazione automatica delle funzioni AWS Lambda e delle relative richieste downstream utilizzando i runtime Java, Node.js e Python, tramite ADOT Managed Lambda Layers.

Gli SDK ADOT per Java e Go supportano le regole di campionamento centralizzato X-Ray. Se hai bisogno di supporto per le regole di campionamento a raggi X in altre lingue, prendi in considerazione l'utilizzo di un SDK. AWS X-Ray

> **Note**
>
> Ora puoi inviare e inviare gli ID di traccia W3C a X-Ray. Per impostazione predefinita, le tracce create con OpenTelemetry hanno un formato ID di traccia basato sulla specifica W3C Trace Context. Questo è diverso dal formato per gli ID di traccia creati utilizzando un SDK X-Ray o dai AWS servizi integrati con X-Ray. Per garantire che gli ID di traccia in formato W3C siano accettati da X-Ray, è necessario utilizzare la versione AWS X-Ray Exporter 0.86.0 o successiva, inclusa in ADOT Collector versione 0.34.0 e successive. Le versioni precedenti dell'esportatore convalidano i timestamp degli ID di traccia, il che potrebbe causare il rifiuto degli ID di traccia W3C.

AWS X-Ray Strumentazione dell'applicazione con gli SDK

AWS X-Ray include una serie di SDK specifici per la lingua per strumentare l'applicazione per inviare tracce a X-Ray. Ogni X-Ray SDK offre quanto segue:

- Moduli di intercettazione da aggiungere al tuo codice per il tracciamento delle richieste HTTP in ingresso
- Da gestori client a client Instrument AWS SDK utilizzati dall'applicazione per chiamare altri Servizi AWS
- Un client HTTP per effettuare chiamate strumentali ad altri servizi Web HTTP interni ed esterni

Gli SDK X-Ray supportano anche le chiamate strumentali ai database SQL, la strumentazione client AWS SDK automatica e altre funzionalità. Invece di inviare i dati di traccia direttamente a X-Ray, l'SDK invia i documenti del segmento JSON a un processo demone che ascolta il traffico UDP. Il demone X-Ray memorizza i segmenti in una coda e li carica su X-Ray in batch.

Sono disponibili i seguenti SDK specifici per ogni lingua:

- AWS X-Ray SDK for Go
- AWS X-Ray SDK per Java
- AWS X-Ray SDK per Node.js
- AWS X-Ray SDK per Python
- AWS X-Ray SDK per.NET
- AWS X-Ray SDK per Ruby

X-Ray attualmente include il supporto per la strumentazione automatica per Java.

Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray

Gli SDK inclusi in X-Ray fanno parte di una soluzione di strumentazione strettamente integrata offerta da. AWS AWS Distro for OpenTelemetry fa parte di una soluzione di settore più ampia in cui X-Ray è solo una delle tante soluzioni di tracciamento. È possibile implementare il end-to-end tracciamento in X-Ray utilizzando entrambi gli approcci, ma è importante comprendere le differenze per determinare l'approccio più utile.

Ti consigliamo di strumentare la tua applicazione con AWS Distro OpenTelemetry se hai bisogno di quanto segue:

- La possibilità di inviare tracce a più backend di tracciamento diversi senza dover ristrumentare il codice
- Support per un gran numero di strumentazioni di libreria per ogni lingua, gestite dalla comunità OpenTelemetry
- Layer Lambda completamente gestiti che racchiudono tutto il necessario per raccogliere dati di telemetria, senza richiedere modifiche al codice quando si utilizza Java, Python o Node.js

> **Note**
>
> AWS Distro for OpenTelemetry offre un'esperienza introduttiva più semplice per la strumentazione delle funzioni Lambda. Tuttavia, a causa della flessibilità OpenTelemetry offerta, la funzione Lambda richiederà memoria aggiuntiva e le chiamate potrebbero subire un aumento della latenza di avvio a freddo, il che può comportare costi aggiuntivi. Se stai ottimizzando per una bassa latenza e non hai bisogno delle funzionalità avanzate

come destinazioni OpenTelemetry di backend configurabili dinamicamente, potresti voler utilizzare l'SDK AWS X-Ray per strumentare la tua applicazione.

Ti consigliamo di scegliere un SDK X-Ray per la strumentazione della tua applicazione se hai bisogno di quanto segue:

- Una soluzione monovendor strettamente integrata
- Integrazione con le regole di campionamento centralizzate X-Ray, inclusa la possibilità di configurare le regole di campionamento dalla console X-Ray e di utilizzarle automaticamente su più host, quando si utilizza Node.js, Python, Ruby o.NET

Strumentate la vostra applicazione con Go

Esistono due modi per strumentare Go l'applicazione per inviare tracce a X-Ray:

- AWS Distro for OpenTelemetry Go: una AWS distribuzione che fornisce un set di librerie open source per l'invio di metriche e tracce correlate a più soluzioni di AWS monitoraggio tra cui Amazon AWS X-Ray e Amazon OpenSearch Service CloudWatch, tramite AWS Distro for Collector. OpenTelemetry
- AWS X-Ray SDK per Go: un set di librerie per la generazione e l'invio di tracce a X-Ray tramite il demone X-Ray.

Per ulteriori informazioni, consulta Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray.

AWSDistro perOpenTelemetryVai

ConAWSDistro perOpenTelemetryVai, puoi strumentare le tue applicazioni una sola volta e inviare metriche e tracce correlate a piùAWSsoluzioni di monitoraggio, tra cui AmazonCloudWatch,AWS X-Raye AmazonOpenSearchServizio. Utilizzo di X-Ray conAWSDistro perOpenTelemetryrichiede due componenti: unOpenTelemetrySDKabilitato per l'uso con X-Ray eAWSDistro perOpenTelemetryCollezionistaabilitato per l'uso con X-Ray.

Per iniziare, consulta ilAWSDistro perOpenTelemetryVai alla documentazione.

Per ulteriori informazioni sull'utilizzo diAWSDistro perOpenTelemetryconAWS X-Raye altroServizi AWS, vediAWSDistro perOpenTelemetryo ilAWSDistro perOpenTelemetryDocumentazione.

Per ulteriori informazioni sul supporto e l'utilizzo delle lingue, vedereAWSOsservabilità suGitHub.

SDK AWS X-Ray per Go

L'SDK X-Ray per Go è un insieme di librerie per applicazioni Go che offre classi e metodi per generare e inviare dati di tracciamento al daemon X-Ray. I dati di tracciamento includono informazioni sulle richieste HTTP in ingresso elaborate dall'applicazione e sulle chiamate che l'applicazione invoca verso i servizi a valle utilizzando l'AWSSDK, client HTTP o connettore di database SQL. Puoi anche possibile creare manualmente dei segmenti e aggiungere informazioni di debug in annotazioni e metadati.

Scarica l'SDK dal repository GitHub con `go get`:

```
$ go get -u github.com/aws/aws-xray-sdk-go/...
```

In caso di applicazioni web, per tracciare le richieste in entrata inizia utilizzando la funzione `xray.Handler`. Il gestore dei messaggi crea un segmento per ogni richiesta tracciata e completa il segmento quando viene inviata la risposta. Fino a che il segmento è aperto puoi usare i metodi del client dell'SDK per aggiungere informazioni al segmento e creare sottosegmenti per tracciare le chiamate a valle. L'SDK, inoltre, registra automaticamente le eccezioni sollevate dall'applicazione per il tempo durante il quale il segmento è aperto.

Per le funzioni Lambda chiamate da un'applicazione o un servizio strumentati, Lambda legge ilintestazione di tracciamentoe traccia automaticamente le richieste campionate. Per altre funzioni, è possibileconfigura Lambdaper campionare e tracciare le richieste in ingresso. In entrambi i casi, Lambda crea il segmento e lo fornisce all'SDK X-Ray.

> **Note**
>
> Su Lambda, l'SDK X-Ray è facoltativo. Se non lo usi nella funzione, la mappa di servizio includerà comunque un nodo per il servizio Lambda e uno per ogni funzione Lambda. Aggiungendo l'SDK, è possibile strumentare il codice funzione per aggiungere sottosegmenti al segmento di funzione registrato da Lambda. Per ulteriori informazioni, consultare AWS Lambda e AWS X-Ray.

Quindi, includi il tuo client all'interno di una chiamata alla funzione `AWS`. Questa fase assicura che X-Ray analizzi le chiamate verso qualsiasi metodo client. Puoi anche analizzare le chiamate verso database SQL.

Dopo aver iniziato a utilizzare l'SDK, personalizza il suo comportamento conconfigurazione del registratore e del middleware. Puoi aggiungere dei plugin per memorizzare i dati sulle risorse di elaborazione sulle quali è eseguita la tua applicazione, personalizzare il comportamento di campionamento definendo regole di campionatura e impostare il livello di log per visualizzare più o meno informazioni generate dall'SDK nei log dell'applicazione.

Registra ulteriori informazioni sulle richieste e sull'attività svolta dalla tua applicazione in annotazioni e metadati. Le annotazioni sono semplici coppie chiave-valore indicizzati per l'uso con espressioni filtro, in modo da poter cercare tracce che contengono dati specifici. Gli elementi metadati sono meno restrittivi e possono memorizzare oggetti completi e vettori, tutto ciò che può essere serializzato in formato JSON.

> **Annotazioni e metadata**
>
> Le annotazioni e i metadati sono testo arbitrario che aggiungi ai segmenti con l'SDK X-Ray. Le annotazioni sono indicizzate per l'uso con espressioni filtro. I metadati non sono indicizzati, ma possono essere visualizzati nel segmento raw con la console o l'API X-Ray. Chiunque conceda l'accesso in lettura a X-Ray può visualizzare questi dati.

Quando disponi di una notevole quantità di client analizzati nel tuo codice, un singolo segmento per la richiesta può contenere un numero elevato di sottosegmenti, uno per ciascuna delle chiamate effettuate con un client analizzato. Puoi organizzare e raggruppare i sottosegmenti inglobando le chiamate del client sottosegmenti personalizzati. Puoi creare un sottosegmento personalizzato per un'intera funzione o qualsiasi porzione di codice, e memorizzare metadati e annotazioni sul sottosegmento invece di scrivere tutto all'interno del segmento padre.

Requisiti

L'SDK X-Ray per Go richiede Go 1.9 o versione più recente.

L'SDK dipende dalle seguenti librerie per la compilazione e il runtime:

- AWSSDK for Go 1.10.0 o versione più recente

Queste dipendenze sono dichiarate nel file `README.md` dell'SDK.

Documentazione di riferimento

Una volta scaricato l'SDK, crea e conserva la documentazione in locale per visualizzarla in un browser Web.

Per visualizzare la documentazione di riferimento

1. Spostarsi nella cartella `$GOPATH/src/github.com/aws/aws-xray-sdk-go` (Linux o Mac) o nella cartella `%GOPATH%\src\github.com\aws\aws-xray-sdk-go` (Windows)
2. Esegui il comando `godoc`.

   ```
   $ godoc -http=:6060
   ```

3. Aprire il browser all'indirizzo `http://localhost:6060/pkg/github.com/aws/aws-xray-sdk-go/`.

Configurazione dell'SDK X-Ray per Go

È possibile specificare la configurazione per X-Ray SDK for Go tramite variabili di ambiente, chiamando `Configure` con un `Config` oggetto o assumendo valori predefiniti. Le variabili di ambiente prevalgono sui valori di `Config`, che hanno la priorità su qualsiasi valore predefinito.

Sections

- Plugin di servizio
- Regole di campionamento
- Registrazione
- Variabili di ambiente
- Utilizzo della configurazione

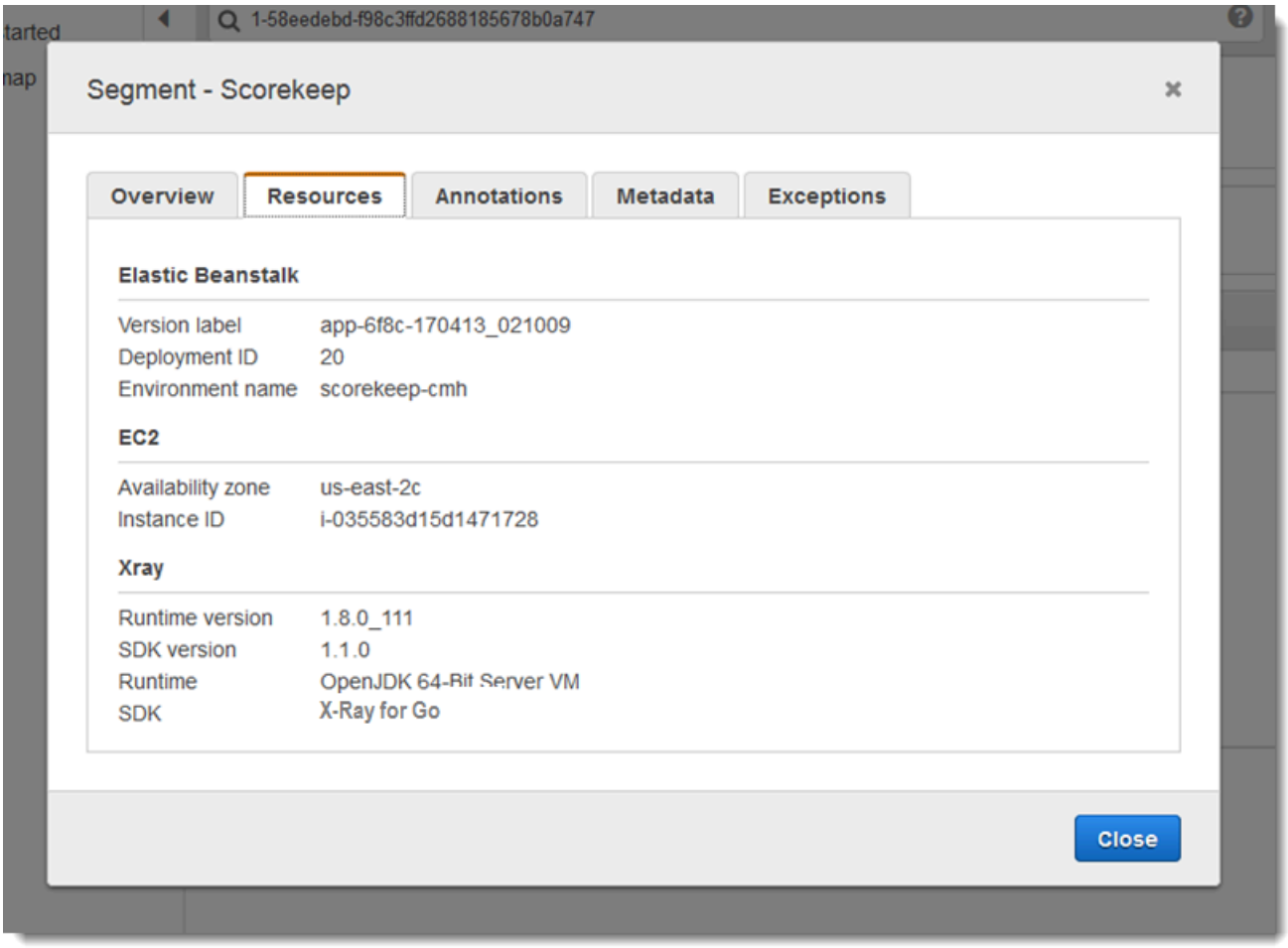
Plugin di servizio

`plugins`Utilizzatelo per registrare informazioni sul servizio che ospita l'applicazione.

Plug-in

- Amazon EC2: `EC2Plugin` aggiunge l'ID dell'istanza, la zona di disponibilità e il gruppo di CloudWatch log.

- Elastic Beanstalk: `ElasticBeanstalkPlugin` aggiunge il nome dell'ambiente, l'etichetta della versione e l'ID di distribuzione.
- Amazon ECS: `ECSPlugin` aggiunge l'ID del contenitore.



Per usare un plugin, importare uno dei seguenti pacchetti.

```
"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"
```

Ogni plugin ha una chiamata di funzione `Init()` esplicita che carica il plugin.

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",
  })
}
```

L'SDK utilizza anche le impostazioni del plug-in per impostare il campo sul segmento. `origin` Indica il tipo di AWS risorsa che esegue l'applicazione. Quando utilizzi più plugin, l'SDK utilizza il seguente ordine di risoluzione per determinare l'origine: ElasticBeanstalk > EKS > ECS > EC2.

Regole di campionamento

L'SDK utilizza le regole di campionamento definite nella console X-Ray per determinare quali richieste registrare. La regola predefinita tiene traccia della prima richiesta ogni secondo e del cinque percento di eventuali richieste aggiuntive su tutti i servizi che inviano tracce a X-Ray. Crea regole aggiuntive nella console X-Ray per personalizzare la quantità di dati registrati per ciascuna delle tue applicazioni.

L'SDK applica le regole personalizzate nell'ordine in cui sono definite. Se una richiesta corrisponde a più regole personalizzate, l'SDK applica solo la prima regola.

> **Note**
>
> Se l'SDK non riesce a contattare X-Ray per ottenere le regole di campionamento, torna a una regola locale predefinita della prima richiesta ogni secondo e del cinque percento di eventuali richieste aggiuntive per host. Ciò può verificarsi se l'host non dispone dell'autorizzazione per chiamare le API di campionamento o non può connettersi al demone X-Ray, che funge da proxy TCP per le chiamate API effettuate dall'SDK.

Puoi anche configurare l'SDK per caricare le regole di campionamento da un documento JSON. L'SDK può utilizzare le regole locali come backup per i casi in cui il campionamento a raggi X non è disponibile o utilizzare esclusivamente regole locali.

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
  }
}
```

Questo esempio definisce una regola personalizzata e una regola predefinita. La regola personalizzata applica una frequenza di campionamento del cinque percento senza alcun numero minimo di richieste da tracciare per i percorsi. `/api/move/` La regola predefinita tiene traccia della prima richiesta ogni secondo e del 10% delle richieste aggiuntive.

Lo svantaggio della definizione locale delle regole è che l'obiettivo fisso viene applicato da ciascuna istanza del registratore in modo indipendente, anziché essere gestito dal servizio X-Ray. Man mano che si installano più host, la tariffa fissa si moltiplica, rendendo più difficile il controllo della quantità di dati registrati.

Sì AWS Lambda, non è possibile modificare la frequenza di campionamento. Se la funzione viene chiamata da un servizio strumentato, le chiamate che hanno generato richieste campionate da quel servizio verranno registrate da Lambda. Se il tracciamento attivo è abilitato e non è presente alcuna intestazione di tracciamento, Lambda prende la decisione di campionamento.

Per fornire le regole di backup, scegli il file JSON per il campionamento locale utilizzando `NewCentralizedStrategyWithFilePath`.

Example main.go — Regola di campionamento locale

```
s, _ := sampling.NewCentralizedStrategyWithFilePath("sampling.json") // path to local
 sampling json
xray.Configure(xray.Config{SamplingStrategy: s})
```

Per usare solo regole locali, seleziona il JSON per il campionamento locale utilizzando `NewLocalizedStrategyFromFilePath`.

Example main.go — Disabilita il campionamento

```
s, _ := sampling.NewLocalizedStrategyFromFilePath("sampling.json") // path to local
 sampling json
xray.Configure(xray.Config{SamplingStrategy: s})
```

Registrazione

> **Note**
>
> I campi `xray.Config{}` `LogLevel` e `LogFormat` sono considerati obsoleti a partire dalla versione 1.0.0-rc.10.

X-Ray utilizza la seguente interfaccia per la registrazione. Il logger predefinito scrive su `stdout` a livello `LogLevelInfo` e superiori.

```
type Logger interface {
 Log(level LogLevel, msg fmt.Stringer)
}

const (
 LogLevelDebug LogLevel = iota + 1
 LogLevelInfo
 LogLevelWarn
 LogLevelError
)
```

Example scrivere su **`io.Writer`**

```
xray.SetLogger(xraylog.NewDefaultLogger(os.Stderr, xraylog.LogLevelError))
```

Variabili di ambiente

È possibile utilizzare le variabili di ambiente per configurare X-Ray SDK for Go. L'SDK supporta le seguenti variabili.

- `AWS_XRAY_CONTEXT_MISSING`— Imposta per `RUNTIME_ERROR` generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.

  Valori validi

  - `RUNTIME_ERROR`— Genera un'eccezione di runtime.
  - `LOG_ERROR`— Registra un errore e continua (impostazione predefinita).
  - `IGNORE_ERROR`— Ignora l'errore e continua.

  Gli errori relativi a segmenti o sottosegmenti mancanti possono verificarsi quando si tenta di utilizzare un client con strumenti nel codice di avvio che viene eseguito quando non è aperta alcuna richiesta o nel codice che genera un nuovo thread.
- `AWS_XRAY_TRACING_NAME`— Imposta il nome del servizio che l'SDK utilizza per i segmenti.
- `AWS_XRAY_DAEMON_ADDRESS`— Imposta l'host e la porta del demone X-Ray. Per impostazione predefinita, l'SDK invia i dati di traccia a. `127.0.0.1:2000` Utilizzate questa variabile se avete configurato il demone per l'ascolto su una porta diversa o se è in esecuzione su un host diverso.

Le variabili di ambiente sostituiscono i valori equivalenti impostati nel codice.

Utilizzo della configurazione

Puoi anche configurare X-Ray SDK for Go utilizzando il metodo. `Configure` `Configure`accetta un argomento, un `Config` oggetto, con i seguenti campi opzionali.

DaemonAddr

Questa stringa specifica l'host e la porta del listener del demone X-Ray. Se non viene specificato, X-Ray utilizza il valore della variabile di `AWS_XRAY_DAEMON_ADDRESS` ambiente. Se il valore non è impostato, utilizza "127.0.0.1:2000".

ServiceVersion

Questa stringa specificata la versione del servizio. Se non viene specificato, X-Ray utilizza la stringa vuota («»).

SamplingStrategy

Questo oggetto `SamplingStrategy` specifica quali delle chiamate della tua applicazione sono tracciate. Se non specificato, X-Ray utilizza a`LocalizedSamplingStrategy`, che utilizza la strategia definita in. `xray/resources/DefaultSamplingRules.json`

StreamingStrategy

Questo **`StreamingStrategy`** oggetto specifica se eseguire lo streaming di un segmento quando RequiresStreamingrestituisce true. Se non specificato, X-Ray utilizza un segnale `DefaultStreamingStrategy` che trasmette un segmento campionato se il numero di sottosegmenti è maggiore di 20.

ExceptionFormattingStrategy

Questo oggetto `ExceptionFormattingStrategy` specifica il modo in cui desideri gestire varie eccezioni. Se non specificato, X-Ray utilizza un tipo `DefaultExceptionFormattingStrategy` con un `XrayError` o`error`, il messaggio di errore e la traccia dello stack.

Strumentazione delle richieste HTTP in ingresso con l'SDK X-Ray per Go

Puoi usare l'SDK X-Ray per tracciare le richieste HTTP in ingresso che l'applicazione elabora su un'istanza EC2 in Amazon EC2,AWS Elastic Beanstalko Amazon ECS.

Utilizza `xray.Handler` per analizzare le richieste HTTP in entrata. SDK X-Ray per Go implementa la libreria Go standard`http.Handler`interfaccia nel`xray.Handler`classe per intercettare le richieste web. La classe `xray.Handler` esegue il wrapping del gestore `http.Handler` fornito con `xray.Capture` utilizzando il contesto della richiesta, analizzando le intestazioni in ingresso, aggiungendo le intestazioni di risposta se necessario e impostando i campi HTTP specifici per il tracciamento.

Quando utilizzi questa classe per gestire le richieste e le risposte HTTP, l'SDK X-Ray per Go crea un segmento per ogni richiesta campionata. Questo segmento include durata, metodo e conclusione della richiesta HTTP. Analisi ulteriori creano sottosegmenti associati a questo segmento.

> **Note**
>
> PerAWS Lambdafunzioni, Lambda crea un segmento per ogni richiesta campionata. Per ulteriori informazioni, consultare AWS Lambda e AWS X-Ray.

L'esempio seguente intercetta le richieste sulla porta 8000 e restituisce «Hello!» come risposta. Crea il segmento myApp e le chiamate di analisi tramite qualsiasi applicazione.

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)
}
```

Ogni segmento ha un nome che identifica l'applicazione nella mappa del servizio. Il segmento può essere denominato staticamente oppure è possibile configurare l'SDK per nominarlo dinamicamente in base all'intestazione host nella richiesta in entrata. La denominazione dinamica consente di raggruppare le tracce in base al nome di dominio nella richiesta e di applicare un nome predefinito se il nome non corrisponde a uno schema previsto (ad esempio, se l'intestazione host è falsificata).

> **Richieste inoltrate**
>
> Se un sistema di bilanciamento del carico o un altro intermediario inoltra una richiesta all'applicazione, X-Ray preleva l'IP client dal`X-Forwarded-For`intestazione nella richiesta anziché dall'IP di origine nel pacchetto IP. L'IP client registrato per una richiesta inoltrata può essere falsificato, quindi non dovrebbe essere attendibile.

Quando una richiesta viene inoltrata, l'SDK imposta un campo aggiuntivo nel segmento per indicarlo. Se il segmento contiene il campo`x_forwarded_for`impostato su`true`, l'IP del client è stato prelevato dal`X-Forwarded-For`intestazione nella richiesta HTTP.

Il gestore crea un segmento per ogni richiesta in entrata con un blocco `http` che contiene le informazioni riportate qui di seguito:

- HTTP method (Metodo HTTP)— GET, POST, PUT, DELETE, ecc.
- Indirizzo client— Indirizzo IP del client che ha inviato la richiesta.
- Codice di risposta— Codice HTTP di risposta per la richiesta completata.

- Timing (Tempo)— Ora di inizio (quando è stata ricevuta la richiesta) e di fine (quando è stata inviata la risposta).
- Agente utente— Il`user-agent`dalla richiesta.
- Lunghezza del contenuto— Il`content-length`della risposta.

Configurazione di una strategia di denominazione dei segmenti

AWS X-Rayusa unnome servizioper identificare l'applicazione e distinguerla dalle altre applicazioni, database, API esterne eAWSrisorse utilizzate dalla tua applicazione. Quando X-Ray SDK genera segmenti per le richieste in arrivo, registra il nome del servizio dell'applicazione nel segmentocampo nome.

L'SDK X-Ray può denominare i segmenti dopo il nome host nell'intestazione della richiesta HTTP. Tuttavia, questa intestazione può essere falsificata, il che potrebbe causare nodi imprevisti nella mappa del servizio. Per impedire all'SDK di assegnare nomi ai segmenti in modo errato a causa di richieste con intestazioni host falsificate, è necessario specificare un nome predefinito per le richieste in arrivo.

Se l'applicazione soddisfa le richieste per più domini, è possibile configurare l'SDK in modo che utilizzi una strategia di denominazione dinamica per rifletterlo nei nomi dei segmenti. Una strategia di denominazione dinamica consente all'SDK di utilizzare il nome host per le richieste che corrispondono a un modello previsto e di applicare il nome predefinito alle richieste che non lo fanno.

Ad esempio, è possibile avere una singola applicazione che elabora le richieste a tre sottodomini:`www.example.com`,`api.example.com`, `estatic.example.com`. È possibile utilizzare una strategia di denominazione dinamica con il modello`*.example.com`per identificare i segmenti per ogni sottodominio con un nome diverso, risultando in tre nodi di servizio sulla mappa del servizio. Se l'applicazione riceve richieste con un nome host che non corrisponde al modello, verrà visualizzato un quarto nodo sulla mappa del servizio con un nome di fallback specificato.

Per utilizzare lo stesso nome per tutti i segmenti della richiesta, specifica il nome della tua applicazione quando crei il gestore, come illustrato nella sezione precedente.

> **Note**
>
> È possibile sovrascrivere il nome di servizio predefinito definito nel codice con il`AWS_XRAY_TRACING_NAME` Variabile di ambiente.

Una strategia di denominazione dinamica definisce un modello al quale devono corrispondere i nomi degli host e un nome di default per l'utilizzo qualora il nome dell'host nella richiesta HTTP non corrisponda al modello. Per denominare segmenti in modo dinamico, utilizza `NewDynamicSegmentNamer` per configurare il nome di default e modello per la corrispondenza.

Example main.go

Se il nome host nella richiesta corrispondente al modello `*.example.com`, utilizza il nome dell'host. In caso contrario, utilizzare `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)
}
```

Tracciamento delle chiamate AWS SDK con X-Ray SDK for Go

Quando l'applicazione effettua chiamate per Servizi AWS archiviare dati, scrivere in una coda o inviare notifiche, X-Ray SDK for Go tiene traccia delle chiamate downstream in sottosegmenti. Le risorse tracciate Servizi AWS e a cui accedi all'interno di tali servizi (ad esempio, un bucket Amazon S3 o una coda Amazon SQS) vengono visualizzate come nodi downstream sulla mappa di traccia nella console X-Ray.

Per tracciare i client AWS SDK, ingloba l'oggetto client nella chiamata `xray.AWS()` come mostrato nel seguente esempio.

Example main.go

```
var dynamo *dynamodb.DynamoDB
func main() {
  dynamo = dynamodb.New(session.Must(session.NewSession()))
  xray.AWS(dynamo.Client)
}
```

Quindi, quando utilizzi il client SDK AWS, utilizza la versione `withContext` del metodo chiamato e passa ad essa il `context` ricavato dall'oggetto `http.Request` passato al gestore.

Example main.go — chiamata SDK AWS

```
func listTablesWithContext(ctx context.Context) {
  output := dynamo.ListTablesWithContext(ctx, &dynamodb.ListTablesInput{})
  doSomething(output)
}
```

Per tutti i servizi, puoi vedere il nome dell'API richiamata nella console X-Ray. Per un sottoinsieme di servizi, l'SDK X-Ray aggiunge informazioni al segmento per fornire una maggiore granularità nella mappa dei servizi.

Ad esempio, quando si effettua una chiamata con un client DynamoDB con strumentazione, l'SDK aggiunge il nome della tabella al segmento per le chiamate destinate a una tabella. Nella console, ogni tabella appare come un nodo separato nella mappa dei servizi, con un nodo DynamoDB generico per le chiamate che non hanno come destinazione una tabella.

Example Sottosegmento per una chiamata a DynamoDB per salvare un elemento

```
{
  "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",
  }
}
```

Quando si accede alle risorse con nome, le chiamate ai seguenti servizi creano ulteriori nodi della mappa del servizio. Le chiamate che non sono hanno come obiettivo risorse specifiche creano un nodo generico per il servizio.

- Amazon DynamoDB: nome della tabella

- Amazon Simple Storage Service: nome del bucket e della chiave
- Amazon Simple Queue Service: nome della coda

Tracciamento delle chiamate verso Web Services HTTP a valle con l'SDK X-Ray per Go

Quando la tua applicazione esegue chiamate verso microservizi o API HTTP pubbliche, puoi utilizzare `xray.Client` per analizzare tali chiamate come sottosegmenti della tua applicazione Go, come mostrato nell'esempio seguente, dove http-client è un client HTTP.

Il client crea una copia shallow del client HTTP fornito, passando per default a`http.DefaultClient`, con round tripper avvolto`xray.RoundTripper`.

Example

<caption>main.go — HTTP client</caption>

```
myClient := xray.Client(http-client)
```

<caption>main.go — Traccia la chiamata HTTP a valle con la libreria ctxhttp</caption>

L'esempio seguente strumento la chiamata HTTP in uscita con la libreria ctxhttp utilizzando`xray.Client.ctx`può essere passato dalla chiamata a monte. In questo modo viene utilizzato il contesto del segmento esistente. Ad esempio, X-Ray non consente la creazione di un nuovo segmento all'interno di una funzione Lambda, pertanto è necessario utilizzare il contesto del segmento Lambda esistente.

```
resp, err := ctxhttp.Get(ctx, xray.Client(nil), url)
```

Tracciamento delle query SQL con l'SDK X-Ray per Go

Per tracciare le chiamate SQL verso PostgreSQL o MySQL, sostituisci le chiamate a `sql.Open` con chiamate a `xray.SQLContext`, come nell'esempio seguente. Se possibile, utilizza le URL al posto delle stringhe di configurazione.

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
```

```
}
```

Generazione di sottosegmenti personalizzati con l'SDK X-Ray per Go

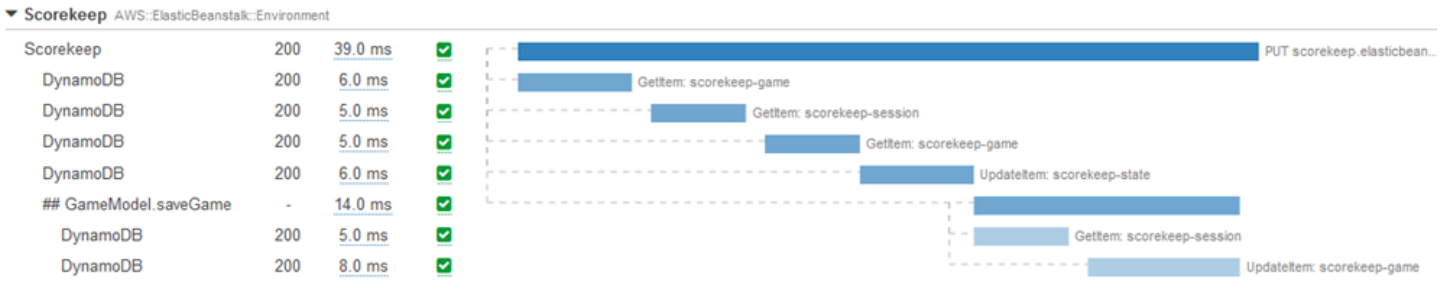
I sottosegmenti estendono una tracciasegmentocon dettagli sul lavoro svolto per soddisfare una richiesta. Ogni volta che si effettua una chiamata con un client strumentato, l'SDK X-Ray registra le informazioni generate in un sottosegmento. È possibile creare sottosegmenti aggiuntivi per raggruppare altri segmenti secondari, per misurare le prestazioni di una sezione di codice o per registrare annotazioni e metadati.

Utilizza il metodo `Capture` per creare un sottosegmento attinente a una funzione.

Example main.go — Sottosegmento personalizzato

```
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)
  })
```

La seguente schermata mostra un esempio di come il sottosegmento `saveGame` può essere visualizzato nei tracciamenti per l'applicazione `Scorekeep`.



Aggiungi annotazioni e metadati ai segmenti con X-Ray SDK for Go

Puoi utilizzare annotazioni e metadati per registrare informazioni aggiuntive sulle richieste, sull'ambiente o sull'applicazione. È possibile aggiungere annotazioni e metadati ai segmenti creati da X-Ray SDK o ai sottosegmenti personalizzati creati dall'utente.

Le annotazioni sono coppie chiave-valore con stringhe, numeri o valori booleani. Le annotazioni sono indicizzate per essere utilizzate con le espressioni di filtro. Utilizzale per registrare i dati che desideri utilizzare per raggruppare le tracce nella console oppure per chiamare l'API `GetTraceSummaries`.

I metadati sono coppie chiave-valore che possono avere valori di qualsiasi tipo, inclusi oggetti ed elenchi, ma non sono indicizzati per essere utilizzati con le espressioni di filtro. Utilizzate i metadati per registrare dati aggiuntivi che desiderate archiviare nella traccia ma che non è necessario utilizzare con la ricerca.

Oltre ad annotazioni e metadati, sui segmenti puoi anche registrare le stringhe degli ID utente. Gli ID utente vengono memorizzati in un campo separato su segmenti e sono indicizzati per l'uso nelle ricerche.

Sections

- Registrazione delle annotazioni con X-Ray SDK for Go
- Registrazione di metadati con X-Ray SDK for Go
- Registrazione degli ID utente con X-Ray SDK for Go

Registrazione delle annotazioni con X-Ray SDK for Go

Utilizza le annotazioni per memorizzare le informazioni sui segmenti che desideri siano indicizzate per la ricerca.

Requisiti per le annotazioni

- Chiavi: la chiave per un'annotazione a raggi X può contenere fino a 500 caratteri alfanumerici. Non è possibile utilizzare spazi o simboli diversi dal simbolo di sottolineatura (_).
- Valori: il valore di un'annotazione X-Ray può contenere fino a 1.000 caratteri Unicode.
- Il numero di annotazioni: è possibile utilizzare fino a 50 annotazioni per traccia.

Per registrare le annotazioni, invoca `AddAnnotation` con una stringa che contiene i metadati che desideri associare al segmento.

```
xray.AddAnnotation(key string, value interface{})
```

L'SDK memorizza le annotazioni come coppie chiave-valore in un oggetto `annotations` all'interno del documento di segmento. Chiamando `AddAnnotation` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento viene sovrascritto.

Per trovare tracciamenti con annotazioni contenenti valori specifici, utilizza la parola chiave `annotations.key` in un'espressione filtro.

Registrazione di metadati con X-Ray SDK for Go

Utilizza i metadati per memorizzare le informazioni su segmenti che non è necessario che siano indicizzate per la ricerca.

Per registrare i metadati, invoca `AddMetadata` con una stringa che contiene i metadati che desideri associare al segmento.

```
xray.AddMetadata(key string, value interface{})
```

Registrazione degli ID utente con X-Ray SDK for Go

Memorizza gli ID utente sui segmenti di richiesta per identificare l'utente che ha inviato la richiesta.

Per registrare gli ID degli utenti

1. Ottenere un riferimento al segmento corrente da `AWSXRay`.

   ```
   import (
     "context"
     "github.com/aws/aws-xray-sdk-go/xray"
   )

   mySegment := xray.GetSegment(context)
   ```

2. Chiamare `setUser` con una stringa che rappresenta l'ID dell'utente che ha inviato la richiesta.

   ```
   mySegment.User = "U12345"
   ```

Per trovare tracciamenti associati ad un ID utente, utilizza la parola chiave `user` in un'espressione filtro.

Strumentate la vostra applicazione con Java

Esistono due modi per strumentare Java l'applicazione per inviare tracce a X-Ray:

- AWS Distro for OpenTelemetry Java: una AWS distribuzione che fornisce un set di librerie open source per l'invio di metriche e tracce correlate a più soluzioni di AWS monitoraggio, tra cui Amazon AWS X-Ray e Amazon OpenSearch Service CloudWatch, tramite AWS Distro for Collector. OpenTelemetry
- AWS X-Ray SDK per Java: un set di librerie per la generazione e l'invio di tracce a X-Ray tramite il demone X-Ray.

Per ulteriori informazioni, consulta Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray.

AWSDistro perOpenTelemetryGiava

ConAWSDistro perOpenTelemetry(ADOT) Java, puoi strumentare le tue applicazioni una sola volta e inviare metriche e tracce correlate a piùAWSsoluzioni di monitoraggio, tra cui AmazonCloudWatch,AWS X-Raye AmazonOpenSearchServizio. L'utilizzo di X-Ray con ADOT richiede due componenti: unOpenTelemetrySDKabilitato per l'uso con X-Ray eAWSDistro perOpenTelemetryCollezionistaabilitato per l'uso con X-Ray. ADOT Java include il supporto per la strumentazione automatica, che consente all'applicazione di inviare tracce senza modifiche al codice.

Per iniziare, consulta ilAWSDistro perOpenTelemetryDocumentazione Java.

Per ulteriori informazioni sull'utilizzo diAWSDistro perOpenTelemetryconAWS X-Raye altroServizi AWS, vediAWSDistro perOpenTelemetryo ilAWSDistro perOpenTelemetryDocumentazione.

Per ulteriori informazioni sul supporto e l'utilizzo delle lingue, vedereAWSOsservabilità suGitHub.

AWS X-Ray SDK per Java

L'X-Ray SDK for Java è un insieme di librerie per applicazioni Java Web che forniscono classi e metodi per generare e inviare dati di traccia al demone X-Ray. I dati di traccia includono informazioni sulle richieste HTTP in entrata fornite dall'applicazione e sulle chiamate effettuate dall'applicazione ai servizi downstream utilizzando l' AWS SDK, i client HTTP o un connettore di database SQL. Puoi anche possibile creare manualmente dei segmenti e aggiungere informazioni di debug in annotazioni e metadati.

X-Ray SDK for Java è un progetto open source. Puoi seguire il progetto e inviare problemi e richieste di pull su GitHub: github.com/aws/ aws-xray-sdk-java

Inizia aggiungendo `AWSXRayServletFilter` come filtro del servlet per tracciare le richieste in entrata. Un filtro servlet crea un segmento. Fino a che il segmento è aperto, puoi usare i metodi

del client dell'SDK per aggiungere informazioni al segmento e creare sottosegmenti per tracciare le chiamate a valle. L'SDK, inoltre, registra automaticamente le eccezioni sollevate dall'applicazione per il tempo durante il quale il segmento è aperto.

A partire dalla versione 1.3, è possibile analizzare l'applicazione utilizzando il sistema aspect-oriented programming (AOP) in Spring. Ciò significa che puoi strumentalizzare la tua applicazione, mentre è in esecuzione AWS, senza aggiungere alcun codice al runtime dell'applicazione.

Successivamente, utilizzate l'SDK X-Ray per Java per strumentare AWS SDK for Java i vostri client includendo il sottomodulo SDK Instrumentor nella configurazione di build. Ogni volta che effettui una chiamata a un downstream Servizio AWS o a una risorsa con un client strumentato, l'SDK registra le informazioni sulla chiamata in un sottosegmento. Servizi AWS e le risorse a cui accedi all'interno dei servizi vengono visualizzate come nodi a valle sulla mappa di traccia per aiutarti a identificare errori e problemi di limitazione sulle singole connessioni.

Se non vuoi strumentalizzare tutte le chiamate downstream Servizi AWS, puoi tralasciare il sottomodulo Instrumentor e scegliere quali client strumentare. Strumenta i singoli client aggiungendo un client di servizio `TracingHandler` SDK. AWS

Altri sottomoduli X-Ray SDK for Java forniscono la strumentazione per le chiamate downstream alle API Web HTTP e ai database SQL. È possibile utilizzare le versioni X-Ray SDK for Java di `HTTPClient` e `HTTPClientBuilder` nel sottomodulo HTTP Apache per strumentare i client HTTP Apache. Per analizzare le query SQL, aggiungi lo strumento di intercettazione dell'SDK all'origine dei dati.

Dopo aver iniziato a utilizzare l'SDK, personalizzane il comportamento configurando il registratore e il filtro servlet. Puoi aggiungere dei plugin per memorizzare i dati sulle risorse di elaborazione sulle quali è eseguita la tua applicazione, personalizzare il comportamento di campionamento definendo regole di campionatura e impostare il livello di log per visualizzare più o meno informazioni generate dall'SDK nei log dell'applicazione.

Registra ulteriori informazioni sulle richieste e sull'attività svolta dalla tua applicazione in annotazioni e metadati. Le annotazioni sono semplici coppie chiave-valore indicizzati per l'uso con espressioni filtro, in modo da poter cercare tracce che contengono dati specifici. Le immissioni di metadati sono meno restrittive e possono registrare interi oggetti e array, ovvero tutto ciò che può essere serializzato in JSON.

> **Annotazioni e metadata**
>
> Le annotazioni e i metadati sono testo arbitrario che aggiungi ai segmenti con X-Ray SDK. Le annotazioni vengono indicizzate per essere utilizzate con le espressioni di filtro. I metadati non sono indicizzati, ma possono essere visualizzati nel segmento non elaborato con la console X-Ray o l'API. Chiunque conceda l'accesso in lettura a X-Ray può visualizzare questi dati.

Quando disponi di una notevole quantità di client analizzati nel tuo codice, un singolo segmento per la richiesta può contenere molti sottosegmenti, uno per ciascuna delle chiamate effettuate con un client analizzato. Puoi organizzare e raggruppare i sottosegmenti inglobando le chiamate del client sottosegmenti personalizzati. Puoi creare un sottosegmento personalizzato per un'intera funzione o qualsiasi porzione di codice, e memorizzare metadati e annotazioni sul sottosegmento invece di scrivere tutto all'interno del segmento padre.

Sottomoduli

Puoi scaricare l'X-Ray SDK per Java da Maven. L'X-Ray SDK for Java è suddiviso in sottomoduli per caso d'uso, con una distinta dei materiali per la gestione delle versioni:

- `aws-xray-recorder-sdk-core`(obbligatorio): funzionalità di base per la creazione e la trasmissione di segmenti. Include `AWSXRayServletFilter` per l'analisi delle richieste in entrata.
- `aws-xray-recorder-sdk-aws-sdk`— Instruments: chiamate da Servizi AWS effettuare con AWS SDK for Java i clienti aggiungendo un client di tracciamento come gestore delle richieste.
- `aws-xray-recorder-sdk-aws-sdk-v2`— Chiamate Instruments Servizi AWS effettuate con client AWS SDK for Java 2.2 e versioni successive aggiungendo un client di tracciamento come intercettore delle richieste.
- `aws-xray-recorder-sdk-aws-sdk-instrumentor`— Con`aws-xray-recorder-sdk-aws-sdk`, Instruments tutti i client vengono automaticamente. AWS SDK for Java
- `aws-xray-recorder-sdk-aws-sdk-v2-instrumentor`— Con`aws-xray-recorder-sdk-aws-sdk-v2`, instruments tutti i client AWS SDK for Java 2.2 e versioni successive automaticamente.
- `aws-xray-recorder-sdk-apache-http`— Chiamate HTTP in uscita di Instruments effettuate con client HTTP Apache.

- `aws-xray-recorder-sdk-spring`— Fornisce intercettori per le applicazioni Spring AOP Framework.
- `aws-xray-recorder-sdk-sql-postgres`— Instruments: chiamate in uscita a un database PostgreSQL realizzate con JDBC.
- `aws-xray-recorder-sdk-sql-mysql`— Chiamate in uscita di Instruments a un database MySQL realizzate con JDBC.
- `aws-xray-recorder-sdk-bom`— Fornisce una distinta dei materiali che è possibile utilizzare per specificare la versione da utilizzare per tutti i sottomoduli.
- `aws-xray-recorder-sdk-metrics`— Pubblica CloudWatch metriche Amazon non campionate dai segmenti X-Ray raccolti.

Se usi Maven o Gradle per creare la tua applicazione, aggiungi l'X-Ray SDK for Java alla configurazione di build.

Per la documentazione di riferimento sulle classi e sui metodi dell'SDK, consulta SDK for API Reference.AWS X-RayJava

Requisiti

L'X-Ray SDK per Java richiede Java 8 o versioni successive, Servlet API 3, SDK e Jackson. AWS

L'SDK dipende dalle seguenti librerie per la compilazione e il runtime:

- AWS SDK per la versione 1.11.398 o successiva Java
- API Servlet 3.1.0

Queste dipendenze sono dichiarate nel file `pom.xml` dell'SDK e vengono automaticamente incluse se compili utilizzando Maven o Gradle.

Se si utilizza una libreria inclusa in X-Ray SDK for Java, è necessario utilizzare la versione inclusa. Ad esempio, se fai già affidamento su Jackson in fase di runtime e includi dei file JAR nella distribuzione per tale dipendenza, devi rimuovere i file JAR perché il JAR dell'SDK include le proprie versioni delle librerie Jackson.

Gestione delle dipendenze

L'X-Ray SDK per Java è disponibile presso Maven:

- Gruppo — `com.amazonaws`
- Artefatto — `aws-xray-recorder-sdk-bom`
- Version (Versione): `2.11.0`

Se utilizzi Maven per compilare l'applicazione, aggiungi l'SDK come dipendenza nel tuo file `pom.xml`.

Example pom.xml - dipendenze

```
<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>
```

Nel caso di Gradle, aggiungi l'SDK come dipendenza in fase di compilazione nel tuo file `build.gradle`.

Example build.gradle - dipendenze

```
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')
    }
}
```

Se utilizzi Elastic Beanstalk per distribuire la tua applicazione, puoi usare Maven o Gradle per creare su istanza ogni volta che esegui la distribuzione, invece di creare e caricare un archivio di grandi dimensioni che include tutte le tue dipendenze. Vedi l'applicazione di esempio per un esempio che utilizza Gradle.

AWS X-Rayagente di auto-strumentazione per Java

LaAWS X-Rayauto-instrumentation agent for Java è una soluzione di tracciamento che strumenta le tue applicazioni Web Java con il minimo sforzo di sviluppo. L'agente consente di tracciare le applicazioni basate su servlet e tutte le richieste a valle dell'agente effettuate con framework e librerie supportati. Ciò include le richieste HTTP di Apache downstream,AWSRichieste SDK e query SQL eseguite utilizzando un driver JDBC. L'agente propaga il contesto a X-Ray, inclusi tutti i segmenti e i sottosegmenti attivi, tra i thread. Tutte le configurazioni e la versatilità dell'SDK X-Ray sono ancora

disponibili con l'agente Java. Sono stati scelti i valori predefiniti adeguati per garantire che l'agente funzioni con il minimo sforzo.

La soluzione agente X-Ray è più adatta per server applicazioni Web Java basati su servlet e richiesta-risposta. Se l'applicazione utilizza un framework asincrono o non è ben modellato come servizio richiesta-risposta, è consigliabile considerare invece la strumentazione manuale con l'SDK.

L'agente X-Ray è creato utilizzando il toolkit di Comprehension Distributed Systems o DisCo. Disco è un framework open source per la creazione di agenti Java che possono essere utilizzati nei sistemi distribuiti. Anche se non è necessario comprendere DisCo per utilizzare l'agente X-Ray, è possibile saperne di più sul progetto visitandonehomepage di su GitHub. Anche l'agente a X-Ray è completamente open source. Per visualizzare il codice sorgente, apportare contributi o sollevare problemi relativi all'agente, visitarlorepository di in GitHub.

Applicazione di esempio

Laeb-java-scorekeepl'applicazione del campione è adattata per essere strumentata con l'agente a X-Ray X. Questo ramo non contiene filtri servlet o configurazione del registratore, poiché queste funzioni vengono eseguite dall'agente. Per eseguire l'applicazione localmente o utilizzandoAWSReadme dell'applicazione di esempio, segui le fasi descritte nel file readme dell'applicazione di esempio. Le istruzioni per l'utilizzo dell'app di esempio per generare tracce a X-Ray sono contenute nellatutorial dell'app di esempio.

Nozioni di base

Per iniziare a utilizzare l'agente Java di auto-strumentazione X-Ray nella tua applicazione, segui questi passaggi.

1. Esegui il daemon X-Ray nel tuo ambiente. Per ulteriori informazioni, consultaDaemon X-Ray.
2. Download diultima distribuzione dell'agente. Decomprimi l'archivio e annota la sua posizione nel file system. Il suo contenuto sarà simile al seguente:

```
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. Modificare gli argomenti JVM dell'applicazione per includere quanto segue, che abilita l'agente. Garantire che`-javaagent`viene inserito l'argomentoprimalo`-jar`argomento se applicabile. Il processo di modifica degli argomenti JVM varia a seconda degli strumenti e dei framework utilizzati per avviare il server Java. Consulta la documentazione del tuo framework server per una guida specifica.

```
-javaagent:/<path-to-disco>/disco-java-agent.jar=pluginPath=/<path-to-disco>/disco-plugins
```

4. Per specificare come viene visualizzato il nome dell'applicazione sulla console X-Ray, impostare il`AWS_XRAY_TRACING_NAME`variabile di ambiente o`com.amazonaws.xray.strategy.tracingName`proprietà di sistema. Se non viene fornito alcun nome, viene utilizzato un nome predefinito.
5. Riavvia il server o il contenitore. Le richieste in arrivo e le chiamate a valle sono ora tracciate. Se non vedi i risultati attesi, consultathe section called “Risoluzione dei problemi”.

Configurazione

L'agente X-Ray è configurato da un file JSON esterno fornito dall'utente. Per impostazione predefinita, questo file si trova alla radice del classpath dell'utente (ad esempio, nel loro`resources`directory) denominata`xray-agent.json`. È possibile configurare una posizione personalizzata per il file di configurazione impostando`com.amazonaws.xray.configFile`proprietà system per il percorso assoluto del file system del file di configurazione.

In seguito viene visualizzato un esempio di file di configurazione.

```
{
    "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
}
```

Specifiche di configurazione

Nella tabella seguente vengono descritti i valori validi per ogni proprietà. I nomi delle proprietà sono sensibili alle maiuscole e minuscole, ma le relative chiavi Per le proprietà che possono essere sovrascritte dalle variabili di ambiente e dalle proprietà di sistema, l'ordine di priorità è sempre variabile d'ambiente, quindi proprietà di sistema e quindi file di configurazione. Consulta laVariabili di ambienteper informazioni sulle proprietà che è possibile ignorare. Tutti i campi sono facoltativi.

Nome proprietà	Tipo	Valori validi	Descrizione	Variabile di ambiente	Proprietà di sistema	Default (predefinito)
serviceName	Stringa	Qualsiasi stringa	Il nome del servizio strumentato verrà visualizzato nella console X-Ray.	AWS_XRAY_TRACING_NAME	com.amazonaws.xray.strategy.tracingName	Servizio strumentato Xray
Strategia mancante del contesto	Stringa	LOG_ERROR, IGNORE_ERROR	L'azione intrapresa dall'agente quando tenta di utilizzare il contesto del segmento dei X-Ray, ma nessuna è presente.	AWS_XRAY_CONTEXT_MISSING	com.amazonaws.xray.strategy.contestoStrategiamancante	LOG_ERROR

Nome proprietà	Tipo	Valori validi	Descrizione	Variabile di ambiente	Proprietà di sistema	Default (predefinito)
Indirizzo Daemon	Stringa	Indirizzo IP e porta formattati o elenco di indirizzi TCP e UDP	L'indirizzo utilizzato dall'agente per comunicare con il daemon X-Ray.	AWS_XRAY_DAEMON_ADDRESS	com.amazonaws.xray.emitter.daemon Indirizzo	127.0.0.1:2000
TracingEnabled	Booleano	true, false	Attiva la strumentazione da parte dell'agente radiografico.	AWS_XRAY_TRACING_ENABLED	com.amazonaws.xray.tracing abilitato	TRUE

Nome proprietà	Tipo	Valori validi	Descrizione	Variabile di ambiente	Proprietà di sistema	Default (predefinito)
SamplingStrategy	Stringa	CENTRALE, LOCALE, NESSUNO, TUTTO	La strategia di campionamento utilizzata dall'agente. ALL acquisisce tutte le richieste, NESSUNA cattura nessuna richiesta. Consulta .reg di campionamento.	N/A	N/D	CENTRALE
Prefisso di iniezione TraceID	Stringa	Qualsiasi stringa	Include il prefisso fornito prima di iniettare gli ID di traccia nei registri.	N/A	N/D	Nessuna (stringa vuota)

Nome proprietà	Tipo	Valori validi	Descrizione	Variabile di ambiente	Proprietà di sistema	Default (predefinito)
Manifest delle regole di campionamento	Stringa	Percorso file assoluto	Il percorso di un file di regole di campionamento personalizzato da utilizzare come fonte di regole di campionamento per la strategia di campionamento locale o le regole di fallback per la strategia centrale.	N/A	N/D	Regules.json predefinito di samplingrules.json

Nome proprietà	Tipo	Valori validi	Descrizione	Variabile di ambiente	Proprietà di sistema	Default (predefinito)
Manifest AWS Service Handler	Stringa	Percorso file assoluto	Il percorso di un parametro personalizzato consente l'elenco, che acquisisce informazioni aggiuntive daAWSClient SDK.	N/A	N/D	Whitelist.json del parametro di funzionamento predefinito
awsSdkVersion	Numero intero	1, 2	Versione dellaAWSSDK per Javastai usando. Ignorare seawsServiceHandlerManifest non è impostato.	N/A	N/D	2

Nome proprietà	Tipo	Valori validi	Descrizione	Variabile di ambiente	Proprietà di sistema	Default (predefinito)
Lunghezza di traccia Max Stack	Numero intero	Numeri interi non negativi	Le righe massime di una traccia di stack da registrare in una traccia.	N/A	N/D	50
Soglia di streaming	Numero intero	Numeri interi non negativi	Dopo che almeno questi molti sottosegmenti sono stati chiusi, vengono trasmessi in streaming al demone fuori banda per evitare che i blocchi siano troppo grandi.	N/A	N/D	100

Nome proprietà	Tipo	Valori validi	Descrizione	Variabile di ambiente	Proprietà di sistema	Default (predefinito)
Iniezione TraceID	Booleano	true, false	Consente l'iniezione di ID di traccia X-Ray nei registri se le dipendenze e la configurazione descritte inregistrazione della configurazionevengono aggiunte anche. Altrimenti, non fa niente.	N/A	N/D	TRUE
Plugin abilitati	Booleano	true, false	Attiva plugin che registrano i metadati relativiA WSambienti in cui stai operando. Consulta .plu	N/A	N/D	TRUE

Nome proprietà	Tipo	Valori validi	Descrizione	Variabile di ambiente	Proprietà di sistema	Default (predefinito)
collectSqlQueries	Booleano	true, false	Registra le stringhe di query SQL nei sottosegmenti SQL in base al miglior tentativo.	N/A	N/D	FALSE
Propagazione del contesto	Booleano	true, false	Propaga automaticamente il contesto a X-Ray tra thread se true. In caso contrario ,utilizza Thread Local per memorizzare il contesto e la propagazione manuale tra i thread è necessaria.	N/A	N/D	TRUE

Registrazione della configurazione

Il livello di registro dell'agente X-Ray può essere configurato allo stesso modo dell'SDK X-Ray per Java. Consulta .Registrazioneper ulteriori informazioni sulla configurazione della registrazione con l'SDK X-Ray per Java.

Strumentazione manuale

Se desideri eseguire la strumentazione manuale oltre alla strumentazione automatica dell'agente, aggiungi l'SDK X-Ray come dipendenza dal tuo progetto. Si noti che i filtri servlet personalizzati dell'SDK menzionati inTracciamento delle richieste in entratanon sono compatibili con l'agente X-Ray.

> **Note**
>
> È necessario utilizzare la versione più recente di X-Ray SDK per eseguire la strumentazione manuale mentre si utilizza anche l'agente.

Se stai lavorando a un progetto Maven, aggiungi le seguenti dipendenze al tuo`pom.xml`file.

```
<dependencies>
  <dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-core</artifactId>
    <version>2.11.0</version>
  </dependency>
  </dependencies>
```

Se stai lavorando a un progetto Gradle, aggiungi le seguenti dipendenze al tuo`build.gradle`file.

```
implementation 'com.amazonaws:aws-xray-recorder-sdk-core:2.11.0'
```

È possibile aggiungereSottosegmenti personalizzatiin aggiunta aannotazioni, metadati e ID utentedurante l'utilizzo dell'agente, proprio come faresti con il normale SDK. L'agente propaga automaticamente il contesto tra i thread, quindi non dovrebbero essere necessarie soluzioni alternative per la propagazione del contesto quando si lavora con applicazioni multithread.

Risoluzione dei problemi

Poiché l'agente offre una strumentazione completamente automatica, può essere difficile identificare la causa principale di un problema quando si verificano problemi. Se l'agente a X-Ray non funziona

come previsto, esaminare i seguenti problemi e soluzioni. L'agente X-Ray e l'SDK utilizzano Jakarta Commons Logging (JCL). Per visualizzare l'output di registrazione, assicurarsi che un bridge che collega JCL al back-end di registrazione si trovi sul classpath, come nell'esempio seguente:`log4j-jclojcl-over-slf4j`.

Problema Ho abilitato l'agente Java sulla mia applicazione ma non vedo nulla sulla console X-Ray

Il daemon X-Ray è in esecuzione sulla stessa macchina?

In caso contrario, vedere ilDocumentazione daemon X-Rayper configurarlo.

Nei registri delle applicazioni, viene visualizzato un messaggio come «Inizializzazione del registratore agente X-Ray»?

Se hai aggiunto correttamente l'agente all'applicazione, questo messaggio viene registrato a livello INFO all'avvio dell'applicazione, prima che inizi a ricevere le richieste. Se questo messaggio non è presente, l'agente Java non è in esecuzione con il processo Java. Assicurati di aver seguito tutti i passaggi di configurazione correttamente senza errori di battitura.

Nei registri delle applicazioni, vedi diversi messaggi di errore che dicono qualcosa come «SoppressioneAWS X-Raycontesto mancante eccezione»?

Questi errori si verificano perché l'agente sta cercando di strumentare le richieste a valle, comeAWSRichieste SDK o query SQL, ma l'agente non è stato in grado di creare automaticamente un segmento. Se riscontri molti di questi errori, l'agente potrebbe non essere lo strumento migliore per il tuo caso d'uso e potresti invece prendere in considerazione la strumentazione manuale con X-Ray SDK. In alternativa, è possibile abilitare l'SDK X-Raydebug dei logper vedere la traccia dello stack di dove si verificano le eccezioni mancanti nel contesto. È possibile avvolgere queste parti del codice con segmenti personalizzati, che dovrebbero risolvere questi errori. Per un esempio di avvolgimento delle richieste a valle con segmenti personalizzati, consulta il codice di esempio instrumenting di codice di avvio.

Problema Alcuni dei segmenti che mi aspetto non appaiono sulla console X-Ray

La tua applicazione utilizza il multithreading?

Se alcuni segmenti che ci si aspetta di essere creati non appaiono nella console, potrebbe essere la causa dei thread in background dell'applicazione. Se l'applicazione esegue attività utilizzando thread in background che sono «fuoco e dimentica», come effettuare una chiamata una tantum a una funzione Lambda conAWSSDK, o polling periodicamente di un endpoint HTTP, che potrebbe confondere l'agente durante la propagazione del contesto tra thread. Per verificare che questo sia il

tuo problema, abilita i registri di debug di X-Ray SDK e verifica la presenza di messaggi come: Non emettere segmenti denominati <NAME >come sottosegmenti in corso. Per aggirare questo problema, puoi provare a unire i thread in background prima che il server ritorni per assicurarti che tutto il lavoro svolto in essi sia registrato. Oppure, puoi impostare l'agente`contextPropagation`configurazione per`false`per disabilitare la propagazione del contesto nei thread in background. Se lo fai, dovrai strumentarli manualmente con segmenti personalizzati o ignorare le eccezioni mancanti del contesto che producono.

Hai configurato le regole di campionamento?

Se sulla console X-Ray appaiono segmenti apparentemente casuali o imprevisti o se i segmenti che si aspettano di trovarsi sulla console non lo sono, potresti riscontrare un problema di campionamento. L'agente X-Ray applica il campionamento centralizzato a tutti i segmenti creati, utilizzando le regole della console X-Ray. La regola predefinita è 1 segmento al secondo, più il 5% dei segmenti successivamente vengono campionati. Ciò significa che i segmenti creati rapidamente con l'agente potrebbero non essere campionati. Per risolvere questo problema, è necessario creare regole di campionamento personalizzate sulla console a X-Ray che campionano in modo appropriato i segmenti desiderati. Per ulteriori informazioni, consultacampionamento.

Configurazione dell'X-Ray SDK per Java

L'X-Ray SDK for Java include una classe denominata `AWSXRay` che fornisce il registratore globale. Questo è un `TracingHandler` che puoi usare per strumentare il tuo codice. Puoi configurare la registrazione globale per personalizzare il `AWSXRayServletFilter` che crea i segmenti relativi alle chiamate HTTP in entrata.

Sections

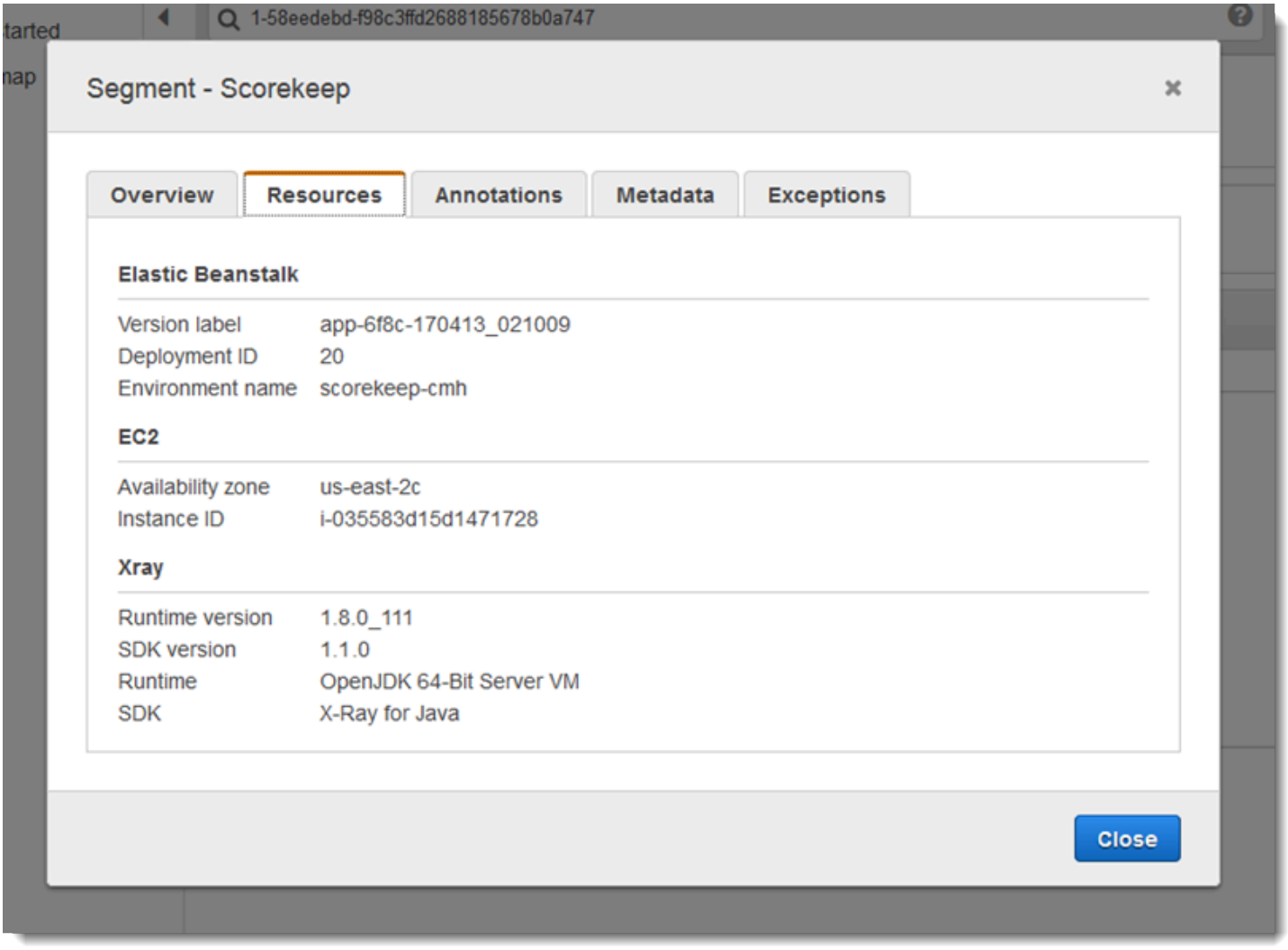
- Plugin di servizio
- Regole di campionamento
- Registrazione
- Listener di segmenti
- Variabili di ambiente
- Proprietà di sistema

Plugin di servizio

`pluginsUtilizzatelo` per registrare informazioni sul servizio che ospita l'applicazione.

Plug-in

- Amazon EC2: `EC2Plugin` aggiunge l'ID dell'istanza, la zona di disponibilità e il gruppo di CloudWatch log.
- Elastic `ElasticBeanstalkPlugin` Beanstalk: aggiunge il nome dell'ambiente, l'etichetta della versione e l'ID di distribuzione.
- Amazon ECS: `ECSPlugin` aggiunge l'ID del contenitore.
- Amazon EKS: `EKSPlugin` aggiunge l'ID del contenitore, il nome del cluster, l'ID del pod e il gruppo CloudWatch Logs.



Per usare un plugin, chiama `withPlugin` sul tuo `AWSXRayRecorderBuilder`.

Example WebConfigsrc/main/java/scorekeep/ .java - registratore

```
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());
  }
}
```

L'SDK utilizza anche le impostazioni del plug-in per impostare il campo sul segmento. `origin` Indica il tipo di AWS risorsa che esegue l'applicazione. Quando utilizzi più plugin, l'SDK utilizza il seguente ordine di risoluzione per determinare l'origine: ElasticBeanstalk > EKS > ECS > EC2.

Regole di campionamento

L'SDK utilizza le regole di campionamento definite nella console X-Ray per determinare quali richieste registrare. La regola predefinita tiene traccia della prima richiesta ogni secondo e del cinque percento di eventuali richieste aggiuntive su tutti i servizi che inviano tracce a X-Ray. Crea regole aggiuntive nella console X-Ray per personalizzare la quantità di dati registrati per ciascuna delle tue applicazioni.

L'SDK applica regole personalizzate nell'ordine in cui sono definite. Se una richiesta corrisponde a più regole personalizzate, l'SDK applica solo la prima regola.

> **Note**
>
> Se l'SDK non riesce a contattare X-Ray per ottenere le regole di campionamento, torna a una regola locale predefinita della prima richiesta ogni secondo e del cinque percento di eventuali

richieste aggiuntive per host. Ciò può verificarsi se l'host non dispone dell'autorizzazione per chiamare le API di campionamento o non può connettersi al demone X-Ray, che funge da proxy TCP per le chiamate API effettuate dall'SDK.

Puoi anche configurare l'SDK per caricare le regole di campionamento da un documento JSON. L'SDK può utilizzare le regole locali come backup per i casi in cui il campionamento a raggi X non è disponibile o utilizzare esclusivamente regole locali.

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
  }
}
```

Questo esempio definisce una regola personalizzata e una regola predefinita. La regola personalizzata applica una frequenza di campionamento del cinque percento senza alcun numero minimo di richieste da tracciare per i percorsi. `/api/move/` La regola predefinita tiene traccia della prima richiesta ogni secondo e del 10% delle richieste aggiuntive.

Lo svantaggio della definizione locale delle regole è che l'obiettivo fisso viene applicato da ciascuna istanza del registratore in modo indipendente, anziché essere gestito dal servizio X-Ray. Man mano che si installano più host, la tariffa fissa si moltiplica, rendendo più difficile il controllo della quantità di dati registrati.

Sì AWS Lambda, non è possibile modificare la frequenza di campionamento. Se la funzione viene chiamata da un servizio strumentato, le chiamate che hanno generato richieste campionate da quel

servizio verranno registrate da Lambda. Se il tracciamento attivo è abilitato e non è presente alcuna intestazione di tracciamento, Lambda prende la decisione di campionamento.

Per fornire le regole di backup in Spring, configura il registratore globale con un codice `CentralizedSamplingStrategy` in una classe di configurazione:

Example src/main/java/myapp/ .java - configurazione del registratore WebConfig

```
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());
}
```

Per Tomcat, aggiungi un listener che estenda `ServletContextListener` e registra il listener nel descrittore della distribuzione.

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>
```

Per utilizzare solo regole locali, sostituisci `CentralizedSamplingStrategy` con una `LocalizedSamplingStrategy`.

```
builder.withSamplingStrategy(new LocalizedSamplingStrategy(ruleFile));
```

Registrazione

Per impostazione predefinita, l'SDK invia messaggi a livello di file nei log delle applicazioni. `ERROR` È possibile abilitare la registrazione a livello di debug sull'SDK per generare log più dettagliati nel file di registro dell'applicazione. I livelli di registro validi sono`DEBUG`,, e. `INFO` `WARN` `ERROR` `FATAL` `FATAL`Il livello di registro silenzia tutti i messaggi di registro perché l'SDK non esegue la registrazione a livello fatale.

Example application.properties

Imposta il livello di log tramite la proprietà `logging.level.com.amazonaws.xray`.

```
logging.level.com.amazonaws.xray = DEBUG
```

Utilizza i log di debug per identificare i problemi, come ad esempio dei sottosegmenti non chiusi, quando generi dei sottosegmenti manualmente.

Inserimento dell'ID di tracciamento nei log

Per esporre l'ID di tracciamento completo corrente alle istruzioni di log, puoi inserire l'ID nel contesto diagnostico mappato (MDC). Utilizzando l'interfaccia `SegmentListener`, i metodi vengono chiamati dal registratore di X-Ray durante gli eventi del ciclo di vita del segmento. Quando inizia un segmento o un sottosegmento, l'ID di traccia qualificato viene inserito nell'MDC con la chiave `AWS-XRAY-TRACE-ID`. Quando tale segmento termina, la chiave viene rimossa dall'MDC. Questo espone l'ID di tracciamento alla libreria di registrazione in uso. Quando un sottosegmento termina, il relativo ID padre viene inserito nell'MDC.

Example ID di tracciamento completo

L'ID completo è rappresentato come `TraceID@EntityID`

```
1-5df42873-011e96598b447dfca814c156@541b3365be3dafc3
```

Questa funzionalità funziona con le applicazioni Java dotate di AWS X-Ray SDK for Java e supporta le seguenti configurazioni di registrazione:

- API front-end SLF4J con backend Logback
- API front-end SLF4J con backend Log4J2
- API front-end Log4J2 con backend Log4J2

Consulta le seguenti schede per le esigenze di ogni front-end e ogni back-end.

SLF4J Frontend

1. Aggiungi la seguente dipendenza Maven al tuo progetto.

```
<dependency>
    <groupId>com.amazonaws</groupId>
    <artifactId>aws-xray-recorder-sdk-slf4j</artifactId>
    <version>2.11.0</version>
</dependency>
```

2. Includi il metodo `withSegmentListener` durante la costruzione di `AWSXRayRecorder`. Questo aggiunge una classe `SegmentListener`, che inserisce automaticamente nuovi ID di tracciamento nell'MDC SLF4J.

   `SegmentListener` accetta una stringa facoltativa come parametro per configurare il prefisso dell'istruzione di log. Il prefisso può essere configurato nei seguenti modi:

   - Nessuno: utilizza il prefisso predefinito. `AWS-XRAY-TRACE-ID`
   - Vuoto: utilizza una stringa vuota (ad esempio`""`).
   - Personalizzato: utilizza un prefisso personalizzato come definito nella stringa.

   Example Dichiarazione **`AWSXRayRecorderBuilder`**

   ```
   AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder
           .standard().withSegmentListener(new SLF4JSegmentListener("CUSTOM-
   PREFIX"));
   ```

Log4J2 front end

1. Aggiungi la seguente dipendenza Maven al tuo progetto.

   ```
   <dependency>
       <groupId>com.amazonaws</groupId>
       <artifactId>aws-xray-recorder-sdk-log4j</artifactId>
       <version>2.11.0</version>
   </dependency>
   ```

2. Includi il metodo `withSegmentListener` durante la costruzione di `AWSXRayRecorder`. Questa operazione aggiungerà una classe `SegmentListener`, che inserisce automaticamente nuovi ID di tracciamento completi nell'MDC SLF4J.

   `SegmentListener` accetta una stringa facoltativa come parametro per configurare il prefisso dell'istruzione di log. Il prefisso può essere configurato nei seguenti modi:

   - Nessuno: utilizza il `AWS-XRAY-TRACE-ID` prefisso predefinito.
   - Vuoto: utilizza una stringa vuota (ad esempio`""`) e rimuove il prefisso.
   - Personalizzato: utilizza il prefisso personalizzato definito nella stringa.

Example Dichiarazione **`AWSXRayRecorderBuilder`**

```
AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder
        .standard().withSegmentListener(new Log4JSegmentListener("CUSTOM-
PREFIX"));
```

Logback backend

Per inserire l'ID di tracciamento negli eventi di registro, è necessario modificare il logger `PatternLayout`, che formatta ogni istruzione di registrazione.

1. Trova dove è configurato `patternLayout`. Puoi farlo a livello di codice o tramite un file di configurazione XML. Per ulteriori informazioni, consulta configurazione di Logback.
2. Inserisci `%X{AWS-XRAY-TRACE-ID}` in punto qualsiasi di `patternLayout` per inserire l'ID di tracciamento nelle istruzioni di registrazione future. `%X{}` indica che si sta recuperando un valore con la chiave fornita dall'MDC. Per ulteriori informazioni su PatternLayouts Logback, consulta. PatternLayout

Log4J2 backend

1. Trova dove è configurato `patternLayout`. Puoi eseguire questa operazione a livello di codice o tramite un file di configurazione scritto in formato XML, JSON, YAML o proprietà.

   Per ulteriori informazioni sulla configurazione di Log4J2 tramite un file di configurazione, consulta Configurazione.

   Per ulteriori informazioni sulla configurazione di Log4J2 a livello di codice, consulta Configurazione programmatica.
2. Inserisci `%X{AWS-XRAY-TRACE-ID}` in punto qualsiasi di `PatternLayout` per inserire l'ID di tracciamento nelle istruzioni di registrazione future. `%X{}` indica che si sta recuperando un valore con la chiave fornita dall'MDC. Per ulteriori informazioni su PatternLayouts Log4J2, vedi Pattern Layout.

Esempio di inserimento dell'ID di tracciamento

Di seguito viene mostrata una stringa `PatternLayout` modificata per includere l'ID di tracciamento. L'ID di tracciamento viene stampato dopo il nome del thread (`%t`) e prima del livello di log (`%-5p`).

Example **`PatternLayout`** con inserimento dell'ID

```
%d{HH:mm:ss.SSS} [%t] %X{AWS-XRAY-TRACE-ID} %-5p %m%n
```

AWS X-Ray stampa automaticamente la chiave e l'ID di traccia nell'istruzione di registro per una facile analisi. Di seguito viene illustrata un'istruzione log utilizzando la modifica `PatternLayout`.

Example Istruzione log con inserimento dell'ID

```
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
```

Il messaggio di registrazione stesso è alloggiato nel modello `%m` e viene impostato quando si chiama il logger.

Listener di segmenti

I listener di segmenti sono un'interfaccia per l'intercettazione degli eventi del ciclo di vita, ad esempio l'inizio e la fine dei segmenti prodotti da `AWSXRayRecorder`. L'implementazione di una funzione evento listener di segmenti potrebbe essere quella di aggiungere la stessa annotazione a tutti i sottosegmenti quando vengono creati con `onBeginSubsegment`, registrare un messaggio dopo che ogni segmento è stato inviato al daemon utilizzando `afterEndSegment`, o per registrare le query inviate dagli intercettori SQL che utilizzano `beforeEndSubsegment` per verificare se il sottosegmento rappresenta una query SQL e, in tal caso, aggiungendo ulteriori metadati.

Per visualizzare l'elenco completo delle `SegmentListener` funzioni, consulta la documentazione dell'API AWS X-Ray Recorder SDK for Java.

Nell'esempio seguente viene illustrato come aggiungere un'annotazione coerente a tutti i sottosegmenti durante la creazione con `onBeginSubsegment` e per stampare un messaggio di log alla fine di ogni segmento con `afterEndSegment`.

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());
    }
}
```

Durante la creazione di `AWSXRayRecorder`, viene fatto quindi riferimento a questo listener.

Example AWSXRayRecorderBuilder dichiarazione

```
AWSXRayRecorderBuilder builder = AWSXRayRecorderBuilder
        .standard().withSegmentListener(new MySegmentListener());
```

Variabili di ambiente

È possibile utilizzare le variabili di ambiente per configurare l'X-Ray SDK for Java. L'SDK supporta le seguenti variabili.

- `AWS_XRAY_TRACING_NAME`— Imposta un nome di servizio che l'SDK utilizza per i segmenti. Sostituisce il nome del servizio impostato sulla strategia di denominazione dei segmenti del filtro servlet.
- `AWS_XRAY_DAEMON_ADDRESS`— Imposta l'host e la porta del demone X-Ray. Per impostazione predefinita, l'SDK utilizza `127.0.0.1:2000` sia i dati di traccia (UDP) che il campionamento (TCP). Utilizzate questa variabile se avete configurato il demone per l'ascolto su una porta diversa o se è in esecuzione su un host diverso.

  Formato

  - Stessa porta — *`address`*`:`*`port`*
  - Porte diverse: `tcp:`*`address`*`:`*`port`* `udp:`*`address`*`:`*`port`*

- `AWS_XRAY_CONTEXT_MISSING`— Imposta `RUNTIME_ERROR` per generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.

  Valori validi

  - `RUNTIME_ERROR`— Genera un'eccezione di runtime.
  - `LOG_ERROR`— Registra un errore e continua (impostazione predefinita).
  - `IGNORE_ERROR`— Ignora l'errore e continua.

  Gli errori relativi a segmenti o sottosegmenti mancanti possono verificarsi quando si tenta di utilizzare un client con strumenti nel codice di avvio che viene eseguito quando non è aperta alcuna richiesta o nel codice che genera un nuovo thread.

Le variabili di ambiente sostituiscono i valori equivalenti per le proprietà di sistema e i valori impostati nel codice.

Proprietà di sistema

Puoi utilizzare le proprietà del sistema come alternativa alle variabili d'ambiente per una specifica JVM. L'SDK supporta le seguenti proprietà.

- `com.amazonaws.xray.strategy.tracingName`— Equivalente a. `AWS_XRAY_TRACING_NAME`
- `com.amazonaws.xray.emitters.daemonAddress`— Equivalente a`AWS_XRAY_DAEMON_ADDRESS`.
- `com.amazonaws.xray.strategy.contextMissingStrategy`— Equivalente a`AWS_XRAY_CONTEXT_MISSING`.

Se sono impostate sia una proprietà di sistema che la variabili di ambiente equivalente, sono utilizzati i valori della variabile di ambiente. Entrambi i metodi sostituiscono i valori impostati nel codice.

Tracciamento delle richieste in arrivo con X-Ray SDK for Java

Puoi utilizzare l'SDK X-Ray per tracciare le richieste HTTP in entrata che la tua applicazione serve su un'istanza EC2 in Amazon EC2 o Amazon ECS. AWS Elastic Beanstalk

Utilizza un `Filter` per analizzare le richieste HTTP in entrata. Quando aggiungete il filtro servlet X-Ray all'applicazione, l'X-Ray SDK for Java crea un segmento per ogni richiesta campionata. Questo segmento include durata, metodo e conclusione della richiesta HTTP. Analisi ulteriori creano sottosegmenti associati a questo segmento.

> **Note**
>
> Per quanto riguarda AWS Lambda le funzioni, Lambda crea un segmento per ogni richiesta campionata. Per ulteriori informazioni, consulta AWS Lambda e AWS X-Ray.

Ogni segmento ha un nome che identifica l'applicazione nella mappa dei servizi. Il segmento può essere denominato staticamente oppure è possibile configurare l'SDK per denominarlo dinamicamente in base all'intestazione dell'host nella richiesta in entrata. La denominazione dinamica consente di raggruppare le tracce in base al nome di dominio nella richiesta e di applicare un nome predefinito se il nome non corrisponde a uno schema previsto (ad esempio, se l'intestazione dell'host è falsificata).

> **Richieste inoltrate**
>
> Se un sistema di bilanciamento del carico o un altro intermediario inoltra una richiesta all'applicazione, X-Ray prende l'IP del client dall'`X-Forwarded-For`intestazione della richiesta anziché dall'IP di origine nel pacchetto IP. L'IP del client registrato per una richiesta inoltrata può essere falsificato, quindi non dovrebbe essere considerato attendibile.

Quando viene inoltrata una richiesta, l'SDK imposta un campo aggiuntivo nel segmento per indicarlo. Se il segmento contiene il campo `x_forwarded_for` impostato su`true`, l'IP del client è stato preso dall'`X-Forwarded-For`intestazione della richiesta HTTP.

Il gestore dei messaggi crea un segmento per ogni richiesta in entrata con un blocco `http` che contiene le informazioni riportate qui di seguito:

- Metodo HTTP: GET, POST, PUT, DELETE, ecc.
- Indirizzo client: l'indirizzo IP del client che ha inviato la richiesta.
- Codice di risposta: il codice di risposta HTTP per la richiesta completata.
- Tempistica: l'ora di inizio (quando è stata ricevuta la richiesta) e l'ora di fine (quando è stata inviata la risposta).
- Agente utente: il `user-agent` codice della richiesta.
- Lunghezza del contenuto: il `content-length` risultato della risposta.

Sections

- Aggiunta di un filtro di tracciamento all'applicazione (Tomcat)
- Aggiunta di un filtro di tracciamento all'applicazione (Spring)
- Configurazione di una strategia di denominazione dei segmenti

Aggiunta di un filtro di tracciamento all'applicazione (Tomcat)

Per Tomcat, aggiungere un `<filter>` al file `web.xml` del progetto. Utilizza il parametro `fixedName` per specificare un nome di servizio da applicare ai segmenti creati per le richieste in entrata.

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>
```

Aggiunta di un filtro di tracciamento all'applicazione (Spring)

Per Spring, aggiungere un `Filter` alla classe `WebConfig`. Passare il nome del segmento al costruttore `AWSXRayServletFilter` come valore di tipo stringa.

Example WebConfigsrc/main/java/myapp/ .java - primavera

```
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");
  }
}
```

Configurazione di una strategia di denominazione dei segmenti

AWS X-Ray utilizza un nome di servizio per identificare l'applicazione e distinguerla dalle altre applicazioni, database, API esterne e risorse utilizzate dall'applicazione. AWS Quando X-Ray SDK genera segmenti per le richieste in entrata, registra il nome del servizio dell'applicazione nel campo del nome del segmento.

L'X-Ray SDK può denominare i segmenti dopo il nome host nell'intestazione della richiesta HTTP. Tuttavia, questa intestazione può essere falsificata, il che potrebbe causare nodi imprevisti nella mappa dei servizi. Per evitare che l'SDK nomini i segmenti in modo errato a causa di richieste con intestazioni host contraffatte, è necessario specificare un nome predefinito per le richieste in entrata.

Se la tua applicazione soddisfa le richieste per più domini, puoi configurare l'SDK in modo che utilizzi una strategia di denominazione dinamica che rifletta questo aspetto nei nomi dei segmenti. Una strategia di denominazione dinamica consente all'SDK di utilizzare il nome host per le richieste che corrispondono a uno schema previsto e di applicare il nome predefinito alle richieste che non lo fanno.

Ad esempio, potresti avere una singola applicazione che serve le richieste a tre sottodomini:, e. `www.example.com` `api.example.com` `static.example.com` È possibile utilizzare una strategia di denominazione dinamica con lo schema `*.example.com` per identificare i segmenti per ogni sottodominio con un nome diverso, ottenendo tre nodi di servizio sulla mappa dei servizi. Se l'applicazione riceve richieste con un nome host che non corrisponde allo schema, sulla mappa dei servizi verrà visualizzato un quarto nodo con un nome di fallback specificato dall'utente.

Per utilizzare lo stesso nome per tutti i segmenti della richiesta, specifica il nome della tua applicazione quando inizializzi il filtro servlet, come illustrato nella sezione precedente. Ciò ha lo stesso effetto della creazione di una correzione SegmentNamingStrategychiamandola `SegmentNamingStrategy.fixed()` e passandola al `AWSXRayServletFilter`costruttore.

> **Note**
>
> È possibile sovrascrivere il nome di servizio predefinito definito nel codice con la variabile di `AWS_XRAY_TRACING_NAME` ambiente.

Una strategia di denominazione dinamica definisce un modello al quale devono corrispondere i nomi degli host e un nome di default per l'utilizzo qualora il nome dell'host nella richiesta HTTP non corrisponda al modello. Per denominare segmenti in modo dinamico in Tomcat, utilizzare la `dynamicNamingRecognizedHosts` e `dynamicNamingFallbackName` per definire rispettivamente i modelli e il nome di default.

Example WEB-INF/web.xml - Filtro Servlet con assegnazione dinamica dei nomi

```
<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>
```

Per Spring, crea una dinamica [SegmentNamingStrategy](#)`SegmentNamingStrategy.dynamic()`chiamando e passala al `AWSXRayServletFilter` costruttore.

Example WebConfigsrc/main/java/myapp/ .java - filtro servlet con denominazione dinamica

```
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"));
  }
}
```

Tracciamento delle chiamate AWS SDK con X-Ray SDK for Java

Quando l'applicazione effettua chiamate per Servizi AWS archiviare dati, scrivere in una coda o inviare notifiche, X-Ray SDK for Java tiene traccia delle chiamate downstream in sottosegmenti. Le risorse tracciate Servizi AWS e a cui accedi all'interno di tali servizi (ad esempio, un bucket Amazon S3 o una coda Amazon SQS) vengono visualizzate come nodi downstream sulla mappa di traccia nella console X-Ray.

L'SDK X-Ray per Java strumenta automaticamente tutti i client SDK AWS quando includi i sottomoduli `aws-sdk` e un `aws-sdk-instrumentor` nella build. Se non includi il modulo Instrumentor, puoi scegliere di analizzare alcuni client escludendo gli altri.

Per strumentare i singoli client, rimuovi il `aws-sdk-instrumentor` sottomodulo dalla build e aggiungi un client `XRayClient` as a `TracingHandler` sul tuo AWS SDK utilizzando il client builder del servizio.

Ad esempio, per analizzare un client `AmazonDynamoDB`, passa un gestore del tracciamento `AmazonDynamoDBClientBuilder`.

Example MyModel.java - Client DynamoDB

```
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();
...
```

Per tutti i servizi, puoi vedere il nome dell'API richiamata nella console X-Ray. Per un sottoinsieme di servizi, l'SDK X-Ray aggiunge informazioni al segmento per fornire una maggiore granularità nella mappa dei servizi.

Ad esempio, quando si effettua una chiamata con un client DynamoDB con strumentazione, l'SDK aggiunge il nome della tabella al segmento per le chiamate destinate a una tabella. Nella console, ogni tabella appare come un nodo separato nella mappa dei servizi, con un nodo DynamoDB generico per le chiamate che non hanno come destinazione una tabella.

Example Sottosegmento per una chiamata a DynamoDB per salvare un elemento

```
{
  "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",
  }
}
```

Quando si accede alle risorse con nome, le chiamate ai seguenti servizi creano ulteriori nodi della mappa del servizio. Le chiamate che non sono hanno come obiettivo risorse specifiche creano un nodo generico per il servizio.

- Amazon DynamoDB: nome della tabella
- Amazon Simple Storage Service: nome del bucket e della chiave
- Amazon Simple Queue Service: nome della coda

Per strumentare le chiamate downstream a Servizi AWS with AWS SDK for Java 2.2 e versioni successive, puoi omettere il `aws-xray-recorder-sdk-aws-sdk-v2-instrumentor` modulo dalla configurazione di build. Al suo posto, includi il `aws-xray-recorder-sdk-aws-sdk-v2 module`, quindi analizza i singoli client configurandoli con un `TracingInterceptor`.

Example AWS SDK for Java 2.2 e versioni successive - 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();
//...
```

Tracciamento delle chiamate ai servizi Web HTTP downstream con X-Ray SDK for Java

Quando l'applicazione effettua chiamate a microservizi o API HTTP pubbliche, è possibile utilizzare la versione di X-Ray SDK per Java per `HttpClient` strumentare tali chiamate e aggiungere l'API al service graph come servizio downstream.

L'X-Ray SDK for Java include `DefaultHttpClient` `HttpClientBuilder` classi che possono essere utilizzate al posto degli HttpComponents equivalenti di Apache per strumentare le chiamate HTTP in uscita.

- `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`

Queste librerie sono contenute nel sottomodulo [aws-xray-recorder-sdk-apache-http].

È possibile sostituire le istruzioni di importazione esistenti con l'equivalente X-Ray di Instrument all client o utilizzare il nome completo quando si inizializza un client su client specifici dello strumento.

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();
    }
  }
```

Quando si effettua una chiamata a un'API Web downstream, l'X-Ray SDK for Java registra un sottosegmento con informazioni sulla richiesta e sulla risposta HTTP. X-Ray utilizza il sottosegmento per generare un segmento dedotto per l'API remota.

Example Sottosegmento per una chiamata HTTP a valle

```
{
  "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 Segmento dedotto per una chiamata HTTP a valle

```
{
  "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
}
```

Tracciamento delle query SQL con X-Ray SDK for Java

Interceptori SQL

Strumenta le query del database SQL aggiungendo l'interceptor JDBC X-Ray SDK for Java alla configurazione dell'origine dati.

- PostgreSQL – `com.amazonaws.xray.sql.postgres.TracingInterceptor`
- MySQL – `com.amazonaws.xray.sql.mysql.TracingInterceptor`

Questi collettori si trovano rispettivamente nei sottomoduli `aws-xray-recorder-sql-postgres` e `aws-xray-recorder-sql-mysql`. Implementano `org.apache.tomcat.jdbc.pool.JdbcInterceptor` e sono compatibili con i pool di connessione di Tomcat.

> **Note**
>
> Gli intercettori SQL non registrano la query SQL stessa all'interno dei sottosegmenti per scopi di sicurezza.

Nel caso di Spring, aggiungi il collettore in un file delle proprietà e compila la sorgente dati con il `DataSourceBuilder` di 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`** - Origine dati

```
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();
  }
...
}
```

Per Tomcat, `setJdbcInterceptors` richiamate l'origine dati JDBC con un riferimento alla classe X-Ray SDK for Java.

Example **`src/main/myapp/model.java`** - Origine dati

```
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;");
```

La libreria Tomcat JDBC Data Source è inclusa nell'X-Ray SDK for Java, ma è possibile dichiararla come dipendenza fornita dal documento che la si utilizza.

Example **`pom.xml`**- Origine dati JDBC

```
<dependency>
  <groupId>org.apache.tomcat</groupId>
  <artifactId>tomcat-jdbc</artifactId>
  <version>8.0.36</version>
  <scope>provided</scope>
</dependency>
```

Decoratore di tracciamento SQL nativo

- Aggiungilo `aws-xray-recorder-sdk-sql`alle tue dipendenze.
- Decora l'origine dati, la connessione o la dichiarazione del database.

```
dataSource = TracingDataSource.decorate(dataSource)
```

```
connection = TracingConnection.decorate(connection)
statement = TracingStatement.decorateStatement(statement)
preparedStatement = TracingStatement.decoratePreparedStatement(preparedStatement,
 sql)
callableStatement = TracingStatement.decorateCallableStatement(callableStatement,
 sql)
```

Generazione di sottosegmenti personalizzati con X-Ray SDK for Java

I sottosegmenti estendono il segmento di una traccia con dettagli sul lavoro svolto per soddisfare una richiesta. Ogni volta che si effettua una chiamata con un client dotato di strumentazione, l'X-Ray SDK registra le informazioni generate in un sottosegmento. È possibile creare sottosegmenti aggiuntivi per raggruppare altri sottosegmenti, misurare le prestazioni di una sezione di codice o registrare annotazioni e metadati.

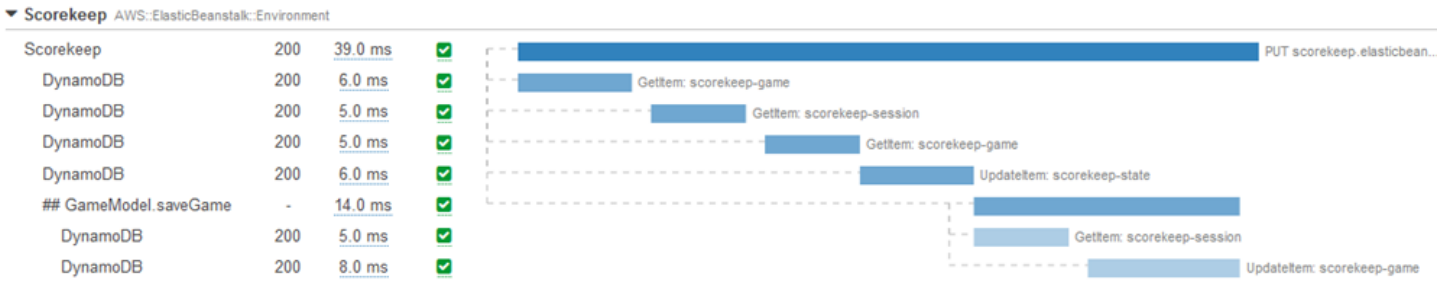
Per gestire i sottosegmenti, utilizza i metodi `beginSubsegment` e `endSubsegment`.

Example GameModel.java - sottosegmento personalizzato

```
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 questo esempio, il codice all'interno del sottosegmento carica la sessione del gioco da DynamoDB con un metodo sul modello di sessione e utilizza il mapper DynamoDB per salvare il AWS SDK for

Java gioco. Inserendo questo codice in un sottosegmento, le chiamate DynamoDB sono figli del `Save Game` sottosegmento nella vista trace della console.



Se il codice nel tuo sottosegmento genera eccezioni gestite, eseguine il wrapping in un blocco `try` e chiama `AWSXRay.endSubsegment()` in un blocco `finally` per assicurare che il sottosegmento sia sempre chiuso. Se un sottosegmento non è chiuso, il segmento principale non può essere completato e non verrà inviato a X-Ray.

Per il codice che non genera eccezioni controllate, puoi passare il codice a `AWSXRay.CreateSubsegment` come funzione Lambda.

Example Funzione Lambda del sottosegmento

```
import com.amazonaws.xray.AWSXRay;

AWSXRay.createSubsegment("getMovies", (subsegment) -> {
    // function code
});
```

Quando si crea un sottosegmento all'interno di un segmento o di un altro sottosegmento, X-Ray SDK for Java genera un ID per tale sottosegmento e registra l'ora di inizio e l'ora di fine.

Example Sottosegmento con metadati

```
"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"
    }
  },
```

Per la programmazione asincrona e multithread, è necessario passare manualmente il sottosegmento al `endSubsegment()` metodo per assicurarsi che sia chiuso correttamente, poiché il contesto X-Ray può essere modificato durante l'esecuzione asincrona. Se un sottosegmento asincrono viene chiuso dopo la chiusura del segmento principale, questo metodo trasmetterà automaticamente l'intero segmento al demone X-Ray.

Example Sottosegmento asincrono

```
@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();
}
```

Aggiungi annotazioni e metadati ai segmenti con X-Ray SDK for Java

È possibile utilizzare annotazioni e metadati per registrare informazioni aggiuntive sulle richieste, sull'ambiente o sull'applicazione. È possibile aggiungere annotazioni e metadati ai segmenti creati da X-Ray SDK o ai sottosegmenti personalizzati creati dall'utente.

Le annotazioni sono coppie chiave-valore con stringhe, numeri o valori booleani. Le annotazioni sono indicizzate per essere utilizzate con le espressioni di filtro. Utilizzale per registrare i dati che desideri utilizzare per raggruppare le tracce nella console oppure per chiamare l'API `GetTraceSummaries`.

I metadati sono coppie chiave-valore che possono avere valori di qualsiasi tipo, inclusi oggetti ed elenchi, ma non sono indicizzati per essere utilizzati con le espressioni di filtro. Utilizzate i metadati per registrare dati aggiuntivi che desiderate archiviare nella traccia ma che non è necessario utilizzare con la ricerca.

Oltre ad annotazioni e metadati, sui segmenti puoi anche registrare le stringhe degli ID utente. Gli ID utente vengono memorizzati in un campo separato su segmenti e sono indicizzati per l'uso nelle ricerche.

Sections

- Registrazione delle annotazioni con X-Ray SDK for Java
- Registrazione di metadati con X-Ray SDK for Java
- Registrazione degli ID utente con X-Ray SDK for Java

Registrazione delle annotazioni con X-Ray SDK for Java

Utilizza le annotazioni per memorizzare le informazioni su segmenti o sottosegmenti che desideri siano indicizzate per la ricerca.

Requisiti per le annotazioni

- Chiavi: la chiave per un'annotazione a raggi X può contenere fino a 500 caratteri alfanumerici. Non è possibile utilizzare spazi o simboli diversi dal simbolo di sottolineatura (_).
- Valori: il valore di un'annotazione X-Ray può contenere fino a 1.000 caratteri Unicode.
- Il numero di annotazioni: è possibile utilizzare fino a 50 annotazioni per traccia.

Per registrare le annotazioni

1. Ottenere un riferimento al segmento o sottosegmento corrente da `AWSXRay`.

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Segment;
   ...
   Segment document = AWSXRay.getCurrentSegment();
   ```

   oppure

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Subsegment;
   ...
   Subsegment document = AWSXRay.getCurrentSubsegment();
   ```

2. Chiamare `putAnnotation` con una chiave di tipo Stringa e un valore booleano, numerico o di tipo Stringa.

```
document.putAnnotation("mykey", "my value");
```

L'SDK memorizza le annotazioni come coppie chiave-valore in un oggetto `annotations` all'interno del documento di segmento. Se chiami `putAnnotation` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento o sottosegmento viene sovrascritto.

Per trovare tracciamenti con annotazioni contenenti valori specifici, utilizza la parola chiave `annotations.`*`key`* in un'espressione filtro.

Example **src/main/java/scorekeep/GameModel.java**— Annotazioni e metadati

```
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();
    }
  }
```

Registrazione di metadati con X-Ray SDK for Java

Utilizza i metadati per memorizzare le informazioni su segmenti o sottosegmenti che non è necessario che siano indicizzate per la ricerca. I valori dei metadati possono essere stringhe, numeri, valori booleani o qualsiasi oggetto che possa essere serializzato in un oggetto o in un vettore JSON.

Per registrare i metadati

1. Ottenere un riferimento al segmento o sottosegmento corrente da `AWSXRay`.

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Segment;
   ...
   Segment document = AWSXRay.getCurrentSegment();
   ```

   oppure

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Subsegment;
   ...
   Subsegment document = AWSXRay.getCurrentSubsegment();
   ```

2. Chiamare `putMetadata` con un namespace di tipo stringa, una chiave di tipo Stringa e un valore booleano, numerico, di tipo Stringa o di tipo oggetto.

   ```
   document.putMetadata("my namespace", "my key", "my value");
   ```

   oppure

   Invocare `putMetadata` semplicemente con una chiave e un valore.

   ```
   document.putMetadata("my key", "my value");
   ```

Se non specifichi un namespace, l'SDK utilizza `default`. Se chiami `putMetadata` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento o sottosegmento viene sovrascritto.

Example **`src/main/java/scorekeep/GameModel.java`**— Annotazioni e metadati

```
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();
    }
  }
```

Registrazione degli ID utente con X-Ray SDK for Java

Memorizza gli ID utente sui segmenti di richiesta per identificare l'utente che ha inviato la richiesta.

Per registrare gli ID degli utenti

1. Ottenere un riferimento al segmento corrente da `AWSXRay`.

   ```
   import com.amazonaws.xray.AWSXRay;
   import com.amazonaws.xray.entities.Segment;
   ...
   Segment document = AWSXRay.getCurrentSegment();
   ```

2. Chiamare `setUser` con una stringa che rappresenta l'ID dell'utente che ha inviato la richiesta.

   ```
   document.setUser("U12345");
   ```

Puoi chiamare `setUser` nel tuo controller per registrare l'ID utente non appena l'applicazione inizia ad elaborare una richiesta. Se usi il segmento solo per impostare l'ID utente, puoi concatenare le chiamate in un'unica riga.

Example MoveControllersrc/main/java/scorekeep/ .java — ID utente

```
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);
  }
```

Per trovare tracciamenti associati ad un ID utente, utilizza la parola chiave `user` in un'espressione filtro.

AWS X-Ray metriche per X-Ray SDK for Java

Questo argomento descrive lo spazio dei AWS X-Ray nomi, le metriche e le dimensioni. Puoi utilizzare X-Ray SDK for Java per pubblicare metriche CloudWatch Amazon non campionate dai segmenti X-Ray raccolti. Questi parametri sono derivati dall'ora di inizio e di fine del segmento e dai flag di stato di errore, errore e limitazione. Utilizzare questi parametri di tracciamento per esporre tentativi e problemi di dipendenza all'interno di sottosegmenti.

CloudWatch è essenzialmente un archivio di metriche. Una metrica è il concetto fondamentale CloudWatch e rappresenta un insieme di punti dati ordinati nel tempo. Pubblichi (o Servizi AWS) punti dati delle metriche CloudWatch e recuperi le statistiche su tali punti dati come un insieme ordinato di dati di serie temporali.

I parametri sono definiti in modo univoco da un nome, un namespace e una o più dimensioni. A ogni punto dati è associato un timestamp e, facoltativamente, un'unità di misura. Quando si richiedono statistiche, il flusso di dati restituito viene identificato da namespace, nome parametro e dimensione.

Per ulteriori informazioni CloudWatch, consulta la Amazon CloudWatch User Guide.

Metriche X-Ray CloudWatch

Lo spazio dei nomi `ServiceMetrics/SDK` include le metriche descritte di seguito.

Metrica	Statistiche disponibili	Descrizione	unità
`Latency`	Media, Minimo, Massimo, Conteggio	La differenza tra l'ora di inizio e di fine. Media, minima e massima descrivono la latenza operativa. Il conteggio descrive il conteggio delle chiamate.	Millisecondi
`ErrorRate`	Average (Media), Sum (Somma)	La percentuale di richieste non riuscite con un codice di stato `4xx Client Error`, causando un errore.	Percentuale
`FaultRate`	Average (Media), Sum (Somma)	Il tasso di tracciamenti non riusciti con un codice di stato `5xx Server Error`, causando un errore.	Percentuale
`ThrottleRate`	Average (Media), Sum (Somma)	La velocità di tracciamento limitate che restituiscono un codice di stato `429`. Questo è un sottoinsieme della metrica `ErrorRate` .	Percentuale
`OkRate`	Average (Media), Sum (Somma)	Il tasso di richieste tracciate risultante in un codice di stato `OK`.	Percentuale

Dimensioni dei raggi X CloudWatch

Utilizzate le dimensioni nella tabella seguente per rifinire le metriche restituite per le vostre applicazioni Java con strumentazione X-Ray.

Dimensione	Descrizione
`ServiceType`	Il tipo di servizio, ad esempio, `AWS::EC2::Instance` o `NONE`, se non noto.
`ServiceName`	Il nome canonico del servizio.

Abilita le metriche X-Ray CloudWatch

Utilizza la procedura seguente per abilitare i parametri di tracciamento nell'applicazione Java strumentata.

Per configurare i parametri di tracciamento

1. Aggiungi il pacchetto `aws-xray-recorder-sdk-metrics` come dipendenza Maven. Per ulteriori informazioni, vedere X-Ray SDK for Java Submodules.
2. Attiva un nuovo `MetricsSegmentListener()` come parte della build del registratore globale.

   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. Implementa l' CloudWatch agente per raccogliere i parametri utilizzando Amazon Elastic Compute Cloud (Amazon EC2), Amazon Elastic Container Service (Amazon ECS) o Amazon Elastic Kubernetes Service (Amazon EKS):

   - Per configurare Amazon EC2, consulta Implementazione dell' CloudWatch agente e del demone X-Ray su Amazon EC2.
   - Per configurare Amazon ECS, consulta Distribuzione dell' CloudWatch agente e del demone X-Ray su Amazon ECS.
   - Per configurare Amazon EKS, consulta Implementazione dell' CloudWatch agente e del demone X-Ray su Amazon EKS.

4. Configura l'SDK per comunicare con l'agente. CloudWatch Per impostazione predefinita, l'SDK comunica con l' CloudWatch agente sull'indirizzo. `127.0.0.1` Puoi configurare indirizzi alternativi impostando la variabile di ambiente o la proprietà Java su `address:port`.

   Example Variabile di ambiente

   ```
   AWS_XRAY_METRICS_DAEMON_ADDRESS=address:port
   ```

   Example Proprietà Java

   ```
   com.amazonaws.xray.metrics.daemonAddress=address:port
   ```

Per convalidare la configurazione

1. Accedi AWS Management Console e apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Apri la scheda Metrics (Parametri) per osservare l'afflusso dei tuoi parametri.

3. (Facoltativo) Nella CloudWatch console, nella scheda Registri, apri il gruppo di `ServiceMetricsSDK` log. Cerca un flusso di log che corrisponda ai parametri host e conferma i messaggi di registro.

Passaggio del contesto del segmento tra thread in un'applicazione multithread

Quando crei un nuovo thread nella tua applicazione, l'`AWSXRayRecorder` non mantenere un riferimento all'Entità del segmento o sottosegmento correnti. Se utilizzi un client con strumentazione nel nuovo thread, l'SDK tenta di scrivere su un segmento che non esiste, causando un. SegmentNotFoundException

Per evitare di generare eccezioni durante lo sviluppo, puoi configurare il registratore con un comando ContextMissingStrategyche gli indichi invece di registrare un errore. È possibile configurare la strategia in codice con SetContextMissingStrategyo configurare opzioni equivalenti con una variabile di ambiente o una proprietà di sistema.

Un modo di risolvere l'errore è utilizzare un nuovo segmento chiamando beginSegment quando inizia il thread e endSegment quando viene chiuso. Questo funziona se si sta analizzando una porzione di codice che non viene eseguita in risposta ad una richiesta HTTP, come il codice eseguito all'avvio dell'applicazione.

Se utilizzi più thread per gestire le richieste in entrata, puoi passare il segmento o il sottosegmento correnti al nuovo thread e fornirli al registratore globale. In questo modo le informazioni registrate all'interno del nuovo thread sono associate allo stesso segmento delle altre informazioni registrate sulla richiesta. Una volta che il segmento è disponibile nel nuovo thread, puoi eseguire qualsiasi eseguibile con accesso al contesto di quel segmento utilizzando il `segment.run(() -> { ... })` metodo.

Consulta Utilizzo dei client analizzati nei thread worker per un esempio.

Utilizzo di X-Ray con programmazione asincrona

L'X-Ray SDK for Java può essere utilizzato in programmi Java asincroni con. SegmentContextExecutors SegmentContextExecutor Implementa l'interfaccia Executor, il che significa che può essere passata a tutte le operazioni asincrone di un. CompletableFuture Ciò garantisce che qualsiasi operazione asincrona venga eseguita con il segmento corretto nel suo contesto.

Example Esempio App.java: passaggio a 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 con Spring e X-Ray SDK per Java

Questo argomento descrive come utilizzare X-Ray SDK e Spring Framework per strumentare l'applicazione senza modificarne la logica di base. Ciò significa che ora esiste un modo non invasivo per strumentare le applicazioni in esecuzione in remoto. AWS

Per abilitare AOP in Spring

1. Configurare Spring
2. Aggiungi un filtro di tracciamento alla tua applicazione
3. Annotare il codice o implementare un'interfaccia
4. Attivare X-Ray nell'applicazione

Configurazione di Spring

Per configurare Spring affinché utilizzi AOP per l'analisi della tua applicazione, puoi utilizzare Maven o Gradle.

Se utilizzi Maven per compilare l'applicazione, aggiungi la seguente dipendenza nel file `pom.xml`.

```
<dependency>
     <groupId>com.amazonaws</groupId>
     <artifactId>aws-xray-recorder-sdk-spring</artifactId>
     <version>2.11.0</version>
</dependency>
```

Se utilizzi Gradle, aggiungi la seguente dipendenze nel file `build.gradle`.

```
compile 'com.amazonaws:aws-xray-recorder-sdk-spring:2.11.0'
```

Configurazione di Spring Boot

Oltre alla dipendenza Spring descritta nella sezione precedente, se stai usando Spring Boot, aggiungi la seguente dipendenza se non è già nel tuo classpath.

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'
```

Aggiungere un filtro di tracciamento all'applicazione

Aggiungi un `Filter` alla tua `WebConfig` classe. Passare il nome del segmento al costruttore `AWSXRayServletFilter` come valore di tipo stringa. Per ulteriori informazioni sul tracciamento dei filtri e sulla strumentazione delle richieste in arrivo, consulta. Tracciamento delle richieste in arrivo con X-Ray SDK for Java

Example src/main/java/myapp/ .java - primavera WebConfig

```
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 Support

Spring 6 utilizza Jakarta anziché Javax per la sua Enterprise Edition. Per supportare questo nuovo spazio dei nomi, X-Ray ha creato un set parallelo di classi che risiedono nel proprio spazio dei nomi di Jakarta.

Per le classi di filtro, sostituirle con. `javax` `jakarta` Quando configurate una strategia di denominazione dei segmenti, aggiungete `jakarta` prima del nome della classe di strategia di denominazione, come nell'esempio seguente:

```
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"));
    }
}
```

Annotazione del codice o implementazione di un'interfaccia

Le tue classi devono essere annotate con l'`@XRayEnabled`annotazione o implementare l'interfaccia. `XRayTraced` Questo indica al sistema AOP di eseguire il wrapping delle funzioni della classe interessata per l'analisi di X-Ray.

Attivazione di X-Ray nell'applicazione

Per attivare il tracciamento X-Ray nell'applicazione, il codice deve estendere la classe astratta `BaseAbstractXRayInterceptor` sovrascrivendo i seguenti metodi.

- `generateMetadata`—Questa funzione consente la personalizzazione dei metadati allegati alla traccia della funzione corrente. Per impostazione predefinita, il nome della classe della funzione in

esecuzione viene registrato nei metadati. Puoi aggiungere altri dati se hai bisogno di informazioni aggiuntive.

- `xrayEnabledClasses`—Questa funzione è vuota e dovrebbe rimanere tale. Serve come host per un pointcut che indica ai collettori di quali metodi effettuare il wrapping. Definisci il pointcut specificando quali delle classi che annotate con `@XRayEnabled` tracciare. La seguente istruzione pointcut indica al collettore di eseguire il wrapping di tutti i bean controller annotati con l'annotazione `@XRayEnabled`.

```
@Pointcut(“@within(com.amazonaws.xray.spring.aop.XRayEnabled) && bean(*Controller)”)
```

Se il tuo progetto utilizza Spring Data JPA, valuta la possibilità di estendere from `AbstractXRayInterceptor` anziché. `BaseAbstractXRayInterceptor`

Esempio

Il codice seguente estende la classe astratta. `BaseAbstractXRayInterceptor`

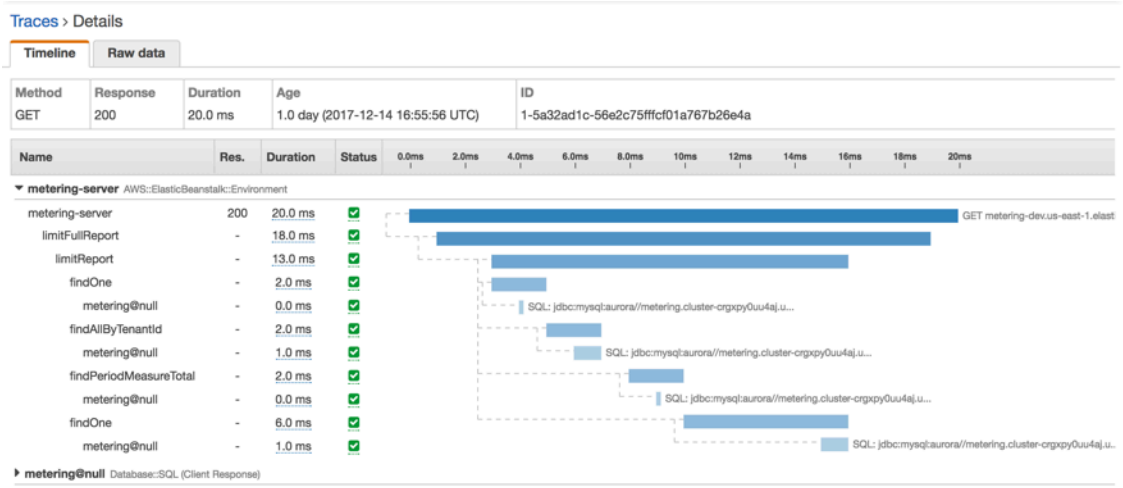
```
@Aspect
@Component
public class XRayInspector extends BaseAbstractXRayInterceptor {
    @Override
    protected Map<String, Map<String, Object>> generateMetadata(ProceedingJoinPoint
 proceedingJoinPoint, Subsegment subsegment) throws Exception {
        return super.generateMetadata(proceedingJoinPoint, subsegment);
    }

  @Override
  @Pointcut("@within(com.amazonaws.xray.spring.aop.XRayEnabled) && bean(*Controller)")

  public void xrayEnabledClasses() {}

}
```

Il codice seguente è una classe che sarà analizzata da X-Ray.

```
@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());
        }
    }
}
```

Se hai configurato la tua applicazione correttamente, dovresti vedere l'intero stack di chiamate dell'applicazione, dai controller fino alle chiamate al servizio, come illustrato nella seguente schermata della console.



Strumentate la vostra applicazione con Node.js

Esistono due modi per utilizzare l'applicazione Node.js per inviare tracce a X-Ray:

- AWS Distro for OpenTelemetry JavaScript: una AWS distribuzione che fornisce un set di librerie open source per l'invio di metriche e tracce correlate a più soluzioni di AWS monitoraggio, tra cui Amazon AWS X-Ray e Amazon OpenSearch Service CloudWatch, tramite AWS Distro for Collector. OpenTelemetry
- AWS X-Ray SDK per Node.js — Un set di librerie per la generazione e l'invio di tracce a X-Ray tramite il demone X-Ray.

Per ulteriori informazioni, consulta Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray.

AWSDistro perOpenTelemetry JavaScript

ConAWSDistro perOpenTelemetry(ADOTTARE)JavaScript, puoi strumentare le tue applicazioni una sola volta e inviare metriche e tracce correlate a piùAWSsoluzioni di monitoraggio, tra cui AmazonCloudWatch,AWS X-Raye AmazonOpenSearchServizio. Utilizzo di X-Ray conAWSDistro perOpenTelemetryrichiede due componenti: unOpenTelemetrySDKabilitato per l'uso con X-Ray eAWSDistro perOpenTelemetryCollezionistaabilitato per l'uso con X-Ray.

Per iniziare, consulta ilAWSDistro perOpenTelemetry JavaScriptdocumentazione.

> **Note**
>
> ADOTTAREJavaScriptè supportato per tutte le applicazioni Node.js lato server. ADOTTAREJavaScriptnon è in grado di esportare dati in X-Ray dai client browser.

Per ulteriori informazioni sull'utilizzo diAWSDistro perOpenTelemetryconAWS X-Raye altroServizi AWS, vediAWSDistro perOpenTelemetryo ilAWSDistro perOpenTelemetryDocumentazione.

Per ulteriori informazioni sul supporto e l'utilizzo delle lingue, vedereAWSOsservabilità suGitHub.

AWS X-Ray SDK per Node.js

L'SDK X-Ray per Node.js è una libreria per le applicazioni Web Express e le funzioni Lambda Node.js che fornisce classi e metodi per generare e inviare dati di traccia al demone X-Ray. I dati di tracciamento includono informazioni sulle richieste HTTP in entrata servite dall'applicazione e sulle chiamate effettuate dall'applicazione ai servizi downstream utilizzando i client SDK o HTTP. AWS

> **Note**
>
> L'X-Ray SDK per Node.js è un progetto open source. Puoi seguire il progetto e inviare problemi e richieste di pull su GitHub: github.com/aws/ aws-xray-sdk-node

Se utilizzi Express, per tracciare le richieste in entrata inizia aggiungendo l'SDK come middleware sul tuo application server. Il middleware crea un segmento per ogni richiesta tracciata e completa

il segmento quando viene inviata la risposta. Fino a che il segmento è aperto puoi usare i metodi del client dell'SDK per aggiungere informazioni al segmento e creare sottosegmenti per tracciare le chiamate a valle. L'SDK, inoltre, registra automaticamente le eccezioni sollevate dall'applicazione per il tempo durante il quale il segmento è aperto.

Per le funzioni Lambda richiamate da un'applicazione o un servizio strumentato, Lambda legge l'intestazione di tracciamento e traccia automaticamente le richieste campionate. Per altre funzioni, puoi configurare Lambda per campionare e tracciare le richieste in arrivo. In entrambi i casi, Lambda crea il segmento e lo fornisce all'X-Ray SDK.

> **Note**
>
> Su Lambda, l'SDK X-Ray è opzionale. Se non lo usi nella tua funzione, la mappa dei servizi includerà comunque un nodo per il servizio Lambda e uno per ogni funzione Lambda. Aggiungendo l'SDK, puoi utilizzare il codice della funzione per aggiungere sottosegmenti al segmento di funzione registrato da Lambda. Per ulteriori informazioni, consulta AWS Lambda e AWS X-Ray.

Successivamente, utilizzate l'X-Ray SDK per Node.js per strumentare il vostro AWS SDK nei client Node.js. JavaScript Ogni volta che effettui una chiamata a un downstream Servizio AWS o a una risorsa con un client dotato di strumenti, l'SDK registra le informazioni sulla chiamata in un sottosegmento. Servizi AWS e le risorse a cui accedi all'interno dei servizi vengono visualizzate come nodi a valle sulla mappa di traccia per aiutarti a identificare errori e problemi di limitazione sulle singole connessioni.

L'X-Ray SDK per Node.js fornisce anche la strumentazione per le chiamate downstream alle API Web HTTP e alle query SQL. Per memorizzare le informazioni sulle chiamate HTTP in uscita includi il client HTTP nel metodo di tracciamento dell'SDK. Nel caso di client SQL, utilizza il metodo di tracciamento per il tuo tipo di database.

Pr determinare quali richieste tracciare, il middleware applica le regole di campionamento alle richieste in entrata. È possibile configurare X-Ray SDK per Node.js per regolare il comportamento di campionamento o per registrare informazioni sulle risorse di AWS calcolo su cui viene eseguita l'applicazione.

Registra ulteriori informazioni sulle richieste e sull'attività svolta dalla tua applicazione in annotazioni e metadati. Le annotazioni sono semplici coppie chiave-valore indicizzati per l'uso con espressioni

filtro, in modo da poter cercare tracce che contengono dati specifici. Le immissioni di metadati sono meno restrittive e possono registrare interi oggetti e matrici, tutto ciò che può essere serializzato in JSON.

> **Annotazioni e metadata**
>
> Le annotazioni e i metadati sono testo arbitrario che aggiungi ai segmenti con X-Ray SDK. Le annotazioni vengono indicizzate per essere utilizzate con le espressioni di filtro. I metadati non sono indicizzati, ma possono essere visualizzati nel segmento non elaborato con la console X-Ray o l'API. Chiunque conceda l'accesso in lettura a X-Ray può visualizzare questi dati.

Quando disponi di una notevole quantità di client analizzati nel tuo codice, un singolo segmento per la richiesta può contenere un numero elevato di sottosegmenti, uno per ciascuna delle chiamate effettuate con un client analizzato. Puoi organizzare e raggruppare i sottosegmenti inglobando le chiamate del client sottosegmenti personalizzati. Puoi creare un sottosegmento personalizzato per un'intera funzione o qualsiasi porzione di codice, e memorizzare metadati e annotazioni sul sottosegmento invece di scrivere tutto all'interno del segmento padre.

Per la documentazione di riferimento sulle classi e i metodi dell'SDK, consulta l'AWS X-Ray SDK for Node.js API Reference.

Requisiti

L'X-Ray SDK per Node.js richiede Node.js e le seguenti librerie:

- `atomic-batcher`— 1.0.2
- `cls-hooked`— 4.2.2
- `pkginfo`— 0.4.0
- `semver`— 5.3.0

L'SDK preleva queste librerie durante la sua installazione con NPM.

Per tracciare i client AWS SDK, X-Ray SDK per Node.js richiede una versione minima AWS dell' JavaScript SDK per Node.js.

- `aws-sdk`— 2.7.15

Gestione delle dipendenze

L'X-Ray SDK per Node.js è disponibile presso NPM.

- Package — aws-xray-sdk

Per lo sviluppo in locale, installa l'SDK nella cartella del tuo progetto con 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
```

Utilizza l'opzione `--save` per salvare l'SDK come dipendenza nel file `package.json` della tua applicazione.

```
~/nodejs-xray$ npm install aws-xray-sdk --save
aws-xray-sdk@3.3.3
```

Se l'applicazione presenta dipendenze le cui versioni sono in conflitto con le dipendenze dell'SDK X-Ray, verranno installate entrambe le versioni per garantire la compatibilità. Per maggiori dettagli, consulta la documentazione ufficiale di NPM per la risoluzione delle dipendenze.

Esempi di Node.js

Utilizza l' AWS X-Ray SDK per Node.js per avere una end-to-end visione delle richieste mentre viaggiano attraverso le tue applicazioni Node.js.

- Applicazione di esempio Node.js attiva. GitHub

Configurazione dell'SDK X-Ray per Node.js

È possibile configurare X-Ray SDK per Node.js con plug-in per includere informazioni sul servizio su cui viene eseguita l'applicazione, modificare il comportamento di campionamento predefinito o aggiungere regole di campionamento che si applicano alle richieste a percorsi specifici.

Sections

- Plugin di servizio
- Regole di campionamento
- Registrazione
- Indirizzo del demone X-Ray
- Variabili di ambiente

Plugin di servizio

Utilizzatelo `plugins` per registrare informazioni sul servizio che ospita l'applicazione.

Plug-in

- Amazon EC2: `EC2Plugin` aggiunge l'ID dell'istanza, la zona di disponibilità e il gruppo di CloudWatch log.
- Elastic `ElasticBeanstalkPlugin` Beanstalk: aggiunge il nome dell'ambiente, l'etichetta della versione e l'ID di distribuzione.
- Amazon ECS: `ECSPlugin` aggiunge l'ID del contenitore.

Per utilizzare un plug-in, configura il client X-Ray SDK per Node.js utilizzando il metodo. `config`

Example app.js - Plugin

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.config([AWSXRay.plugins.EC2Plugin,AWSXRay.plugins.ElasticBeanstalkPlugin]);
```

L'SDK utilizza anche le impostazioni del plug-in per impostare il `origin` campo sul segmento. Indica il tipo di AWS risorsa che esegue l'applicazione. Quando utilizzi più plugin, l'SDK utilizza il seguente ordine di risoluzione per determinare l'origine: ElasticBeanstalk > EKS > ECS > EC2.

Regole di campionamento

L'SDK utilizza le regole di campionamento definite nella console X-Ray per determinare quali richieste registrare. La regola predefinita tiene traccia della prima richiesta ogni secondo e del cinque percento di eventuali richieste aggiuntive su tutti i servizi che inviano tracce a X-Ray. Crea regole aggiuntive nella console X-Ray per personalizzare la quantità di dati registrati per ciascuna delle tue applicazioni.

L'SDK applica le regole personalizzate nell'ordine in cui sono definite. Se una richiesta corrisponde a più regole personalizzate, l'SDK applica solo la prima regola.

> **Note**
>
> Se l'SDK non riesce a contattare X-Ray per ottenere le regole di campionamento, torna a una regola locale predefinita della prima richiesta ogni secondo e del cinque percento di eventuali richieste aggiuntive per host. Ciò può verificarsi se l'host non dispone dell'autorizzazione per chiamare le API di campionamento o non può connettersi al demone X-Ray, che funge da proxy TCP per le chiamate API effettuate dall'SDK.

Puoi anche configurare l'SDK per caricare le regole di campionamento da un documento JSON. L'SDK può utilizzare le regole locali come backup per i casi in cui il campionamento a raggi X non è disponibile o utilizzare esclusivamente regole locali.

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
  }
}
```

Questo esempio definisce una regola personalizzata e una regola predefinita. La regola personalizzata applica una frequenza di campionamento del cinque percento senza alcun numero minimo di richieste da tracciare per i percorsi. `/api/move/` La regola predefinita tiene traccia della prima richiesta ogni secondo e del 10% delle richieste aggiuntive.

Lo svantaggio della definizione locale delle regole è che l'obiettivo fisso viene applicato da ciascuna istanza del registratore in modo indipendente, anziché essere gestito dal servizio X-Ray. Man mano che si installano più host, la tariffa fissa si moltiplica, rendendo più difficile il controllo della quantità di dati registrati.

Sì AWS Lambda, non è possibile modificare la frequenza di campionamento. Se la funzione viene chiamata da un servizio strumentato, le chiamate che hanno generato richieste campionate da quel servizio verranno registrate da Lambda. Se il tracciamento attivo è abilitato e non è presente alcuna intestazione di tracciamento, Lambda prende la decisione di campionamento.

Per configurare le regole di backup, chiedi a X-Ray SDK per Node.js di caricare le regole di campionamento da un file con. `setSamplingRules`

Example app.js - regole di campionamento da un file

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.middleware.setSamplingRules('sampling-rules.json');
```

Puoi anche definire le regole nel codice e passarle a `setSamplingRules` come oggetto.

Example app.js - Regole di campionamento da un oggetto

```
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);
```

Per utilizzare solo le regole locali, chiama `disableCentralizedSampling`.

```
AWSXRay.middleware.disableCentralizedSampling()
```

Registrazione

Per generare i log sull'output dell'SDK, chiama `AWSXRay.setLogger(logger)`, dove `logger` è un oggetto che fornisce i metodi di log standard (`warn`, `info`, ecc.).

Per impostazione predefinita, l'SDK registrerà i messaggi di errore sulla console utilizzando i metodi standard sull'oggetto console. Il livello di registro del logger integrato può essere impostato utilizzando le variabili di `AWS_XRAY_LOG_LEVEL` ambiente `AWS_XRAY_DEBUG_MODE` o. Per un elenco di valori validi a livello di registro, vedete Variabili di ambiente.

Se desideri fornire un formato o una destinazione diversi per i log, puoi fornire all'SDK la tua implementazione dell'interfaccia del logger come mostrato di seguito. È possibile utilizzare qualsiasi oggetto che implementa questa interfaccia. Ciò significa che molte librerie di registrazione, ad esempio Winston, possono essere utilizzate e passate direttamente all'SDK.

Example app.js - logging

```
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]);
```

Chiama `setLogger` prima di eseguire altri metodi di configurazione per assicurarti che l'acquisizione dell'output di tali operazioni funzioni.

Indirizzo del demone X-Ray

Se il demone X-Ray è in ascolto su una porta o su un host diverso`127.0.0.1:2000`, è possibile configurare X-Ray SDK per Node.js per inviare i dati di traccia a un indirizzo diverso.

```
AWSXRay.setDaemonAddress('host:port');
```

Puoi specificare l'host in base al nome o all'indirizzo IPv4.

Example app.js - Indirizzo daemon

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.setDaemonAddress('daemonhost:8082');
```

Se hai configurato il daemon per ascoltare su diverse porte per TCP e UDP, puoi specificare entrambe nelle impostazioni dell'indirizzo del daemon.

Example app.js - Indirizzo del daemon su porte separate

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.setDaemonAddress('tcp:daemonhost:8082 udp:daemonhost:8083');
```

È inoltre possibile impostare l'indirizzo del demone utilizzando la variabile di ambiente. `AWS_XRAY_DAEMON_ADDRESS`

Variabili di ambiente

È possibile utilizzare le variabili di ambiente per configurare l'SDK X-Ray per Node.js. L'SDK supporta le seguenti variabili.

- `AWS_XRAY_CONTEXT_MISSING`— Imposta per `RUNTIME_ERROR` generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.

  Valori validi

  - `RUNTIME_ERROR`— Genera un'eccezione di runtime.
  - `LOG_ERROR`— Registra un errore e continua (impostazione predefinita).
  - `IGNORE_ERROR`— Ignora l'errore e continua.

  Gli errori relativi a segmenti o sottosegmenti mancanti possono verificarsi quando si tenta di utilizzare un client con strumenti nel codice di avvio che viene eseguito quando non è aperta alcuna richiesta o nel codice che genera un nuovo thread.
- `AWS_XRAY_DAEMON_ADDRESS`— Imposta l'host e la porta del demone X-Ray. Per impostazione predefinita, l'SDK utilizza `127.0.0.1:2000` sia i dati di traccia (UDP) che il campionamento

(TCP). Utilizzate questa variabile se avete configurato il demone per l'ascolto su una porta diversa o se è in esecuzione su un host diverso.

Formato

- Stessa porta — *address*:*port*
- Porte diverse: tcp:*address*:*port* udp:*address*:*port*

- AWS_XRAY_DEBUG_MODE— Imposta su TRUE per configurare l'SDK per inviare i log alla console, a livellodebug.
- AWS_XRAY_LOG_LEVEL — Imposta un livello di registro per il logger predefinito. I valori validi sono debug, info, warn, error e silent. Questo valore viene ignorato quando AWS_XRAY_DEBUG_MODE è impostato su TRUE.
- AWS_XRAY_TRACING_NAME— Imposta un nome di servizio che l'SDK utilizza per i segmenti. Sovrascrive il nome del segmento impostato sul middleware di Express.

Tracciamento delle richieste in arrivo con X-Ray SDK per Node.js

Puoi utilizzare l'SDK X-Ray per Node.js per tracciare le richieste HTTP in entrata che le tue applicazioni Express e Restify servono su un'istanza EC2 in Amazon EC2 o Amazon ECS. AWS Elastic Beanstalk

L'X-Ray SDK per Node.js fornisce middleware per applicazioni che utilizzano i framework Express e Restify. Quando aggiungete il middleware X-Ray all'applicazione, l'SDK X-Ray per Node.js crea un segmento per ogni richiesta campionata. Questo segmento include durata, metodo e conclusione della richiesta HTTP. Analisi ulteriori creano sottosegmenti associati a questo segmento.

> **Note**
>
> Per quanto riguarda AWS Lambda le funzioni, Lambda crea un segmento per ogni richiesta campionata. Per ulteriori informazioni, consulta AWS Lambda e AWS X-Ray.

Ogni segmento ha un nome che identifica l'applicazione nella mappa dei servizi. Il segmento può essere denominato staticamente oppure è possibile configurare l'SDK per denominarlo dinamicamente in base all'intestazione dell'host nella richiesta in entrata. La denominazione dinamica consente di raggruppare le tracce in base al nome di dominio nella richiesta e di applicare un nome predefinito se il nome non corrisponde a uno schema previsto (ad esempio, se l'intestazione dell'host è falsificata).

> **Richieste inoltrate**
>
> Se un sistema di bilanciamento del carico o un altro intermediario inoltra una richiesta all'applicazione, X-Ray prende l'IP del client dall'`X-Forwarded-For`intestazione della richiesta anziché dall'IP di origine nel pacchetto IP. L'IP del client registrato per una richiesta inoltrata può essere falsificato, quindi non dovrebbe essere considerato attendibile.

Quando viene inoltrata una richiesta, l'SDK imposta un campo aggiuntivo nel segmento per indicarlo. Se il segmento contiene il campo `x_forwarded_for` impostato su`true`, l'IP del client è stato preso dall'`X-Forwarded-For`intestazione della richiesta HTTP.

Il gestore dei messaggi crea un segmento per ogni richiesta in entrata con un blocco `http` che contiene le informazioni riportate qui di seguito:

- Metodo HTTP: GET, POST, PUT, DELETE, ecc.
- Indirizzo client: l'indirizzo IP del client che ha inviato la richiesta.
- Codice di risposta: il codice di risposta HTTP per la richiesta completata.
- Tempistica: l'ora di inizio (quando è stata ricevuta la richiesta) e l'ora di fine (quando è stata inviata la risposta).
- Agente utente: il `user-agent` codice della richiesta.
- Lunghezza del contenuto: il `content-length` risultato della risposta.

Sections

- Tracciamento delle richieste in entrata con Express
- Tracciamento delle richieste in entrata con Restify
- Configurazione di una strategia di denominazione dei segmenti

Tracciamento delle richieste in entrata con Express

Per usare il middleware Express, inizializzare il client SDK e utilizzare il middleware restituito dalla funzione `express.openSegment` prima di definire gli instradamenti.

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());
```

Dopo aver definito i percorsi, utilizzate l'output di `express.closeSegment` come mostrato per gestire eventuali errori restituiti da X-Ray SDK per Node.js.

Tracciamento delle richieste in entrata con Restify

Per usare il middleware Restify, inizializzare il client SDK client e eseguire `enable`. Passare ad esso il server Restify e il nome del segmento.

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');
});
```

Configurazione di una strategia di denominazione dei segmenti

AWS X-Ray utilizza un nome di servizio per identificare l'applicazione e distinguerla dalle altre applicazioni, database, API esterne e AWS risorse utilizzate dall'applicazione. Quando X-Ray SDK genera segmenti per le richieste in entrata, registra il nome del servizio dell'applicazione nel campo del nome del segmento.

L'X-Ray SDK può denominare i segmenti dopo il nome host nell'intestazione della richiesta HTTP. Tuttavia, questa intestazione può essere falsificata, il che potrebbe causare nodi imprevisti nella mappa dei servizi. Per evitare che l'SDK nomini i segmenti in modo errato a causa di richieste con intestazioni host contraffatte, è necessario specificare un nome predefinito per le richieste in entrata.

Se la tua applicazione soddisfa le richieste per più domini, puoi configurare l'SDK in modo che utilizzi una strategia di denominazione dinamica che rifletta questo aspetto nei nomi dei segmenti. Una strategia di denominazione dinamica consente all'SDK di utilizzare il nome host per le richieste che corrispondono a uno schema previsto e di applicare il nome predefinito alle richieste che non lo fanno.

Ad esempio, potresti avere una singola applicazione che serve le richieste a tre sottodomini:, e. `www.example.com` `api.example.com` `static.example.com` È possibile utilizzare una strategia di denominazione dinamica con lo schema `*.example.com` per identificare i segmenti per ogni sottodominio con un nome diverso, ottenendo tre nodi di servizio sulla mappa dei servizi. Se l'applicazione riceve richieste con un nome host che non corrisponde allo schema, sulla mappa dei servizi verrà visualizzato un quarto nodo con un nome di fallback specificato dall'utente.

Per utilizzare lo stesso nome per tutti i segmenti della richiesta, specifica il nome della tua applicazione quando inizializzi il middleware, come illustrato nelle sezioni precedenti.

> **Note**
>
> È possibile sovrascrivere il nome di servizio predefinito definito nel codice con la `AWS_XRAY_TRACING_NAME` variabile di ambiente.

Una strategia di denominazione dinamica definisce un modello al quale devono corrispondere i nomi degli host e un nome di default per l'utilizzo qualora il nome dell'host nella richiesta HTTP non corrisponda al modello. Per denominare i segmenti in modo dinamico, utilizza `AWSXRay.middleware.enableDynamicNaming`.

Example app.js - Nomi dei segmenti dinamici

Se il nome host nella richiesta corrispondente al modello `*.example.com`, utilizza il nome dell'host. In caso contrario, utilizzare `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());
```

Tracciamento delle chiamate AWS SDK con X-Ray SDK per Node.js

Quando l'applicazione effettua chiamate per Servizi AWS archiviare dati, scrivere in una coda o inviare notifiche, X-Ray SDK per Node.js tiene traccia delle chiamate a valle in sottosegmenti. Le risorse tracciate Servizi AWS e a cui accedi all'interno di tali servizi (ad esempio, un bucket Amazon S3 o una coda Amazon SQS) vengono visualizzate come nodi downstream sulla mappa di traccia nella console X-Ray.

Client Instrument AWS SDK creati tramite V2 o V3.AWS SDK for JavaScriptAWS SDK for JavaScript Ogni versione AWS SDK offre metodi diversi per la strumentazione dei client SDK. AWS

> **Note**
>
> Attualmente, l' AWS X-Ray SDK per Node.js restituisce meno informazioni sui segmenti durante la strumentazione dei client AWS SDK for JavaScript V3, rispetto alla strumentazione dei client V2. Ad esempio, i sottosegmenti che rappresentano le chiamate a DynamoDB non restituiranno il nome della tabella. Se hai bisogno di queste informazioni sui segmenti nelle tue tracce, prendi in considerazione l'utilizzo della versione V2. AWS SDK for JavaScript

AWS SDK for JavaScript V2

Puoi utilizzare tutti i client AWS SDK V2 inserendo l'istruzione `aws-sdk` require in una chiamata a. `AWSXRay.captureAWS`

Example app.js - Strumentazione SDK AWS

```
const AWS = AWSXRay.captureAWS(require('aws-sdk'));
```

Per strumentare i singoli client, inserisci il tuo client AWS SDK in una chiamata a. `AWSXRay.captureAWSClient` Ad esempio, per analizzare un client `AmazonDynamoDB`:

Example app.js - Strumentazione client DynamoDB

```
    const AWSXRay = require('aws-xray-sdk');
```

```
...
    const ddb = AWSXRay.captureAWSClient(new AWS.DynamoDB());
```

> **Warning**
>
> Non utilizzare `captureAWS` e `captureAWSClient` insieme. Ciò causerà sottosegmenti duplicati.

Se desideri utilizzarla TypeScriptcon i moduli ECMAScript (ESM) per caricare il JavaScript codice, usa il seguente esempio per importare le librerie:

Example app.js - strumentazione SDK AWS

```
import * as AWS from 'aws-sdk';
import * as AWSXRay from 'aws-xray-sdk';
```

Per strumentare tutti i AWS client con ESM, usa il seguente codice:

Example app.js - Strumentazione AWS SDK

```
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();
```

Per tutti i servizi, puoi vedere il nome dell'API richiamata nella console X-Ray. Per un sottoinsieme di servizi, l'SDK X-Ray aggiunge informazioni al segmento per fornire una maggiore granularità nella mappa dei servizi.

Ad esempio, quando si effettua una chiamata con un client DynamoDB con strumentazione, l'SDK aggiunge il nome della tabella al segmento per le chiamate destinate a una tabella. Nella console, ogni tabella appare come un nodo separato nella mappa dei servizi, con un nodo DynamoDB generico per le chiamate che non hanno come destinazione una tabella.

Example Sottosegmento per una chiamata a DynamoDB per salvare un elemento

```
{
```

```
  "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",
  }
}
```

Quando si accede alle risorse con nome, le chiamate ai seguenti servizi creano ulteriori nodi della mappa del servizio. Le chiamate che non sono hanno come obiettivo risorse specifiche creano un nodo generico per il servizio.

- Amazon DynamoDB: nome della tabella
- Amazon Simple Storage Service: nome del bucket e della chiave
- Amazon Simple Queue Service: nome della coda

AWS SDK for JavaScript V3

La AWS SDK for JavaScript V3 è modulare, quindi il codice carica solo i moduli necessari. Per questo motivo, non è possibile strumentare tutti i client AWS SDK poiché la V3 non supporta il metodo. `captureAWS`

Se desideri utilizzarlo TypeScript con i moduli ECMAScript (ESM) per caricare il JavaScript codice, puoi utilizzare il seguente esempio per importare le librerie:

```
import * as AWS from 'aws-sdk';
import * as AWSXRay from 'aws-xray-sdk';
```

Strumenta ogni client AWS SDK utilizzando il metodo. `AWSXRay.captureAWSv3Client` Ad esempio, per analizzare un client `AmazonDynamoDB`:

Example app.js - Strumentazione client DynamoDB che utilizza SDK per Javascript V3

```
    const AWSXRay = require('aws-xray-sdk');
    const { DynamoDBClient } = require("@aws-sdk/client-dynamodb");
...
    const ddb = AWSXRay.captureAWSv3Client(new DynamoDBClient({ region:
 "region" }));
```

Quando si utilizza AWS SDK for JavaScript V3, i metadati come il nome della tabella, il nome del bucket e della chiave o il nome della coda non vengono attualmente restituiti, pertanto la mappa di traccia non conterrà nodi discreti per ogni risorsa denominata, come accade quando si strumentano i client SDK utilizzando la V2. AWS AWS SDK for JavaScript

Example Sottosegmento per una chiamata a DynamoDB per salvare un elemento, quando si utilizza la V3 AWS SDK for JavaScript

```
{
  "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",
  }
}
```

Tracciamento delle chiamate ai servizi Web HTTP downstream utilizzando l'SDK X-Ray per Node.js

Quando l'applicazione effettua chiamate a microservizi o API HTTP pubbliche, è possibile utilizzare il client X-Ray SDK for Node.js per strumentare tali chiamate e aggiungere l'API al grafico del servizio come servizio downstream.

Passa il tuo `http` `https` client al `captureHTTPs` metodo X-Ray SDK for Node.js per tracciare le chiamate in uscita.

> **Note**
>
> Le chiamate che utilizzano librerie di richieste HTTP di terza parte, ad esempio Axios o Superagent, sono supportate tramite l'API `captureHTTPsGlobal()` e verranno comunque tracciate quando utilizzano il modulo `http` nativo.

Example app.js - Client HTTP

```
var AWSXRay = require('aws-xray-sdk');
var http = AWSXRay.captureHTTPs(require('http'));
```

Per abilitare il tracciamento su tutti i client HTTP, chiama `captureHTTPsGlobal` prima di caricare `http`.

Example app.js - Client HTTP (Globale)

```
var AWSXRay = require('aws-xray-sdk');
AWSXRay.captureHTTPsGlobal(require('http'));
var http = require('http');
```

Quando si effettua una chiamata a un'API Web downstream, l'SDK X-Ray per Node.js registra un sottosegmento che contiene informazioni sulla richiesta e sulla risposta HTTP. X-Ray utilizza il sottosegmento per generare un segmento dedotto per l'API remota.

Example Sottosegmento per una chiamata HTTP a valle

```
{
  "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 Segmento dedotto per una chiamata HTTP a valle

```
{
  "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
}
```

Tracciamento delle query SQL con X-Ray SDK per Node.js

Strumenta le query del database SQL inserendo il client SQL nel metodo client X-Ray SDK for Node.js corrispondente.

- 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);
```

Quando utilizzi un client analizzato per eseguire query SQL, X-Ray SDK per Node.js memorizza le informazioni sulla connessione e sulla query in un sottosegmento

Includere dati aggiuntivi nei sottosegmenti SQL

È possibile aggiungere informazioni aggiuntive ai sottosegmenti generati per le query SQL, purché siano mappate su un campo SQL consentito. Ad esempio, per registrare la stringa di query SQL sanificata in un sottosegmento, è possibile aggiungerla direttamente all'oggetto SQL del sottosegmento.

Example Assegna SQL al sottosegmento

```
    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;
}
```

Per un elenco completo dei campi SQL consentiti, consulta SQL Queries nella Developer Guide.AWS X-Ray

Generazione di sottosegmenti personalizzati con X-Ray SDK per Node.js

I sottosegmenti estendono il segmento di una traccia con dettagli sul lavoro svolto per soddisfare una richiesta. Ogni volta che si effettua una chiamata con un client dotato di strumentazione, l'X-Ray SDK registra le informazioni generate in un sottosegmento. È possibile creare sottosegmenti aggiuntivi per raggruppare altri sottosegmenti, misurare le prestazioni di una sezione di codice o registrare annotazioni e metadati.

Sottosegmenti Express personalizzati

Per creare un sottosegmento personalizzato per una funzione che effettua chiamate a servizi a valle, utilizza la funzione `captureAsyncFunc`.

Example app.js - Sottosegmenti personalizzati 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 questo esempio, l'applicazione crea un sottosegmento personalizzato denominato `send` per le chiamate alla funzione `sendRequest`. `captureAsyncFunc` trasferisce un sottosegmento che è necessario chiudere all'interno della funzione di callback quando le chiamate asincrone invocate vengono completate.

Per le funzioni sincrone, puoi utilizzare la funzione `captureFunc` che chiude il sottosegmento automaticamente non appena il blocco della funzione si conclude.

Quando si crea un sottosegmento all'interno di un segmento o di un altro sottosegmento, X-Ray SDK per Node.js genera un ID per tale sottosegmento e registra l'ora di inizio e l'ora di fine.

Example Sottosegmento con metadati

```
"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"
    }
  },
```

Sottosegmenti Lambda personalizzati

L'SDK è configurato per creare automaticamente un segmento di facciata segnaposto quando rileva che è in esecuzione in Lambda. Per creare un sottosegmento di base, che creerà un singolo `AWS::Lambda::Function` nodo sulla mappa di tracciamento a raggi X, richiamate e riutilizzate il segmento di facciata. Se crei manualmente un nuovo segmento con un nuovo ID (mentre condividi l'ID di tracciamento, l'ID padre e la decisione di campionamento), sarai in grado di inviare un nuovo segmento.

Example app.js - sottosegmenti personalizzati manualmente

```
const segment = AWSXRay.getSegment(); //returns the facade segment
const subsegment = segment.addNewSubsegment('subseg');
...
```

```
subsegment.close();
//the segment is closed by the SDK automatically
```

Aggiungi annotazioni e metadati ai segmenti con l'SDK X-Ray per Node.js

È possibile utilizzare annotazioni e metadati per registrare informazioni aggiuntive sulle richieste, sull'ambiente o sull'applicazione. Puoi aggiungere annotazioni e metadati ai segmenti creati da X-Ray SDK o ai sottosegmenti personalizzati che crei.

Le annotazioni sono coppie chiave-valore con stringhe, numeri o valori booleani. Le annotazioni sono indicizzate per essere utilizzate con le espressioni di filtro. Utilizzale per registrare i dati che desideri utilizzare per raggruppare le tracce nella console oppure per chiamare l'API `GetTraceSummaries`.

I metadati sono coppie chiave-valore che possono avere valori di qualsiasi tipo, inclusi oggetti ed elenchi, ma non sono indicizzati per essere utilizzati con le espressioni di filtro. Utilizzate i metadati per registrare dati aggiuntivi che desiderate archiviare nella traccia ma che non è necessario utilizzare con la ricerca.

Oltre ad annotazioni e metadati, sui segmenti puoi anche registrare le stringhe degli ID utente. Gli ID utente vengono memorizzati in un campo separato su segmenti e sono indicizzati per l'uso nelle ricerche.

Sections

- Registrazione delle annotazioni con X-Ray SDK per Node.js
- Registrazione di metadati con X-Ray SDK per Node.js
- Registrazione degli ID utente con X-Ray SDK per Node.js

Registrazione delle annotazioni con X-Ray SDK per Node.js

Utilizza le annotazioni per memorizzare le informazioni su segmenti o sottosegmenti che desideri siano indicizzate per la ricerca.

Requisiti per le annotazioni

- Chiavi: la chiave per un'annotazione a raggi X può contenere fino a 500 caratteri alfanumerici. Non è possibile utilizzare spazi o simboli diversi dal simbolo di sottolineatura (_).
- Valori: il valore di un'annotazione X-Ray può contenere fino a 1.000 caratteri Unicode.
- Il numero di annotazioni: è possibile utilizzare fino a 50 annotazioni per traccia.

Per registrare le annotazioni

1. Ottenere un riferimento al segmento o sottosegmento corrente.

```
var AWSXRay = require('aws-xray-sdk');
...
var document = AWSXRay.getSegment();
```

2. Chiamare `addAnnotation` con una chiave di tipo Stringa e un valore booleano, numerico o di tipo Stringa.

```
document.addAnnotation("mykey", "my value");
```

L'SDK memorizza le annotazioni come coppie chiave-valore in un oggetto `annotations` all'interno del documento di segmento. Se chiami `addAnnotation` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento o sottosegmento viene sovrascritto.

Per trovare tracciamenti con annotazioni contenenti valori specifici, utilizza la parola chiave `annotations.`*`key`* in un'espressione filtro.

Example app.js - Annotazioni

```
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) {
...
```

Registrazione di metadati con X-Ray SDK per Node.js

Utilizza i metadati per memorizzare le informazioni su segmenti o sottosegmenti che non è necessario che siano indicizzate per la ricerca. I valori dei metadati possono essere stringhe, numeri, valori booleani o qualsiasi altro oggetto che può essere serializzato in un oggetto o in un vettore JSON.

Per registrare i metadati

1. Ottenere un riferimento al segmento o sottosegmento corrente.

   ```
   var AWSXRay = require('aws-xray-sdk');
   ...
   var document = AWSXRay.getSegment();
   ```

2. Chiamare `addMetadata` con una chiave di tipo Stringa, un valore booleano, numerico, di tipo Stringa o di tipo oggetto e un namespace di tipo stringa.

   ```
   document.addMetadata("my key", "my value", "my namespace");
   ```

   oppure

   Invocare `addMetadata` semplicemente con una chiave e un valore.

   ```
   document.addMetadata("my key", "my value");
   ```

Se non specifichi un namespace, l'SDK utilizza `default`. Se chiami `addMetadata` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento o sottosegmento viene sovrascritto.

Registrazione degli ID utente con X-Ray SDK per Node.js

Memorizza gli ID utente sui segmenti di richiesta per identificare l'utente che ha inviato la richiesta. Questa operazione non è compatibile con AWS Lambda le funzioni perché i segmenti negli ambienti Lambda sono immutabili. La chiamata `setUser` può essere applicata solo ai segmenti, non ai sottosegmenti.

Per registrare gli ID degli utenti

1. Ottenere un riferimento al segmento o sottosegmento corrente.

```
var AWSXRay = require('aws-xray-sdk');
...
var document = AWSXRay.getSegment();
```

2. Chiamare `setUser()` con una stringa che rappresenta l'ID dell'utente che ha inviato la richiesta.

```
var user = 'john123';

AWSXRay.getSegment().setUser(user);
```

È possibile chiamare `setUser` per registrare l'ID utente non appena l'applicazione Express inizia l'elaborazione di una richiesta. Se usi il segmento per impostare l'ID utente, puoi concatenare le chiamate in un'unica riga.

Example app.js - ID utente

```
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) {
...
```

Per trovare tracciamenti associati ad un ID utente, utilizza la parola chiave `user` in un'espressione filtro.

Strumentate la vostra applicazione con Python

Esistono due modi per strumentare Python l'applicazione per inviare tracce a X-Ray:

- AWS Distro for OpenTelemetry Python: una AWS distribuzione che fornisce un set di librerie open source per l'invio di metriche e tracce correlate a più soluzioni di AWS monitoraggio, tra cui Amazon AWS X-Ray e Amazon OpenSearch Service CloudWatch, tramite AWS Distro for Collector. OpenTelemetry
- AWS X-Ray SDK per Python: un set di librerie per la generazione e l'invio di tracce a X-Ray tramite il demone X-Ray.

Per ulteriori informazioni, consulta Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray.

AWS Distro per OpenTelemetry Python

Con AWS Distro for OpenTelemetry (ADOT)Python, puoi strumentare le tue applicazioni una sola volta e inviare metriche e tracce correlate a più AWS soluzioni di monitoraggio tra cui Amazon AWS X-Ray e Amazon CloudWatch Service. OpenSearch L'utilizzo di X-Ray con ADOT richiede due componenti: un OpenTelemetry SDK abilitato per l'uso con X-Ray e AWS Distro for OpenTelemetry Collector abilitato per l'uso con X-Ray. ADOT Python include il supporto per la strumentazione automatica, che consente all'applicazione di inviare tracce senza modifiche al codice.

Per iniziare, consulta la Distro per la AWS documentazione. OpenTelemetry Python

Per ulteriori informazioni sull'utilizzo di AWS Distro for OpenTelemetry AWS X-Ray e altro Servizi AWS, consulta AWS Distro for OpenTelemetry o Distro for DocumentationAWS . OpenTelemetry

Per ulteriori informazioni sul supporto e l'utilizzo della lingua, vedere AWS Observability on. GitHub

AWS X-Ray SDK per Python

X-Ray SDK for Python è una libreria per applicazioni Python Web che fornisce classi e metodi per generare e inviare dati di traccia al demone X-Ray. I dati di traccia includono informazioni sulle richieste HTTP in entrata fornite dall'applicazione e sulle chiamate effettuate dall'applicazione ai servizi downstream utilizzando l' AWS SDK, i client HTTP o un connettore di database SQL. Puoi

anche possibile creare manualmente dei segmenti e aggiungere informazioni di debug in annotazioni e metadati.

Puoi scaricare l'SDK tramite `pip`.

```
$ pip install aws-xray-sdk
```

> **Note**
>
> L'X-Ray SDK for Python è un progetto open source. Puoi seguire il progetto e inviare problemi e richieste di pull su: github.com/aws/ GitHub aws-xray-sdk-python

Se utilizzi Django o Flask, per tracciare le richieste in entrata inizia aggiungendo il middleware dell'SDK alla tua applicazione. Il middleware crea un segmento per ogni richiesta tracciata e completa il segmento quando viene inviata la risposta. Fino a che il segmento è aperto, puoi usare i metodi del client dell'SDK per aggiungere informazioni al segmento e creare sottosegmenti per tracciare le chiamate a valle. L'SDK, inoltre, registra automaticamente le eccezioni sollevate dall'applicazione per il tempo durante il quale il segmento è aperto. Per altre applicazioni, puoi creare i segmenti manualmente.

Per le funzioni Lambda richiamate da un'applicazione o un servizio strumentato, Lambda legge l'intestazione di tracciamento e traccia automaticamente le richieste campionate. Per altre funzioni, puoi configurare Lambda per campionare e tracciare le richieste in arrivo. In entrambi i casi, Lambda crea il segmento e lo fornisce all'X-Ray SDK.

> **Note**
>
> Su Lambda, l'SDK X-Ray è opzionale. Se non lo usi nella tua funzione, la mappa dei servizi includerà comunque un nodo per il servizio Lambda e uno per ogni funzione Lambda. Aggiungendo l'SDK, puoi utilizzare il codice della funzione per aggiungere sottosegmenti al segmento di funzione registrato da Lambda. Per ulteriori informazioni, consulta AWS Lambda e AWS X-Ray.

Vedi un Worker esempio di Python funzione strumentata in Lambda.

Successivamente, usa l'X-Ray SDK per Python per strumentare le chiamate downstream applicando patch alle librerie dell'applicazione. L'SDK supporta le seguenti librerie.

Librerie supportate

- `botocore`, — Client per strumenti. `boto3` AWS SDK for Python (Boto)
- `pynamodb`— La versione del client Amazon DynamoDB di Instrument PynamoDB.
- `aiobotocore`, `aioboto3` — Versioni integrate di Instrument Asyncio di SDK per client Python.
- `requests`, `aiohttp` — Client HTTP di alto livello dello strumento.
- `httplib`, `http.client`— I client HTTP di basso livello dello strumento e le librerie di livello superiore che li utilizzano.
- `sqlite3`— Client SQLite di Instrument.
- `mysql-connector-python`— Client MySQL di Instrument.
- `pg8000`— Interfaccia PostgreSQL dello strumento Pure-Python.
- `psycopg2`— Adattatore per database PostgreSQL Instrument.
- `pymongo`— Client Instrument MongoDB.
- `pymysql`— Client basati su Instrument PyMy SQL per MySQL e MariaDB.

Ogni volta che l'applicazione effettua chiamate a AWS un database SQL o ad altri servizi HTTP, l'SDK registra le informazioni sulla chiamata in un sottosegmento. Servizi AWS e le risorse a cui accedi all'interno dei servizi vengono visualizzate come nodi a valle sulla mappa di traccia per aiutarti a identificare errori e problemi di limitazione sulle singole connessioni.

Dopo aver iniziato a utilizzare l'SDK, personalizzane il comportamento configurando il registratore e il middleware. Puoi aggiungere dei plugin per memorizzare i dati sulle risorse di elaborazione sulle quali è eseguita la tua applicazione, personalizzare il comportamento di campionamento definendo regole di campionatura e impostare il livello di log per visualizzare più o meno informazioni generate dall'SDK nei log dell'applicazione.

Registra ulteriori informazioni sulle richieste e sull'attività svolta dalla tua applicazione in annotazioni e metadati. Le annotazioni sono semplici coppie chiave-valore indicizzati per l'uso con espressioni filtro, in modo da poter cercare tracce che contengono dati specifici. Le immissioni di metadati sono meno restrittive e possono registrare interi oggetti e array, ovvero tutto ciò che può essere serializzato in JSON.

> **Annotazioni e metadata**
>
> Le annotazioni e i metadati sono testo arbitrario che aggiungi ai segmenti con X-Ray SDK. Le annotazioni vengono indicizzate per essere utilizzate con le espressioni di filtro. I metadati

non sono indicizzati, ma possono essere visualizzati nel segmento non elaborato con la console X-Ray o l'API. Chiunque conceda l'accesso in lettura a X-Ray può visualizzare questi dati.

Quando disponi di una notevole quantità di client analizzati nel tuo codice, un singolo segmento per la richiesta può contenere un numero elevato di sottosegmenti, uno per ciascuna delle chiamate effettuate con un client analizzato. Puoi organizzare e raggruppare i sottosegmenti inglobando le chiamate del client sottosegmenti personalizzati. Puoi creare un sottosegmento personalizzato per un'intera funzione o per qualsiasi parte del codice. Puoi quindi memorizzare metadati e annotazioni nel sottosegmento invece di scrivere tutto all'interno del segmento padre.

Per la documentazione di riferimento per le classi e i metodi dell'SDK, consulta AWS X-Ray SDK for Python API Reference.

Requisiti

L'X-Ray SDK per Python supporta le seguenti versioni di linguaggio e libreria.

- Python — 2.7, 3.4 e versioni successive
- Django — 1.10 e versioni successive
- Flask — 0.10 e versioni successive
- aiohttp — 2.3.0 e versioni successive
- AWS SDK for Python (Boto)— 1.4.0 e versioni successive
- botocore — 1.5.0 e versioni successive
- enum — 0.4.7 e versioni successive, per le versioni 3.4.0 e precedenti Python
- jsonpickle — 1.0.0 e versioni successive
- setuptools — 40.6.3 e versioni successive
- wrapt — 1.11.0 e versioni successive

Gestione delle dipendenze

L'X-Ray SDK per Python è disponibile da. `pip`

- Package — `aws-xray-sdk`

Aggiungi l'SDK come dipendenza nel file `requirements.txt`.

Example requirements.txt

```
aws-xray-sdk==2.4.2
boto3==1.4.4
botocore==1.5.55
Django==1.11.3
```

Se utilizzi Elastic Beanstalk per distribuire l'applicazione, Elastic Beanstalk installa automaticamente tutti i pacchetti. `requirements.txt`

Configurazione dell'SDK X-Ray per Python

l'SDK X-Ray per Python ha una classe denominata`xray_recorder` che fornisce il registratore globale. Puoi configurare il registratore globale per personalizzare il middleware che crea i segmenti relativi alle chiamate HTTP in entrata.

Sezioni

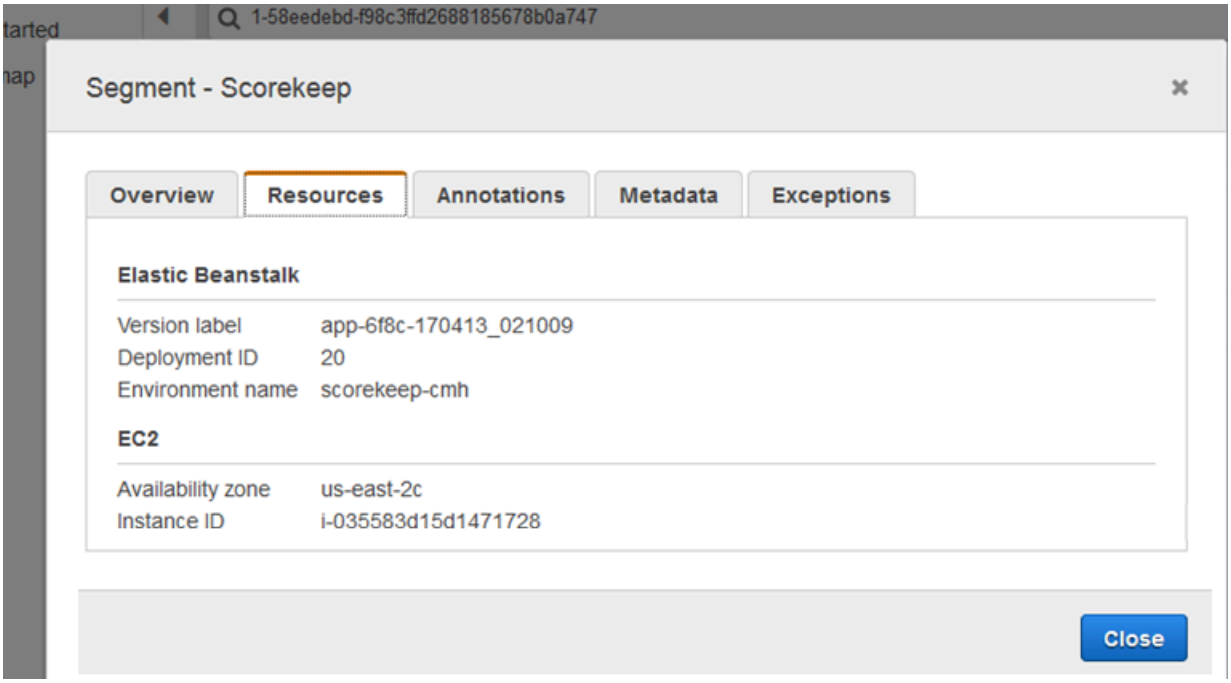
- Plugin di servizio
- Regole di campionamento
- Registrazione
- Configurazione del registratore nel codice
- Configurazione del registratore con Django
- Variabili di ambiente

Plugin di servizio

`plugins`Da utilizzare per registrare informazioni sul servizio che ospita l'applicazione.

Plug-in

- Amazon EC2:`EC2Plugin` aggiunge l'ID dell'istanza, la zona di disponibilità e il gruppo di CloudWatch log.
- Elastic Beanstalk:`ElasticBeanstalkPlugin` aggiunge il nome dell'ambiente, l'etichetta della versione e l'ID di distribuzione.
- Amazon ECS:`ECSPlugin` aggiunge l'ID del contenitore.



Per usare un plugin, chiama `configure` sul tuo `xray_recorder`.

```
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**
>
> Poiché`plugins` vengono passati come tupla, assicurati di includere un finale`,` quando specifichi un singolo plugin. Ad esempio, `plugins = ('EC2Plugin',)`

Per configurare il registratore puoi anche utilizzare le variabili di ambiente, che hanno la precedenza sui valori impostati nel codice.

Configura i plugin prima applicare le patch alle librerie per memorizzare le chiamate a valle.

L'SDK utilizza anche le impostazioni del plugin per impostare il`origin` campo sul segmento. Indica il tipo diAWS risorsa che esegue l'applicazione. Quando si utilizzano più plugin, l'SDK utilizza il seguente ordine di risoluzione per determinare l'origine: ElasticBeanstalk > EKS > ECS > EC2.

Regole di campionamento

L'SDK utilizza le regole di campionamento definite nella console X-Ray per determinare quali richieste registrare. La regola predefinita tiene traccia della prima richiesta ogni secondo e del cinque percento di tutte le richieste aggiuntive su tutti i servizi che inviano tracce a X-Ray. Crea regole aggiuntive nella console X-Ray per personalizzare la quantità di dati registrati per ciascuna delle tue applicazioni.

L'SDK applica le regole personalizzate nell'ordine in cui sono definite. Se una richiesta corrisponde a più regole personalizzate, l'SDK applica solo la prima regola.

> **Note**
>
> Se l'SDK non riesce a raggiungere X-Ray per ottenere le regole di campionamento, torna a una regola locale predefinita della prima richiesta ogni secondo e del cinque percento di eventuali richieste aggiuntive per host. Ciò può verificarsi se l'host non dispone dell'autorizzazione per chiamare le API di campionamento o non riesce a connettersi al demone X-Ray, che funge da proxy TCP per le chiamate API effettuate dall'SDK.

Puoi anche configurare l'SDK per caricare le regole di campionamento da un documento JSON. L'SDK può utilizzare regole locali come backup nei casi in cui il campionamento a X-Ray non è disponibile o utilizzare esclusivamente regole locali.

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
  }
}
```

Questo esempio definisce una regola personalizzata e una regola predefinita. La regola personalizzata applica una frequenza di campionamento del cinque percento senza un numero minimo di richieste da tracciare per i percorsi`/api/move/`. La regola predefinita traccia la prima richiesta ogni secondo e il 10 percento delle altre richieste.

Lo svantaggio della definizione locale delle regole è che l'obiettivo fisso viene applicato da ciascuna istanza del registratore in modo indipendente, anziché essere gestito dal servizio X-Ray. Man mano che si installano più host, la velocità fissa si moltiplica, rendendo più difficile il controllo della quantità di dati registrati.

SìAWS Lambda, non è possibile modificare la frequenza di campionamento. Se la funzione viene chiamata da un servizio strumentato, le chiamate che hanno generato richieste campionate da quel servizio verranno registrate da Lambda. Se il tracciamento attivo è abilitato e non è presente un'intestazione di tracciamento, Lambda prende la decisione di campionamento.

Per configurare le regole di campionamento di backup, chiama `xray_recorder.configure`, come nell'esempio seguente, dove *`regole`* è un dizionario di regole o il percorso assoluto di un file JSON che contiene le regole di campionamento.

```
xray_recorder.configure(sampling_rules=rules)
```

Per utilizzare solo regole locali, configura un registratore con una `LocalSampler`.

```
from aws_xray_sdk.core.sampling.local.sampler import LocalSampler
xray_recorder.configure(sampler=LocalSampler())
```

Puoi anche configurare il registratore globale per disabilitare il campionamento e analizzare tutte le richieste in entrata.

Example main.py — Disabilita il campionamento

```
xray_recorder.configure(sampling=False)
```

Registrazione

L'SDK utilizza il`logging` modulo integrato di Python con un livello di`WARNING` registrazione predefinito. Ottieni un riferimento al logger per la classe `aws_xray_sdk` e chiama `setLevel` per configurare il livello di log differenti per la libreria e il resto dell'applicazione.

Example app.py — Registrazione

```
logging.basicConfig(level='WARNING')
logging.getLogger('aws_xray_sdk').setLevel(logging.ERROR)
```

Utilizza i log di debug per identificare i problemi, come ad esempio dei sottosegmenti non chiusi, quando generi dei sottosegmenti manualmente.

Configurazione del registratore nel codice

Impostazioni aggiuntive sono disponibili tramite il metodo `configure` su `xray_recorder`.

- `context_missing`— ImpostatoLOG_ERROR per evitare di generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.
- `daemon_address`— Imposta l'host e la porta del listener del demone X-Ray.
- `service`— Imposta un nome di servizio che l'SDK utilizza per i segmenti.
- `plugins`— Registra informazioni sulleAWS risorse dell'applicazione.
- `sampling`— Impostare su`False` per disabilitare il campionamento.
- `sampling_rules`— Imposta il percorso del file JSON contenente le regole di campionamento.

Example main.py — Disabilita le eccezioni mancanti del contesto

```
from aws_xray_sdk.core import xray_recorder

xray_recorder.configure(context_missing='LOG_ERROR')
```

Configurazione del registratore con Django

Se utilizzi il framework Django, puoi utilizzare il file `settings.py` di Django per configurare le opzioni del registratore globale.

- `AUTO_INSTRUMENT`(Solo Django): registra i sottosegmenti per le operazioni di rendering integrate del database e dei modelli.

- `AWS_XRAY_CONTEXT_MISSING`— Impostato`LOG_ERROR` per evitare di generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.
- `AWS_XRAY_DAEMON_ADDRESS`— Imposta l'host e la porta del listener del demone X-Ray.
- `AWS_XRAY_TRACING_NAME`— Imposta un nome di servizio che l'SDK utilizza per i segmenti.
- `PLUGINS`— Registra informazioni sulleAWS risorse dell'applicazione.
- `SAMPLING`— Impostare su`False` per disabilitare il campionamento.
- `SAMPLING_RULES`— Imposta il percorso del file JSON contenente le regole di campionamento.

Per abilitare la configurazione del registratore in `settings.py`, aggiungi il middleware Django all'elenco delle applicazioni.

Example settings.py — App installate

```
INSTALLED_APPS = [
    ...
    'django.contrib.sessions',
    'aws_xray_sdk.ext.django',
]
```

Configura le impostazioni disponibili in un dizionario denominato `XRAY_RECORDER`.

Example settings.py — App installate

```
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,
}
```

Variabili di ambiente

Puoi utilizzare le variabili di ambiente per Python X-Ray per Python. L'SDK supporta le seguenti variabili:

- `AWS_XRAY_TRACING_NAME`— Imposta un nome di servizio che l'SDK utilizza per i segmenti. Sostituisce il nome del servizio impostato a livello di programmazione.

- `AWS_XRAY_SDK_ENABLED`— Se impostato su`false`, disabilita l'SDK. Per impostazione predefinita, l'SDK è abilitato solo se la variabile di ambiente è impostata su false.
  - Quando l'SDK è disabilitato, il registratore globale genera automaticamente segmenti o sottosegmenti fittizi che non vengono inviati al daemon e l'applicazione automatica di patch è disattivata. I middleware vengono scritti come wrapper per il registratore globale. Anche tutti i segmenti e sottosegmenti generati tramite il middleware diventano segmenti e sottosegmenti fittizi.
  - Imposta il valore di `AWS_XRAY_SDK_ENABLED` tramite la variabile di ambiente o l'interazione diretta con l'oggetto `global_sdk_config` dalla libreria `aws_xray_sdk`. Le impostazioni con la variabile di ambiente sostituiscono le interazioni.
- `AWS_XRAY_DAEMON_ADDRESS`— Imposta l'host e la porta del listener del demone X-Ray. Per impostazione predefinita, l'SDK utilizza sia`127.0.0.1:2000` per i dati di traccia (UDP) che per il campionamento (TCP). Usa questa variabile se hai configurato il demone per l'ascolto su una porta diversa o se è in esecuzione su un host diverso.

  Formato

  - Stessa porta:*`address`*`:`*`port`*
  - Porte diverse:`tcp:`*`address`*`:`*`port`* `udp:`*`address`*`:`*`port`*
- `AWS_XRAY_CONTEXT_MISSING`— ImpostatoRUNTIME_ERROR per generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.

  Valori validi

  - `RUNTIME_ERROR`— Genera un'eccezione di runtime.
  - `LOG_ERROR`— Registra un errore e continua (impostazione predefinita).
  - `IGNORE_ERROR`— Ignora l'errore e continua.

  Gli errori relativi a segmenti o sottosegmenti mancanti possono verificarsi quando si tenta di utilizzare un client strumentato nel codice di avvio che viene eseguito quando nessuna richiesta è aperta o nel codice che genera un nuovo thread.

Le variabili di ambiente sostituiscono i valori impostati nel codice.

Tracciamento delle richieste in arrivo con l'SDK X-Ray per Python

Quando aggiungi il middleware all'applicazione e configuri il nome di un segmento, l'SDK X-Ray per Python crea un segmento per ogni richiesta campionata. Questo segmento include durata, metodo e conclusione della richiesta HTTP. Analisi ulteriori creano sottosegmenti associati a questo segmento.

l'SDK X-Ray per SDK for Python per Python per strumentare le richieste HTTP in entrata:

- Django
- Flask
- Bottle

> **Note**
>
> PerAWS Lambda le funzioni, Lambda crea un segmento per ogni richiesta campionata. Per ulteriori informazioni, consulta AWS Lambda e AWS X-Ray.

VediWorker per un esempio di funzione Python strumentata in Lambda.

Per gli script o le applicazioni Python su altri framework, puoi creare i segmenti manualmente.

Ogni segmento ha un nome che identifica l'applicazione nella mappa dei servizi. Il segmento può essere denominato staticamente oppure è possibile configurare l'SDK per denominarlo in modo dinamico in base all'intestazione dell'host nella richiesta in arrivo. La denominazione dinamica consente di raggruppare le tracce in base al nome di dominio nella richiesta e di applicare un nome predefinito se il nome non corrisponde a uno schema previsto (ad esempio, se l'intestazione dell'host è falsificata).

> **Richieste inoltrate**
>
> Se un sistema di bilanciamento del carico o un altro intermediario inoltra una richiesta all'applicazione, X-Ray prende l'IP del client dall'`X-Forwarded-For`intestazione della richiesta anziché dall'IP di origine nel pacchetto IP. L'IP del client registrato per una richiesta inoltrata può essere falsificato, quindi non dovrebbe essere considerato affidabile.

Quando viene inoltrata una richiesta, l'SDK imposta un campo aggiuntivo nel segmento per indicarlo. Se il segmento contiene il campo`x_forwarded_for` impostato su`true`, l'IP del client è stato preso dall'`X-Forwarded-For`intestazione della richiesta HTTP.

Il middleware crea un segmento per ogni richiesta in entrata con un blocco `http` che contiene le informazioni riportate qui di seguito:

- Metodo HTTP: GET, POST, PUT, DELETE, ecc.
- Indirizzo client: l'indirizzo IP del client che ha inviato la richiesta.
- Codice di risposta: il codice di risposta HTTP per la richiesta completata.
- Tempistica: l'ora di inizio (quando è stata ricevuta la richiesta) e l'ora di fine (quando è stata inviata la risposta).
- Agente utente: la`user-agent` parte della richiesta.
- Lunghezza del contenuto: la lunghezza`content-length` della risposta.

Sezioni

- Aggiunta del middleware all'applicazione (Django)
- Aggiunta del middleware all'applicazione (flask)
- Aggiunta del middleware all'applicazione (Bottle)
- Analisi manuale del codice Python
- Configurazione di una strategia di denominazione dei segmenti

Aggiunta del middleware all'applicazione (Django)

Aggiungi il middleware all'elenco `MIDDLEWARE` nel tuo file `settings.py`. Il middleware X-Ray dovrebbe essere presente nella prima riga del tuo file `settings.py` per assicurarti che le richieste che non ricadano sotto il controllo di altri middleware quando vengono registrate.

Example settings.py - SDK X-Ray per Python

```
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'
]
```

Aggiungi l'app X-Ray SDK Django all'`INSTALLED_APP`Selenco del tuo`settings.py` file. Ciò consentirà di configurare il registratore a X-Ray all'avvio dell'app.

Example settings.py - l'SDK X-Ray per Python Django

```
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',
]
```

Configurazione di un nome di segmento nel tuo file `settings.py`.

Example settings.py — nome del segmento

```
XRAY_RECORDER = {
    'AWS_XRAY_TRACING_NAME': 'My application',
    'PLUGINS': ('EC2Plugin',),
}
```

Questo indica al registratore X-Ray di tracciare le richieste servite dall'applicazione Django con la frequenza di campionamento predefinita. Puoi configurare il file delle impostazioni del registratore Django affinché applichi le regole di campionatura personalizzate o modificare altre impostazioni.

> **Note**
>
> Poiché`plugins` vengono passati come tupla, assicurati di includere un finale`,` quando specifichi un singolo plugin. Ad esempio, `plugins = ('EC2Plugin',)`

Aggiunta del middleware all'applicazione (flask)

Per analizzare la tua applicazione, prima di tutto configura un nome di segmento in `xray_recorder`. Quindi, utilizza la funzione `XRayMiddleware` per applicare una patch al codice dell'applicazione Flask.

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)
```

Questo indica al registratore a X-Ray di tracciare le richieste servite dall'applicazione Flask con la frequenza di campionamento predefinita. Puoi configurare il registratore all'interno del codice affinché applichi le regole di campionatura personalizzate o modificare altre impostazioni.

Aggiunta del middleware all'applicazione (Bottle)

Per analizzare l'applicazione Bottle, prima di tutto configura un nome di segmento in `xray_recorder`. Quindi, utilizza la funzione `XRayMiddleware` per applicare una patch al codice dell'applicazione Bottle.

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))
```

Questo indica al registratore a X-Ray di tracciare le richieste inviate dall'applicazione Bottle con la frequenza di campionamento predefinita. Puoi configurare il registratore all'interno del codice affinché applichi le regole di campionatura personalizzate o modificare altre impostazioni.

Analisi manuale del codice Python

Se non utilizzi Django o Flask, puoi creare i segmenti manualmente. È possibile creare un segmento per ogni richiesta in arrivo o creare segmenti attorno ai client HTTP oAWS SDK con patch per fornire al registratore il contesto in cui aggiungere sottosegmenti.

Example main.py — Strumentazione manuale

```
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()
```

Configurazione di una strategia di denominazione dei segmenti

AWS X-Rayutilizza un nome di servizio per identificare l'applicazione e distinguerla dalle altre applicazioni, database, API esterne eAWS risorse utilizzate dall'applicazione. Quando l'SDK X-Ray genera segmenti per le richieste in entrata, registra il nome del servizio dell'applicazione nel campo del nome del segmento.

L'SDK X-Ray può assegnare ai segmenti il nome dell'host nell'intestazione della richiesta HTTP. Tuttavia, questa intestazione può essere falsificata, il che potrebbe causare nodi imprevisti nella mappa dei servizi. Per evitare che l'SDK nomini i segmenti in modo errato a causa di richieste con intestazioni host contraffatte, è necessario specificare un nome predefinito per le richieste in entrata.

Se la tua applicazione soddisfa richieste per più domini, puoi configurare l'SDK in modo che utilizzi una strategia di denominazione dinamica per rispecchiarla nei nomi dei segmenti. Una strategia di denominazione dinamica consente all'SDK di utilizzare il nome host per le richieste che corrispondono a uno schema previsto e di applicare il nome predefinito alle richieste che non lo soddisfano.

Ad esempio, potresti avere una singola applicazione che invia richieste a tre sottodomini:`www.example.com`api.example.com, `estatic.example.com`. È possibile utilizzare una strategia di denominazione dinamica con il modello`*.example.com` per identificare i segmenti per ogni sottodominio con un nome diverso, ottenendo tre nodi di servizio sulla mappa dei servizi. Se la tua applicazione riceve richieste con un nome host che non corrisponde al modello, vedrai un quarto nodo sulla mappa del servizio con un nome di fallback specificato.

Per utilizzare lo stesso nome per tutti i segmenti della richiesta, specifica il nome della tua applicazione quando configuri il registratore, come illustrato nelle sezioni precedenti.

Una strategia di denominazione dinamica definisce un modello al quale devono corrispondere i nomi degli host e un nome di default per l'utilizzo qualora il nome dell'host nella richiesta HTTP non corrisponda al modello. Per denominare in modo dinamico i segmenti in Django, aggiungere l'impostazione `DYNAMIC_NAMING` al tuo file settings.py.

Example settings.py — Denominazione dinamica

```
XRAY_RECORDER = {
    'AUTO_INSTRUMENT': True,
    'AWS_XRAY_TRACING_NAME': 'My application',
    'DYNAMIC_NAMING': '*.example.com',
    'PLUGINS': ('ElasticBeanstalkPlugin', 'EC2Plugin')
}
```

Puoi utilizzare '*' nel modello per una corrispondenza con qualsiasi stringa o '?' per una corrispondenza con qualsiasi carattere singolo. Per Flask, configura il registratore nel codice.

Example main.py — nome del segmento

```
from aws_xray_sdk.core import xray_recorder
xray_recorder.configure(service='My application')
xray_recorder.configure(dynamic_naming='*.example.com')
```

> **Note**
>
> È possibile sovrascrivere il nome del servizio predefinito definito nel codice con la variabile di`AWS_XRAY_TRACING_NAME` ambiente.

Applicare le patch alle librerie per analizzare le chiamate a valle

Per strumentare le chiamate downstream, usa X-Ray SDK for Python per applicare patch alle librerie utilizzate dall'applicazione. L'X-Ray SDK per Python può applicare patch alle seguenti librerie.

Librerie supportate

- `botocore`,`boto3`— StrumentoAWS SDK for Python (Boto)clienti.
- `pynamodb`— La versione del client Amazon DynamoDB di Instrument PynamoDB.
- `aiobotocore`,`aioboto3`— Strumentoasincrono-versioni integrate di SDK per client Python.
- `requests`,`aiohttp`— Client HTTP di alto livello dello strumento.
- `httplib`,`http.client`— Strumenta i client HTTP di basso livello e le librerie di livello superiore che li utilizzano.
- `sqlite3`— Client SQLite di Instrument.
- `mysql-connector-python`— Client Instrument MySQL.
- `pg8000`— Interfaccia PostgreSQL dello strumento Pure-Python.
- `psycopg2`— Adattatore per database Instrument PostgreSQL.
- `pymongo`— Client Instrument MongoDB.
- `pymysql`— StrumentoPyMyClient basati su SQL per MySQL e MariaDB.

Quando si utilizza una libreria con patch, X-Ray SDK per Python crea un sottosegmento per la chiamata e registra le informazioni della richiesta e della risposta. Un segmento devono essere disponibili affinché l'SDK possa creare il sottosegmento, sia dal middleware dell'SDK che da AWS Lambda.

> **Note**
>
> Se utilizzi SQLAlchemy ORM, puoi analizzare le tue query SQL importando la versione dell'SDK delle classi della sessione e delle query di SQLAlchemy. Consulta Utilizzare SQLAlchemy ORM per le relative istruzioni.

Per installare le patch disponibili per tutte le librerie, utilizza la funzione `patch_all` in `aws_xray_sdk.core`. Alcune librerie, ad esempio `httplib` e `urllib`, potrebbero dover abilitare l'applicazione di patch doppie chiamando `patch_all(double_patch=True)`.

Example main.py — Patch tutte le librerie supportate

```
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()
```

Per applicare patch a una singola libreria, chiamare `patch` con una tupla del nome della libreria. Per raggiungere questo risultato, è necessario fornire un singolo elenco di elementi.

Example main.py: applica patch a librerie specifiche

```
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 alcuni casi, la chiave utilizzata per installare le patch di una libreria non corrisponde al nome della libreria. Alcune chiavi servono come alias per una o più librerie.
>
> Alias librerie
>
> - `httplib`—`httplib` e `http.client`
> - `mysql` – `mysql-connector-python`

Tracciamento del contesto per le attività asincrone

Per`asyncio`librerie integrate, oppurecreare sottosegmenti per funzioni asincrone, è inoltre necessario configurare X-Ray SDK per Python con un contesto asincrono. Importa il`AsyncContext`classe e ne passa un'istanza al registratore a raggi X.

> **Note**
>
> Le librerie di supporto del framework Web, come AIOHTTP, non vengono gestite tramite il modulo `aws_xray_sdk.core.patcher`. Non sono presenti nel catalogo `patcher` delle librerie supportate.

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)
```

Tracciamento delle chiamate AWS SDK con X-Ray SDK per Python

Quando l'applicazione effettua chiamate per Servizi AWS archiviare dati, scrivere in una coda o inviare notifiche, X-Ray SDK for Python tiene traccia delle chiamate a valle in sottosegmenti. Le risorse tracciate Servizi AWS e a cui accedi all'interno di tali servizi (ad esempio, un bucket Amazon S3 o una coda Amazon SQS) vengono visualizzate come nodi downstream sulla mappa di traccia nella console X-Ray.

L'X-Ray SDK per Python strumenta automaticamente tutti i client AWS SDK quando applichi una patch alla libreria. `botocore` Non puoi analizzare singoli client.

Per tutti i servizi, puoi vedere il nome dell'API richiamata nella console X-Ray. Per un sottoinsieme di servizi, l'SDK X-Ray aggiunge informazioni al segmento per fornire una maggiore granularità nella mappa dei servizi.

Ad esempio, quando si effettua una chiamata con un client DynamoDB con strumentazione, l'SDK aggiunge il nome della tabella al segmento per le chiamate destinate a una tabella. Nella console, ogni tabella appare come un nodo separato nella mappa dei servizi, con un nodo DynamoDB generico per le chiamate che non hanno come destinazione una tabella.

Example Sottosegmento per una chiamata a DynamoDB per salvare un elemento

```
{
  "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",
  }
}
```

Quando si accede alle risorse con nome, le chiamate ai seguenti servizi creano ulteriori nodi della mappa del servizio. Le chiamate che non sono hanno come obiettivo risorse specifiche creano un nodo generico per il servizio.

- Amazon DynamoDB: nome della tabella
- Amazon Simple Storage Service: nome del bucket e della chiave
- Amazon Simple Queue Service: nome della coda

Tracciamento delle chiamate verso Web Services HTTP a valle con l'utilizzo dell'SDK X-Ray per Python

Quando la tua applicazione esegue chiamate verso microservizi o API HTTP pubbliche, puoi utilizzare l'SDK X-Ray per Python per analizzare tali chiamate e aggiungere l'API al grafo del servizio come servizio a valle.

Per analizzare i client HTTP, applica le patch alla libreria che utilizzi per effettuare le chiamate in uscita. Se utilizzi `requests` o il client HTTP integrato in Python, questo è tutto ciò che devi fare. Per `aiohttp`, inoltre, configura il registratore con un contesto async.

Se utilizzi le API del client `aiohttp` 3 devi anche configurare la `ClientSession` con un'istanza della configurazione di tracciamento fornita dall'SDK.

Example Client API **`aiohttp`** 3

```
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()
```

Quando analizzi una chiamata a un'API web a valle, l'SDK X-Ray per Python memorizza un sottosegmento che contiene le informazioni sulla richiesta HTTP e sulla relativa risposta. X-Ray utilizza il sottosegmento per generare un segmento dedotto per l'API remota.

Example Sottosegmento per una chiamata HTTP a valle

```
{
  "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 Segmento dedotto per una chiamata HTTP a valle

```
{
  "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
}
```

Generazione di sottosegmenti personalizzati con l'SDK X-Ray per Python

I sottosegmenti estendono una tracciasegmentocon dettagli sul lavoro svolto per soddisfare una richiesta. Ogni volta che si effettua una chiamata con un client strumentato, l'SDK X-Ray registra le informazioni generate in un sottosegmento. È possibile creare sottosegmenti aggiuntivi per raggruppare altri segmenti secondari, per misurare le prestazioni di una sezione di codice o per registrare annotazioni e metadati.

Per gestire i sottosegmenti, utilizza i metodi `begin_subsegment` e `end_subsegment`.

Example main.py — sottosegmento personalizzato

```
from aws_xray_sdk.core import xray_recorder

subsegment = xray_recorder.begin_subsegment('annotations')
subsegment.put_annotation('id', 12345)
xray_recorder.end_subsegment()
```

Per creare un sottosegmento personalizzato per una funzione sincrona, utilizza il decoratore `@xray_recorder.capture`. Puoi passare un nome per il sottosegmento alla funzione di acquisizione o ometterlo per utilizzare il nome della funzione.

Example main.py sottosegmento funzione

```
from aws_xray_sdk.core import xray_recorder

@xray_recorder.capture('## create_user')
def create_user():
...
```

Per una funzione asincrona, utilizza il decoratore `@xray_recorder.capture_async` e passa un contesto async al registratore.

Example main.py — sottosegmento di funzione asincrona

```
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()
```

Quando crei un sottosegmento all'interno di un segmento o di un altro sottosegmento, l'SDK X-Ray per Python genera per esso un ID e memorizza l'ora di inizio e fine.

Example Sottosegmento con metadati

```
"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"
```

```
    }
  },
```

Aggiungi annotazioni e metadati ai segmenti con l'SDK X-Ray per Python

È possibile utilizzare annotazioni e metadati per registrare informazioni aggiuntive sulle richieste, sull'ambiente o sull'applicazione. È possibile aggiungere annotazioni e metadati ai segmenti creati da X-Ray SDK o ai sottosegmenti personalizzati creati dall'utente.

Le annotazioni sono coppie chiave-valore con stringhe, numeri o valori booleani. Le annotazioni sono indicizzate per essere utilizzate con le espressioni di filtro. Utilizzale per registrare i dati che desideri utilizzare per raggruppare le tracce nella console oppure per chiamare l'API `GetTraceSummaries`.

I metadati sono coppie chiave-valore che possono avere valori di qualsiasi tipo, inclusi oggetti ed elenchi, ma non sono indicizzati per essere utilizzati con le espressioni di filtro. Utilizzate i metadati per registrare dati aggiuntivi che desiderate archiviare nella traccia ma che non è necessario utilizzare con la ricerca.

Oltre ad annotazioni e metadati, sui segmenti puoi anche registrare le stringhe degli ID utente. Gli ID utente vengono memorizzati in un campo separato su segmenti e sono indicizzati per l'uso nelle ricerche.

Sections

- Registrazione di annotazioni con X-Ray SDK per Python
- Registrazione di metadati con X-Ray SDK per Python
- Registrazione degli ID utente con X-Ray SDK per Python

Registrazione di annotazioni con X-Ray SDK per Python

Utilizza le annotazioni per memorizzare le informazioni su segmenti o sottosegmenti che desideri siano indicizzate per la ricerca.

Requisiti per le annotazioni

- Chiavi: la chiave per un'annotazione a raggi X può contenere fino a 500 caratteri alfanumerici. Non è possibile utilizzare spazi o simboli diversi dal simbolo di sottolineatura (_).
- Valori: il valore di un'annotazione X-Ray può contenere fino a 1.000 caratteri Unicode.
- Il numero di annotazioni: è possibile utilizzare fino a 50 annotazioni per traccia.

Per registrare le annotazioni

1. Ottenere un riferimento al segmento o sottosegmento corrente da `xray_recorder`.

   ```
   from aws_xray_sdk.core import xray_recorder
   ...
   document = xray_recorder.current_segment()
   ```

   oppure

   ```
   from aws_xray_sdk.core import xray_recorder
   ...
   document = xray_recorder.current_subsegment()
   ```

2. Chiamare `put_annotation` con una chiave di tipo Stringa e un valore booleano, numerico o di tipo Stringa.

   ```
   document.put_annotation("mykey", "my value");
   ```

In alternativa, puoi utilizzare il metodo `put_annotation` sul `xray_recorder`. Questo metodo memorizza le annotazioni sul sottosegmento corrente oppure, se nessun sottosegmento è aperto, sul segmento.

```
xray_recorder.put_annotation("mykey", "my value");
```

L'SDK memorizza le annotazioni come coppie chiave-valore in un oggetto `annotations` all'interno del documento di segmento. Se chiami `put_annotation` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento o sottosegmento viene sovrascritto.

Per trovare tracciamenti con annotazioni contenenti valori specifici, utilizza la parola chiave `annotations.`*`key`* in un'espressione filtro.

Registrazione di metadati con X-Ray SDK per Python

Utilizza i metadati per memorizzare le informazioni su segmenti o sottosegmenti che non è necessario che siano indicizzate per la ricerca. I valori dei metadati possono essere stringhe, numeri, valori booleani o qualsiasi oggetto che possa essere serializzato in un oggetto o in un vettore JSON.

Per registrare i metadati

1. Ottenere un riferimento al segmento o sottosegmento corrente da `xray_recorder`.

   ```
   from aws_xray_sdk.core import xray_recorder
   ...
   document = xray_recorder.current_segment()
   ```

   oppure

   ```
   from aws_xray_sdk.core import xray_recorder
   ...
   document = xray_recorder.current_subsegment()
   ```

2. Chiama `put_metadata` con una chiave di tipo Stringa, un valore booleano, numerico, di tipo Stringa o di tipo oggetto e un namespace di tipo stringa.

   ```
   document.put_metadata("my key", "my value", "my namespace");
   ```

   oppure

   Invocare `put_metadata` semplicemente con una chiave e un valore.

   ```
   document.put_metadata("my key", "my value");
   ```

In alternativa, puoi utilizzare il metodo `put_metadata` sul `xray_recorder`. Questo metodo memorizza i metadati sul sottosegmento corrente oppure, se nessun sottosegmento è aperto, sul segmento.

```
xray_recorder.put_metadata("my key", "my value");
```

Se non specifichi un namespace, l'SDK utilizza `default`. Se chiami `put_metadata` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento o sottosegmento viene sovrascritto.

Registrazione degli ID utente con X-Ray SDK per Python

Memorizza gli ID utente sui segmenti di richiesta per identificare l'utente che ha inviato la richiesta.

Per registrare gli ID degli utenti

1. Ottenere un riferimento al segmento corrente da `xray_recorder`.

```
from aws_xray_sdk.core import xray_recorder
...
document = xray_recorder.current_segment()
```

2. Chiamare `setUser` con una stringa che rappresenta l'ID dell'utente che ha inviato la richiesta.

```
document.set_user("U12345");
```

Puoi chiamare `set_user` nel tuo controller per registrare l'ID utente non appena l'applicazione inizia ad elaborare una richiesta.

Per trovare tracciamenti associati ad un ID utente, utilizza la parola chiave `user` in un'espressione filtro.

Analisi di framework Web distribuiti in ambienti serverless

L' AWS X-Ray SDK per Python supporta la strumentazione di framework web distribuiti in applicazioni serverless. Serverless è l'architettura nativa del cloud che consente di trasferire più responsabilità operative ad AWS, aumentando agilità e innovazione.

L'architettura serverless è un modello di applicazione software che consente di creare ed eseguire applicazioni e servizi senza pensare ai server. Elimina le attività di gestione delle infrastrutture come il provisioning del server o del cluster, l'applicazione di patch, la manutenzione del sistema operativo e il provisioning della capacità. Puoi creare soluzioni serverless per pressoché ogni tipo di applicazione o servizio di back-end e tutte le operazioni necessarie per l'esecuzione e la scalabilità dell'applicazione saranno gestite in automatico.

Questo tutorial mostra come utilizzare automaticamente la strumentazione AWS X-Ray su un framework web, come Flask o Django, distribuito in un ambiente serverless. La strumentazione X-Ray dell'applicazione ti consente di visualizzare tutte le chiamate downstream effettuate, a partire da Amazon API Gateway attraverso la tua AWS Lambda funzione, e le chiamate in uscita effettuate dall'applicazione.

L'X-Ray SDK per Python supporta i seguenti framework applicativi Python:

- Flask 0.8 o versioni successive

- Django 1.0 o versioni successive

Questo tutorial sviluppa un esempio di applicazione serverless che viene distribuita su Lambda e richiamata da API Gateway. Questo tutorial utilizza Zappa per distribuire automaticamente l'applicazione su Lambda e configurare l'endpoint API Gateway.

Prerequisiti

- Zappa
- Python — Versione 2.7 o 3.6.
- AWS CLI— Verifica che il tuo AWS CLI sia configurato con l'account e Regione AWS in cui distribuirai l'applicazione.
- Pip
- Virtualenv

Fase 1: creazione di un ambiente

In questa fase, crei un ambiente virtuale usando `virtualenv` per ospitare un'applicazione.

1. Utilizzando AWS CLI, create una directory per l'applicazione. Quindi, passare alla nuova directory.

   ```
   mkdir serverless_application
   cd serverless_application
   ```

2. Successivamente, creare un ambiente virtuale all'interno della nuova directory. Utilizzare il comando seguente per attivarlo.

   ```
   # Create our virtual environment
   virtualenv serverless_env

   # Activate it
   source serverless_env/bin/activate
   ```

3. Installa X-Ray, Flask, Zappa e la libreria Requests nel tuo ambiente.

   ```
   # Install X-Ray, Flask, Zappa, and Requests into your environment
   pip install aws-xray-sdk flask zappa requests
   ```

4. Aggiungere il codice dell'applicazione nella directory `serverless_application`. In questo caso, si utilizza l'esempio Hello World di Flask.

   Nella directory `serverless_application`, creare un file denominato `my_app.py`. Quindi utilizzare un editor di testo per aggiungere i seguenti comandi. Questa applicazione analizza la libreria delle richieste, applica le patch al middleware dell'applicazione Flask e apre l'endpoint `'/'`.

   ```
   # 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
   ```

Fase 2: creazione e distribuzione di un ambiente Zappa

In questo passaggio utilizzerai Zappa per configurare automaticamente un endpoint API Gateway e poi distribuirlo su Lambda.

1. Inizializzare Zappa dalla directory `serverless_application`. Per questo esempio vengono utilizzate le impostazioni predefinite, ma se si desidera personalizzare le preferenze, è possibile visualizzare le istruzioni di configurazione di Zappa.

   ```
   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. Abilita X-Ray. Aprire il file `zappa_settings.json` e verificare che sia simile all'esempio.

```
{
    "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. Aggiungere la voce `"xray_tracing": true` al file di configurazione.

```
{
    "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. Distribuire l'applicazione. Questo configura automaticamente l'endpoint API Gateway e carica il codice su Lambda.

```
zappa deploy
```

```
...
Deploying API Gateway..
Deployment complete!: https://**********.execute-api.us-west-2.amazonaws.com/dev
```

Fase 3: Abilitare il tracciamento a raggi X per API Gateway

In questo passaggio interagirai con la console API Gateway per abilitare il tracciamento X-Ray.

1. Accedi AWS Management Console e apri la console API Gateway all'indirizzo https://console.aws.amazon.com/apigateway/.
2. Trovare l'API generata di recente. Deve avere un formato analogo a `serverless-exam-dev`.
3. Scegliere Stages (Fasi).
4. Scegliere il nome della fase di distribuzione. Il valore predefinito è `dev`.
5. Nella scheda Log/Tracciamento, selezionare la casella Enable X-Ray Tracing (Abilita tracciamento X-Ray).
6. Seleziona Salva modifiche.
7. Accedere all'endpoint nel browser. Se è stata utilizzata l'applicazione di esempio `Hello World`, viene visualizzato quanto segue.

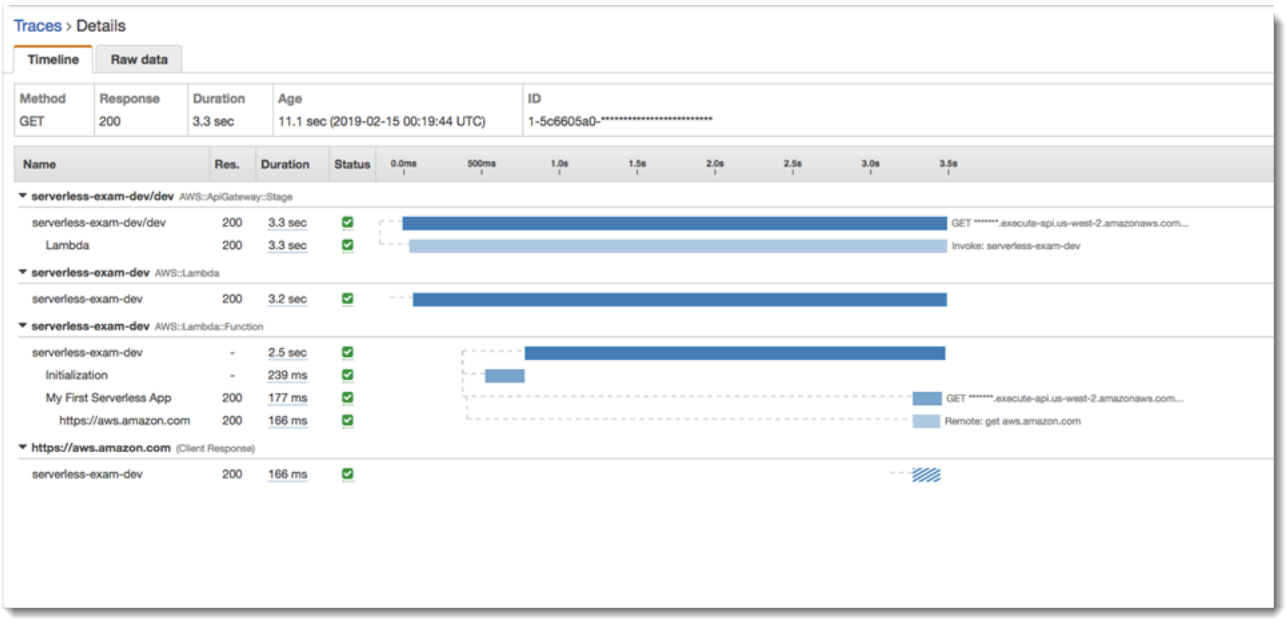
```
"Hello, World: https://aws.amazon.com/"
```

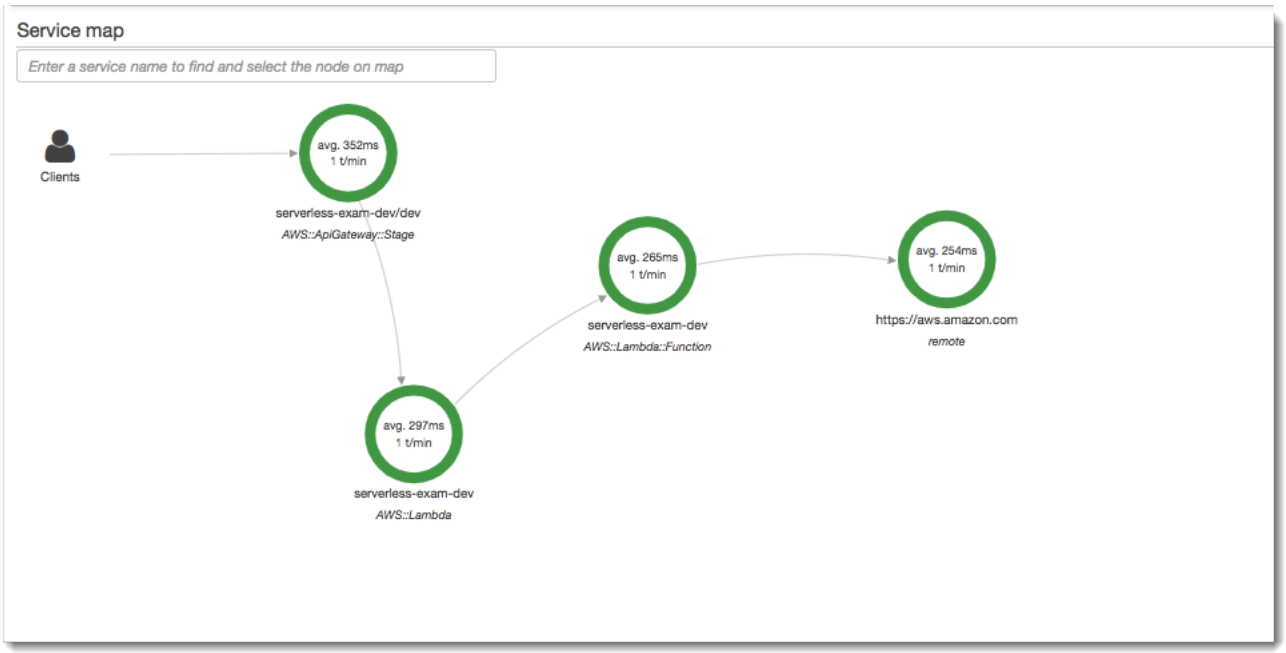
Fase 4: visualizzazione della traccia creata

In questo passaggio interagirai con la console X-Ray per visualizzare la traccia creata dall'applicazione di esempio. Per una spiegazione passo per passo sull'analisi di traccia, consulta Visualizzazione della mappa del servizio.

1. Accedere AWS Management Console e aprire la console X-Ray all'indirizzo https://console.aws.amazon.com/xray/home.

2. Visualizza i segmenti generati da API Gateway, dalla funzione Lambda e dal contenitore Lambda.
3. Nel segmento della funzione Lambda, visualizza un sottosegmento denominato `My First Serverless App` È seguito da un secondo sottosegmento denominato `https://aws.amazon.com`.
4. Durante l'inizializzazione, Lambda potrebbe anche generare un terzo sottosegmento denominato. `initialization`





Fase 5: rimozione

Termina sempre le risorse che non usi più per evitare l'accumulo di costi imprevisti. Come dimostra questo tutorial, gli strumenti come Zappa semplificano la ridistribuzione serverless.

Per rimuovere l'applicazione da Lambda, API Gateway e Amazon S3, esegui il comando seguente nella directory del progetto utilizzando il. AWS CLI

```
zappa undeploy dev
```

Passaggi successivi

Aggiungi altre funzionalità alla tua applicazione aggiungendo AWS client e strumentandoli con X-Ray. Scopri di più sulle opzioni di elaborazione serverless su Serverless on. AWS

Strumentate la vostra applicazione con .NET

Esistono due modi per strumentare .NET l'applicazione per inviare tracce a X-Ray:

- AWS Distro for OpenTelemetry .NET: una AWS distribuzione che fornisce un set di librerie open source per l'invio di metriche e tracce correlate a più soluzioni di AWS monitoraggio tra cui Amazon AWS X-Ray e Amazon OpenSearch Service CloudWatch, tramite AWS Distro for Collector. OpenTelemetry
- AWS X-Ray SDK per .NET: un set di librerie per la generazione e l'invio di tracce a X-Ray tramite il demone X-Ray.

Per ulteriori informazioni, consulta Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray.

AWS Distro per OpenTelemetry .NET

Con AWS Distro for OpenTelemetry .NET, puoi strumentare le tue applicazioni una sola volta e inviare metriche e tracce correlate a più soluzioni di AWS monitoraggio tra cui Amazon CloudWatch AWS X-Ray e Amazon Service. OpenSearch L'utilizzo di X-Ray con AWS Distro for OpenTelemetry richiede due componenti: un OpenTelemetry SDK abilitato per l'uso con X-Ray e AWS Distro for OpenTelemetry Collector abilitato per l'uso con X-Ray.

Per iniziare, consulta la Distro per la documentazione.AWS OpenTelemetry .NET

Per ulteriori informazioni sull'utilizzo di AWS Distro for OpenTelemetry AWS X-Ray e altro Servizi AWS, consulta AWS Distro for OpenTelemetry o Distro for DocumentationAWS . OpenTelemetry

Per ulteriori informazioni sul supporto e l'utilizzo della lingua, vedere AWS Observability on. GitHub

AWS X-Ray SDK per.NET

X-Ray SDK for .NET è una libreria per la strumentazione delle applicazioni Web C# .NET, delle applicazioni Web.NET Core e delle funzioni.NET Core. AWS Lambda Fornisce classi e metodi per generare e inviare dati di traccia al demone X-Ray. Ciò include informazioni sulle richieste in entrata fornite dall'applicazione e sulle chiamate che l'applicazione effettua a valle Servizi AWS, alle API Web HTTP e ai database SQL.

> **Note**
>
> X-Ray SDK for .NET è un progetto open source. Puoi seguire il progetto e inviare problemi e richieste di pull su GitHub: github.com/aws/ aws-xray-sdk-dotnet

In caso di applicazioni web, per tracciare le richieste in entrata inizia aggiungendo un gestore di messaggi nella configurazione web. Il gestore dei messaggi crea un segmento per ogni richiesta tracciata e completa il segmento quando viene inviata la risposta. Fino a che il segmento è aperto puoi usare i metodi del client dell'SDK per aggiungere informazioni al segmento e creare sottosegmenti per tracciare le chiamate a valle. L'SDK, inoltre, registra automaticamente le eccezioni sollevate dall'applicazione per il tempo durante il quale il segmento è aperto.

Per le funzioni Lambda richiamate da un'applicazione o un servizio strumentato, Lambda legge l'intestazione di tracciamento e traccia automaticamente le richieste campionate. Per altre funzioni, puoi configurare Lambda per campionare e tracciare le richieste in arrivo. In entrambi i casi, Lambda crea il segmento e lo fornisce all'X-Ray SDK.

> **Note**
>
> Su Lambda, l'SDK X-Ray è opzionale. Se non lo usi nella tua funzione, la mappa dei servizi includerà comunque un nodo per il servizio Lambda e uno per ogni funzione Lambda. Aggiungendo l'SDK, puoi utilizzare il codice della funzione per aggiungere sottosegmenti al segmento di funzione registrato da Lambda. Per ulteriori informazioni, consulta AWS Lambda e AWS X-Ray.

Successivamente, usa l'X-Ray SDK for .NET per strumentare i tuoi clienti. AWS SDK for .NET Ogni volta che effettui una chiamata a un downstream Servizio AWS o a una risorsa con un client dotato di strumentazione, l'SDK registra le informazioni sulla chiamata in un sottosegmento. AWS i servizi e le risorse a cui accedi all'interno dei servizi vengono visualizzati come nodi a valle sulla mappa di traccia per aiutarti a identificare errori e problemi di limitazione sulle singole connessioni.

L'X-Ray SDK for .NET fornisce anche la strumentazione per le chiamate downstream alle API Web HTTP e ai database SQL. Il metodo di estensione `GetResponseTraced` di `System.Net.HttpWebRequest` traccia le chiamate HTTP in uscita. È possibile utilizzare la versione X-Ray SDK for .NET di per `SqlCommand` strumentare le query SQL.

Dopo aver iniziato a utilizzare l'SDK, personalizzane il comportamento configurando il registratore e il gestore dei messaggi. Puoi aggiungere dei plugin per memorizzare i dati sulle risorse di elaborazione sulle quali è eseguita la tua applicazione, personalizzare il comportamento di campionamento definendo regole di campionatura e impostare il livello di log per visualizzare più o meno informazioni generate dall'SDK nei log dell'applicazione.

Registra ulteriori informazioni sulle richieste e sull'attività svolta dalla tua applicazione in annotazioni e metadati. Le annotazioni sono semplici coppie chiave-valore indicizzati per l'uso con espressioni filtro, in modo da poter cercare tracce che contengono dati specifici. Le immissioni di metadati sono meno restrittive e possono registrare interi oggetti e matrici, ovvero tutto ciò che può essere serializzato in JSON.

> **Annotazioni e metadata**
>
> Le annotazioni e i metadati sono testo arbitrario che aggiungi ai segmenti con X-Ray SDK. Le annotazioni vengono indicizzate per essere utilizzate con le espressioni di filtro. I metadati non sono indicizzati, ma possono essere visualizzati nel segmento non elaborato con la console X-Ray o l'API. Chiunque conceda l'accesso in lettura a X-Ray può visualizzare questi dati.

Quando disponi di una notevole quantità di client analizzati nel tuo codice, un singolo segmento per la richiesta può contenere un numero elevato di sottosegmenti, uno per ciascuna delle chiamate effettuate con un client analizzato. Puoi organizzare e raggruppare i sottosegmenti inglobando le chiamate del client sottosegmenti personalizzati. Puoi creare un sottosegmento personalizzato per un'intera funzione o qualsiasi porzione di codice, e memorizzare metadati e annotazioni sul sottosegmento invece di scrivere tutto all'interno del segmento padre.

Per la documentazione di riferimento sulle classi e sui metodi dell'SDK, consulta quanto indicato qui di seguito:

- AWS X-Ray Guida di riferimento all'API SDK for .NET
- AWS X-Ray Guida di riferimento all'API SDK for .NET Core

Lo stesso pacchetto supporta sia .NET che .NET Core, ma le classi utilizzate sono diverse. Gli esempi in questo capitolo sono collegati alla guida di riferimento per le API .NET a meno che la classe non sia specifica per .NET Core.

Requisiti

X-Ray SDK for .NET richiede .NET Framework 4.5 o versione successiva e. AWS SDK for .NET

In caso di applicazioni e funzioni .NET Core, l'SDK richiede .NET Core 2.0 o versioni successive.

Aggiungere l'X-Ray SDK for .NET all'applicazione

NuGet Usalo per aggiungere l'X-Ray SDK for .NET alla tua applicazione.

Per installare X-Ray SDK per .NET con il gestore di NuGet pacchetti in Visual Studio

1. Scegli Tools, NuGet Package Manager, Manage NuGet Packages for Solution.
2. Cercare AWSXRayRecorder.
3. Scegliere il pacchetto e quindi scegliere Install (Installa).

Gestione delle dipendenze

L'X-Ray SDK for .NET è disponibile presso Nuget. Installa l'SDK utilizzando il gestore di pacchetti:

```
Install-Package AWSXRayRecorder -Version 2.10.1
```

Il pacchetto `AWSXRayRecorder v2.10.1` nuget ha le seguenti dipendenze:

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)
```

Per ulteriori dettagli sulla gestione delle dipendenze, consulta la documentazione di Microsoft sulla dipendenza da Nuget e sulla risoluzione delle dipendenze da Nuget.

Configurazione l'SDK X-Ray per .NET

È possibile configurare X-Ray SDK for .NET con plug-in per includere informazioni sul servizio su cui viene eseguita l'applicazione, modificare il comportamento di campionamento predefinito o aggiungere regole di campionamento applicabili alle richieste a percorsi specifici.

Per le applicazioni web .NET, aggiungi le chiavi alla sezione `appSettings` del file `Web.config`.

Example Web.config

```
<configuration>
  <appSettings>
    <add key="AWSXRayPlugins" value="EC2Plugin"/>
    <add key="SamplingRuleManifest" value="sampling-rules.json"/>
  </appSettings>
</configuration>
```

Per .NET Core, crea un file denominato `appsettings.json` con una chiave di primo livello denominata `XRay`.

Example .NET appsettings.json

```
{
  "XRay": {
    "AWSXRayPlugins": "EC2Plugin",
    "SamplingRuleManifest": "sampling-rules.json"
  }
}
```

Quindi, nel codice dell'applicazione, crea un oggetto di configurazione e usalo per inizializzare il registratore a X-Ray X. Esegui questa operazione prima di inizializzare il registratore.

Example .NET Core Program.cs — Configurazione del registratore

```
using Amazon.XRay.Recorder.Core;
...
AWSXRayRecorder.InitializeInstance(configuration);
```

Se stai analizzando un'applicazione web .NET Core, puoi anche inoltrare l'oggetto di configurazione al metodo `UseXRay` quando configuri il gestore dei messaggi. Per le funzioni Lambda, utilizzate il`InitializeInstance` metodo illustrato in precedenza.

Per ulteriori informazioni sull'API di configurazione.NET Core, vedere Configurare un'app ASP.NET Core su docs.microsoft.com.

Sezioni

- Plug-in
- Regole di campionamento
- Logging (.NET)
- Logging (.NET Core)
- Variabili di ambiente

Plug-in

Utilizza i plug-in per aggiungere i dati sul servizio che ospita l'applicazione.

Plug-in

- Amazon EC2:`EC2Plugin` aggiunge l'ID dell'istanza, la zona di disponibilità e il gruppo di CloudWatch log.
- Elastic Beanstalk:`ElasticBeanstalkPlugin` aggiunge il nome dell'ambiente, l'etichetta della versione e l'ID di distribuzione.
- Amazon ECS:`ECSPlugin` aggiunge l'ID del contenitore.

Per utilizzare un plug-in, configurare X-Ray SDK per client .NET client aggiungendo l'`AWSXRayPlugins`impostazione. Se si applicano più plugin all'applicazione, specificarli tutti nella stessa impostazione, separati da virgole.

Example Web.config - Plugin

```
<configuration>
  <appSettings>
    <add key="AWSXRayPlugins" value="EC2Plugin,ElasticBeanstalkPlugin"/>
  </appSettings>
</configuration>
```

Example .NET Core appsettings.json — Plugin

```
{
  "XRay": {
    "AWSXRayPlugins": "EC2Plugin,ElasticBeanstalkPlugin"
  }
}
```

Regole di campionamento

L'SDK utilizza le regole di campionamento definite nella console X-Ray per determinare quali richieste registrare. La regola predefinita tiene traccia della prima richiesta ogni secondo e del cinque percento di tutte le richieste aggiuntive su tutti i servizi che inviano tracce a X-Ray. Crea regole aggiuntive nella console X-Ray per personalizzare la quantità di dati registrati per ciascuna delle tue applicazioni.

L'SDK applica regole personalizzate nell'ordine in cui sono definite. Se una richiesta corrisponde a più regole personalizzate, l'SDK applica solo la prima regola.

> **Note**
>
> Se l'SDK non riesce a raggiungere X-Ray per ottenere le regole di campionamento, torna a una regola locale predefinita della prima richiesta ogni secondo e del cinque percento di tutte le richieste aggiuntive per host. Ciò può verificarsi se l'host non dispone dell'autorizzazione per chiamare le API di campionamento o non riesce a connettersi al demone X-Ray, che funge da proxy TCP per le chiamate API effettuate dall'SDK.

Puoi anche configurare l'SDK per caricare le regole di campionamento da un documento JSON. L'SDK può utilizzare regole locali come backup nei casi in cui il campionamento a X-Ray non è disponibile o utilizzare esclusivamente regole locali.

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
  }
}
```

Questo esempio definisce una regola personalizzata e una regola predefinita. La regola personalizzata applica una frequenza di campionamento del cinque percento senza un numero minimo di richieste da tracciare per i percorsi`/api/move/`. La regola predefinita traccia la prima richiesta ogni secondo e il 10 percento delle altre richieste.

Lo svantaggio della definizione locale delle regole è che l'obiettivo fisso viene applicato da ciascuna istanza del registratore in modo indipendente, anziché essere gestito dal servizio X-Ray. Man mano che si installano più host, la velocità fissa si moltiplica, rendendo più difficile il controllo della quantità di dati registrati.

SìAWS Lambda, non è possibile modificare la frequenza di campionamento. Se la funzione viene chiamata da un servizio strumentato, le chiamate che hanno generato richieste campionate da quel servizio verranno registrate da Lambda. Se il tracciamento attivo è abilitato e non è presente un'intestazione di tracciamento, Lambda prende la decisione di campionamento.

Per configurare le regole di backup, comunica a X-Ray SDK for .NET di caricare le regole di campionamento da un file con l'`SamplingRuleManifest`impostazione.

Example .NET Web.config - regole di campionamento

```
<configuration>
  <appSettings>
```

```
    <add key="SamplingRuleManifest" value="sampling-rules.json"/>
  </appSettings>
</configuration>
```

Example .NET Core appsettings.json — Regole di campionamento

```
{
  "XRay": {
    "SamplingRuleManifest": "sampling-rules.json"
  }
}
```

Per utilizzare solo regole locali, crea un registratore con una `LocalizedSamplingStrategy`. Se si dispone di regole di backup configurate, rimuovere tale configurazione.

Example .NET global.asax — Regole di campionamento locali

```
var recorder = new AWSXRayRecorderBuilder().WithSamplingStrategy(new
 LocalizedSamplingStrategy("samplingrules.json")).Build();
AWSXRayRecorder.InitializeInstance(recorder: recorder);
```

Example .NET Core Program.cs — Regole di campionamento locali

```
var recorder = new AWSXRayRecorderBuilder().WithSamplingStrategy(new
 LocalizedSamplingStrategy("sampling-rules.json")).Build();
AWSXRayRecorder.InitializeInstance(configuration,recorder);
```

Logging (.NET)

l'SDK X-Ray per .NET utilizza lo stesso meccanismo di registrazione di AWS SDK for .NET. Se l'applicazione è già stata configurata per registrare l'AWS SDK for .NEToutput, la stessa configurazione si applica all'output dall'SDK X-Ray per .NET.

Per configurare il log, aggiungi una sezione di configurazione denominata `aws` al tuo file `App.config` o al tuo file `Web.config`.

Example Web.config - Generazione dei log

```
...
<configuration>
  <configSections>
    <section name="aws" type="Amazon.AWSSection, AWSSDK.Core"/>
```

```
  </configSections>
  <aws>
    <logging logTo="Log4Net"/>
  </aws>
</configuration>
```

Per ulteriori informazioni, consulta Configurazione dell'applicazione AWS SDK for .NET nella Guida per gli sviluppatori di AWS SDK for .NET.

Logging (.NET Core)

l'SDK X-Ray per .NET utilizzare le stesse opzioni di registrazione di AWS SDK for .NET. Per configurare la registrazione per le applicazioni.NET Core, passate l'opzione di registrazione al`AWSXRayRecorder.RegisterLogger` metodo.

Ad esempio, per utilizzare log4net, è necessario creare un file di configurazione che definisca il logger, formato di output e la posizione del file.

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>
```

Quindi creare il logger e applicare la configurazione al codice del programma.

Example .NET Core Program.cs — Registrazione

```
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)
  {
  ...
  }
}
```

Per ulteriori informazioni sulla configurazione di log4net, vedere Configurazione su logging.apache.org.

Variabili di ambiente

È possibile utilizzare variabili di ambiente per configurare l'SDK X-Ray per .NET. L'SDK supporta le seguenti variabili.

- `AWS_XRAY_TRACING_NAME`— Imposta un nome di servizio che l'SDK utilizza per i segmenti. Sostituisce il nome del servizio impostato sulla strategia di denominazione dei segmenti del filtro servlet.
- `AWS_XRAY_DAEMON_ADDRESS`— Imposta l'host e la porta del listener del demone X-Ray. Per impostazione predefinita, l'SDK utilizza sia`127.0.0.1:2000` per i dati di traccia (UDP) che per il campionamento (TCP). Usa questa variabile se hai configurato il demone per l'ascolto su una porta diversa o se è in esecuzione su un host diverso.

  Formato

  - Stessa porta:*`address`*`:`*`port`*
  - Porte diverse:`tcp:`*`address`*`:`*`port`* `udp:`*`address`*`:`*`port`*
- `AWS_XRAY_CONTEXT_MISSING`— Impostato`RUNTIME_ERROR` per generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.

  Valori validi

  - `RUNTIME_ERROR`— Genera un'eccezione di runtime.
  - `LOG_ERROR`— Registra un errore e continua (impostazione predefinita).

- `IGNORE_ERROR`— Ignora l'errore e continua.

Gli errori relativi a segmenti o sottosegmenti mancanti possono verificarsi quando si tenta di utilizzare un client strumentato nel codice di avvio che viene eseguito quando nessuna richiesta è aperta o nel codice che genera un nuovo thread.

Strumentazione delle richieste HTTP in ingresso con l'SDK X-Ray per .NET

Puoi usare l'SDK X-Ray per tracciare le richieste HTTP in ingresso elaborate su un'istanza EC2 in Amazon EC2,AWS Elastic Beanstalko Amazon ECS.

Utilizza un gestore di messaggi per analizzare le richieste HTTP in ingresso. Quando aggiungi il gestore di messaggi X-Ray alla tua applicazione, l'SDK X-Ray per .NET crea un segmento per ogni richiesta campionata. Questo segmento include durata, metodo e conclusione della richiesta HTTP. Analisi ulteriori creano sottosegmenti associati a questo segmento.

> **Note**
>
> PerAWS LambdaFunzioni, Lambda crea un segmento per ogni richiesta campionata. Per ulteriori informazioni, consultare AWS Lambda e AWS X-Ray.

Ogni segmento ha un nome che identifica l'applicazione nella mappa del servizio. Il segmento può essere denominato staticamente oppure è possibile configurare l'SDK per nominarlo dinamicamente in base all'intestazione host nella richiesta in entrata. La denominazione dinamica consente di raggruppare le tracce in base al nome di dominio nella richiesta e di applicare un nome predefinito se il nome non corrisponde a uno schema previsto (ad esempio, se l'intestazione host è falsificata).

> **Richieste inoltrate**
>
> Se un sistema di bilanciamento del carico o un altro intermediario inoltra una richiesta all'applicazione, X-Ray preleva l'IP del client dal`X-Forwarded-For`intestazione nella richiesta anziché dall'IP di origine nel pacchetto IP. L'IP client registrato per una richiesta inoltrata può essere falsificato, quindi non dovrebbe essere attendibile.

Il gestore dei messaggi crea un segmento per ogni richiesta in entrata con un blocco `http` che contiene le informazioni riportate qui di seguito:

- HTTP method (Metodo HTTP)— GET, POST, PUT, DELETE, ecc.
- Indirizzo client— Indirizzo IP del client che ha inviato la richiesta.
- Codice di risposta— Codice HTTP di risposta per la richiesta completata.
- Timing (Tempo)— Ora di inizio (quando è stata ricevuta la richiesta) e di fine (quando è stata inviata la risposta).
- User Agent— Il`user-agent`dalla richiesta.
- Lunghezza del contenuto— Il`content-length`della risposta.

Sezioni

- Analisi delle richieste in entrata (.NET)
- Analisi delle richieste in entrata (.NET Core)
- Configurazione di una strategia di denominazione dei segmenti

Analisi delle richieste in entrata (.NET)

Per analizzare le richieste elaborate dalla tua applicazione, chiama `RegisterXRay` nel metodo `Init` del tuo file `global.asax`.

Example global.asax - Gestore messaggi

```
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");
    }
  }
}
```

Analisi delle richieste in entrata (.NET Core)

Per analizzare le richieste elaborate dalla tua applicazione, chiama`UseXRay`metodo prima di qualsiasi altro middleware nel`Configure`metodo della classe Startup come idealmente middleware X-Ray dovrebbe essere il primo middleware ad elaborare la richiesta e l'ultimo middleware per elaborare la risposta nella pipeline.

> **Note**
>
> Per .NET Core 2.0, se hai un`UseExceptionHandler`metodo nell'applicazione, assicurarsi di chiamare`UseXRay`dopo`UseExceptionHandler`metodo per garantire la registrazione delle eccezioni.

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
    ...
  }
```

Il metodo `UseXRay` può anche ricevere un oggetto di configurazione come secondo argomento.

```
app.UseXRay("MyApp", configuration);
```

Configurazione di una strategia di denominazione dei segmenti

AWS X-Rayusa unnome servizioper identificare l'applicazione e distinguerla dalle altre applicazioni, database, API esterne eAWSrisorse utilizzate dalla tua applicazione. Quando X-Ray SDK genera segmenti per le richieste in arrivo, registra il nome del servizio dell'applicazione nel segmentocampo name.

L'SDK X-Ray può denominare i segmenti dopo il nome host nell'intestazione della richiesta HTTP. Tuttavia, questa intestazione può essere falsificata, il che potrebbe causare nodi imprevisti nella mappa del servizio. Per impedire all'SDK di assegnare nomi ai segmenti in modo errato a causa di richieste con intestazioni host falsificate, è necessario specificare un nome predefinito per le richieste in arrivo.

Se l'applicazione soddisfa le richieste per più domini, è possibile configurare l'SDK in modo che utilizzi una strategia di denominazione dinamica per rifletterlo nei nomi dei segmenti. Una strategia di denominazione dinamica consente all'SDK di utilizzare il nome host per le richieste che corrispondono a un modello previsto e di applicare il nome predefinito alle richieste che non lo fanno.

Ad esempio, è possibile avere una singola applicazione che elavora le richieste su tre sottodomini:`www.example.com`,`api.example.com`, `estatic.example.com`. È possibile utilizzare una strategia di denominazione dinamica con il modello`*.example.com`per identificare i segmenti per ogni sottodominio con un nome diverso, risultando in tre nodi di servizio sulla mappa del servizio. Se l'applicazione riceve richieste con un nome host che non corrisponde al modello, verrà visualizzato un quarto nodo sulla mappa del servizio con un nome di fallback specificato.

Per utilizzare lo stesso nome per tutti i segmenti della richiesta, specifica il nome della tua applicazione quando inizializzi il gestore dei messaggi, come illustrato nella sezione precedente. Ciò ha lo stesso effetto di creare una `FixedSegmentNamingStrategy` e passarla al metodo `RegisterXRay`.

```
AWSXRayASPNET.RegisterXRay(this, new FixedSegmentNamingStrategy("MyApp"));
```

> **Note**
>
> È possibile sovrascrivere il nome di servizio predefinito definito nel codice con il`AWS_XRAY_TRACING_NAME` Variabile di ambiente.

Una strategia di denominazione dinamica definisce un modello al quale devono corrispondere i nomi degli host e un nome di default per l'utilizzo qualora il nome dell'host nella richiesta HTTP non corrisponda al modello. Per denominare i segmenti in modo dinamico, è necessario creare una `DynamicSegmentNamingStrategy` e passarla al metodo `RegisterXRay`.

```
AWSXRayASPNET.RegisterXRay(this, new DynamicSegmentNamingStrategy("MyApp",
 "*.example.com"));
```

Tracciamento delle chiamate AWS SDK con X-Ray SDK for .NET

Quando l'applicazione effettua chiamate per Servizi AWS archiviare dati, scrivere in una coda o inviare notifiche, X-Ray SDK for .NET tiene traccia delle chiamate downstream in sottosegmenti. Le risorse tracciate Servizi AWS e a cui accedi all'interno di tali servizi (ad esempio, un bucket Amazon S3 o una coda Amazon SQS) vengono visualizzate come nodi downstream sulla mappa di traccia nella console X-Ray.

Puoi strumentare tutti i tuoi AWS SDK for .NET clienti chiamando prima di crearli. `RegisterXRayForAllServices`

Example SampleController.cs - Strumentazione client DynamoDB

```
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;
    });
```

Per analizzare il client di alcuni servizi e non di altri, chiama `RegisterXRay` invece di `RegisterXRayForAllServices`. Sostituisci il testo evidenziato con il nome dell'interfaccia client del servizio.

```
AWSSDKHandler.RegisterXRay<IAmazonDynamoDB>()
```

Per tutti i servizi, puoi vedere il nome dell'API richiamata nella console X-Ray. Per un sottoinsieme di servizi, l'SDK X-Ray aggiunge informazioni al segmento per fornire una maggiore granularità nella mappa dei servizi.

Ad esempio, quando si effettua una chiamata con un client DynamoDB con strumentazione, l'SDK aggiunge il nome della tabella al segmento per le chiamate destinate a una tabella. Nella console, ogni tabella appare come un nodo separato nella mappa dei servizi, con un nodo DynamoDB generico per le chiamate che non hanno come destinazione una tabella.

Example Sottosegmento per una chiamata a DynamoDB per salvare un elemento

```
{
  "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",
  }
}
```

Quando si accede alle risorse con nome, le chiamate ai seguenti servizi creano ulteriori nodi della mappa del servizio. Le chiamate che non sono hanno come obiettivo risorse specifiche creano un nodo generico per il servizio.

- Amazon DynamoDB: nome della tabella

- Amazon Simple Storage Service: nome del bucket e della chiave
- Amazon Simple Queue Service: nome della coda

Tracciamento delle chiamate verso Web Services HTTP a valle con l'SDK X-Ray per .NET

Quando la tua applicazione esegue chiamate verso microservizi o API HTTP pubbliche, puoi utilizzare l'SDK X-Ray per .NET`GetResponseTraced`metodo delle estensioni per`System.Net.HttpWebRequest`analizzare tali chiamate e aggiungere l'API al grafo del servizio come servizio a valle.

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();
}
```

Per le chiamate asincrone, utilizza `GetAsyncResponseTraced`.

```
request.GetAsyncResponseTraced();
```

Se utilizzi `system.net.http.httpclient`, usa il gestore della delegazione `HttpClientXRayTracingHandler` per memorizzare le chiamate.

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);
}
```

Quando analizzi una chiamata a un'API web a valle, l'SDK X-Ray per .NET memorizza un sottosegmento con informazioni sulla richiesta HTTP e sulla relativa risposta. X-Ray utilizza il sottosegmento per generare un segmento dedotto per l'API.

Example Sottosegmento per una chiamata HTTP a valle

```
{
  "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 Segmento dedotto per una chiamata HTTP a valle

```
{
  "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
}
```

Tracciamento delle query SQL con l'SDK X-Ray per .NET

X-Ray SDK for .NET fornisce una classe wrapper per`System.Data.SqlClient.SqlCommand`, denominato`TraceableSqlCommand`, che puoi utilizzare al posto di`SqlCommand`. Puoi inizializzare un comando SQL con la classe `TraceableSqlCommand`.

Tracciamento di query SQL con metodi sincroni e asincroni

I seguenti esempi mostrano come utilizzare `TraceableSqlCommand` per tracciare automaticamente query SQL Server in modo sincrono e asincrono.

Example **`Controller.cs`** - Analisi client SQL (sincrono)

```
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();
  }
}
```

Puoi eseguire la query in modo asincrono utilizzando il metodo `ExecuteReaderAsync`.

Example **`Controller.cs`** - Analisi client SQL (Asincrono)

```
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();
  }
}
```

Raccolta di query SQL effettuate su SQL Server

Puoi abilitare l'acquisizione di `SqlCommand.CommandText` come parte del sottosegmento creato dalla query SQL. `SqlCommand.CommandText` viene visualizzato come il campo `sanitized_query` nel sottosegmento JSON. Per impostazione predefinita, questa caratteristica è disabilitata per motivi di sicurezza.

> **Note**
>
> Non abilitare la caratteristica di raccolta se si includono informazioni sensibili come testo in chiaro nelle query SQL.

Puoi abilitare la raccolta di query SQL in due modi:

- Imposta la proprietà `CollectSqlQueries` su `true` nella configurazione globale dell'applicazione.
- Imposta il parametro `collectSqlQueries` nell'istanza `TraceableSqlCommand` su `true` per raccogliere le chiamate all'interno dell'istanza.

Abilitazione della proprietà CollectSqlQueries globale

I seguenti esempi mostrano come abilitare la proprietà `CollectSqlQueries` per .NET e .NET Core.

.NET

Per impostare la proprietà `CollectSqlQueries` su `true` nella configurazione globale dell'applicazione in .NET, modifica la `appsettings` del file `Web.config` o `App.config`, come mostrato.

Example **`App.config`**O**`Web.config`**— Abilitazione della raccolta di query SQL a livello globale

```
<configuration>
<appSettings>
    <add key="CollectSqlQueries" value="true">
</appSettings>
</configuration>
```

.NET Core

Per impostare il`CollectSqlQueries`proprietà `atrue`nella configurazione globale dell'applicazione in .NET Core, modifica la`appsettings.json`file sotto il tasto X-Ray, come mostrato.

Example **`appsettings.json`**— Abilitazione della raccolta di query SQL a livello globale

```
{
  "XRay": {
    "CollectSqlQueries":"true"
  }
}
```

Abilitazione del parametro collectSqlQueries

Puoi impostare il parametro `collectSqlQueries` nell'istanza `TraceableSqlCommand` su per `true` per raccogliere il testo della query SQL per le query SQL Server eseguite utilizzando tale istanza. L'impostazione del parametro su `false` disabilita la caratteristica `CollectSqlQuery` per l'istanza `TraceableSqlCommand`.

> **Note**
>
> Il valore di `collectSqlQueries` nell'istanza `TraceableSqlCommand` sostituisce il valore impostato nella configurazione globale della proprietà `CollectSqlQueries`.

Example Esempio**`Controller.cs`**— Abilitazione della raccolta di query SQL per l'istanza

```
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();
  }
}
```

Creazione di sottosegmenti aggiuntivi

I sottosegmenti estendono una tracciasegmentocon dettagli sul lavoro svolto per soddisfare una richiesta. Ogni volta che si effettua una chiamata con un client strumentato, l'SDK X-Ray registra le informazioni generate in un sottosegmento. È possibile creare sottosegmenti aggiuntivi per raggruppare altri segmenti secondari, per misurare le prestazioni di una sezione di codice o per registrare annotazioni e metadati.

Per gestire i sottosegmenti, utilizza i metodi `BeginSubsegment` e `EndSubsegment`. Esegui qualsiasi attività nel sottosegmento in un blocco `try` e utilizza `AddException` per tracciare le eccezioni. Chiama `EndSubsegment` in un blocco `finally` per assicurarti che il sottosegmento venga chiuso.

Example Controller.cs - Sottosegmento personalizzato

```
AWSXRayRecorder.Instance.BeginSubsegment("custom method");
try
{
  DoWork();
}
catch (Exception e)
{
  AWSXRayRecorder.Instance.AddException(e);
}
finally
{
  AWSXRayRecorder.Instance.EndSubsegment();
}
```

Quando crei un sottosegmento all'interno di un segmento o di un altro sottosegmento, l'SDK X-Ray per .NET genera per esso un ID e memorizza l'ora di inizio e fine.

Example Sottosegmento con metadati

```
"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"
    }
  },
```

Aggiungi annotazioni e metadati ai segmenti con X-Ray SDK for .NET

È possibile utilizzare annotazioni e metadati per registrare informazioni aggiuntive sulle richieste, sull'ambiente o sull'applicazione. Puoi aggiungere annotazioni e metadati ai segmenti creati da X-Ray SDK o ai sottosegmenti personalizzati che crei.

Le annotazioni sono coppie chiave-valore con stringhe, numeri o valori booleani. Le annotazioni sono indicizzate per essere utilizzate con le espressioni di filtro. Utilizzale per registrare i dati che desideri utilizzare per raggruppare le tracce nella console oppure per chiamare l'API `GetTraceSummaries`.

I metadati sono coppie chiave-valore che possono avere valori di qualsiasi tipo, inclusi oggetti ed elenchi, ma non sono indicizzati per essere utilizzati con le espressioni di filtro. Utilizzate i metadati per registrare dati aggiuntivi che desiderate archiviare nella traccia ma che non è necessario utilizzare con la ricerca.

Sections

- Registrazione delle annotazioni con X-Ray SDK for .NET
- Registrazione di metadati con X-Ray SDK for .NET

Registrazione delle annotazioni con X-Ray SDK for .NET

Utilizza le annotazioni per memorizzare le informazioni su segmenti o sottosegmenti che desideri siano indicizzate per la ricerca.

Quanto segue è necessario per tutte le annotazioni in X-Ray:

Requisiti per le annotazioni

- Chiavi: la chiave per un'annotazione a raggi X può contenere fino a 500 caratteri alfanumerici. Non è possibile utilizzare spazi o simboli diversi dal simbolo di sottolineatura (_).
- Valori: il valore di un'annotazione X-Ray può contenere fino a 1.000 caratteri Unicode.
- Il numero di annotazioni: è possibile utilizzare fino a 50 annotazioni per traccia.

Per registrare annotazioni al di fuori di una funzione AWS Lambda

1. Procurarsi un'istanza di `AWSXRayRecorder`.

   ```
   using Amazon.XRay.Recorder.Core;
   ...
   AWSXRayRecorder recorder = AWSXRayRecorder.Instance;
   ```

2. Chiamare `addAnnotation` con una chiave di tipo Stringa e un valore booleano, Int32, Int64, Double o Stringa.

   ```
   recorder.AddAnnotation("mykey", "my value");
   ```

Per registrare annotazioni all'interno di una funzione AWS Lambda

Sia i segmenti che i sottosegmenti all'interno di una funzione Lambda sono gestiti dall'ambiente di runtime Lambda. Se desideri aggiungere un'annotazione a un segmento o sottosegmento all'interno di una funzione Lambda, devi fare quanto segue:

1. Crea il segmento o il sottosegmento all'interno della funzione Lambda.
2. Aggiungi l'annotazione al segmento o al sottosegmento.
3. Termina il segmento o il sottosegmento.

Il seguente esempio di codice mostra come aggiungere un'annotazione a un sottosegmento all'interno di una funzione Lambda:

```
#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();
}
```

L'X-Ray SDK registra le annotazioni come coppie chiave-valore in un `annotations` oggetto nel documento del segmento. La chiamata all'`addAnnotation`operazione due volte con la stessa chiave sovrascrive un valore registrato in precedenza sullo stesso segmento o sottosegmento.

Per trovare tracciamenti con annotazioni contenenti valori specifici, utilizza la parola chiave `annotations.`*`key`* in un'espressione filtro.

Registrazione di metadati con X-Ray SDK for .NET

Usa i metadati per registrare informazioni su segmenti o sottosegmenti che non devi indicizzare per utilizzarli all'interno di una ricerca. I valori dei metadati possono essere stringhe, numeri, valori booleani o qualsiasi altro oggetto che può essere serializzato in un oggetto o array JSON.

Per registrare i metadati

1. Ottieni un'istanza di`AWSXRayRecorder`, come mostrato nel seguente esempio di codice:

   ```
   using Amazon.XRay.Recorder.Core;
   ...
   AWSXRayRecorder recorder = AWSXRayRecorder.Instance;
   ```

2. Chiama `AddMetadata` con uno spazio dei nomi di stringa, una chiave di stringa e un valore di oggetto, come illustrato nel seguente esempio di codice:

   ```
   recorder.AddMetadata("my namespace", "my key", "my value");
   ```

   È inoltre possibile chiamare l'`AddMetadata`operazione utilizzando solo una coppia chiave/valore, come illustrato nel seguente esempio di codice:

```
recorder.AddMetadata("my key", "my value");
```

Se non si specifica un valore per lo spazio dei nomi, viene utilizzato l'SDK X-Ray. `default` La chiamata all'`AddMetadata`operazione due volte con la stessa chiave sovrascrive un valore registrato in precedenza sullo stesso segmento o sottosegmento.

Strumentazione della tua applicazione con Ruby

Esistono due modi per strumentare l'applicazione Ruby per inviare tracce a X-Ray:

- AWS Distro for OpenTelemetry Ruby: una AWS distribuzione che fornisce un set di librerie open source per l'invio di metriche e tracce correlate a più soluzioni di AWS monitoraggio, tra cui Amazon AWS X-Ray e Amazon OpenSearch Service CloudWatch, tramite Distro for Collector.AWS OpenTelemetry
- AWS X-Ray SDK for Ruby — Un insieme di librerie per la generazione e l'invio di tracce a X-Ray tramite il demone X-Ray.

Per ulteriori informazioni, consulta Scelta tra gli AWS SDK Distro for OpenTelemetry e X-Ray.

AWSDistro perOpenTelemetryRubino

ConAWSDistro perOpenTelemetry(ADOT) Ruby, puoi strumentare le tue applicazioni una sola volta e inviare metriche e tracce correlate a piùAWSsoluzioni di monitoraggio, tra cui AmazonCloudWatch,AWS X-Raye AmazonOpenSearchServizio. L'utilizzo di X-Ray con ADOT richiede due componenti: unOpenTelemetrySDKabilitato per l'uso con X-Ray eAWSDistro perOpenTelemetryCollezionistaabilitato per l'uso con X-Ray.

Per iniziare, consulta ilAWSDistro perOpenTelemetryDocumentazione Ruby.

Per ulteriori informazioni sull'utilizzo diAWSDistro perOpenTelemetryconAWS X-Raye altroServizi AWS, vediAWSDistro perOpenTelemetryo ilAWSDistro perOpenTelemetryDocumentazione.

Per ulteriori informazioni sul supporto e l'utilizzo delle lingue, vedereAWSOsservabilità suGitHub.

AWS X-Ray SDK per Ruby

X-Ray SDK è una libreria per applicazioni web Ruby che fornisce classi e metodi per generare e inviare dati di traccia al demone X-Ray. I dati di traccia includono informazioni sulle richieste HTTP in entrata fornite dall'applicazione e sulle chiamate effettuate dall'applicazione ai servizi downstream utilizzando l' AWS SDK, i client HTTP o un client di registrazione attivo. Puoi anche possibile creare manualmente dei segmenti e aggiungere informazioni di debug in annotazioni e metadati.

Puoi scaricare l'SDK aggiungendolo al tuo gemfile ed eseguendo `bundle install`.

Example Gemfile

```
gem 'aws-sdk'
```

Se usi Rails, inizia aggiungendo il middleware X-Ray SDK per tracciare le richieste in arrivo. Un filtro sulla richiesta crea un segmento. Fino a che il segmento è aperto, puoi usare i metodi del client dell'SDK per aggiungere informazioni al segmento e creare sottosegmenti per tracciare le chiamate a valle. L'SDK, inoltre, registra automaticamente le eccezioni sollevate dall'applicazione per il tempo durante il quale il segmento è aperto. Per applicazioni non Rail, puoi creare i segmenti manualmente.

Successivamente, usa l'SDK X-Ray per strumentare i tuoi AWS SDK for Ruby client HTTP e SQL configurando il registratore per applicare patch alle librerie associate. Ogni volta che effettui una chiamata a un downstream Servizio AWS o a una risorsa con un client dotato di strumenti, l'SDK registra le informazioni sulla chiamata in un sottosegmento. Servizi AWS e le risorse a cui accedi all'interno dei servizi vengono visualizzate come nodi a valle sulla mappa di traccia per aiutarti a identificare errori e problemi di limitazione sulle singole connessioni.

Una volta presa confidenza con l'SDK, personalizza il suo comportamento configurando il registratore. Puoi aggiungere dei plugin per memorizzare i dati sulle risorse di elaborazione sulle quali è eseguita la tua applicazione, personalizzare il comportamento di campionamento definendo regole di campionatura e impostare un sistema di generazione dei log per visualizzare più o meno informazioni generate dall'SDK nei log dell'applicazione.

Registra ulteriori informazioni sulle richieste e sull'attività svolta dalla tua applicazione in annotazioni e metadati. Le annotazioni sono semplici coppie chiave-valore indicizzati per l'uso con espressioni filtro, in modo da poter cercare tracce che contengono dati specifici. Le immissioni di metadati sono meno restrittive e possono registrare interi oggetti e array, tutto ciò che può essere serializzato in JSON.

> **Annotazioni e metadata**
>
> Le annotazioni e i metadati sono testo arbitrario che aggiungi ai segmenti con X-Ray SDK. Le annotazioni vengono indicizzate per essere utilizzate con le espressioni di filtro. I metadati non sono indicizzati, ma possono essere visualizzati nel segmento non elaborato con la console X-Ray o l'API. Chiunque conceda l'accesso in lettura a X-Ray può visualizzare questi dati.

Quando disponi di una notevole quantità di client analizzati nel tuo codice, un singolo segmento per la richiesta può contenere un numero elevato di sottosegmenti, uno per ciascuna delle chiamate effettuate con un client analizzato. Puoi organizzare e raggruppare i sottosegmenti inglobando le chiamate del client sottosegmenti personalizzati. Puoi creare un sottosegmento personalizzato per un'intera funzione o qualsiasi porzione di codice, e memorizzare metadati e annotazioni sul sottosegmento invece di scrivere tutto all'interno del segmento padre.

Per la documentazione di riferimento per le classi e i metodi dell'SDK, consulta l'AWS X-Ray SDK for Ruby API Reference.

Requisiti

L'X-Ray SDK richiede Ruby 2.3 o successivo ed è compatibile con le seguenti librerie:

- AWS SDK for Ruby versione 3.0 o successiva
- Rails versione 5.1 o successive

Configurazione dell'SDK for Ruby by by by by by by by by by by by by by

X-Ray SDK for Ruby per Ruby.`XRay.recorder` Puoi configurare il registratore globale per personalizzare il middleware che crea i segmenti relativi alle chiamate HTTP in entrata.

Sezioni

- Plugin di servizio
- Regole di campionamento
- Registrazione
- Configurazione del registratore nel codice
- Configurazione del registratore con Rails

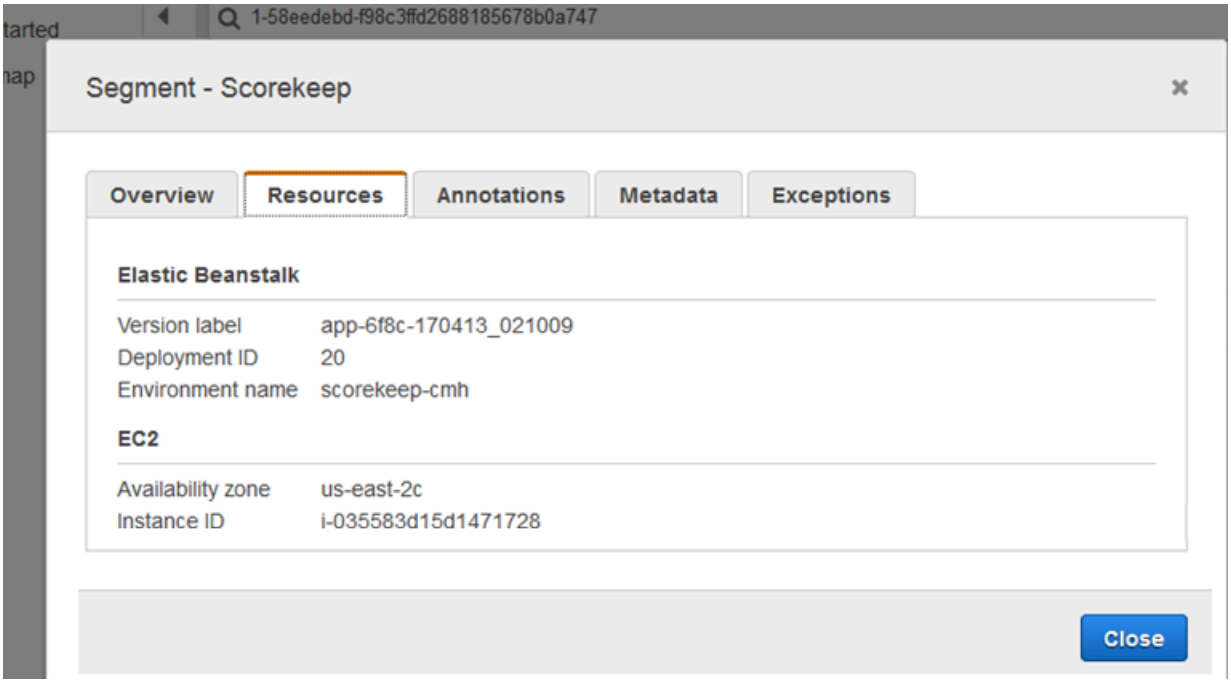
- Variabili di ambiente

Plugin di servizio

`plugins`Da utilizzare per registrare informazioni sul servizio che ospita l'applicazione.

Plug-in

- Amazon EC2:`ec2` aggiunge l'ID istanza e la zona di disponibilità di.
- Elastic Beanstalk:`elastic_beanstalk` aggiunge il nome dell'ambiente, l'etichetta della versione e l'ID di distribuzione.
- Amazon ECS:`ecs` aggiunge l'ID del contenitore.



Per utilizzare i plugin, specificalo nell'oggetto di configurazione che passi al registratore.

Example main.rb: configurazione del plugin

```
my_plugins = %I[ec2 elastic_beanstalk]

config = {
  plugins: my_plugins,
  name: 'my app',
}
```

```
XRay.recorder.configure(config)
```

Per configurare il registratore puoi anche utilizzare le variabili di ambiente, che hanno la precedenza sui valori impostati nel codice.

L'SDK utilizza anche le impostazioni del plugin per impostare il`origin` campo sul segmento. Indica il tipo diAWS risorsa che esegue l'applicazione. Quando si utilizzano più plugin, l'SDK utilizza il seguente ordine di risoluzione per determinare l'origine:ElasticBeanstalk > EKS > ECS > EC2.

Regole di campionamento

L'SDK utilizza le regole di campionamento definite nella console X-Ray per determinare quali richieste registrare. La regola predefinita tiene traccia della prima richiesta ogni secondo e del cinque percento di tutte le richieste aggiuntive su tutti i servizi che inviano tracce a X-Ray. Crea regole aggiuntive nella console X-Ray per personalizzare la quantità di dati registrati per ciascuna delle tue applicazioni.

L'SDK applica le regole personalizzate nell'ordine in cui sono definite. Se una richiesta corrisponde a più regole personalizzate, l'SDK applica solo la prima regola.

> **Note**
>
> Se l'SDK non riesce a raggiungere X-Ray per ottenere le regole di campionamento, torna a una regola locale predefinita della prima richiesta ogni secondo e del cinque percento di eventuali richieste aggiuntive per host. Ciò può verificarsi se l'host non dispone dell'autorizzazione per chiamare le API di campionamento o non riesce a connettersi al demone X-Ray, che funge da proxy TCP per le chiamate API effettuate dall'SDK.

Puoi anche configurare l'SDK per caricare le regole di campionamento da un documento JSON. L'SDK può utilizzare regole locali come backup nei casi in cui il campionamento a X-Ray non è disponibile o utilizzare esclusivamente regole locali.

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
  }
}
```

Questo esempio definisce una regola personalizzata e una regola predefinita. La regola personalizzata applica una frequenza di campionamento del cinque percento senza un numero minimo di richieste da tracciare per i percorsi`/api/move/`. La regola predefinita tiene traccia della prima richiesta ogni secondo e del 10 percento delle altre richieste.

Lo svantaggio della definizione locale delle regole è che l'obiettivo fisso viene applicato da ciascuna istanza del registratore in modo indipendente, anziché essere gestito dal servizio X-Ray. Man mano che si installano più host, la velocità fissa si moltiplica, rendendo più difficile il controllo della quantità di dati registrati.

Per configurare le regole di backup, definire un hash per il documento dell'oggetto di configurazione che si passa al registratore.

Example main.rb — Configurazione delle regole di Backup

```
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)
```

Per archiviare le regole di campionamento in modo indipendente, definire l'hash in un file separato e richiedere i file per inserirli all'interno della propria applicazione.

Example config/sampling-rules.rb

```
my_sampling_rules =  {
  version: 1,
  default: {
    fixed_target: 1,
    rate: 0.1
  }
}
```

Example main.rb — Regola di campionamento da un file

```
require 'aws-xray-sdk'
require 'config/sampling-rules.rb'

config = {
  sampling_rules: my_sampling_rules,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Per utilizzare solo regole locali, richiedere le regole di campionamento e configurare il `LocalSampler`.

Example main.rb — Campionamento di regole locali

```
require 'aws-xray-sdk'
require 'aws-xray-sdk/sampling/local/sampler'

config = {
  sampler: LocalSampler.new,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Puoi anche configurare il registratore globale per disabilitare il campionamento e analizzare tutte le richieste in entrata.

Example main.rb — Disabilita il campionamento

```
require 'aws-xray-sdk'
config = {
  sampling: false,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Registrazione

Per impostazione predefinita, il registratore genera gli eventi a livello di informazioni su `$stdout`. Puoi personalizzare la generazione dei log definendo un logger nell'oggetto di configurazione da passare al registratore.

Example main.rb — Registrazione

```
require 'aws-xray-sdk'
config = {
  logger: my_logger,
  name: 'my app',
}
XRay.recorder.configure(config)
```

Utilizza i log di debug per identificare i problemi, come ad esempio dei sottosegmenti non chiusi, quando generi dei sottosegmenti manualmente.

Configurazione del registratore nel codice

Impostazioni aggiuntive sono disponibili tramite il metodo `configure` su `XRay.recorder`.

- `context_missing`— Impostato`LOG_ERROR` per evitare di generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.
- `daemon_address`— Imposta l'host e la porta del listener del demone X-Ray.
- `name`— Imposta un nome di servizio che l'SDK utilizza per i segmenti.
- `naming_pattern`— Imposta un modello di nome di dominio per utilizzare la denominazione dinamica.
- `plugins`— Registra informazioni sulleAWS risorse dell'applicazione con i plugin.
- `sampling`— Impostare su`false` per disabilitare il campionamento.
- `sampling_rules`— Imposta l'hash contenente le tue regole di campionamento.

Example main.rb — Disabilita le eccezioni mancanti del contesto

```
require 'aws-xray-sdk'
config = {
  context_missing: 'LOG_ERROR'
}

XRay.recorder.configure(config)
```

Configurazione del registratore con Rails

Se utilizzi il framework Rails, puoi configurare le opzioni sul registratore globale in un file Ruby sotto `app_root/initializers`. L'SDK X-Ray supporta una chiave di configurazione aggiuntiva da utilizzare con Rails.

- `active_record`— Impostato per`true` registrare sottosegmenti per le transazioni del database Active Record.

Configura le impostazioni disponibili in un oggetto di configurazione denominato `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
}
```

Variabili di ambiente

Puoi utilizzare le variabili di ambiente per configurare l'SDK for Ruby by by by Ruby. L'SDK supporta le seguenti variabili:

- `AWS_XRAY_TRACING_NAME`— Imposta un nome di servizio che l'SDK utilizza per i segmenti. Sostituisce il nome del servizio impostato sulla strategia di denominazione dei segmenti del filtro servlet.
- `AWS_XRAY_DAEMON_ADDRESS`— Imposta l'host e la porta del listener del demone X-Ray. Per impostazione predefinita, l'SDK invia i dati di traccia a`127.0.0.1:2000`. Usa questa variabile

se hai configurato il demone per l'ascolto su una porta diversa o se è in esecuzione su un host diverso.

- `AWS_XRAY_CONTEXT_MISSING`— Impostato`RUNTIME_ERROR` per generare eccezioni quando il codice strumentato tenta di registrare dati quando nessun segmento è aperto.

  Valori validi

  - `RUNTIME_ERROR`— Genera un'eccezione di runtime.
  - `LOG_ERROR`— Registra un errore e continua (impostazione predefinita).
  - `IGNORE_ERROR`— Ignora l'errore e continua.

  Gli errori relativi a segmenti o sottosegmenti mancanti possono verificarsi quando si tenta di utilizzare un client strumentato nel codice di avvio che viene eseguito quando nessuna richiesta è aperta o nel codice che genera un nuovo thread.

Le variabili di ambiente sostituiscono i valori impostati nel codice.

Tracciamento delle richieste in arrivo con il middleware X-Ray SDK for Ruby

Puoi utilizzare l'SDK X-Ray per tracciare le richieste HTTP in entrata che la tua applicazione serve su un'istanza EC2 in Amazon EC2 o Amazon ECS. AWS Elastic Beanstalk

Se utilizzi Rails, utilizza il middleware Rails per analizzare le richieste HTTP in entrata. Quando aggiungi il middleware all'applicazione e configuri il nome di un segmento, X-Ray SDK for Ruby crea un segmento per ogni richiesta campionata. Tutti i segmenti creati da analisi aggiuntive diventano sottosegmenti del segmento a livello di richiesta che offre informazioni sulla richiesta e sulla risposta HTTP. Queste informazioni includono durata, metodo e conclusione della richiesta HTTP.

Ogni segmento ha un nome che identifica l'applicazione nella mappa del servizio. Il segmento può essere denominato staticamente oppure è possibile configurare l'SDK per denominarlo dinamicamente in base all'intestazione dell'host nella richiesta in entrata. La denominazione dinamica consente di raggruppare le tracce in base al nome di dominio nella richiesta e di applicare un nome predefinito se il nome non corrisponde a uno schema previsto (ad esempio, se l'intestazione dell'host è falsificata).

> **Richieste inoltrate**
>
> Se un sistema di bilanciamento del carico o un altro intermediario inoltra una richiesta all'applicazione, X-Ray prende l'IP del client dall'`X-Forwarded-For`intestazione della

richiesta anziché dall'IP di origine nel pacchetto IP. L'IP del client registrato per una richiesta inoltrata può essere falsificato, quindi non dovrebbe essere considerato attendibile.

Quando viene inoltrata una richiesta, l'SDK imposta un campo aggiuntivo nel segmento per indicarlo. Se il segmento contiene il campo `x_forwarded_for` impostato su`true`, l'IP del client è stato preso dall'`X-Forwarded-For`intestazione della richiesta HTTP.

Il middleware crea un segmento per ogni richiesta in entrata con un blocco `http` che contiene le informazioni riportate qui di seguito:

- Metodo HTTP: GET, POST, PUT, DELETE, ecc.
- Indirizzo client: l'indirizzo IP del client che ha inviato la richiesta.
- Codice di risposta: il codice di risposta HTTP per la richiesta completata.
- Tempistica: l'ora di inizio (quando è stata ricevuta la richiesta) e l'ora di fine (quando è stata inviata la risposta).
- Agente utente: il `user-agent` codice della richiesta.
- Lunghezza del contenuto: il `content-length` risultato della risposta.

Utilizzo del middleware Rails

Per usare il middleware, aggiorna il tuo gemfile per includere il railtie necessario.

Example Gemfile - rails

```
gem 'aws-xray-sdk', require: ['aws-xray-sdk/facets/rails/railtie']
```

Per utilizzare il middleware, è inoltre necessario configurare il registratore con un nome che rappresenti l'applicazione nella mappa di traccia.

Example config/initializers/aws_xray.rb

```
Rails.application.config.xray = {
  name: 'my app'
}
```

Analisi manuale del codice

Se non utilizzi Rails, crea i segmenti manualmente. È possibile creare un segmento per ogni richiesta in arrivo o creare segmenti attorno ai client HTTP o AWS SDK con patch per fornire al registratore il contesto in cui aggiungere sottosegmenti.

```
# 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
```

Configurazione di una strategia di denominazione dei segmenti

AWS X-Ray utilizza un nome di servizio per identificare l'applicazione e distinguerla dalle altre applicazioni, database, API esterne e risorse utilizzate dall'applicazione. AWS Quando X-Ray SDK genera segmenti per le richieste in entrata, registra il nome del servizio dell'applicazione nel campo del nome del segmento.

L'X-Ray SDK può denominare i segmenti dopo il nome host nell'intestazione della richiesta HTTP. Tuttavia, questa intestazione può essere falsificata, il che potrebbe causare nodi imprevisti nella mappa dei servizi. Per evitare che l'SDK nomini i segmenti in modo errato a causa di richieste con intestazioni host contraffatte, è necessario specificare un nome predefinito per le richieste in entrata.

Se la tua applicazione soddisfa le richieste per più domini, puoi configurare l'SDK in modo che utilizzi una strategia di denominazione dinamica che rifletta questo aspetto nei nomi dei segmenti. Una

strategia di denominazione dinamica consente all'SDK di utilizzare il nome host per le richieste che corrispondono a uno schema previsto e di applicare il nome predefinito alle richieste che non lo fanno.

Ad esempio, potresti avere una singola applicazione che serve le richieste a tre sottodomini:, e. `www.example.com` `api.example.com` `static.example.com` È possibile utilizzare una strategia di denominazione dinamica con lo schema `*.example.com` per identificare i segmenti per ogni sottodominio con un nome diverso, ottenendo tre nodi di servizio sulla mappa dei servizi. Se l'applicazione riceve richieste con un nome host che non corrisponde allo schema, sulla mappa dei servizi verrà visualizzato un quarto nodo con un nome di fallback specificato dall'utente.

Per utilizzare lo stesso nome per tutti i segmenti della richiesta, specifica il nome della tua applicazione quando configuri il registratore, come illustrato nelle sezioni precedenti.

Una strategia di denominazione dinamica definisce un modello al quale devono corrispondere i nomi degli host e un nome di default per l'utilizzo qualora il nome dell'host nella richiesta HTTP non corrisponda al modello. Per denominare segmenti in modo dinamico, specificare un modello di denominazione nel config hash.

Example main.rb — Denominazione dinamica

```
config = {
  naming_pattern: '*mydomain*',
  name: 'my app',
}

XRay.recorder.configure(config)
```

Puoi utilizzare '*' nel modello per una corrispondenza con qualsiasi stringa o '?' per una corrispondenza con qualsiasi carattere singolo.

> **Note**
>
> È possibile sovrascrivere il nome di servizio predefinito definito nel codice con la variabile di ambiente. `AWS_XRAY_TRACING_NAME`

Applicare le patch alle librerie per analizzare le chiamate a valle

Per strumentare le chiamate downstream, utilizzate l'X-Ray SDK per Ruby per applicare patch alle librerie utilizzate dall'applicazione. L'X-Ray SDK per Ruby può applicare patch alle seguenti librerie.

Librerie supportate

- `net/http`— Client HTTP dello strumento.
- `aws-sdk`— StrumentoAWS SDK for Rubyclienti.

Quando si utilizza una libreria con patch, X-Ray SDK per Ruby crea un sottosegmento per la chiamata e registra le informazioni della richiesta e della risposta. Un segmento deve essere disponibile affinché l'SDK possa creare il sottosegmento, sia tramite il middleware dell'SDK che tramite una chiamata a `XRay.recorder.begin_segment`.

Per applicare patch alle librerie, specificatele nell'oggetto di configurazione che passate al registratore X-Ray.

Example main.rb — Librerie di patch

```
require 'aws-xray-sdk'

config = {
  name: 'my app',
  patch: %I[net_http aws_sdk]
}

XRay.recorder.configure(config)
```

Tracciamento delle chiamate AWS SDK con X-Ray SDK for Ruby

Quando l'applicazione effettua chiamate per Servizi AWS archiviare dati, scrivere in una coda o inviare notifiche, X-Ray SDK for Ruby tiene traccia delle chiamate downstream in sottosegmenti. Le risorse tracciate Servizi AWS e a cui accedi all'interno di tali servizi (ad esempio, un bucket Amazon S3 o una coda Amazon SQS) vengono visualizzate come nodi downstream sulla mappa di traccia nella console X-Ray.

L'X-Ray SDK for Ruby strumenta automaticamente AWS tutti i client SDK quando applichi una patch alla libreria. `aws-sdk` Non puoi analizzare singoli client.

Per tutti i servizi, puoi vedere il nome dell'API richiamata nella console X-Ray. Per un sottoinsieme di servizi, l'SDK X-Ray aggiunge informazioni al segmento per fornire una maggiore granularità nella mappa dei servizi.

Ad esempio, quando si effettua una chiamata con un client DynamoDB con strumentazione, l'SDK aggiunge il nome della tabella al segmento per le chiamate destinate a una tabella. Nella console, ogni tabella appare come un nodo separato nella mappa dei servizi, con un nodo DynamoDB generico per le chiamate che non hanno come destinazione una tabella.

Example Sottosegmento per una chiamata a DynamoDB per salvare un elemento

```
{
  "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",
  }
}
```

Quando si accede alle risorse con nome, le chiamate ai seguenti servizi creano ulteriori nodi della mappa del servizio. Le chiamate che non sono hanno come obiettivo risorse specifiche creano un nodo generico per il servizio.

- Amazon DynamoDB: nome della tabella
- Amazon Simple Storage Service: nome del bucket e della chiave
- Amazon Simple Queue Service: nome della coda

Generazione di sottosegmenti personalizzati con l'SDK X-Ray

I sottosegmenti estendono una tracciasegmentocon dettagli sul lavoro svolto per soddisfare una richiesta. Ogni volta che si effettua una chiamata con un client strumentato, l'SDK X-Ray registra le informazioni generate in un sottosegmento. È possibile creare sottosegmenti aggiuntivi per raggruppare altri segmenti secondari, per misurare le prestazioni di una sezione di codice o per registrare annotazioni e metadati.

Per gestire i sottosegmenti, utilizza i metodi `begin_subsegment` e `end_subsegment`.

```
subsegment = XRay.recorder.begin_subsegment name: 'annotations', namespace: 'remote'
my_annotations = { id: 12345 }
subsegment.annotations.update my_annotations
XRay.recorder.end_subsegment
```

Per creare un subsegment per una funzione, eseguire il wrapping in una chiamata a `XRay.recorder.capture`.

```
XRay.recorder.capture('name_for_subsegment') do |subsegment|
  resp = myfunc() # myfunc is your function
  subsegment.annotations.update k1: 'v1'
  resp
end
```

Quando crei un sottosegmento all'interno di un segmento o di un altro sottosegmento, l'SDK X-Ray genera per esso un ID e memorizza l'ora di inizio e fine.

Example Sottosegmento con metadati

```
"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"
    }
  },
```

Aggiungi annotazioni e metadati ai segmenti con l'X-Ray SDK for Ruby

È possibile utilizzare annotazioni e metadati per registrare informazioni aggiuntive sulle richieste, sull'ambiente o sull'applicazione. Puoi aggiungere annotazioni e metadati ai segmenti creati da X-Ray SDK o ai sottosegmenti personalizzati che crei.

Le annotazioni sono coppie chiave-valore con stringhe, numeri o valori booleani. Le annotazioni sono indicizzate per essere utilizzate con le espressioni di filtro. Utilizzale per registrare i dati che desideri utilizzare per raggruppare le tracce nella console oppure per chiamare l'API `GetTraceSummaries`.

I metadati sono coppie chiave-valore che possono avere valori di qualsiasi tipo, inclusi oggetti ed elenchi, ma non sono indicizzati per essere utilizzati con le espressioni di filtro. Utilizzate i metadati per registrare dati aggiuntivi che desiderate archiviare nella traccia ma che non è necessario utilizzare con la ricerca.

Oltre ad annotazioni e metadati, sui segmenti puoi anche registrare le stringhe degli ID utente. Gli ID utente vengono memorizzati in un campo separato su segmenti e sono indicizzati per l'uso nelle ricerche.

Sections

- Registrazione delle annotazioni con X-Ray SDK for Ruby
- Registrazione di metadati con X-Ray SDK for Ruby
- Registrazione degli ID utente con X-Ray SDK for Ruby

Registrazione delle annotazioni con X-Ray SDK for Ruby

Utilizza le annotazioni per memorizzare le informazioni su segmenti o sottosegmenti che desideri siano indicizzate per la ricerca.

Requisiti per le annotazioni

- Chiavi: la chiave per un'annotazione a raggi X può contenere fino a 500 caratteri alfanumerici. Non è possibile utilizzare spazi o simboli diversi dal simbolo di sottolineatura (_).
- Valori: il valore di un'annotazione X-Ray può contenere fino a 1.000 caratteri Unicode.
- Il numero di annotazioni: è possibile utilizzare fino a 50 annotazioni per traccia.

Per registrare le annotazioni

1. Ottenere un riferimento al segmento o sottosegmento corrente da `xray_recorder`.

   ```
   require 'aws-xray-sdk'
   ...
   document = XRay.recorder.current_segment
   ```

   oppure

   ```
   require 'aws-xray-sdk'
   ...
   document = XRay.recorder.current_subsegment
   ```

2. Chiama `update` con un valore hash.

   ```
   my_annotations = { id: 12345 }
   document.annotations.update my_annotations
   ```

L'SDK memorizza le annotazioni come coppie chiave-valore in un oggetto `annotations` all'interno del documento di segmento. Se chiami `add_annotations` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento o sottosegmento viene sovrascritto.

Per trovare tracciamenti con annotazioni contenenti valori specifici, utilizza la parola chiave `annotations.`*`key`* in un'espressione filtro.

Registrazione di metadati con X-Ray SDK for Ruby

Utilizza i metadati per memorizzare le informazioni su segmenti o sottosegmenti che non è necessario che siano indicizzate per la ricerca. I valori dei metadati possono essere stringhe, numeri, valori booleani o qualsiasi oggetto che possa essere serializzato in un oggetto o in un vettore JSON.

Per registrare i metadati

1. Ottenere un riferimento al segmento o sottosegmento corrente da `xray_recorder`.

   ```
   require 'aws-xray-sdk'
   ...
   document = XRay.recorder.current_segment
   ```

oppure

```
require 'aws-xray-sdk'
...
document = XRay.recorder.current_subsegment
```

2. Chiama `metadata` con una chiave di tipo Stringa, un valore booleano, numerico, di tipo Stringa o di tipo oggetto e un namespace di tipo stringa.

```
my_metadata = {
  my_namespace: {
    key: 'value'
  }
}
subsegment.metadata my_metadata
```

Se chiami `metadata` due volte con la stessa chiave, il valore precedentemente memorizzato nello stesso segmento o sottosegmento viene sovrascritto.

Registrazione degli ID utente con X-Ray SDK for Ruby

Memorizza gli ID utente sui segmenti di richiesta per identificare l'utente che ha inviato la richiesta.

Per registrare gli ID degli utenti

1. Ottenere un riferimento al segmento corrente da `xray_recorder`.

```
require 'aws-xray-sdk'
...
document = XRay.recorder.current_segment
```

2. Imposta il campo utente sul segmento a un ID di tipo stringa dell'utente che ha inviato la richiesta.

```
segment.user = 'U12345'
```

Puoi impostare l'utente nei tuoi controller per registrare l'ID utente non appena l'applicazione inizia ad elaborare una richiesta.

Per trovare tracciamenti associati ad un ID utente, utilizza la parola chiave `user` in un'espressione filtro.

Integrazione AWS X-Ray con altri Servizi AWS

Molti Servizi AWS offrono diversi livelli di integrazione con i raggi X, tra cui il campionamento e l'aggiunta di intestazioni alle richieste in arrivo, l'esecuzione del demone X-Ray e l'invio automatico di dati di traccia a X-Ray. L'integrazione con X-Ray può includere quanto segue:

- Strumentazione attiva: campioni e strumenti (richieste in arrivo)
- Strumentazione passiva: richieste di strumenti che sono state campionate da un altro servizio
- Tracciamento delle richieste: aggiunge un'intestazione di tracciamento a tutte le richieste in arrivo e la propaga a valle
- Strumenti: esegue il demone X-Ray per ricevere segmenti dall'SDK X-Ray

> **Note**
>
> Gli X-Ray SDK includono plugin per un'ulteriore integrazione con. Servizi AWS Ad esempio, puoi utilizzare il plug-in X-Ray SDK for Java Elastic Beanstalk per aggiungere informazioni sull'ambiente Elastic Beanstalk che esegue l'applicazione, inclusi il nome e l'ID dell'ambiente.

Ecco alcuni esempi Servizi AWS integrati con X-Ray:

- AWS Distro for OpenTelemetry (ADOT): con ADOT, gli ingegneri possono strumentare le proprie applicazioni una sola volta e inviare metriche e tracce correlate a più AWS soluzioni di monitoraggio tra cui Amazon CloudWatch, AWS X-Ray Amazon Service e Amazon Managed Service for OpenSearch Prometheus.
- AWS Lambda— Strumentazione attiva e passiva delle richieste in arrivo in tutte le fasi di esecuzione. AWS Lambda aggiunge due nodi alla mappa di traccia, uno per il AWS Lambda servizio e uno per la funzione. Quando si abilita la strumentazione, esegue AWS Lambda anche il demone X-Ray su Java e i runtime Node.js da utilizzare con l'SDK X-Ray.
- Amazon API Gateway: strumentazione attiva e passiva. API Gateway utilizza regole di campionamento per determinare quali richieste registrare e aggiunge un nodo per la fase gateway alla mappa dei servizi.
- AWS Elastic Beanstalk— Utensili. Elastic Beanstalk include il demone X-Ray sulle seguenti piattaforme:
  - Java SE — 2.3.0 e configurazioni successive

- Tomcat — 2.4.0 e configurazioni successive
- Node.js — 3.2.0 e configurazioni successive
- Windows Server: tutte le configurazioni diverse da Windows Server Core rilasciate dopo il 9 dicembre 2016

  Puoi usare la console Elastic Beanstalk per dire a Elastic Beanstalk di eseguire il demone su queste piattaforme oppure utilizzare l'opzione nel namespace. `XRayEnabled` `aws:elasticbeanstalk:xray`

- Elastic Load Balancing: traccia delle richieste sugli Application Load Balancer. L'Application Load Balancer aggiunge l'ID di traccia all'intestazione della richiesta prima di inviarlo a un gruppo target.
- Amazon EventBridge — Strumentazione passiva. Se un servizio che pubblica eventi su EventBridge è dotato di strumentazione con X-Ray SDK, le destinazioni degli eventi riceveranno l'intestazione di tracciamento e potranno continuare a propagare l'ID di traccia originale.
- Amazon Simple Notification Service — Strumentazione passiva. Se un publisher Amazon SNS traccia il proprio client con l'SDK X-Ray, gli abbonati possono recuperare l'intestazione di tracciamento e continuare a propagare la traccia originale dell'editore con lo stesso ID di traccia.
- Amazon Simple Queue Service: strumentazione passiva. Se un servizio traccia le richieste utilizzando l'SDK X-Ray, Amazon SQS può inviare l'intestazione di tracciamento e continuare a propagare la traccia originale dal mittente al consumatore con un ID di traccia coerente.

Scegli tra i seguenti argomenti per esplorare il set completo di soluzioni integrate. Servizi AWS

Argomenti

- AWSDistro perOpenTelemetryeAWS X-Ray
- Supporto di tracciamento attivo di Amazon API Gateway per AWS X-Ray
- Amazon EC2 e AWS App Mesh
- AWSApp Runner e X-Ray
- AWS AppSync e AWS X-Ray
- Registrazione delle chiamate all'API X-Ray con AWS CloudTrail
- CloudWatch integrazione con X-Ray
- Tracciamento delle modifiche alla configurazione della crittografia a X-Ray conAWS Config
- Amazon Elastic Compute Cloud eAWS X-Ray
- AWS Elastic Beanstalk e AWS X-Ray

- Elastic Load Balancing e AWS X-Ray
- Amazon EventBridge e AWS X-Ray
- AWS Lambda e AWS X-Ray
- Amazon SNS e AWS X-Ray
- AWS Step Functions e AWS X-Ray
- Amazon SQS e AWS X-Ray
- Amazon S3 e AWS X-Ray

AWSDistro perOpenTelemetryeAWS X-Ray

Usa ilAWSDistro perOpenTelemetry(ADOT) per raccogliere e inviare metriche e tracce aAWS X-Raye altre soluzioni di monitoraggio, come AmazonCloudWatch, AmazonOpenSearchService e Amazon Managed Service per Prometheus.

AWS Distro per OpenTelemetry

IlAWSDistro perOpenTelemetry(ADOT) è unAWSdistribuzione basata sulla Cloud Native Computing Foundation (CNCF)OpenTelemetryprogetto. OpenTelemetryfornisce un unico set di API, librerie e agenti open source per raccogliere tracce e metriche distribuite. Questo toolkit è una distribuzione di upstreamOpenTelemetrycomponenti tra cui SDK, agenti di strumentazione automatica e raccoglitori testati, ottimizzati, protetti e supportati daAWS.

Con ADOT, gli ingegneri possono strumentare le loro applicazioni una sola volta e inviare metriche e tracce correlate a piùAWSsoluzioni di monitoraggio tra cui AmazonCloudWatch,AWS X-Ray, AmazonOpenSearchService e Amazon Managed Service per Prometheus.

ADOT è integrato con un numero crescente diServizi AWSper semplificare l'invio di tracce e metriche a soluzioni di monitoraggio come X-Ray. Alcuni esempi di servizi integrati con ADOT includono:

- AWS Lambda—AWSi livelli Lambda gestiti per ADOT forniscono unplug-and-playesperienza utente mediante la strumentazione automatica di una funzione Lambda, la creazione di pacchettiOpenTelemetryinsieme a unout-of-the-boxconfigurazione perAWS Lambdae X-Ray in un livello facile da configurare. Gli utenti possono abilitare e disabilitareOpenTelemetryper la loro funzione Lambda senza modificare il codice. Per ulteriori informazioni, vedereAWSDistro perOpenTelemetryLambda
- Amazon Elastic Container Service (ECS)— Raccogli metriche e tracce dalle applicazioni Amazon ECS utilizzando ilAWSDistro perOpenTelemetryCollector, da inviare a X-Ray e altre

soluzioni di monitoraggio. Per ulteriori informazioni, vedereRaccolta dei dati di tracciamento delle applicazioninella guida per sviluppatori di Amazon ECS.

- AWSApp Runner— App Runner supporta l'invio di tracce a X-Ray utilizzando ilAWSDistro perOpenTelemetry(ADOTTARE). Usa gli SDK ADOT per raccogliere dati di traccia per le tue applicazioni containerizzate e usa X-Ray per analizzare e ottenere informazioni dettagliate sulla tua applicazione strumentata. Per ulteriori informazioni, consultaAWSApp Runner e X-Ray.

Per ulteriori informazioni sulAWSDistro perOpenTelemetry, inclusa l'integrazione con altriServizi AWS, vedi ilAWSDistro perOpenTelemetryDocumentazione.

Per ulteriori informazioni sulla strumentazione dell'applicazione conAWSDistro perOpenTelemetrye X-Ray, vediStrumentate la vostra applicazione conAWSDistro perOpenTelemetry.

Supporto di tracciamento attivo di Amazon API Gateway per AWS X-Ray

Puoi usare X-Ray per tracciare e analizzare le richieste degli utenti mentre viaggiano attraverso le API di Amazon API Gateway verso i servizi sottostanti. API Gateway supporta il tracciamento a raggi X per tutti i tipi di endpoint API Gateway: regionali, ottimizzati per l'edge e privati. Puoi usare X-Ray con Amazon API Gateway ovunque Regioni AWS X-Ray sia disponibile. Per ulteriori informazioni, consulta Trace API Gateway API Execution with AWS X-Ray nella Amazon API Gateway Developer Guide.

> **Note**
>
> X-Ray supporta solo il tracciamento per le API REST tramite API Gateway.

Amazon API Gateway fornisce supporto di tracciamento attivo per AWS X-Ray. Abilita il tracciamento attivo sulle fasi dell'API per campionare le richieste in arrivo e inviare tracce a X-Ray.

Per abilitare il tracciamento attivo su una fase API

1. Aprire la console Gateway API all'indirizzo https://console.aws.amazon.com/apigateway/.
2. Scegliere un'API.
3. Scegliere una fase.

4. Nella scheda Log/Tracciamento, scegli Abilita tracciamento X-Ray, quindi scegli Salva modifiche.
5. Scegliere Resources (Risorse) sul pannello di navigazione a sinistra.
6. Per ridistribuire l'API con le nuove impostazioni, scegli il menu a discesa Azioni, quindi scegli Deploy API.

API Gateway utilizza le regole di campionamento definite nella console X-Ray per determinare quali richieste registrare. È possibile creare regole che si applicano solo alle API o che si applicano solo alle richieste che contengono determinate intestazioni. API Gateway registra le intestazioni negli attributi del segmento, insieme ai dettagli sulla fase e sulla richiesta. Per ulteriori informazioni, consulta Configurazione delle regole di campionamento di .

> **Note**
>
> Quando si tracciano le API REST con l'integrazione HTTP di API Gateway, il nome di servizio di ogni segmento viene impostato sul percorso dell'URL della richiesta da API Gateway all'endpoint di integrazione HTTP, ottenendo un nodo di servizio sulla mappa di traccia X-Ray per ogni percorso URL univoco. Un numero elevato di percorsi URL può far sì che la mappa di traccia superi il limite di 10.000 nodi, con conseguente errore.
> Per ridurre al minimo il numero di nodi di servizio creati da API Gateway, valuta la possibilità di passare i parametri all'interno della stringa di query dell'URL o nel corpo della richiesta tramite POST. Entrambi gli approcci garantiranno che i parametri non facciano parte del percorso URL, il che potrebbe comportare un minor numero di percorsi URL e nodi di servizio distinti.

Per tutte le richieste in entrata, API Gateway aggiunge un'intestazione di tracciamento alle richieste HTTP in entrata che non ne hanno già una.

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

Formato ID di traccia X-Ray

Una radiografia `trace_id` è composta da tre numeri separati da trattini. Ad esempio, `1-58406520-a006649127e371903a2de979`. Questo include:

- Il numero di versione, che è. 1
- L'ora della richiesta originale in Unix epoch time utilizzando 8 cifre esadecimali.

Ad esempio, il fuso orario PST delle 10:00 del 1° dicembre 2016 è espresso in secondi o in cifre esadecimali. `1480615200` `58406520`

- Un identificatore a 96 bit univoco a livello globale per la traccia in 24 cifre esadecimali.

Se il tracciamento attivo è disattivato, la fase memorizza comunque un segmento se la richiesta proviene da un servizio che ha campionato la richiesta e ha avviato un tracciamento. Ad esempio, un'applicazione Web strumentata può chiamare un'API API Gateway con un client HTTP. Quando si strumenta un client HTTP con l'SDK X-Ray, aggiunge un'intestazione di tracciamento alla richiesta in uscita che contiene la decisione di campionamento. API Gateway legge l'intestazione di tracciamento e crea un segmento per le richieste campionate.

Se utilizzi API Gateway per generare un SDK Java per la tua API, puoi strumentare il client SDK aggiungendo un gestore di richieste con il client builder, nello stesso modo in cui strumenteresti manualmente un client SDK. AWS Per istruzioni, consulta Tracciamento delle chiamate AWS SDK con X-Ray SDK for Java.

Amazon EC2 e AWS App Mesh

AWS X-Ray si integra con AWS App Meshper gestire i proxy Envoy per i microservizi. App Mesh fornisce una versione di Envoy che è possibile configurare per inviare dati di traccia al demone X-Ray in esecuzione in un contenitore dello stesso task o pod. X-Ray supporta il tracciamento con i seguenti servizi compatibili con App Mesh:

- Amazon Elastic Container Service (Amazon ECS)
- Amazon Elastic Kubernetes Service (Amazon EKS)
- Amazon Elastic Compute Cloud (Amazon EC2)

Utilizza le seguenti istruzioni per abilitare il tracciamento X-Ray tramite App Mesh.

Per configurare il proxy Envoy per inviare dati a X-Ray, imposta la variabile di `ENABLE_ENVOY_XRAY_TRACING` ambiente nella definizione del contenitore.

> **Note**
>
> La versione App Mesh di Envoy attualmente non invia tracce in base alle regole di campionamento configurate. Utilizza invece una frequenza di campionamento fissa del 5%

per la versione 1.16.3 o successiva di Envoy o una frequenza di campionamento del 50% per le versioni di Envoy precedenti alla 1.16.3.

Example Definizione del contenitore Envoy per 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

Per ulteriori informazioni sugli indirizzi delle regioni Envoy disponibili, consulta l'immagine di Envoy nella Guida per l'utente. AWS App Mesh

Per i dettagli sull'esecuzione del demone X-Ray in un contenitore, vedere. Esecuzione del daemon X-Ray su Amazon ECECS ECS ECS ECS EC Per un'applicazione di esempio che include una service mesh, un microservice, un proxy Envoy e un demone X-Ray, distribuisci l'esempio `colorapp` nel repository App Mesh Examples. GitHub

Ulteriori informazioni

- Nozioni di base su AWS App Mesh
- Guida introduttiva AWS App Mesh e Amazon ECS

AWSApp Runner e X-Ray

AWSApp Runner è unServizio AWSche offre un modo rapido, semplice ed economico per la distribuzione dal codice sorgente o da un'immagine del contenitore direttamente a un'applicazione web scalabile e sicura inCloud AWS. Non è necessario apprendere nuove tecnologie, decidere quale servizio di elaborazione utilizzare o sapere come eseguire il provisioning e la configurazione delle risorse AWS. VediChe cos'èAWSApp Runnerper ulteriori informazioni.

AWSApp Runner invia tracce a X-Ray integrandosi conAWSDistro perOpenTelemetry(ADOTTARE). Usa gli SDK ADOT per raccogliere dati di traccia per le tue applicazioni containerizzate e usa X-Ray per analizzare e ottenere informazioni dettagliate sulla tua applicazione strumentata. Per ulteriori informazioni, consultaTracciamento per l'applicazione App Runner con X-Ray.

AWS AppSync e AWS X-Ray

È possibile abilitare e tracciare le richieste perAWSAppSync. Per ulteriori informazioni, consultaTracciamento conAWSX-Rayper istruzioni.

Quando il tracciamento X-Ray è abilitato per unAWSAPI AppSync, unAWSIdentity and Access ManagementRuolo collegato ai serviziviene creato automaticamente nel tuo account con le autorizzazioni appropriate. Ciò consente adAWSAppSync per inviare tracce a X-Ray in modo sicuro.

Registrazione delle chiamate all'API X-Ray con AWS CloudTrail

AWS X-Ray è integrato con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, ruolo o un Servizio AWS. CloudTrail acquisisce tutte le chiamate API per X-Ray come eventi. Le chiamate acquisite includono chiamate dalla console X-Ray e chiamate in codice alle operazioni dell'API X-Ray. Utilizzando le informazioni raccolte da CloudTrail, è possibile determinare la richiesta effettuata a X-Ray, l'indirizzo IP da cui è stata effettuata la richiesta, quando è stata effettuata e ulteriori dettagli.

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con le credenziali utente root o utente.
- Se la richiesta è stata effettuata per conto di un utente di IAM Identity Center.
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro Servizio AWS.

CloudTrail è attivo nel tuo account Account AWS quando crei l'account e hai automaticamente accesso alla cronologia degli CloudTrail eventi. La cronologia CloudTrail degli eventi fornisce un record visualizzabile, ricercabile, scaricabile e immutabile degli ultimi 90 giorni di eventi di gestione registrati in un. Regione AWSPer ulteriori informazioni, consulta Lavorare con la cronologia degli CloudTrail eventi nella Guida per l'utente.AWS CloudTrail Non sono CloudTrail previsti costi per la visualizzazione della cronologia degli eventi.

Per una registrazione continua degli eventi degli Account AWS ultimi 90 giorni, crea un trail o un data store di eventi CloudTrailLake.

CloudTrail sentieri

Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Tutti i percorsi creati utilizzando il AWS Management Console sono multiregionali. È possibile creare un percorso a regione singola o multiregione utilizzando. AWS CLI La creazione di un percorso multiregionale è consigliata in quanto consente di registrare l'intera attività del proprio account Regioni AWS . Se crei un percorso a regione singola, puoi visualizzare solo gli eventi registrati nel percorso. Regione AWS Per ulteriori informazioni sui sentieri, consulta Creazione di un percorso per te Account AWS e Creazione di un percorso per un'organizzazione nella Guida per l'AWS CloudTrail utente.

Puoi inviare gratuitamente una copia dei tuoi eventi di gestione in corso al tuo bucket Amazon S3 CloudTrail creando un percorso, tuttavia ci sono costi di storage di Amazon S3. Per ulteriori informazioni sui CloudTrail prezzi, consulta la pagina Prezzi.AWS CloudTrail Per informazioni sui prezzi di Amazon S3, consulta Prezzi di Amazon S3.

CloudTrail Archivi di dati sugli eventi di Lake

CloudTrail Lake ti consente di eseguire query basate su SQL sui tuoi eventi. CloudTrail Lake converte gli eventi esistenti in formato JSON basato su righe in formato Apache ORC. ORC è un formato di archiviazione a colonne ottimizzato per il recupero rapido dei dati. Gli eventi vengono aggregati in archivi di dati degli eventi, che sono raccolte di eventi immutabili basate sui criteri selezionati applicando i selettori di eventi avanzati. I selettori applicati a un archivio di dati degli eventi controllano quali eventi persistono e sono disponibili per l'esecuzione della query. Per

ulteriori informazioni su CloudTrail Lake, consulta Working with AWS CloudTrail Lake nella Guida per l'utente.AWS CloudTrail

CloudTrail Gli archivi e le richieste di dati sugli eventi di Lake comportano dei costi. Quando crei un datastore di eventi, scegli l'opzione di prezzo da utilizzare per tale datastore. L'opzione di prezzo determina il costo per l'importazione e l'archiviazione degli eventi, nonché il periodo di conservazione predefinito e quello massimo per il datastore di eventi. Per ulteriori informazioni sui CloudTrail prezzi, consulta Prezzi.AWS CloudTrail

Argomenti

- Eventi di gestione dei raggi X in CloudTrail
- Eventi relativi ai dati a raggi X in CloudTrail
- Esempi di eventi X-Ray

Eventi di gestione dei raggi X in CloudTrail

AWS X-Ray si integra con AWS CloudTrail per registrare le azioni API eseguite da un utente, un ruolo o un utente Servizio AWS in X-Ray. Puoi utilizzarlo CloudTrail per monitorare le richieste API X-Ray in tempo reale e archiviare i log in Amazon S3, Amazon Logs e Amazon Events. CloudWatch CloudWatch X-Ray supporta la registrazione delle seguenti azioni come eventi nei CloudTrail file di registro:

Operazioni API supportate

- PutEncryptionConfig
- GetEncryptionConfig
- CreateGroup
- UpdateGroup
- DeleteGroup
- GetGroup
- GetGroups
- GetInsight
- GetInsightEvents
- GetInsightImpactGraph
- GetInsightSummaries

- GetSamplingStatisticSummaries

Eventi relativi ai dati a raggi X in CloudTrail

Gli eventi di dati forniscono informazioni sulle operazioni sulle risorse eseguite su o in una risorsa (ad esempio PutTraceSegments, che carica i documenti dei segmenti su X-Ray).

Queste operazioni sono definite anche operazioni del piano dei dati. Gli eventi di dati sono spesso attività che interessano volumi elevati di dati. Per impostazione predefinita, CloudTrail non registra gli eventi relativi ai dati. La cronologia CloudTrail degli eventi non registra gli eventi relativi ai dati.

Per gli eventi di dati sono previsti costi aggiuntivi. Per ulteriori informazioni sui CloudTrail prezzi, consulta la sezione AWS CloudTrail Prezzi.

È possibile registrare gli eventi relativi ai dati per i tipi di risorse X-Ray utilizzando la CloudTrail console o AWS CLI le operazioni CloudTrail API. Per ulteriori informazioni su come registrare gli eventi relativi ai dati, vedere Registrazione degli eventi relativi ai dati con AWS Management Console e Registrazione degli eventi relativi ai dati con the AWS Command Line Interface nella Guida per l'utente.AWS CloudTrail

La tabella seguente elenca i tipi di risorse X-Ray per i quali è possibile registrare gli eventi relativi ai dati. La colonna Data event type (console) mostra il valore da scegliere dall'elenco Data event type (console) sulla CloudTrail console. La colonna del valore resources.type mostra il `resources.type` valore da specificare durante la configurazione dei selettori di eventi avanzati utilizzando le API o. AWS CLI CloudTrail La CloudTrail colonna Data API loggate mostra le chiamate API registrate per il tipo di risorsa. CloudTrail

Tipo di evento di dati (console)	valore resources.type	API di dati registrate su CloudTrail
Traccia a raggi X	`AWS::XRay::Trace`	• PutTraceSegments • GetTraceSummaries • GetTraceGraph • GetServiceGraph • BatchGetTraces • GetTimeSeriesServiceStatistics

Tipo di evento di dati (console)	valore resources.type	API di dati registrate su CloudTrail
		• PutTelemetryRecords • GetSamplingTargets

Puoi configurare selettori di eventi avanzati per filtrare `readOnly` i campi `eventName` e registrare solo gli eventi che ritieni importanti. Tuttavia, non è possibile selezionare gli eventi aggiungendo il selettore di `resources.ARN` campo, poiché le tracce X-Ray non hanno ARN. Per ulteriori informazioni su questi campi, consulta l'AWS CloudTrail API AdvancedFieldSelectorReference. Di seguito è riportato un esempio di come eseguire il `put-event-selectors` AWS CLI comando per registrare gli eventi relativi ai dati su un CloudTrail percorso. È necessario eseguire il comando o specificare la regione in cui è stato creato il trail; in caso contrario, l'operazione restituisce un'`InvalidHomeRegionException`eccezione.

```
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"
      }
   ]
}'
```

Esempi di eventi X-Ray

Esempio di evento gestionale, **`GetEncryptionConfig`**

Di seguito è riportato un esempio di registrazione del GetEncryptionConfig registro X-Ray. 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"
}
```

Esempio di evento relativo ai dati, **`PutTraceSegments`**

Di seguito è riportato un esempio di immissione nel registro degli eventi PutTraceSegments dei dati X-Ray. 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 integrazione con X-Ray

AWS X-Ray si integra con CloudWatch Application Signals, CloudWatch RUM e CloudWatch Synthetics per semplificare il monitoraggio dello stato delle applicazioni. Consenti alla tua applicazione Application Signals di monitorare e risolvere i problemi operativi dei tuoi servizi, delle pagine client, dei canali Synthetics e delle dipendenze dei servizi.

Correlando CloudWatch metriche, log e tracce a raggi X, la mappa di tracciamento a raggi X fornisce una end-to-end panoramica dei tuoi servizi per aiutarti a individuare rapidamente i punti deboli delle prestazioni e identificare gli utenti interessati.

Con CloudWatch RUM, è possibile eseguire il monitoraggio reale degli utenti per raccogliere e visualizzare dati lato client sulle prestazioni delle applicazioni Web dalle sessioni utente effettive in tempo quasi reale. Con AWS X-Ray and CloudWatch RUM, è possibile analizzare ed eseguire il debug del percorso della richiesta a partire dagli utenti finali dell'applicazione tramite servizi gestiti a valle. AWS Ciò consente di identificare le tendenze e gli errori di latenza che influiscono sugli utenti finali.

Argomenti

- CloudWatch RUM e AWS X-Ray
- Eseguire il debug di canarini CloudWatch sintetici utilizzando X-Ray

CloudWatch RUM e AWS X-Ray

Con Amazon CloudWatch RUM, puoi eseguire il monitoraggio reale degli utenti per raccogliere e visualizzare dati lato client sulle prestazioni delle tue applicazioni Web da sessioni utente effettive in tempo quasi reale. Con AWS X-Ray and CloudWatch RUM, puoi analizzare ed eseguire il debug del percorso della richiesta partendo dagli utenti finali della tua applicazione attraverso servizi gestiti a valle. AWS Ciò consente di identificare le tendenze e gli errori di latenza che influiscono sugli utenti finali.

Dopo aver attivato il tracciamento a raggi X delle sessioni utente, CloudWatch RUM aggiunge un'intestazione di traccia a raggi X alle richieste HTTP consentite e registra un segmento a raggi X per le richieste HTTP consentite. È quindi possibile visualizzare le tracce e i segmenti di queste sessioni utente in X-Ray CloudWatch e nelle console, inclusa la mappa di tracciamento X-Ray.

> **Note**
>
> CloudWatch RUM non si integra con le regole di campionamento a raggi X. Scegliete invece una percentuale di campionamento quando configurate l'applicazione per utilizzare RUM. CloudWatch Le tracce inviate dal CloudWatch RUM potrebbero comportare costi aggiuntivi. Per ulteriori informazioni, consultare Prezzi di AWS X-Ray.

Per impostazione predefinita, le tracce lato client inviate da CloudWatch RUM non sono connesse alle tracce lato server. Per connettere le tracce lato client con le tracce lato server, configura il client web CloudWatch RUM per aggiungere un'intestazione di traccia X-Ray a queste richieste HTTP.

> **Warning**
>
> La configurazione del client web CloudWatch RUM per aggiungere un'intestazione di traccia X-Ray alle richieste HTTP può causare il fallimento della condivisione delle risorse tra le origini (CORS). Per evitare ciò, aggiungete l'intestazione `X-Amzn-Trace-Id` HTTP all'elenco delle intestazioni consentite nella configurazione CORS del servizio downstream. Se utilizzi API Gateway come downstream, consulta Abilitazione di CORS per una risorsa API REST. Si consiglia vivamente di testare l'applicazione prima di aggiungere un'intestazione di traccia X-Ray lato client in un ambiente di produzione. Per ulteriori informazioni, consultate la documentazione del client web CloudWatch RUM.

Per ulteriori informazioni sul monitoraggio degli utenti reali in CloudWatch, consulta Use CloudWatch RUM. Per configurare l'applicazione per l'utilizzo di CloudWatch RUM, incluso il tracciamento delle sessioni utente con X-Ray, consulta Configurare un'applicazione per CloudWatch utilizzare RUM.

Eseguire il debug di canarini CloudWatch sintetici utilizzando X-Ray

CloudWatch Synthetics è un servizio completamente gestito che consente di monitorare gli endpoint e le API utilizzando canary con script che funzionano 24 ore al giorno, una volta al minuto.

È possibile personalizzare gli script Canary per verificare le modifiche in:

- Disponibilità
- Latenza
- Transazioni
- Link interrotti o morti
- Completamento delle attività S tep-by-step
- Errori di caricamento della pagina
- Latenze di carico per le risorse dell'interfaccia utente
- Flussi della procedura guidata complessi
- Flussi di checkout nella tua applicazione

I Canary seguono gli stessi percorsi, eseguono le stesse azioni e comportamenti dei tuoi clienti e verificano continuamente l'esperienza del cliente.

Per ulteriori informazioni sulla configurazione dei test, consulta Utilizzo di Synthetics per creare e gestire i Canary.

Gli esempi seguenti mostrano casi d'uso comuni per problemi di debug che i Canary di Synthetics generano. Ogni esempio dimostra una strategia chiave per il debug utilizzando la mappa di traccia o la console X-Ray Analytics.

Per ulteriori informazioni su come leggere e interagire con la mappa di traccia, vedere Visualizzazione della mappa dei servizi.

Per ulteriori informazioni su come leggere e interagire con la console di X-Ray Analytics, consulta Interagire con la AWS X-Ray console Analytics.

Argomenti

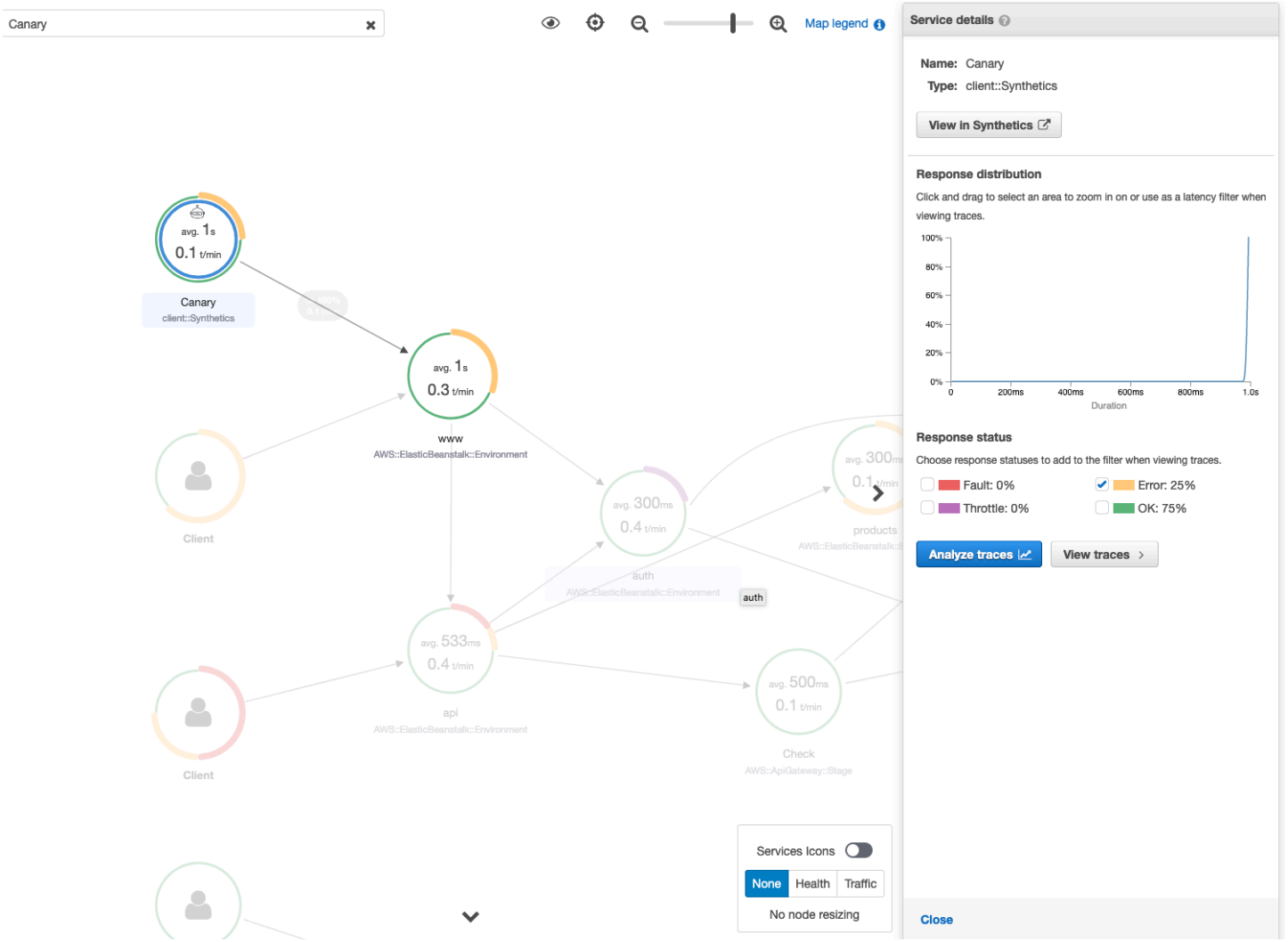
- Visualizza i canarini con una maggiore segnalazione degli errori nella mappa di tracciamento

- Utilizza le mappe dei dettagli di traccia per le singole tracce per visualizzare ogni richiesta in dettaglio
- Determinare la causa principale degli errori in corso nei servizi upstream e downstream
- Identificare i colli di bottiglia e le tendenze delle prestazioni
- Confrontare latenza ed errori o tassi di errore prima e dopo le modifiche
- Determinare la copertura Canary richiesta per tutte le API e gli URL
- Utilizzare i gruppi per concentrarsi sui test Synthetics.

Visualizza i canarini con una maggiore segnalazione degli errori nella mappa di tracciamento

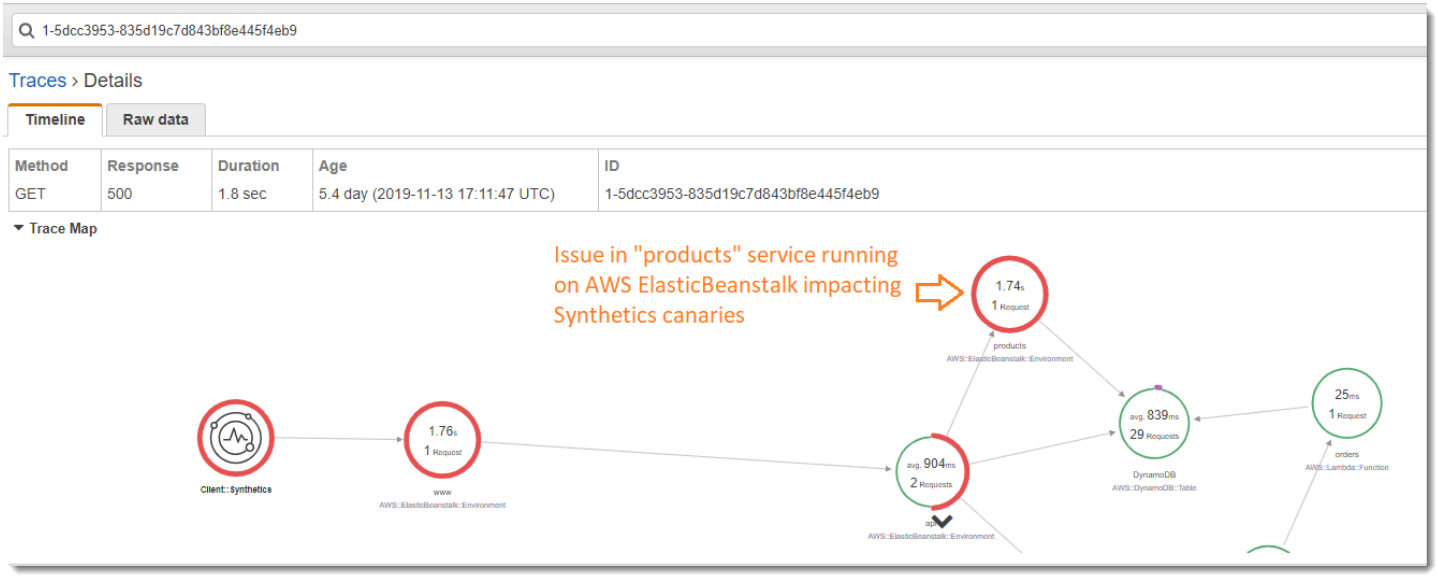
Per vedere quali canarini presentano un aumento di errori, guasti, velocità di limitazione o tempi di risposta lenti all'interno della mappa di tracciamento X-Ray, puoi evidenziare i nodi client Synthetics Canary utilizzando il filtro. `Client::Synthetic` Facendo clic su un nodo viene visualizzata la distribuzione del tempo di risposta dell'intera richiesta. Facendo clic su un bordo tra due nodi vengono visualizzati i dettagli sulle richieste che hanno percorso quella connessione. È inoltre possibile visualizzare i nodi dedotti «remoti» per i servizi downstream correlati nella mappa di traccia.

Quando fai clic sul nodo Synthetics, c'è un pulsante Visualizza in Synthetics sul pannello laterale che ti reindirizza alla console Synthetics dove puoi controllare i dettagli del canarino.



Utilizza le mappe dei dettagli di traccia per le singole tracce per visualizzare ogni richiesta in dettaglio

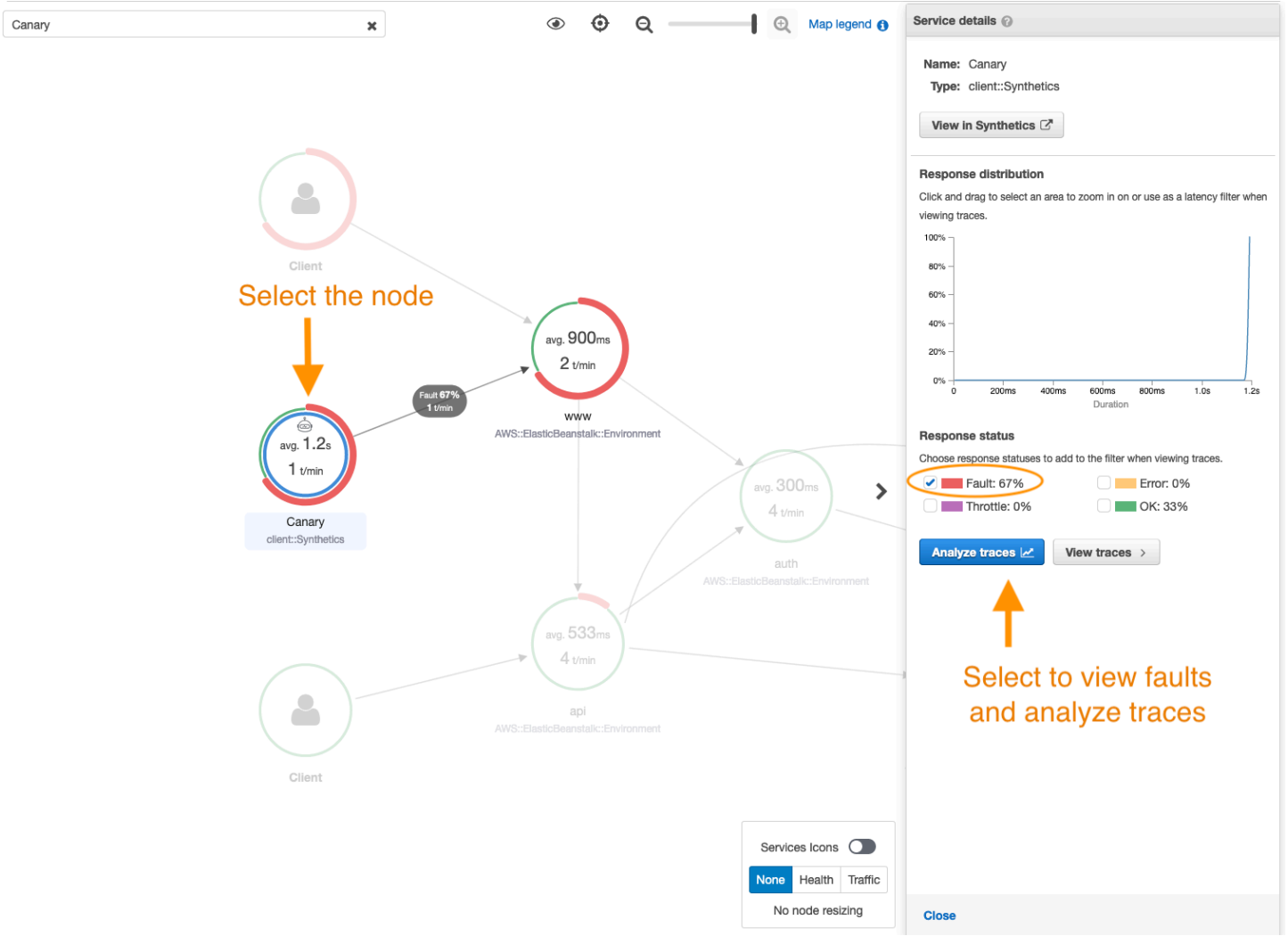
Per determinare quale servizio genera la maggiore latenza o causa un errore, richiama la mappa dei dettagli di traccia selezionando la traccia nella mappa di traccia. Le singole mappe dei dettagli di traccia mostrano il end-to-end percorso di una singola richiesta. Utilizza questa opzione per comprendere i servizi richiamati e visualizzare i servizi upstream e downstream.

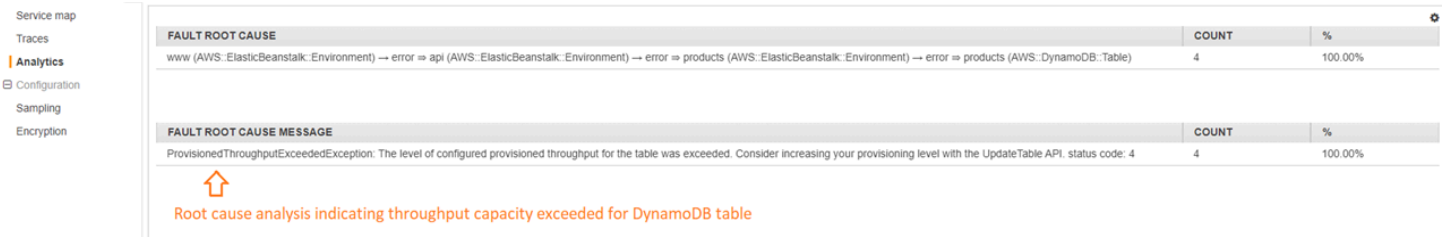


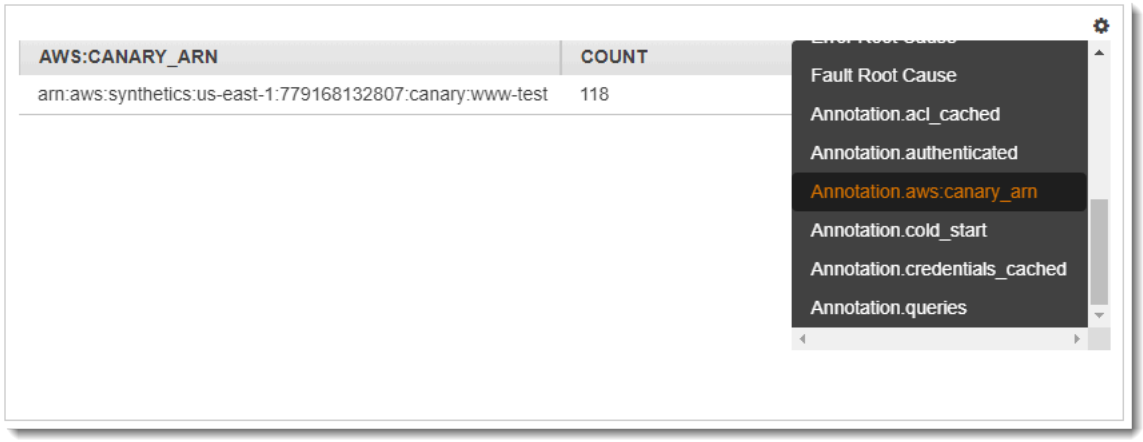
Determinare la causa principale degli errori in corso nei servizi upstream e downstream

Una volta ricevuto un CloudWatch allarme per guasti in un sistema Synthetics Canary, utilizzate la modellazione statistica sui dati di traccia in X-Ray per determinare la probabile causa principale del problema all'interno della console X-Ray Analytics. Nella console Analytics, la tabella Response Time Root Cause mostra i percorsi delle entità registrate. X-Ray determina quale percorso della traccia è la causa più probabile del tempo di risposta. Il formato indica una gerarchia di entità incontrate che termina in una causa principale del tempo di risposta.

L'esempio seguente mostra che il test Synthetics per l'API «XXX» in esecuzione su API Gateway non riesce a causa di un'eccezione di capacità di throughput nella tabella Amazon DynamoDB.

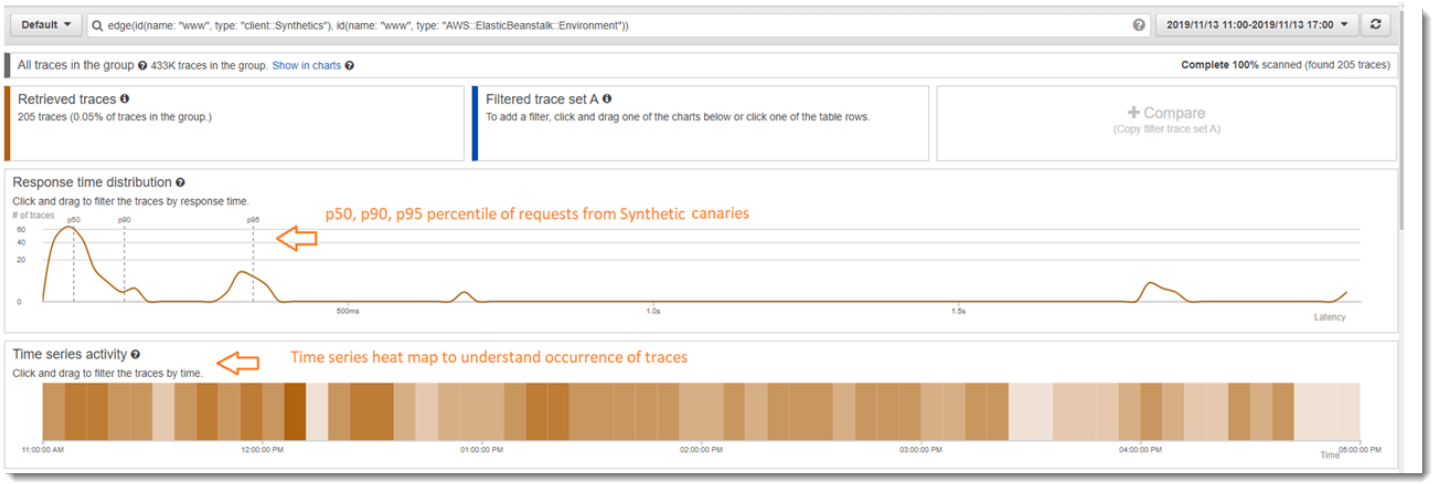

Canary
Map legend
Service details
Name: Canary
Type: client::Synthetics
View in Synthetics
Response distribution
Click and drag to select an area to zoom in on or use as a latency filter when viewing traces.
100%
80%
60%
40%
20%
0%
0
200ms
400ms
600ms
800ms
1.0s
1.2s
Duration
Response status
Choose response 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



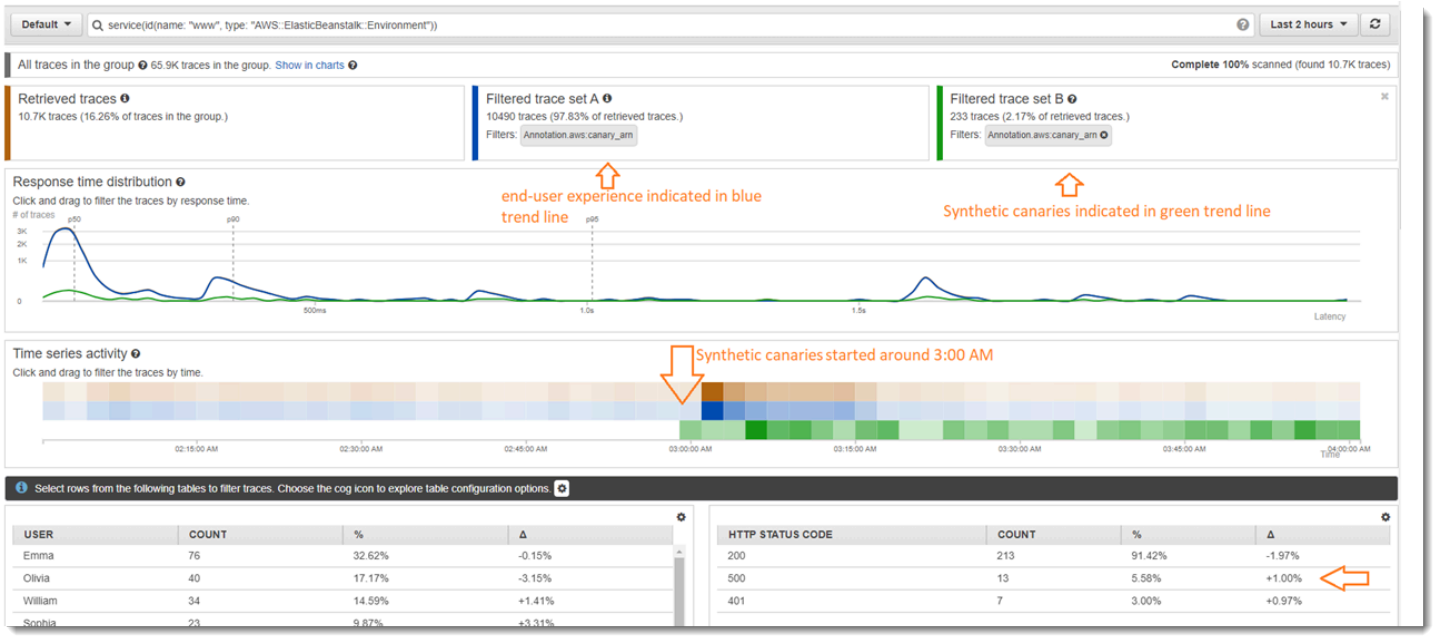
Identificare i colli di bottiglia e le tendenze delle prestazioni

Puoi visualizzare le tendenze delle prestazioni del tuo endpoint nel tempo utilizzando il traffico continuo proveniente dai tuoi canali Synthetics per compilare una mappa dei dettagli di tracciamento per un periodo di tempo.



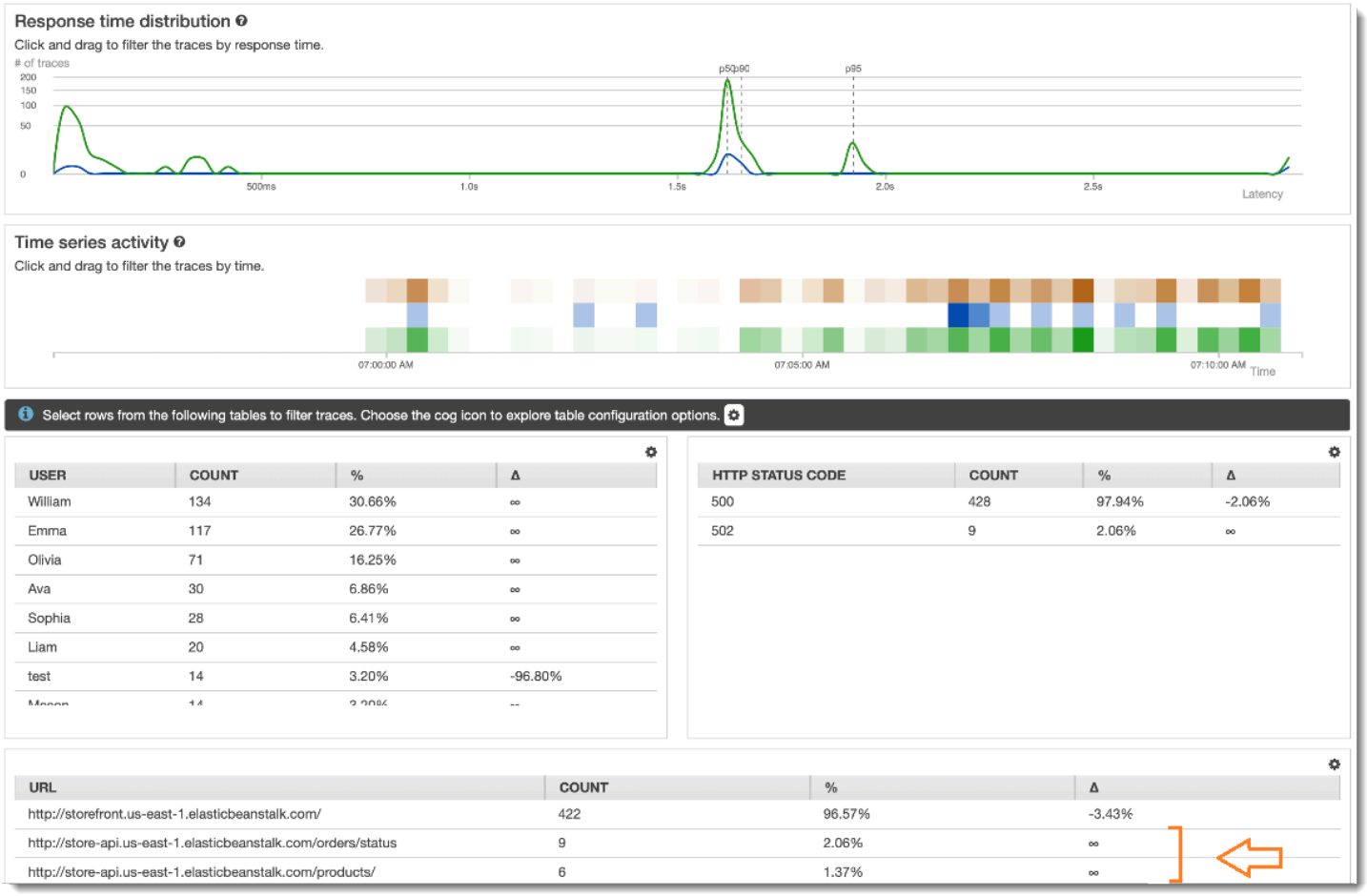
Confrontare latenza ed errori o tassi di errore prima e dopo le modifiche

Pinpoint l'ora in cui si è verificata una modifica per correlarla all'aumento dei problemi riscontrati dai vostri canarini. Usa la console X-Ray Analytics per definire gli intervalli di tempo prima e dopo come set di tracce diversi, creando una differenziazione visiva nella distribuzione del tempo di risposta.



Determinare la copertura Canary richiesta per tutte le API e gli URL

Usa X-Ray Analytics per confrontare l'esperienza dei Canary con gli utenti. L'interfaccia utente seguente mostra una linea di tendenza blu per i Canary e una linea verde per gli utenti. Puoi anche identificare che due URL su tre non dispongono di test dei Canary.

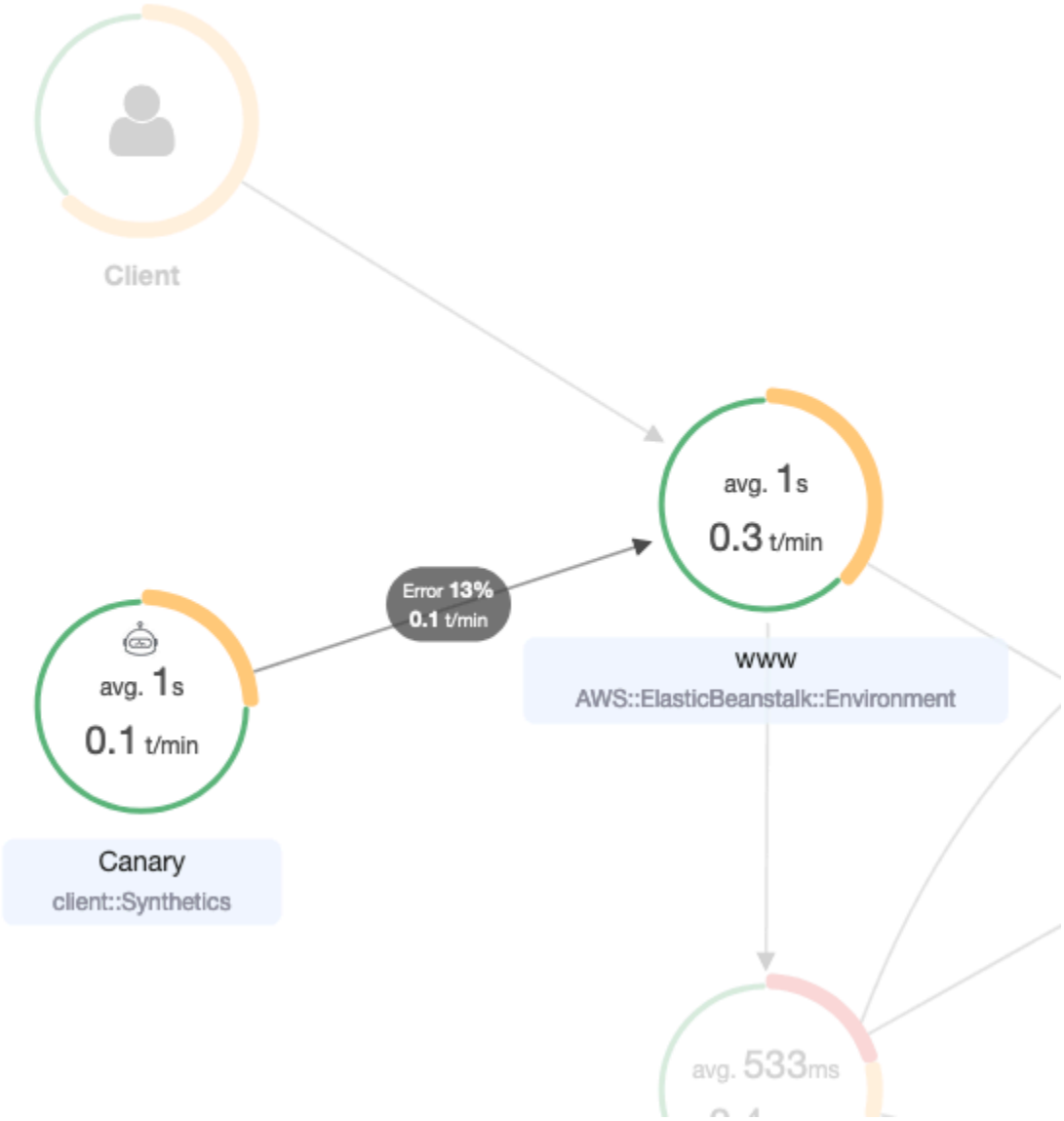


Utilizzare i gruppi per concentrarsi sui test Synthetics.

È possibile creare un gruppo X-Ray utilizzando un'espressione di filtro per concentrarsi su un determinato insieme di flussi di lavoro, ad esempio i test Synthetics per l'applicazione «www» in esecuzione. AWS Elastic Beanstalk Utilizzate le parole chiave complesse e `service()` filtrate attraverso servizi e `edge()` bordi.

Example Espressione filtro gruppo

```
"edge(id(name: "www", type: "client::Synthetics"), id(name: "www", type:
 "AWS::ElasticBeanstalk::Environment"))"
```



Tracciamento delle modifiche alla configurazione della crittografia a X-Ray conAWS Config

AWS X-Raysi integra conAWS Config per registrare le modifiche alla configurazione apportate alle risorse di crittografia X-Ray. È possibile utilizzarloAWS Config per inventariare le risorse di crittografia a raggi X, controllare la cronologia delle configurazioni di X-Ray e inviare notifiche in base alle modifiche delle risorse.

AWS Configsupporta la registrazione delle seguenti modifiche alle risorse di crittografia a X-Ray come eventi:

- Modifiche alla configurazione: modifica o aggiunta di una chiave di crittografia o ripristino dell'impostazione di crittografia a X-Ray predefinita.

Usa le seguenti istruzioni per imparare a creare una connessione di base tra X-Ray eAWS Config.

Creazione di un trigger di una funzione Lambda

Prima di poter generare una regola personalizzata di AWS Config, devi disporre dell'ARN di una funzione AWS Lambda personalizzata. Segui queste istruzioni per creare una funzione di base con Node.js che restituisce un valore conforme o meno a AWS Config in base allo stato della risorsa `XrayEncryptionConfig`.

Per creare una funzione Lambda con un trigger di AWS::XrayEncryptionConfig modifica

1. Aprire la console Lambda. Scegli Create function (Crea funzione).
2. Scegli Blueprints, quindi filtra la libreria dei blueprint per il config-rule-change-triggeredblueprint. Fare clic sul collegamento nel nome del progetto o scegliere Configure (Configura) per proseguire.
3. Definire i seguenti campi per configurare il progetto:
   - Digitare un nome nel campo Name (Nome).
   - In Role (Ruolo), seleziona Create new role from template(s) (Crea nuovo ruolo da modello/i):
   - In Role Name (Nome ruolo) digita un nome.
   - In Policy templates (Modelli di policy), scegliere AWS Config Rules permissions (Regole di autorizzazione).
4. Scegliere Create function (Crea funzione) per creare e visualizzare la funzione nella console AWS Lambda.
5. Modificare il codice della funzione per sostituire `AWS::EC2::Instance` con `AWS::XrayEncryptionConfig`. Puoi anche aggiornare il campo descrizione affinché rifletta questa modifica.

   Codice di default

```
    if (configurationItem.resourceType !== 'AWS::EC2::Instance') {
        return 'NOT_APPLICABLE';
    } else if (ruleParameters.desiredInstanceType ===
configurationItem.configuration.instanceType) {
```

```
        return 'COMPLIANT';
    }
        return 'NON_COMPLIANT';
```

Codice aggiornato

```
    if (configurationItem.resourceType !== 'AWS::XRay::EncryptionConfig') {
        return 'NOT_APPLICABLE';
    } else if (ruleParameters.desiredInstanceType ===
configurationItem.configuration.instanceType) {
        return 'COMPLIANT';
    }
        return 'NON_COMPLIANT';
```

6. Aggiungi quanto segue al tuo ruolo di esecuzione in IAM per accedere a X-Ray. Queste autorizzazioni consentono l'accesso in sola lettura alle risorse X-Ray. La mancata fornitura dell'accesso alle risorse appropriate comporterà un messaggio fuori ambito aAWS Config partire dalla valutazione della funzione Lambda associata alla regola.

```
    {
        "Sid": "Stmt1529350291539",
        "Action": [
            "xray:GetEncryptionConfig"
        ],
        "Effect": "Allow",
        "Resource": "*"
     }
```

Creazione di una regola personalizzata per X-Ray in AWS Config

Quando viene creata la funzione Lambda, annota l'ARN della funzione e vai allaAWS Config console per creare la tua regola personalizzata.

Per creare unaAWS Config regola per X-Ray

1. Aprire la pagina Rules (Regole) della console di AWS Config
2. Scegliere Add rule (Aggiungi regola) e quindi scegliere Add custom rule (Aggiungi regola personalizzata).

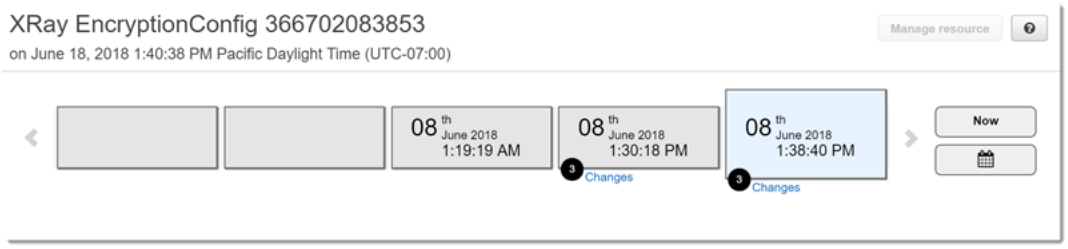
3. Nella AWS LambdaFunzione ARN, inserisci l'ARN associato alla funzione Lambda che desideri utilizzare.
4. Scegliere il tipo di trigger da impostare:
   - Modifiche alla configurazione:AWS Config attiva la valutazione quando una risorsa che corrisponde all'ambito della regola cambia nella configurazione. La valutazione viene eseguita dopo che AWS Config ha inviato una notifica di modifica dell'elemento di configurazione.
   - Periodico:AWS Config esegue le valutazioni della regola con una frequenza selezionata (ad esempio, ogni 24 ore).
5. Per Tipo di risorsa, scegli EncryptionConfignella sezione X-Ray.
6. Seleziona Salva.

La console di AWS Config inizia immediatamente a valutare la conformità della regola. La valutazione può richiedere alcuni minuti.

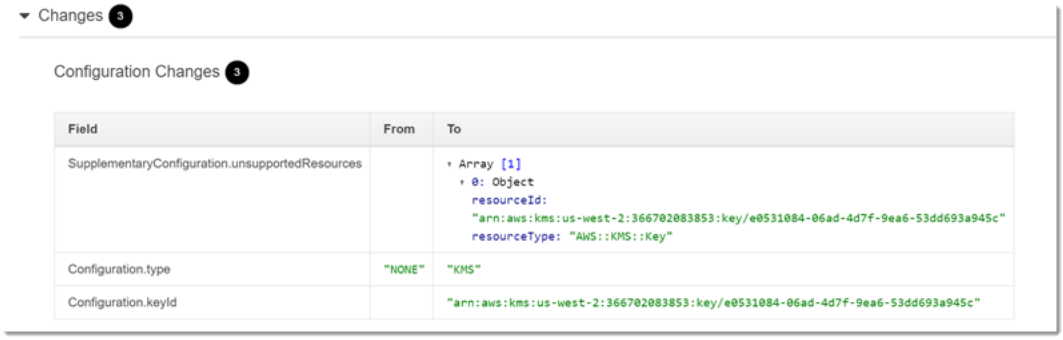
Ora che questa regola è valutata come conforme, AWS Config può iniziare a compilare una cronologia di verifiche. AWS Config registra le modifiche alle risorse sotto forma di sequenza temporale. Per ogni modifica nella sequenza temporale degli eventi, AWS Config genera una tabella in formato da/a per mostrare ciò che è stato modificato nella rappresentazione JSON della chiave crittografica. Le due modifiche ai campi associate EncryptionConfig sono`Configuration.type` e`Configuration.keyID`.

Risultati di esempio

Di seguito è riportato un esempio di una sequenza temporale di AWS Config che mostra le modifiche apportate a date e ore specifiche.



Qui di seguito è riportato un esempio di un elemento che descrive una modifica in AWS Config. Il formato da/a illustra ciò che è stato modificato. Questo esempio mostra che le impostazioni di crittografia X-Ray predefinite sono state modificate in una chiave di crittografia definita.



Notifiche Amazon SNS

Per ricevere una notifica delle modifiche alla configurazione, configuraAWS Config in modo che pubblichi le notifiche Amazon SNS. Per ulteriori informazioni, consulta Monitoraggio delle modifiche alle risorse AWS Config tramite e-mail.

Amazon Elastic Compute Cloud eAWS X-Ray

Puoi installare ed eseguire il daemon X-Ray su un'istanza di Amazon EC2 con uno script dati utente. Per istruzioni, consulta Esecuzione del daemon X-Ray su Amazon EC2.

Utilizza un profilo dell'istanza per concedere al daemon le autorizzazioni necessarie per caricare i dati di tracciamento su X-Ray. Per ulteriori informazioni, consultare Dare al demone il permesso di inviare dati a X-Ray.

AWS Elastic Beanstalk e AWS X-Ray

AWS Elastic Beanstalkpiattaforme includono il daemon X-Ray. È possibileEsecuzione del daemonimpostando un'opzione nella console di Elastic Beanstalk o con un file di configurazione.

Nella piattaforma Java SE, puoi utilizzare un file Buildfile per creare un'applicazione con Maven o Gradle sull'istanza. L'SDK X-Ray per Java eAWS SDK for Javasono disponibili su Maven, perciò puoi distribuire solo il codice della tua applicazione ed eseguire la compilazione sull'istanza per evitare la creazione del pacchetto e il caricamento di tutte le dipendenze.

Puoi utilizzare le proprietà dell'ambiente Elastic Beanstalk per configurare l'SDK X-Ray. Il metodo che Elastic Beanstalk utilizza per passare le proprietà dell'ambiente alla tua applicazione variano in base alla piattaforma. Utilizza le variabili di ambiente o le proprietà di sistema dell'SDK X-Ray a seconda della piattaforma.

- Piattaforma Node.js— Usavariabili di ambiente

- Piattaforma Java SE— Usavariabili di ambiente
- Piattaforma Tomcat— Usaproprietà di sistema

Per ulteriori informazioni, consulta la sezione Configurazione del debugging di AWS X-Ray nella Guida per lo sviluppatore di AWS Elastic Beanstalk.

Elastic Load Balancing e AWS X-Ray

I sistemi di bilanciamento del carico delle applicazioni Elastic Load Balancing aggiungono un ID di traccia alle richieste HTTP in entrata in un'intestazione denominata. `X-Amzn-Trace-Id`

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

Formato ID di traccia X-Ray

Una radiografia `trace_id` è composta da tre numeri separati da trattini. Ad esempio, `1-58406520-a006649127e371903a2de979`. Questo include:

- Il numero di versione, che è. `1`
- L'ora della richiesta originale in Unix epoch time utilizzando 8 cifre esadecimali.

  Ad esempio, il fuso orario PST delle 10:00 del 1° dicembre 2016 è espresso in secondi o in cifre esadecimali. `1480615200` `58406520`
- Un identificatore a 96 bit univoco a livello globale per la traccia in 24 cifre esadecimali.

I sistemi di bilanciamento del carico non inviano dati a X-Ray e non vengono visualizzati come nodo sulla mappa dei servizi.

Per ulteriori informazioni, consulta Request Tracing for Your Application Load Balancer nella Elastic Load Balancing Developer Guide.

Amazon EventBridge e AWS X-Ray

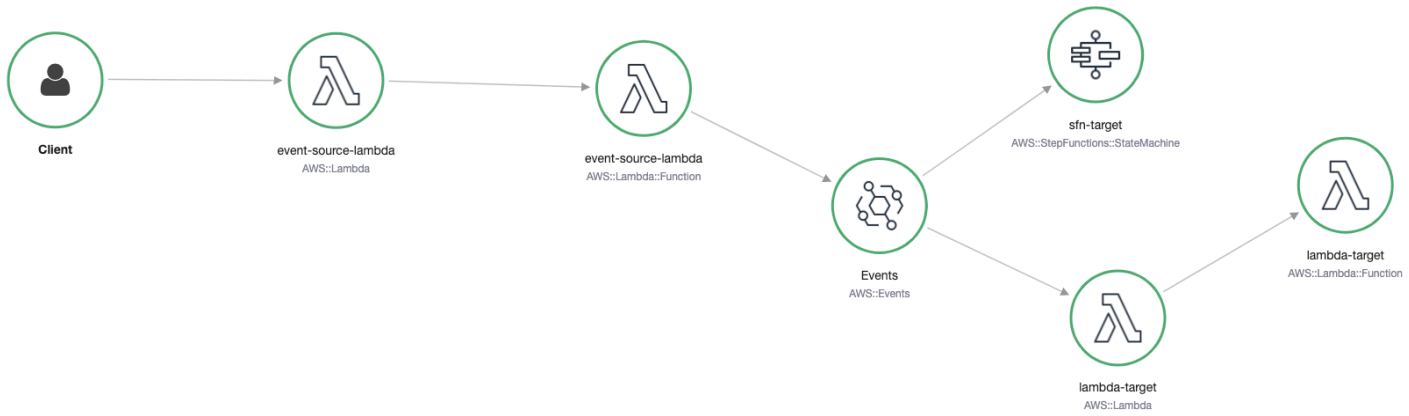
AWS X-Ray si integra con Amazon EventBridge per tracciare gli eventi trasmessi EventBridge. Se un servizio dotato di strumentazione con X-Ray SDK invia eventi EventBridge a, il contesto di traccia viene propagato alle destinazioni degli eventi a valle all'interno dell'intestazione di tracciamento. L'X-Ray SDK rileva automaticamente l'intestazione di tracciamento e la applica a

qualsiasi strumentazione successiva. Questa continuità consente agli utenti di tracciare, analizzare ed eseguire il debug in tutti i servizi a valle e fornisce una visione più completa del sistema.

Per ulteriori informazioni, vedere EventBridge X-Ray Integration nella Guida per l'EventBridge utente.

Visualizzazione della sorgente e delle destinazioni sulla mappa dei servizi X-Ray

La mappa di tracciamento a raggi X mostra un nodo EventBridge evento che collega i servizi di origine e destinazione, come nell'esempio seguente:



Propaga il contesto di traccia alle destinazioni degli eventi

L'X-Ray SDK consente all'origine dell' EventBridge evento di propagare il contesto della traccia alle destinazioni degli eventi a valle. I seguenti esempi specifici del linguaggio dimostrano la chiamata da EventBridge una funzione Lambda su cui è abilitata la traccia attiva:

Java

Aggiungi le dipendenze necessarie per X-Ray:

- AWS X-Ray SDK per Java
- AWS X-Ray SDK Recorder per 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

Aggiungi la seguente dipendenza al tuo file requirements.txt:

```
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#

Aggiungi i seguenti pacchetti X-Ray alle tue dipendenze in C#:

```
<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 e AWS X-Ray

Puoi usarlo AWS X-Ray per tracciare le tue AWS Lambda funzioni. Lambda esegue il demone X-Ray e registra un segmento con dettagli sull'invocazione e l'esecuzione della funzione. Per ulteriore strumentazione, puoi abbinare l'X-Ray SDK alla tua funzione per registrare le chiamate in uscita e aggiungere annotazioni e metadati.

Se la funzione Lambda viene chiamata da un altro servizio strumentato, Lambda traccia le richieste che sono già state campionate senza alcuna configurazione aggiuntiva. Il servizio upstream può

essere un'applicazione Web strumentata o un'altra funzione Lambda. Il tuo servizio può richiamare la funzione direttamente con un client AWS SDK strumentato o chiamando un'API API Gateway con un client HTTP strumentato.

AWS X-Ray supporta il tracciamento di applicazioni basate sugli eventi utilizzando Amazon AWS Lambda SQS. Usa la CloudWatch console per visualizzare una vista connessa di ogni richiesta mentre viene messa in coda con Amazon SQS ed elaborata da una funzione Lambda downstream. Le tracce dei produttori di messaggi upstream vengono automaticamente collegate alle tracce dei nodi consumer Lambda a valle, creando una end-to-end visualizzazione dell'applicazione. Per ulteriori informazioni, consulta la sezione di tracciamento delle applicazioni basate sugli eventi.

> **Note**
>
> Se hai le tracce abilitate per una funzione Lambda downstream, devi avere le tracce abilitate anche per la funzione Lambda root che chiama la funzione downstream affinché la funzione downstream generi tracce.

Se la funzione Lambda viene eseguita in base a una pianificazione o viene richiamata da un servizio non dotato di strumentazione, è possibile configurare Lambda per campionare e registrare le chiamate con tracciamento attivo.

Per configurare l'integrazione X-Ray su una funzione AWS Lambda

1. Apri la AWS Lambda console.
2. Seleziona Funzioni dalla barra di navigazione a sinistra.
3. Scegliere la funzione.
4. Nella scheda Configurazione, scorri verso il basso fino alla scheda Strumenti di monitoraggio aggiuntivi. Puoi trovare questa scheda anche selezionando Strumenti di monitoraggio e operazioni nel riquadro di navigazione a sinistra.
5. Seleziona Edit (Modifica).
6. In AWS X-ray, abilitare Active tracing (Tracciamento attivo).

Nei runtime con un SDK X-Ray corrispondente, Lambda esegue anche il demone X-Ray.

SDK X-Ray su Lambda

- X-Ray SDK for Go — Go 1.7 e runtime più recenti

- X-Ray SDK per Java — Java 8 runtime
- X-Ray SDK per Node.js — Node.js 4.3 e versioni successive
- X-Ray SDK per Python — Python 2.7, Python 3.6 e runtime più recenti
- X-Ray SDK for .NET: .NET Core 2.0 e runtime più recenti

Per utilizzare l'SDK X-Ray su Lambda, aggiungilo al codice funzione ogni volta che crei una nuova versione. Puoi strumentare le tue funzioni Lambda con gli stessi metodi che usi per strumentare le applicazioni in esecuzione su altri servizi. La differenza principale è che non si utilizza l'SDK per analizzare le richieste in entrata, assumere le decisioni di campionamento e creare i segmenti.

L'altra differenza tra la strumentazione delle funzioni Lambda e le applicazioni Web è che il segmento che Lambda crea e invia a X-Ray non può essere modificato dal codice della funzione. Puoi creare sottosegmenti e memorizzare in essi annotazioni e metadati, ma non puoi aggiungere annotazioni e metadati al segmento padre.

Per ulteriori informazioni, consulta Using AWS X-Ray nella AWS Lambda Developer Guide.

Amazon SNS e AWS X-Ray

Puoi utilizzare Amazon AWS X-Ray Simple Notification Service (Amazon SNS) per tracciare e analizzare le richieste man mano che passano dai tuoi argomenti SNS ai servizi in abbonamento supportati da SNS. Usa il tracciamento X-Ray con Amazon SNS per analizzare le latenze nei tuoi messaggi e nei relativi servizi di back-end, ad esempio quanto tempo trascorre una richiesta in un argomento e quanto tempo impiega a recapitare il messaggio a ciascuno degli abbonamenti dell'argomento. Amazon SNS supporta il tracciamento di X-Ray per gli argomenti standard e FIFO.

Se pubblichi su un argomento di Amazon SNS da un servizio già dotato di strumentazione X-Ray, Amazon SNS trasmette il contesto di traccia dall'editore agli abbonati. Inoltre, puoi attivare il tracciamento attivo per inviare a X-Ray i dati dei segmenti relativi ai tuoi abbonamenti Amazon SNS per i messaggi pubblicati da un client SNS dotato di strumentazione. Attiva il tracciamento attivo per un argomento di Amazon SNS utilizzando la console Amazon SNS o l'API o la CLI di Amazon SNS. Per ulteriori informazioni sulla strumentazione dei client SNS, consulta la sezione Strumentazione dell'applicazione.

Configurazione del tracciamento attivo di Amazon SNS

Puoi utilizzare la console Amazon SNS o l'interfaccia a riga di comando o l'SDK per configurare il AWS tracciamento attivo di Amazon SNS.

Quando usi la console Amazon SNS, Amazon SNS tenta di creare le autorizzazioni necessarie affinché SNS possa chiamare X-Ray. Il tentativo può essere rifiutato se non si dispone di autorizzazioni sufficienti per modificare le politiche relative alle risorse X-Ray. Per ulteriori informazioni su queste autorizzazioni, consulta Gestione delle identità e degli accessi in Amazon SNS e Casi di esempio per il controllo degli accessi di Amazon SNS nella Amazon Simple Notification Service Developer Guide. Per ulteriori informazioni sull'attivazione del tracciamento attivo tramite la console Amazon SNS, consulta l'argomento Enabling active tracing on an Amazon SNS nella Amazon Simple Notification Service Developer Guide.

Quando si utilizza la AWS CLI o l'SDK per attivare la traccia attiva, è necessario configurare manualmente le autorizzazioni utilizzando politiche basate sulle risorse. Utilizzalo `PutResourcePolicy`per configurare X-Ray con la politica basata sulle risorse necessaria per consentire ad Amazon SNS di inviare tracce a X-Ray.

Example Esempio di policy basata su risorse X-Ray per il tracciamento attivo di Amazon SNS

Questo documento di policy di esempio specifica le autorizzazioni necessarie ad Amazon SNS per inviare dati di traccia a X-Ray:

```
{
    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"
          }
        }
```

```
      }
    ]
  }
```

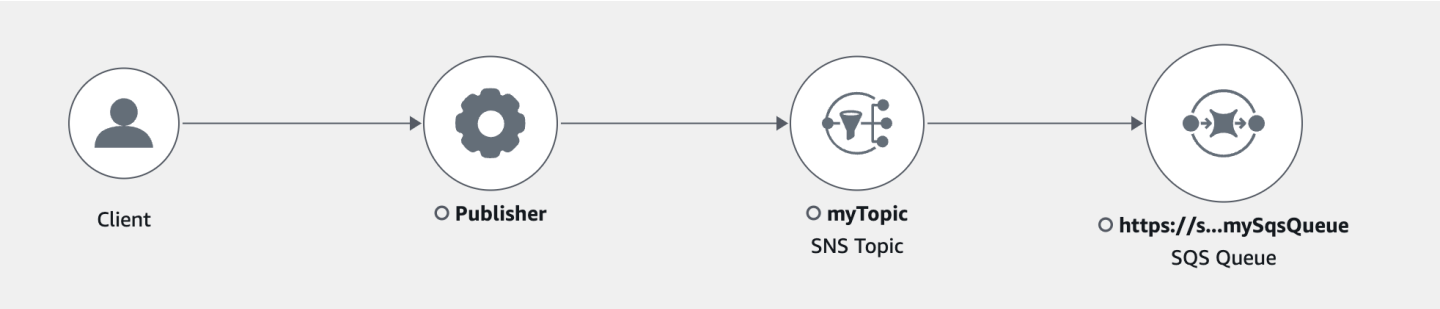
Utilizza la CLI per creare una policy basata sulle risorse che conceda ad Amazon SNS le autorizzazioni per inviare dati di traccia a X-Ray:

```
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" } } } ] }'
```

Per utilizzare questi esempi, sostituisci *partition*, *regionaccount-id*, e *topic-name*con la AWS partizione, la regione, l'ID dell'account e il nome dell'argomento Amazon SNS specifici. Per autorizzare tutti gli argomenti di Amazon SNS a inviare dati di traccia a X-Ray, sostituisci il nome dell'argomento con. *

Visualizza le tracce degli editori e degli abbonati di Amazon SNS nella console X-Ray

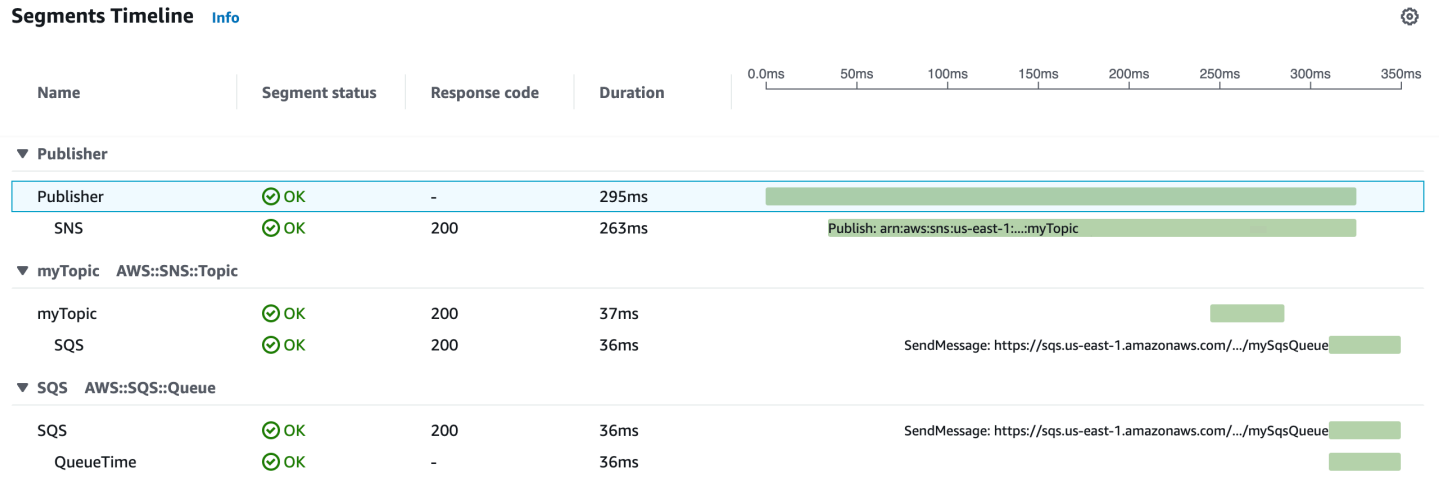
Usa la console X-Ray per visualizzare una mappa di tracciamento e i dettagli di traccia che mostrano una visualizzazione connessa di editori e abbonati Amazon SNS. Quando il tracciamento attivo di Amazon SNS è attivato per un argomento, la mappa di traccia a raggi X e la mappa dei dettagli di traccia mostrano i nodi connessi per gli editori Amazon SNS, l'argomento Amazon SNS e gli abbonati downstream:



Dopo aver scelto una traccia che includa un editore e un abbonato di Amazon SNS, la pagina dei dettagli di tracciamento X-Ray mostra una mappa dei dettagli di tracciamento e una cronologia dei segmenti.

Example Cronologia di esempio con editore e abbonato di Amazon SNS

Questo esempio mostra una sequenza temporale che include un publisher Amazon SNS che invia un messaggio a un argomento Amazon SNS, che viene elaborato da un abbonato Amazon SQS.

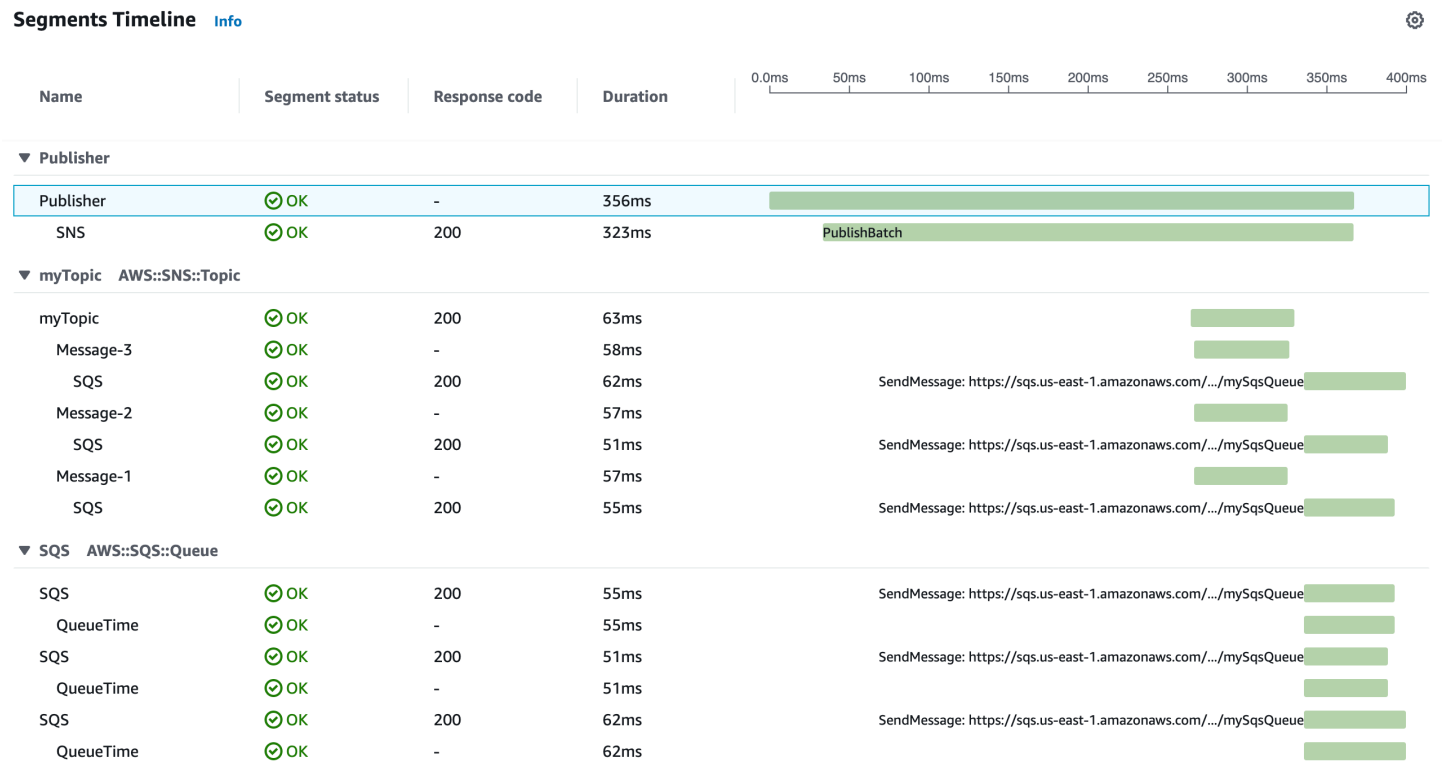


La sequenza temporale di esempio sopra riportata fornisce dettagli sul flusso di messaggi Amazon SNS:

- Il segmento SNS rappresenta la durata di andata e ritorno della chiamata `Publish` API dal client.
- Il segmento MyTopic rappresenta la latenza della risposta di Amazon SNS alla richiesta di pubblicazione.
- Il sottosegmento SQS rappresenta il tempo di andata e ritorno impiegato da Amazon SNS per pubblicare il messaggio in una coda Amazon SQS.
- Il tempo tra il segmento MyTopic e il sottosegmento SQS rappresenta il tempo che il messaggio trascorre nel sistema Amazon SNS.

Example Cronologia di esempio con messaggi Amazon SNS in batch

Se più messaggi Amazon SNS vengono raggruppati in batch all'interno di un'unica traccia, la timeline del segmento mostra i segmenti che rappresentano ogni messaggio elaborato.



AWS Step Functions e AWS X-Ray

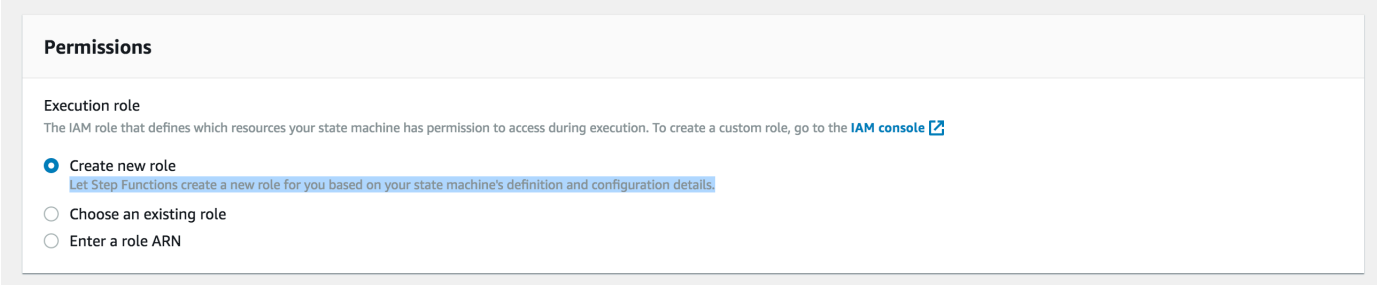
AWS X-RayIntegrazione di conAWS Step Functionsper tracciare e analizzare le richieste di Step Functions. È possibile visualizzare i componenti del tuomacchina a stati, identificare i colli di bottiglia delle prestazioni e risolvere i problemi delle richieste che hanno provocato un errore. Per ulteriori informazioni, consultaAWS X-Raye Step FunctionsnellaAWS Step FunctionsGuida per sviluppatori di .

Per abilitare il tracciamento a X-Ray durante la creazione di una nuova macchina a stato

1. Aprire la console Step Functions all'indirizzohttps://console.aws.amazon.com/states/.
2. ScegliereCreazione di una macchina a stati.
3. SulDefinire la macchina a statipagina, scegliere unaAutore con frammenti di codiceoInizia con un modello. Se scegli di eseguire un progetto di esempio, tunon puòabilitare il monitoraggio di X-Ray durante la creazione. Invece,abilita il monitoraggio di X-Raycrea ilmacchina a stati.
4. Seleziona Next (Successivo).
5. SulSpecificare i dettaglipage, configurare la macchina a stati.
6. ScegliereAbilita il monitoraggio di X-Ray.

Per abilitare il monitoraggio di X-Ray in una macchina a stati esistenti

1. Nella console Step Functions, selezionare la macchina a stato per cui si desidera abilitare il tracciamento.
2. Scegli Modifica.
3. ScegliereAbilita il monitoraggio di X-Ray.
4. (Facoltativo) Genera automaticamente un nuovo ruolo per il computer a stato per includere le autorizzazioni a X-Ray scegliendoCreazione di un nuovo ruolodalla finestra Autorizzazioni.



5. Scegli Salva.

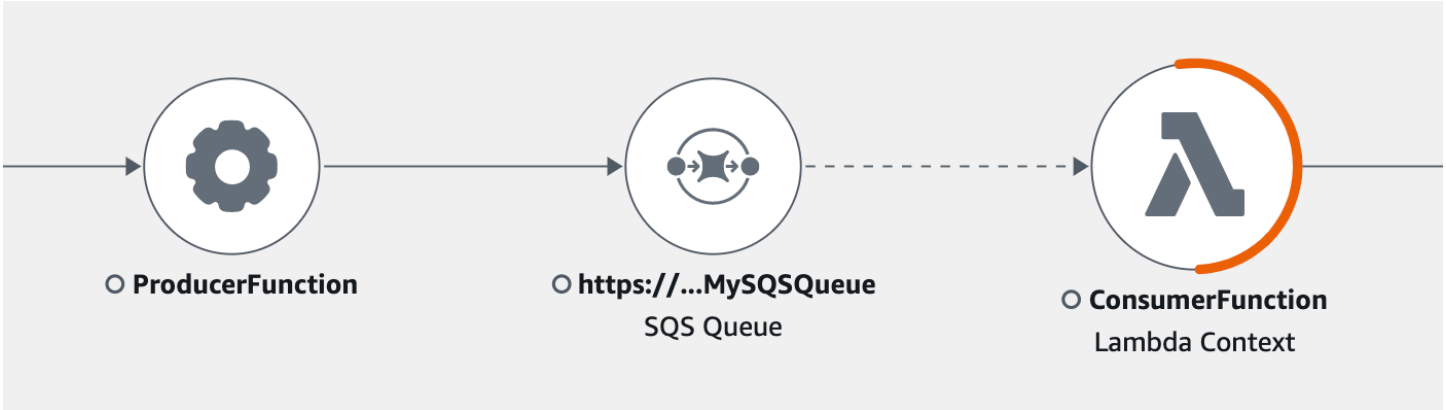
> **Note**
>
> Quando si crea una nuova macchina a stati,esso's tracciato automaticamente se la richiesta viene campionata e l'analisi è abilitata in un servizio upstream come Amazon API Gateway oAWS Lambda. Per qualsiasi macchina a stato esistente non configurata tramite console, ad esempio tramite unAWS CloudFormationtemplate,verifica di disporre di un criterio IAM che conceda autorizzazioni sufficienti per abilitare le tracce a X-Ray.

Amazon SQS e AWS X-Ray

AWS X-Ray si integra con Amazon Simple Queue Service (Amazon SQS) per tracciare i messaggi che vengono passati attraverso una coda Amazon SQS. Se un servizio traccia le richieste utilizzando l'SDK X-Ray, Amazon SQS può inviare l'intestazione di tracciamento e continuare a propagare la traccia originale dal mittente al consumatore con un ID di traccia coerente. Questa continuità consente agli utenti di tracciare, analizzare ed eseguire il debug di tutti i servizi di downstream.

AWS X-Ray supporta il tracciamento di applicazioni basate su eventi utilizzando Amazon SQS e. AWS Lambda Usa la CloudWatch console per visualizzare una vista connessa di ogni richiesta

mentre viene messa in coda con Amazon SQS ed elaborata da una funzione Lambda downstream. Le tracce dei produttori di messaggi upstream vengono automaticamente collegate alle tracce dei nodi consumer Lambda a valle, creando una end-to-end visualizzazione dell'applicazione. Per ulteriori informazioni, consulta la sezione di tracciamento delle applicazioni basate sugli eventi.



Amazon SQS supporta la seguente strumentazione di tracing header:

- Intestazione HTTP predefinita: l'SDK X-Ray popola automaticamente l'intestazione di traccia come intestazione HTTP quando chiami Amazon SQS tramite l'SDK. AWS L'intestazione di traccia predefinita viene trasportata da `X-Amzn-Trace-Id` e corrisponde a tutti i messaggi inclusi in una richiesta `SendMessage` o `SendMessageBatch`. Per ulteriori informazioni sull'intestazione HTTP predefinita, consulta Intestazione di tracciamento.
- **`AWSTraceHeader`**Attributo di sistema: `AWSTraceHeader` è un attributo di sistema di messaggi riservato da Amazon SQS per trasportare l'intestazione di traccia X-Ray con i messaggi in coda. `AWSTraceHeader`è disponibile per l'uso anche quando la strumentazione automatica tramite X-Ray SDK non lo è, ad esempio quando si crea un SDK di tracciamento per una nuova lingua. Quando sono impostate entrambe le strumentazioni delle intestazioni, l'attributo del sistema di messaggi sostituisce l'intestazione di traccia HTTP.

Quando è in esecuzione su Amazon EC2, Amazon SQS supporta l'elaborazione di un messaggio alla volta. Ciò si applica quando viene eseguito su un host locale e quando si utilizzano servizi container AWS Fargate, come Amazon ECS o. AWS App Mesh

L'intestazione di traccia è esclusa sia dalla dimensione dei messaggi di Amazon SQS che dalle quote degli attributi del messaggio. L'abilitazione del tracciamento X-Ray non supererà le quote di Amazon SQS. Per ulteriori informazioni sulle AWS quote, consulta Amazon SQS Quotas.

Invio dell'intestazione di traccia HTTP

I componenti del mittente in Amazon SQS possono inviare automaticamente l'intestazione trace tramite `SendMessageBatch`la chiamata o. `SendMessage` Quando i client AWS SDK sono dotati di strumentazione, possono essere tracciati automaticamente in tutte le lingue supportate tramite X-Ray SDK. Le risorse tracciate Servizi AWS e a cui accedi all'interno di tali servizi (ad esempio, un bucket Amazon S3 o una coda Amazon SQS) vengono visualizzate come nodi downstream sulla mappa di traccia nella console X-Ray.

Per informazioni su come tracciare le chiamate AWS SDK con la tua lingua preferita, consulta i seguenti argomenti negli SDK supportati:

- Go: Tracciamento delle chiamate AWS SDK con X-Ray SDK for Go
- Java — Tracciamento delle chiamate AWS SDK con X-Ray SDK for Java
- Node.js: Tracciamento delle chiamate AWS SDK con X-Ray SDK per Node.js
- Python: Tracciamento delle chiamate AWS SDK con X-Ray SDK per Python
- Ruby: Tracciamento delle chiamate AWS SDK con X-Ray SDK for Ruby
- .NET: Tracciamento delle chiamate AWS SDK con X-Ray SDK for .NET

Recupero dell'intestazione di traccia e recupero del contesto di traccia

Se utilizzi un consumer Lambda downstream, la propagazione del contesto di traccia è automatica. Per continuare la propagazione del contesto con altri consumatori Amazon SQS, devi strumentalizzare manualmente il passaggio al componente ricevente.

Esistono tre fasi principali per recuperare il contesto di traccia:

- Ricezione del messaggio dalla coda per l'attributo `AWSTraceHeader` chiamando l'API `ReceiveMessage`.
- Recupero dell'intestazione di traccia dall'attributo.
- Recupero dell'ID di traccia dall'intestazione. Facoltativamente, aggiunta di ulteriori parametri al segmento.

Di seguito è riportato un esempio di implementazione scritto con X-Ray SDK for Java.

Example : recupero dell'intestazione di traccia e recupero del contesto di traccia

```
// 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 e AWS X-Ray

AWS X-Ray si integra con Amazon S3 per tracciare le richieste upstream per aggiornare i bucket S3 dell'applicazione. Se un servizio traccia le richieste utilizzando l'SDK X-Ray, Amazon S3 può inviare le intestazioni di tracciamento a sottoscrittori di eventi downstream come Amazon SQS e Amazon SNS. AWS Lambda X-Ray consente di tracciare i messaggi per le notifiche degli eventi di Amazon S3.

Puoi utilizzare la mappa di tracciamento a raggi X per visualizzare le connessioni tra Amazon S3 e altri servizi utilizzati dall'applicazione. È inoltre possibile utilizzare la console per visualizzare i parametri come la latenza media e le percentuali di errore. Per ulteriori informazioni sulla console X-Ray, vedere. AWS X-Ray consolle

Amazon S3 supporta la strumentazione di intestazione http predefinita. L'SDK X-Ray compila automaticamente l'intestazione di traccia come intestazione HTTP quando chiami Amazon S3

tramite l'SDK. AWS L'intestazione di traccia predefinita è gestita da. `X-Amzn-Trace-Id` Per ulteriori informazioni sul tracciamento delle intestazioni, consulta la Intestazione di tracciamento pagina concettuale. La propagazione del contesto di traccia di Amazon S3 supporta i seguenti abbonati: Lambda, SQS e SNS. Poiché SQS e SNS non emettono autonomamente i dati dei segmenti, non verranno visualizzati nella traccia o nella mappa di traccia quando vengono attivati da S3, anche se propagheranno l'intestazione di tracciamento ai servizi a valle.

Configurazione delle notifiche degli eventi di Amazon S3

Con la funzione di notifica di Amazon S3, ricevi notifiche quando si verificano determinati eventi nel tuo bucket. Queste notifiche possono quindi essere propagate alle seguenti destinazioni all'interno dell'applicazione:

- Servizio di notifica semplice Amazon (Amazon Simple Notification Service (Amazon SNS))
- Amazon Simple Queue Service (Amazon SQS)
- AWS Lambda

Per un elenco degli eventi supportati, consulta Tipi di eventi supportati nella guida per sviluppatori di Amazon S3.

Amazon SNS e Amazon SQS

Per pubblicare notifiche su un argomento SNS o una coda SQS, devi prima concedere le autorizzazioni ad Amazon S3. Per concedere queste autorizzazioni, alleghi una policy AWS Identity and Access Management (IAM) all'argomento SNS o alla coda SQS di destinazione. Per ulteriori informazioni sulle policy IAM richieste, consulta Concessione delle autorizzazioni per pubblicare messaggi su un argomento SNS o su una coda SQS.

Per informazioni sull'integrazione di SNS e SQS con X-Ray, vedere, e. Amazon SNS e AWS X-Ray Amazon SQS e AWS X-Ray

AWS Lambda

Quando usi la console Amazon S3 per configurare le notifiche di eventi su un bucket S3 per una funzione Lambda, la console imposta le autorizzazioni necessarie sulla funzione Lambda in modo che Amazon S3 disponga delle autorizzazioni per richiamare la funzione dal bucket. Per ulteriori informazioni, consulta Come posso abilitare e configurare le notifiche di eventi per un bucket S3? nella Guida per l'utente della console di Amazon Simple Storage Service.

Puoi anche concedere ad Amazon S3 le autorizzazioni AWS Lambda per richiamare la tua funzione Lambda. Per ulteriori informazioni, consulta Tutorial: Using AWS Lambda with Amazon S3 nella AWS Lambda Developer Guide.

Per ulteriori informazioni sull'integrazione di Lambda con X-Ray, consulta Strumentazione del codice Java in Lambda. AWS

Creazione di risorse radiografiche conAWS CloudFormation

AWS X-Ray è integrato con AWS CloudFormation, un servizio che ti consente di modellare e configurare le tue risorse AWS in modo da dedicare meno tempo alla creazione e alla gestione delle risorse e dell'infrastruttura. Puoi creare un modello che descrive tutte le risorse AWS desiderate e AWS CloudFormation si occuperà del provisioning e della configurazione di queste risorse per tuo conto.

Quando si utilizzaAWS CloudFormation, è possibile riutilizzare il modello per configurare le risorse radiografiche in modo coerente e ripetuto. Basta descrivere le risorse una volta sola, dopodiché si può effettuare il provisioning di tali risorse quante volte si vuole in più Account AWS e regioni.

X-Ray eAWS CloudFormationmodelli

Per fornire e configurare le risorse per X-Ray e i servizi correlati, è necessario comprendereAWS CloudFormationmodelli. I modelli sono file di testo formattati in JSON o YAML. Questi modelli descrivono le risorse di cui intendi effettuare il provisioning negli stack AWS CloudFormation. Se non hai familiarità con JSON o YAML, puoi usare AWS CloudFormation Designer per iniziare a utilizzare i modelli AWS CloudFormation. Per ulteriori informazioni, consulta Che cos'è AWS CloudFormation Designer? nella Guida per l'utente di AWS CloudFormation.

X-Ray supporta la creazione`AWS::XRay::Group`e`AWS::XRay::SamplingRule`risorse inAWS CloudFormation. Per ulteriori informazioni, inclusi esempi di modelli JSON e YAML, consultaRiferimento al tipo di risorsa X-RaynelAWS CloudFormationGuida per l'utente.

Ulteriori informazioni su AWS CloudFormation

Per ulteriori informazioni su AWS CloudFormation, consulta le seguenti risorse:

- AWS CloudFormation
- Guida per l'utente di AWS CloudFormation
- Documentazione di riferimento dell'API AWS CloudFormation
- Guida per l'utente dell'interfaccia a riga di comando di AWS CloudFormation

Tagging delle regole e dei gruppi di campionamento di X-Ray

I tag sono parole o frasi che puoi utilizzare per identificare e organizzare le risorse AWS. È possibile aggiungere più tag a ciascuna risorsa. Ciascun tag include una chiave e un valore opzionale definiti dall'utente. Ad esempio, una chiave tag potrebbe essere**`domain`**e il valore del tag potrebbe essere**`example.com`**. Puoi cercare e filtrare le risorse in base ai tag aggiunti. Per ulteriori informazioni su come utilizzare i tag, consultaApplicazione di tagAWSrisorsenellaAWSRiferimenti generali.

Puoi utilizzare i tag per applicare le autorizzazioni basate su tag alle distribuzioni CloudFront. Per ulteriori informazioni, consultaControllo dell'accesso adAWSRisorse che utilizzano i tag delle risorse.

> **Note**
>
> Editor di tageAWSGruppi di risorseal momento non supporta le risorse X-Ray. Aggiunta e gestione dei tag tramiteAWS X-Rayconsole o API.

Puoi applicare tag a risorse utilizzando la console X-Ray, l'API,AWS CLI, SDK eAWS Tools for Windows PowerShell. Per ulteriori informazioni, consulta la seguente documentazione:

- API X-Ray: vedere le seguenti operazioni nelAWS X-RayDocumentazione di riferimento API:
  - ListTagsForResource
  - Crea una regola di campionamento
  - CreateGroup
  - TagResource
  - UntagResource
- AWS CLI— VedereX-RaynellaAWS CLIRiferimento ai comandi
- SDK: consulta la documentazione relativa all'SDK applicabile nella pagina Documentazione di AWS

> **Note**
>
> Se non è possibile aggiungere o modificare tag in una risorsa a X-Ray o non è possibile aggiungere una risorsa con tag specifici, è possibile che non si disponga delle autorizzazioni per eseguire questa operazione. Per richiedere l'accesso, contatta unAWSutente della tua azienda che haAmministratoreautorizzazioni in X-Ray.

Argomenti

- Limitazioni applicate ai tag
- Gestione dei tag nella console
- Gestione dei tag nellaAWS CLI
- Controlla l'accesso alle risorse X-Ray in base ai tag

Limitazioni applicate ai tag

Ai tag si applicano le limitazioni seguenti.

- Numero massimo di tag per risorsa: 50
- lunghezza massima della chiave: 128 caratteri Unicode;
- lunghezza massima del valore: 256 caratteri Unicode;
- Valori validi per la chiave e il valore - a-z, A-Z, 0-9, spazi e i seguenti caratteri: _ . : / = + - e @
- i valori e le chiavi dei tag rispettano la distinzione tra maiuscole e minuscole;
- Non utilizzare`aws:`come prefisso per le chiavi; l'utilizzo di questo prefisso è esclusivo diAWSutilizzare.

> **Note**
>
> Non è possibile modificare o eliminare i tag di sistema.

Gestione dei tag nella console

È possibile aggiungere tag opzionali durante la creazione di un gruppo di X-Ray o una regola di campionamento. I tag possono anche essere modificati o eliminati nella console in un secondo momento.

Le procedure seguenti descrivono come aggiungere, modificare ed eliminare tag per i gruppi e le regole di campionamento nella console X-Ray.

Argomenti

- Aggiunta di tag a un nuovo gruppo (console)
- Aggiunta di tag a una nuova regola di campionamento (console)
- Modificare o eliminare tag per un gruppo (console)
- Modificare o eliminare tag per una regola di campionamento (console)

Aggiunta di tag a un nuovo gruppo (console)

Mentre crei un nuovo gruppo a X-Ray, puoi aggiungere tag facoltativi sulCreazione di gruppo(Certificato creato).

1. Eseguire l'accesso adAWS Management Consolee aprire la console X-Ray all'indirizzohttps://console.aws.amazon.com/xray/home.
2. Nel riquadro di navigazione, espandiConfigurazionee scegliGruppi.
3. Seleziona Create group (Crea gruppo).
4. SulCreazione di gruppopage, specificare un nome e un'espressione di filtro per il gruppo. Per ulteriori informazioni su queste proprietà, consultare Configurazione dei gruppi.
5. Nello statoTag, immettere una chiave di tag e un valore di tag opzionale. Ad esempio, è anche possibile immettere una chiave del tag di**`Stage`**e un valore di tag di**`Production`**, per indicare che questo gruppo è destinato alla produzione. Quando aggiungi un tag, viene visualizzata una nuova riga per aggiungere un altro tag, se necessario. Consulta .Limitazioni applicate ai tagin questo argomento per limitazioni sui tag.
6. Una volta completata l'aggiunta di tag, scegliereCreazione di gruppo.

Aggiunta di tag a una nuova regola di campionamento (console)

Man mano che crei una nuova regola di campionamento a X-Ray, puoi aggiungere tag nellaCreazione di regole di campionamento(Certificato creato).

1. Eseguire l'accesso adAWS Management Consolee aprire la console X-Ray all'indirizzohttps://console.aws.amazon.com/xray/home.
2. Nel riquadro di navigazione, espandiConfigurazionee scegliCampionamento.
3. ScegliereCreazione di regole di campionamento.
4. SulCreazione di regole di campionamentopagina, specificare un nome, una priorità, i limiti, i criteri corrispondenti e gli attributi corrispondenti. Per ulteriori informazioni su queste proprietà, consultare Configurazione delle regole di campionamento di .
5. Nello statoTag, immettere una chiave di tag e un valore di tag opzionale. Ad esempio, è anche possibile immettere una chiave del tag di**`Stage`**e un valore di tag di**`Production`**, per indicare che questa regola di campionamento è destinata all'uso in produzione. Quando aggiungi un tag, viene visualizzata una nuova riga per aggiungere un altro tag, se necessario. Consulta .Limitazioni applicate ai tagin questo argomento per limitazioni sui tag.
6. Una volta completata l'aggiunta di tag, scegliereCreazione di regole di campionamento.

Modificare o eliminare tag per un gruppo (console)

È possibile modificare o eliminare i tag di un gruppo X-Ray sulModifica del gruppo(Certificato creato).

1. Eseguire l'accesso adAWS Management Consolee aprire la console X-Ray all'indirizzohttps://console.aws.amazon.com/xray/home.
2. Nel riquadro di navigazione, espandiConfigurazionee scegliGruppi.
3. NellaGruppi, scegliere il nome di un gruppo.
4. SulModifica del gruppopagina, inTag, modificare le chiavi e i valori dei tag. Non è possibile disporre di chiavi tag duplicate. I valori dei tag sono facoltativi; è possibile eliminare i valori se lo si desidera. Per ulteriori informazioni su altre proprietà sulModifica del gruppopagina, vediConfigurazione dei gruppi. Consulta .Limitazioni applicate ai tagin questo argomento per limitazioni sui tag.
5. Per rimuovere un tag, scegliXa destra della targhetta.
6. Una volta completata la modifica o l'eliminazione di tag, sceglieriAggiornamento del gruppo.

Modificare o eliminare tag per una regola di campionamento (console)

È possibile modificare o eliminare i tag su una regola di campionamento a X-Ray sulModifica regola di campionamento(Certificato creato).

1. Eseguire l'accesso adAWS Management Consolee aprire la console X-Ray all'indirizzohttps://console.aws.amazon.com/xray/home.
2. Nel riquadro di navigazione, espandiConfigurazionee scegliCampionamento.
3. NellaRegole di campionamento, scegliere il nome di una regola di campionamento.
4. Nello statoTag, modificare le chiavi e i valori dei tag. Non è possibile disporre di chiavi tag duplicate. I valori dei tag sono facoltativi; è possibile eliminare i valori se lo si desidera. Per ulteriori informazioni su altre proprietà sulModifica regola di campionamentopagina, vediConfigurazione delle regole di campionamento di . Consulta .Limitazioni applicate ai tagin questo argomento per limitazioni sui tag.
5. Per rimuovere un tag, scegliXa destra della targhetta.
6. Una volta completata la modifica o l'eliminazione di tag, scegliereAggiornamento della regola di campionamento.

Gestione dei tag nellaAWS CLI

È possibile aggiungere tag quando si crea un gruppo X-Ray o una regola di campionamento. È possibile utilizzare anche laAWS CLIper creare e gestire i tag. Per aggiornare i tag su un gruppo o una regola di campionamento esistente, utilizzare ilAWS X-Rayconsole oTagResourceoUntagResourceAPI.

Argomenti

- Aggiunta di tag a un nuovo gruppo di X-Ray o a una regola di campionamento (CLI)
- Aggiunta di tag a una risorsa esistente (CLI)
- Elenca i tag su una risorsa (CLI)
- Elimina i tag su una risorsa (CLI)

Aggiunta di tag a un nuovo gruppo di X-Ray o a una regola di campionamento (CLI)

Per aggiungere tag opzionali durante la creazione di un nuovo gruppo di X-Ray o una regola di campionamento, utilizzare uno dei seguenti comandi.

- Per aggiungere tag a un nuovo gruppo, eseguire il comando seguente, sostituendo*group_name*con il nome del gruppo,*mydomain.com*con l'endpoint del tuo servizio,*nome_chiave*con un tag key e, facoltativamente,*valore*con un valore del tag. Per ulteriori informazioni su come creare un gruppo, consultaGruppi.

  ```
  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"}]
  ```

  Di seguito è riportato un esempio.

  ```
  aws xray create-group \
     --group-name "AdminGroup" \
     --filter-expression "service(\"mydomain.com\") {fault OR error}" \
     --tags [{"Key": "Stage","Value": "Prod"},{"Key": "Department","Value": "QA"}]
  ```

- Per aggiungere tag a una nuova regola di campionamento, eseguire il comando seguente, sostituendo*nome_chiave*con un tag key e, facoltativamente,*valore*con un valore del tag. Questo comando specifica i valori nella`--sampling-rule`parametro come file JSON. Per ulteriori informazioni su come creare una regola di campionamento, consultaRegole di campionamento.

  ```
  aws xray create-sampling-rule \
     --cli-input-json file://file_name.json
  ```

  Di seguito sono riportati i contenuti del file JSON*nome_file.json*che è specificato dal`--cli-input-json`Parametro .

  ```
  {
      "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"
           }
          ]
}
```

Il comando seguente è un esempio.

```
aws xray create-sampling-rule \
   --cli-input-json file://9000-base-scorekeep.json
```

Di seguito sono riportati i contenuti dell'esempio`9000-base-scorekeep.json`file specificato dal`--cli-input-json`Parametro .

```
{
    "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"
           }
          ]
}
```

Aggiunta di tag a una risorsa esistente (CLI)

È anche possibile emettere`tag-resource`comando per aggiungere tag a un gruppo di X-Ray o a una regola di campionamento esistente Questo metodo potrebbe essere più semplice dell'aggiunta di tag eseguendo`update-group`o`update-sampling-rule`.

Per aggiungere tag a un gruppo o a una regola di campionamento, eseguire il comando seguente, sostituendo l'ARN con l'ARN della risorsa e specificando le chiavi e i valori facoltativi dei tag che si desidera aggiungere.

```
aws xray tag-resource \
   --resource-arn "ARN" \
   --tag-keys [{"Key":"key_name","Value":"value"}, {"Key":"key_name","Value":"value"}]
```

Di seguito è riportato un esempio.

```
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"}]
```

Elenca i tag su una risorsa (CLI)

È anche possibile emettere`list-tags-for-resource`comando per elencare i tag di un gruppo di X-Ray o di una regola di campionamento.

Per elencare i tag associati a un gruppo o a una regola di campionamento, eseguire il comando seguente, sostituendo l'ARN con l'ARN della risorsa.

```
aws xray list-tags-for-resource \
   --resource-arn "ARN"
```

Di seguito è riportato un esempio.

```
aws xray list-tags-for-resource \
   --resource-arn "arn:aws:xray:us-east-2:01234567890:group/AdminGroup"
```

Elimina i tag su una risorsa (CLI)

È anche possibile emettereuntag-resourcecomando per rimuovere tag da un gruppo X-Ray o una regola di campionamento.

Per rimuovere i tag da un gruppo o da una regola di campionamento, eseguire il comando seguente, sostituendo l'ARN con l'ARN della risorsa e specificando le chiavi dei tag che si desidera rimuovere.

È possibile rimuovere solo intere tag conuntag-resourcecomando. Per rimuovere i valori dei tag, utilizzare la console X-Ray oppure eliminare i tag e aggiungere nuovi tag con le stesse chiavi, ma valori diversi o vuoti.

```
aws xray untag-resource \
   --resource-arn "ARN" \
   --tag-keys ["key_name","key_name"]
```

Di seguito è riportato un esempio.

```
aws xray untag-resource \
   --resource-arn "arn:aws:xray:us-east-2:01234567890:group/group_name" \
   --tag-keys ["Stage","Department"]
```

Controlla l'accesso alle risorse X-Ray in base ai tag

È possibile allegare tag a gruppi di raggi X o regole di campionamento o passare i tag in una richiesta a X-Ray. Per controllare l'accesso basato su tag, fornire informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `xray:ResourceTag/key-name`,

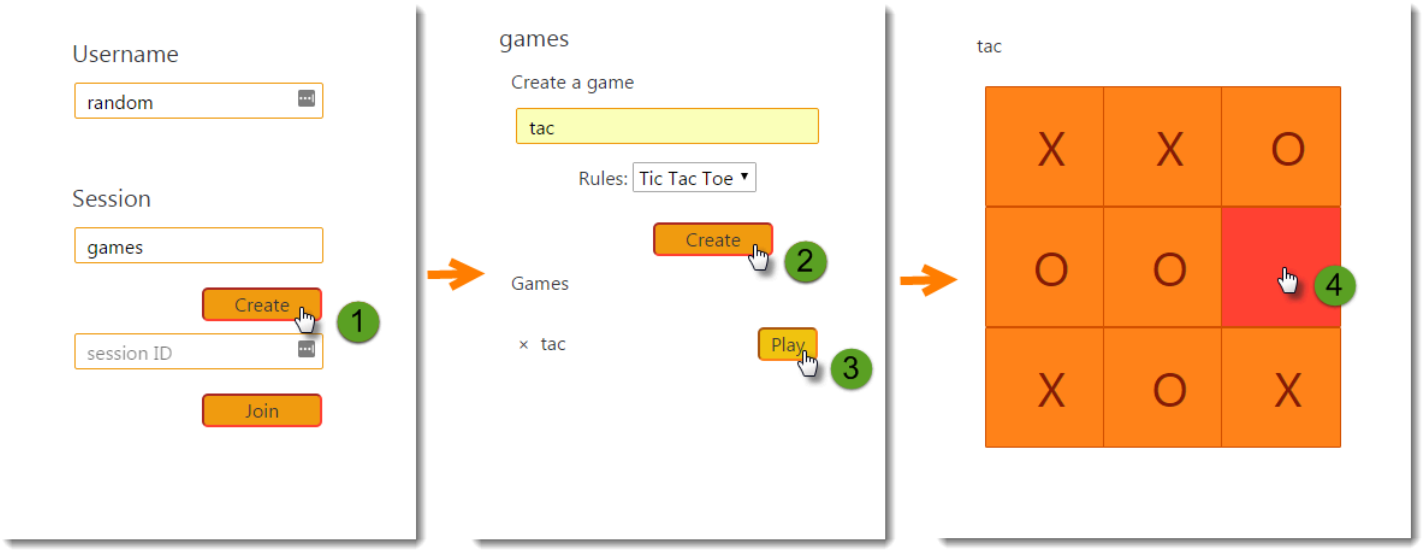
`aws:RequestTag/key-name` o `aws:TagKeys`. Per ulteriori informazioni su queste chiavi di condizione, consultaControllo dell'accesso adAWSrisorse che utilizzano i tag delle risorse.

Per visualizzare una policy basata sulle identità di esempio per limitare l'accesso a una risorsa basata su tag su tale risorsa, consulta Gestione dell'accesso ai gruppi X-Ray e alle regole di campionamento basate sui tag.

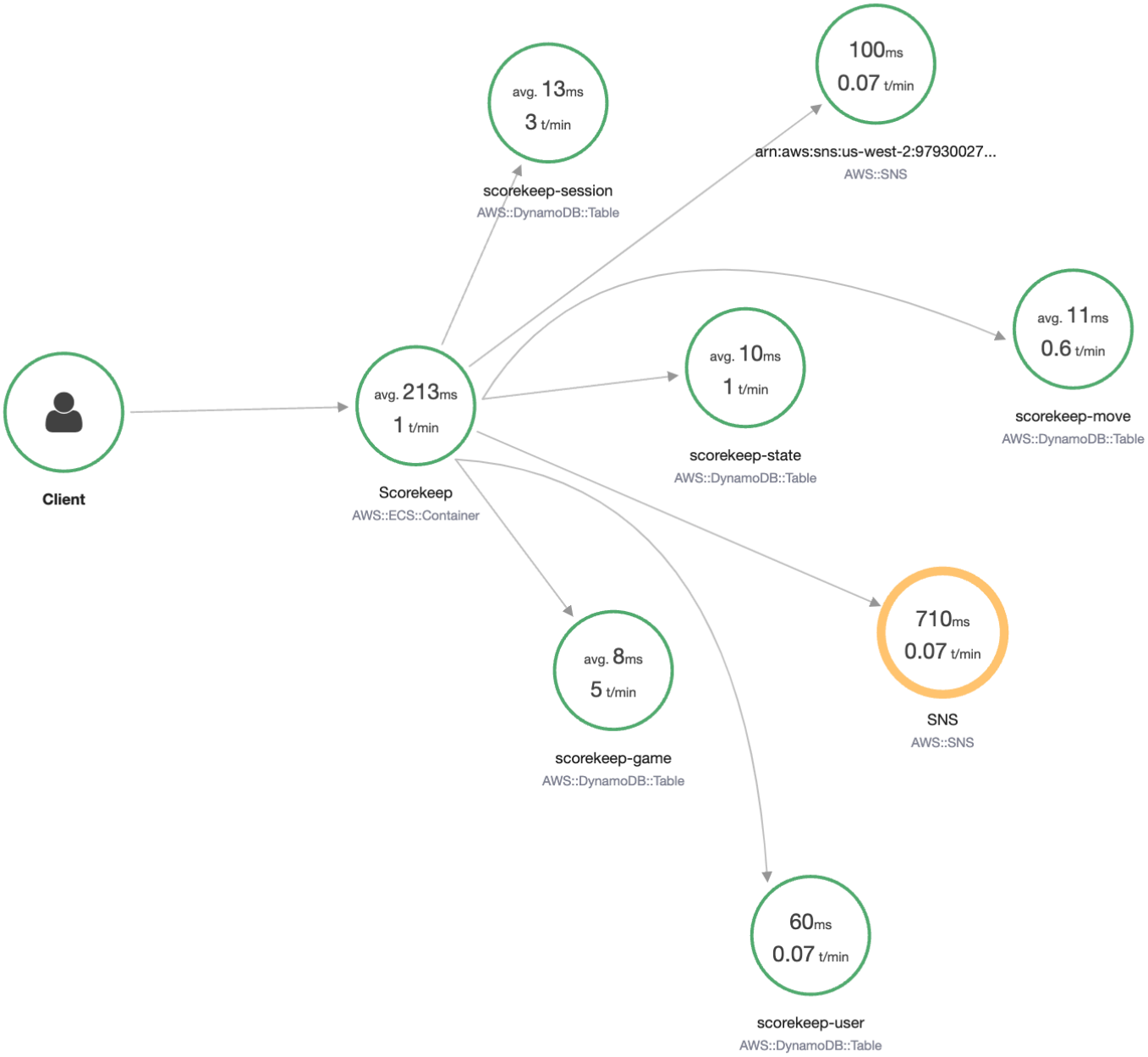
AWS X-Ray applicazione di esempio

L'app di eb-java-scorekeepesempio AWS X-Ray, disponibile su GitHub, mostra l'uso dell'SDK AWS X-Ray per monitorare le chiamate HTTP in entrata, i client SDK DynamoDB e i client HTTP. L'app di esempio consente AWS CloudFormation di creare tabelle DynamoDB, compilare codice Java su istanza ed eseguire il demone X-Ray senza alcuna configurazione aggiuntiva.

Consulta il tutorial Scorekeep per iniziare a installare e utilizzare un'applicazione di esempio strumentata, utilizzando o il. AWS Management Console AWS CLI



L'esempio include un'app Web front-end, l'API che chiama e le tabelle DynamoDB che utilizza per archiviare i dati. La strumentazione di base con filtri, plugin e client AWS SDK strumentati è mostrata nella sezione dedicata al progetto. `xray-gettingstarted` Questo è il ramo che viene distribuito nel tutorial sulle nozioni di base. Poiché questo ramo include solo le nozioni di base, puoi eseguire una diff rispetto al ramo `master` per individuare rapidamente le nozioni di base.



L'applicazione di esempio illustra l'analisi di base in questi file:

- Filtro di richiesta HTTP: **`WebConfig.java`**
- AWS Strumentazione client SDK — `build.gradle`

Il `xray` ramo dell'applicazione include l'uso di HttpClient, Annotations, query SQL, sottosegmenti personalizzati, una funzione strumentata e codice e script di inizializzazione strumentati AWS Lambda.

Per supportare l'accesso e l' AWS SDK for JavaScript utilizzo degli utenti nel browser, la `xray-cognito` filiale aggiunge Amazon Cognito per supportare l'autenticazione e l'autorizzazione degli utenti. Con le credenziali recuperate da Amazon Cognito, l'app Web invia anche i dati di traccia a X-Ray per registrare le informazioni sulla richiesta dal punto di vista del cliente. Il client del browser appare come un nodo a sé stante sulla mappa di tracciamento e registra informazioni aggiuntive, tra cui l'URL della pagina che l'utente sta visualizzando e l'ID dell'utente.

Infine, il `xray-worker` ramo aggiunge una funzione Python Lambda strumentata che viene eseguita in modo indipendente, elaborando gli elementi da una coda Amazon SQS. Scorekeep aggiunge un elemento alla coda ogni volta che termina una partita. L'operatore Lambda, attivato da CloudWatch Events, estrae gli elementi dalla coda ogni pochi minuti e li elabora per archiviare i record di gioco in Amazon S3 per l'analisi.

Argomenti

- Guida introduttiva all'applicazione di esempio Scorekeep
- Strumentazione manuale dei client SDK AWS
- Creazione di sottosegmenti aggiuntivi
- Registrazione di annotazioni, metadati e ID utente
- Analisi delle chiamate HTTP in uscita
- Analizzare le chiamate a un database PostgresSQL
- AWS Lambda Funzioni di strumentazione
- Analisi del codice di avvio
- Analisi degli script
- Analisi di un client di app Web
- Utilizzo dei client analizzati nei thread worker

Guida introduttiva all'applicazione di esempio Scorekeep

Questo tutorial utilizza il `xray-gettingstarted` ramo dell'applicazione di esempio Scorekeep, che utilizza AWS CloudFormation per creare e configurare le risorse che eseguono l'applicazione di esempio e il demone X-Ray su Amazon ECS. L'applicazione utilizza il framework Spring per

implementare un'API Web JSON e AWS SDK for Java per rendere persistenti i dati su Amazon DynamoDB. Un servlet filtra nell'applicazione per tutte le richieste in entrata servite dall'applicazione e un gestore di richieste sugli strumenti client AWS SDK per le chiamate downstream a DynamoDB.

Puoi seguire questo tutorial usando il o il. AWS Management Console AWS CLI

Sections

- Prerequisiti
- Installa l'applicazione Scorekeep utilizzando CloudFormation
- Generare i dati di tracciamento
- Visualizza la mappa di tracciamento nel AWS Management Console
- Configurazione delle notifiche Amazon SNS
- Esplorare l'applicazione di esempio
- Opzionale: policy con privilegio minimo
- Eliminazione
- Passaggi successivi

Prerequisiti

Questo tutorial serve AWS CloudFormation a creare e configurare le risorse che eseguono l'applicazione di esempio e il demone X-Ray. I seguenti prerequisiti sono necessari per l'installazione e l'esecuzione del tutorial:

1. Se utilizzi un utente IAM con autorizzazioni limitate, aggiungi le seguenti politiche utente nella console IAM:

   - `AWSCloudFormationFullAccess`— per accedere e utilizzare CloudFormation
   - `AmazonS3FullAccess`— caricare un file modello da CloudFormation utilizzare AWS Management Console
   - `IAMFullAccess`— per creare i ruoli delle istanze Amazon ECS e Amazon EC2
   - `AmazonEC2FullAccess`— per creare le risorse Amazon EC2
   - `AmazonDynamoDBFullAccess`— per creare le tabelle DynamoDB
   - `AmazonECS_FullAccess`— per creare risorse Amazon ECS
   - `AmazonSNSFullAccess`— per creare l'argomento Amazon SNS

- `AWSXrayReadOnlyAccess`— per il permesso di visualizzare la mappa di tracciamento e le tracce nella console X-Ray

2. Per eseguire il tutorial utilizzando la AWS CLI, installa la CLI versione 2.7.9 o successiva e configura la CLI con l'utente del passaggio precedente. Assicurati che la regione sia configurata durante la configurazione con l'utente. AWS CLI Se una regione non è configurata, sarà necessario aggiungerla `--region` *`AWS-REGION`* a ogni comando CLI.
3. Assicurati che Git sia installato, per clonare il repository dell'applicazione di esempio.
4. Usa il seguente esempio di codice per clonare il `xray-gettingstarted` ramo del repository Scorekeep:

```
git clone https://github.com/aws-samples/eb-java-scorekeep.git xray-scorekeep -b
 xray-gettingstarted
```

Installa l'applicazione Scorekeep utilizzando CloudFormation

AWS Management Console

Installa l'applicazione di esempio utilizzando il AWS Management Console

1. Apri la console CloudFormation
2. Scegli Crea stack, quindi scegli Con nuove risorse dal menu a discesa.
3. Nella sezione Specify template(Specifica il modello) scegliere Upload a template file (Carica un file modello).
4. Seleziona Scegli file, vai alla `xray-scorekeep/cloudformation` cartella che è stata creata quando hai clonato il repository git e scegli il file. `cf-resources.yaml`
5. Seleziona Successivo per continuare.
6. Entra `scorekeep` nella casella di testo Stack name, quindi scegli Avanti nella parte inferiore della pagina per continuare. Nota che il resto di questo tutorial presuppone che lo stack abbia un nome. `scorekeep`
7. Scorri fino alla fine della pagina Configura le opzioni dello stack e scegli Avanti per continuare.
8. Scorri fino alla fine della pagina di revisione, seleziona la casella di controllo che riconosce che CloudFormation potrebbe creare risorse IAM con nomi personalizzati e scegli Create stack.

9. Lo CloudFormation stack è ora in fase di creazione. Lo stato dello stack rimarrà `CREATE_IN_PROGRESS` per circa cinque minuti prima di passare a. `CREATE_COMPLETE` Lo stato verrà aggiornato periodicamente oppure puoi aggiornare la pagina.

AWS CLI

Installa l'applicazione di esempio utilizzando il AWS CLI

1. Accedete alla `cloudformation` cartella del `xray-scorekeep` repository che avete clonato in precedenza nel tutorial:

```
cd xray-scorekeep/cloudformation/
```

2. Immettete il seguente AWS CLI comando per creare lo stack: CloudFormation

```
aws cloudformation create-stack --stack-name scorekeep --capabilities
 "CAPABILITY_NAMED_IAM" --template-body file://cf-resources.yaml
```

3. Attendi che lo stato dello CloudFormation stack sia raggiunto`CREATE_COMPLETE`, operazione che richiederà circa cinque minuti. Usa il seguente AWS CLI comando per controllare lo stato:

```
aws cloudformation describe-stacks --stack-name scorekeep --query
 "Stacks[0].StackStatus"
```

Generare i dati di tracciamento

L'applicazione di esempio include un app web di front-end. Utilizza l'app Web per generare traffico verso l'API e inviare dati di traccia a X-Ray. Innanzitutto, recupera l'URL dell'app Web utilizzando AWS Management Console o: AWS CLI

AWS Management Console

Trovate l'URL dell'applicazione utilizzando il AWS Management Console

1. Apri la console CloudFormation
2. Scegliete lo `scorekeep` stack dall'elenco.
3. Scegli la scheda Output nella pagina dello `scorekeep` stack e scegli il link `LoadBalancerUrl` URL per aprire l'applicazione web.

AWS CLI

Trovate l'URL dell'applicazione utilizzando il AWS CLI

1. Utilizzate il seguente comando per visualizzare l'URL dell'applicazione Web:

```
aws cloudformation describe-stacks --stack-name scorekeep --query
 "Stacks[0].Outputs[0].OutputValue"
```

2. Copia questo URL e aprilo in un browser per visualizzare l'applicazione web Scorekeep.

Utilizza l'applicazione Web per generare dati di traccia

1. Scegliere Create (Crea) per creare un utente e una sessione.
2. Digitare un game name (nome del gioco), impostare Rules (Regole) a Tic Tac Toe (Tris) e quindi scegliere Create (Crea) per creare un gioco.
3. Scegliere Play (Gioca) per iniziare il gioco.
4. Scegliere un riquadro per fare una mossa e modificare lo stato del gioco.

Ciascuno di questi passaggi genera richieste HTTP all'API e chiamate downstream a DynamoDB per leggere e scrivere dati relativi a utenti, sessioni, giochi, spostamenti e stati.

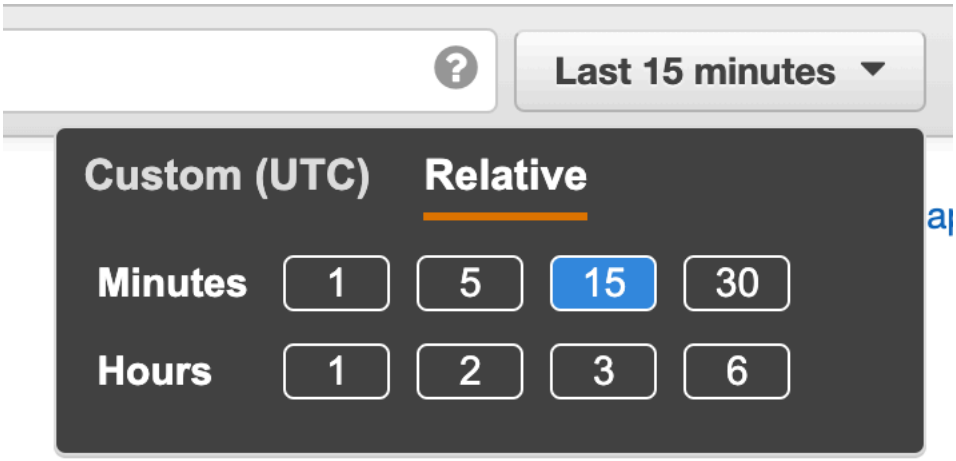
Visualizza la mappa di tracciamento nel AWS Management Console

È possibile visualizzare la mappa di traccia e le tracce generate dall'applicazione di esempio negli X-Ray e CloudWatch nelle console.

X-Ray console

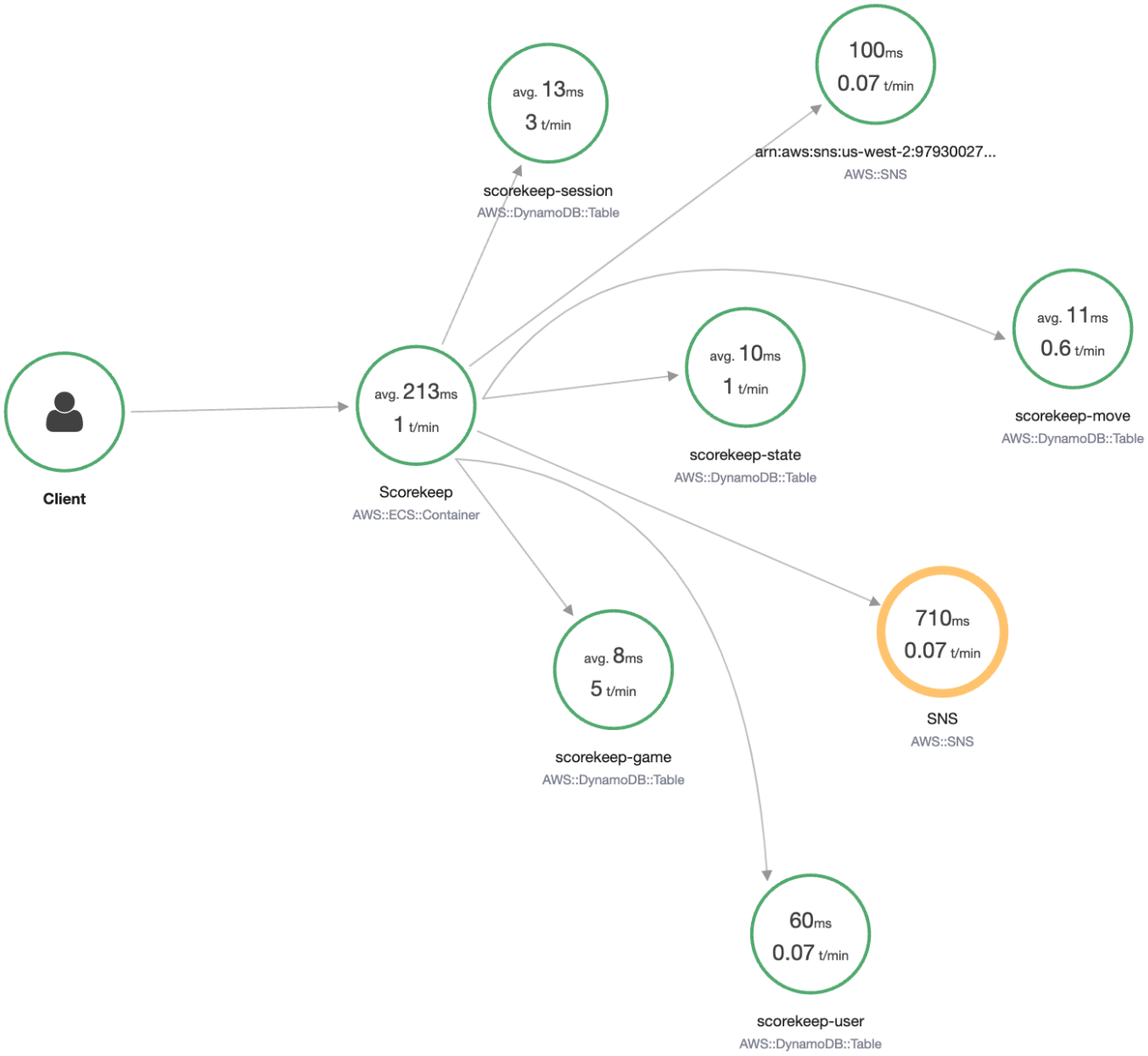
Usa la console X-Ray

1. Apri la pagina della mappa di tracciamento della console X-Ray.
2. La console mostra una rappresentazione del grafico del servizio generato da X-Ray dai dati di traccia inviati dall'applicazione. Assicurati di modificare il periodo di tempo della mappa di traccia, se necessario, per assicurarti che mostri tutte le tracce dal primo avvio dell'applicazione web.



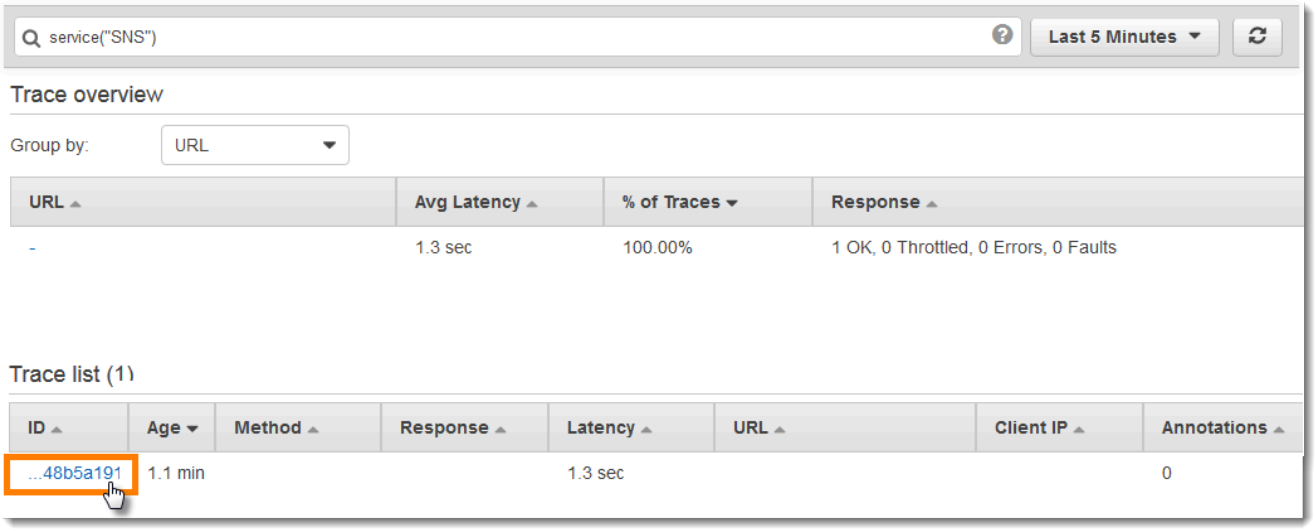
La mappa di traccia mostra il client dell'app Web, l'API in esecuzione in Amazon ECS e ogni tabella DynamoDB utilizzata dall'applicazione. Ogni richiesta all'applicazione, fino a un numero massimo configurabile di richieste al secondo, viene tracciata non appena viene ricevuta dall'API, genera le richieste verso i servizi a valle e si conclude.

Puoi scegliere qualsiasi nodo del grafo del servizio per visualizzare i dati di tracciamento per le richieste che hanno generato il traffico verso tale nodo. Attualmente, il nodo Amazon SNS è giallo. Approfondisci l'esplorazione per scoprire perché.

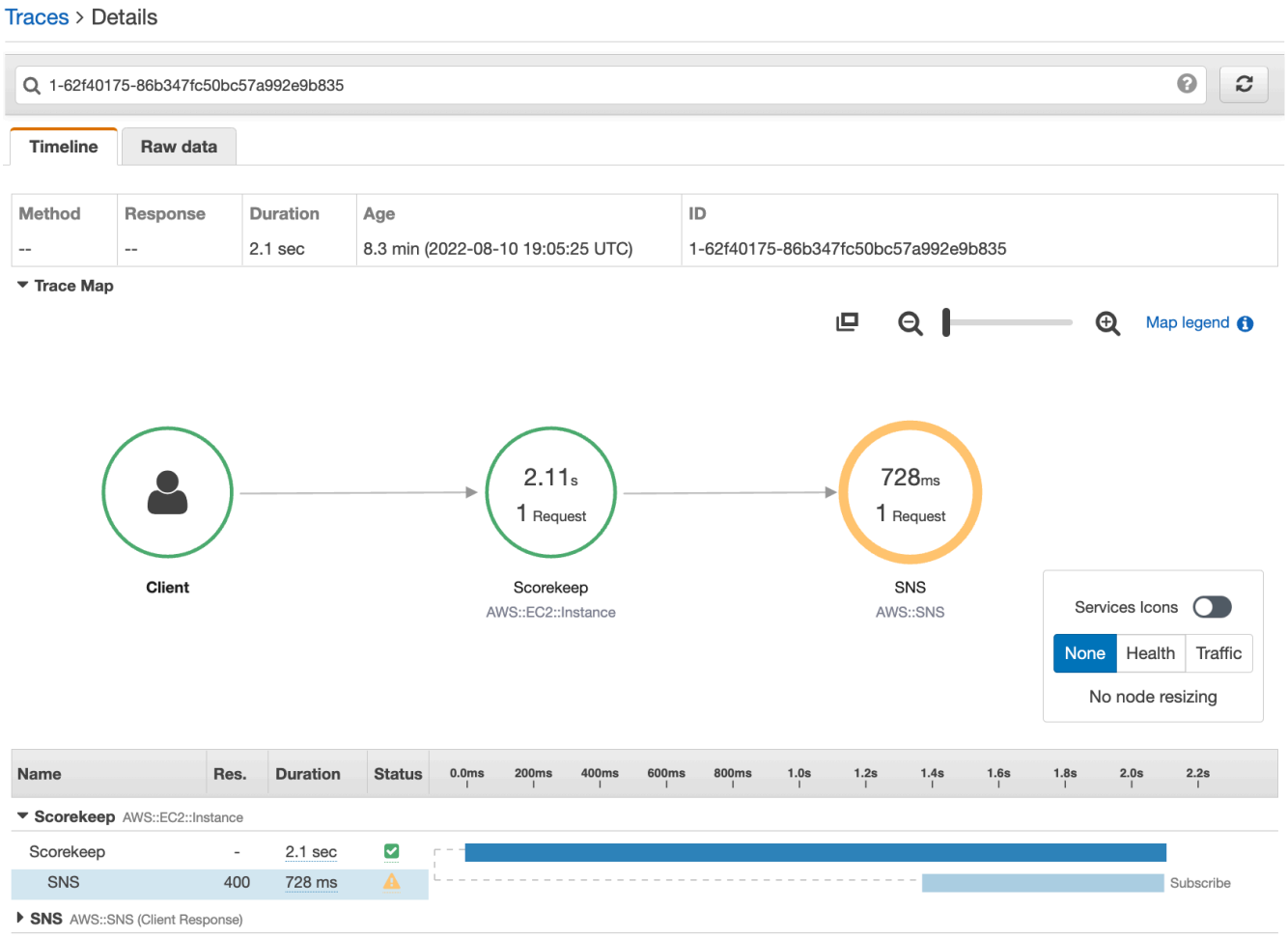


Per trovare la causa dell'errore

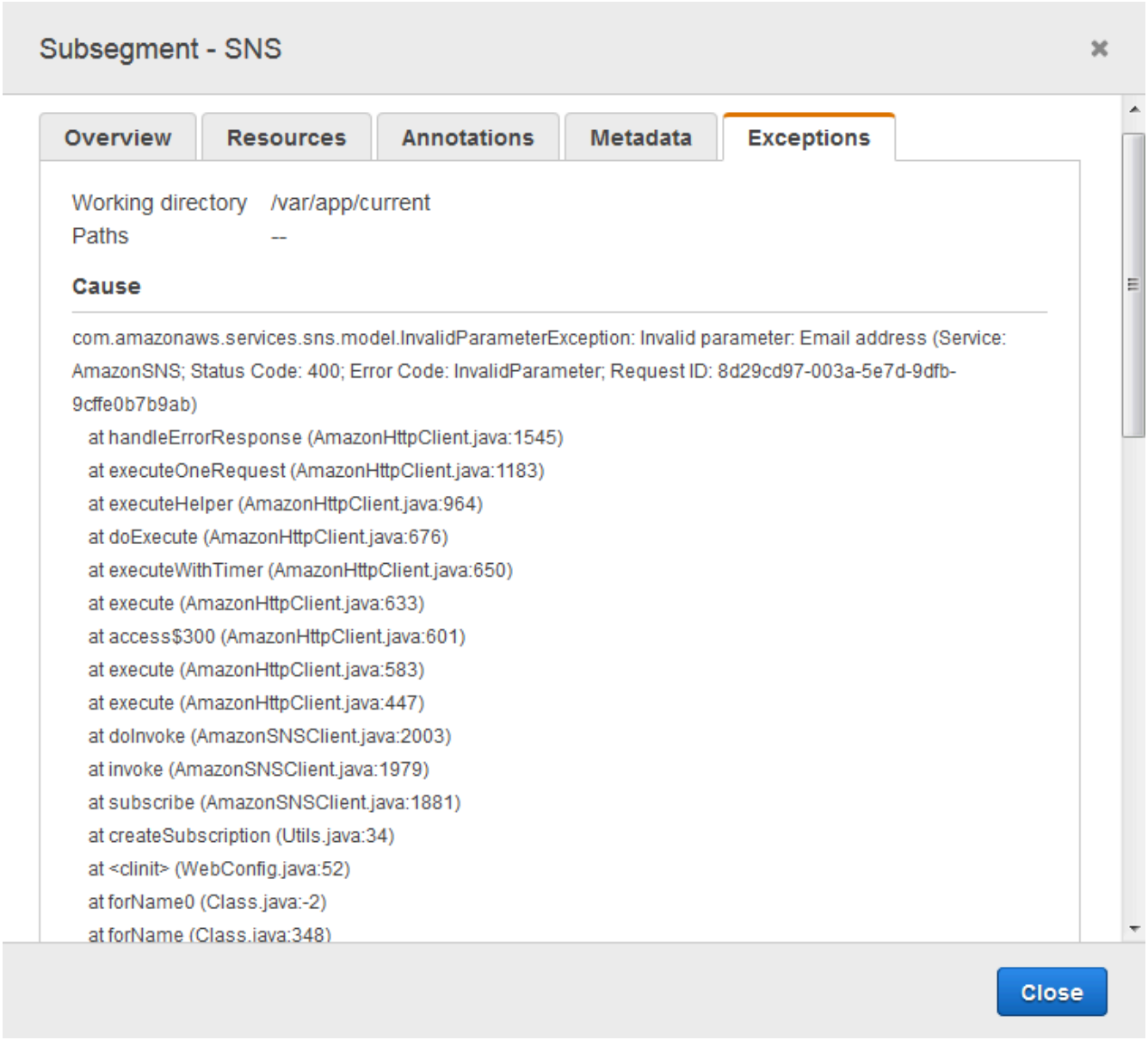
1. Scegliere il nodo denominato SNS. Viene visualizzato il pannello dei dettagli del nodo.
2. Scegliere View traces (Visualizza tracciamenti) per accedere alla schermata Trace overview (Panoramica tracciamenti).
3. Scegliere il tracciamento dalla Trace list (Elenco tracciamenti). A questo tracciamento non sono associati metodo o URL perché è stato registrato durante l'avvio anziché in risposta a una richiesta in entrata.



4. Scegli l'icona dello stato dell'errore all'interno del segmento Amazon SNS nella parte inferiore della pagina per aprire la pagina Eccezioni per il sottosegmento SNS.



5. L'SDK X-Ray acquisisce automaticamente le eccezioni generate dai client AWS SDK strumentati e registra la traccia dello stack.

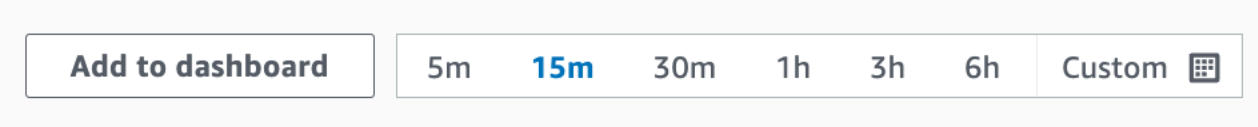


CloudWatch console

Usa la console CloudWatch

1. Apri la pagina della mappa di tracciamento X-Ray della CloudWatch console.
2. La console mostra una rappresentazione del grafico del servizio generato da X-Ray dai dati di traccia inviati dall'applicazione. Assicurati di modificare il periodo di tempo della

mappa di traccia, se necessario, per assicurarti che mostri tutte le tracce dal primo avvio dell'applicazione web.



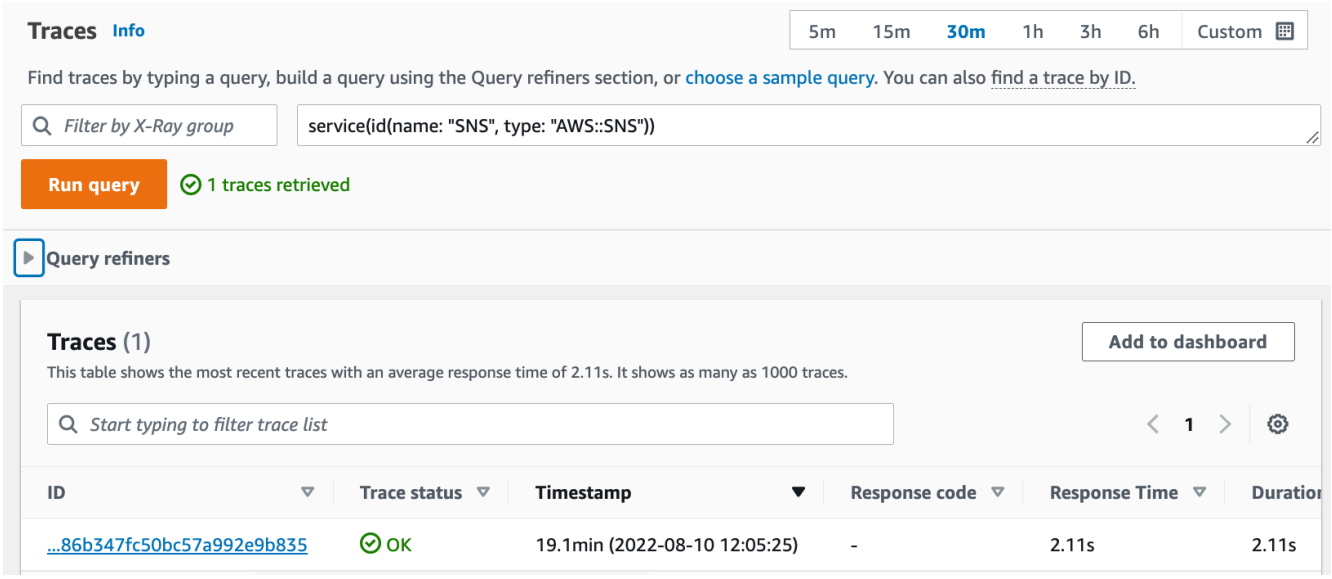
La mappa di traccia mostra il client dell'app Web, l'API in esecuzione in Amazon EC2 e ogni tabella DynamoDB utilizzata dall'applicazione. Ogni richiesta all'applicazione, fino a un numero massimo configurabile di richieste al secondo, viene tracciata non appena viene ricevuta dall'API, genera le richieste verso i servizi a valle e si conclude.

Puoi scegliere qualsiasi nodo del grafo del servizio per visualizzare i dati di tracciamento per le richieste che hanno generato il traffico verso tale nodo. Attualmente, il nodo Amazon SNS è arancione. Approfondisci l'esplorazione per scoprire perché.

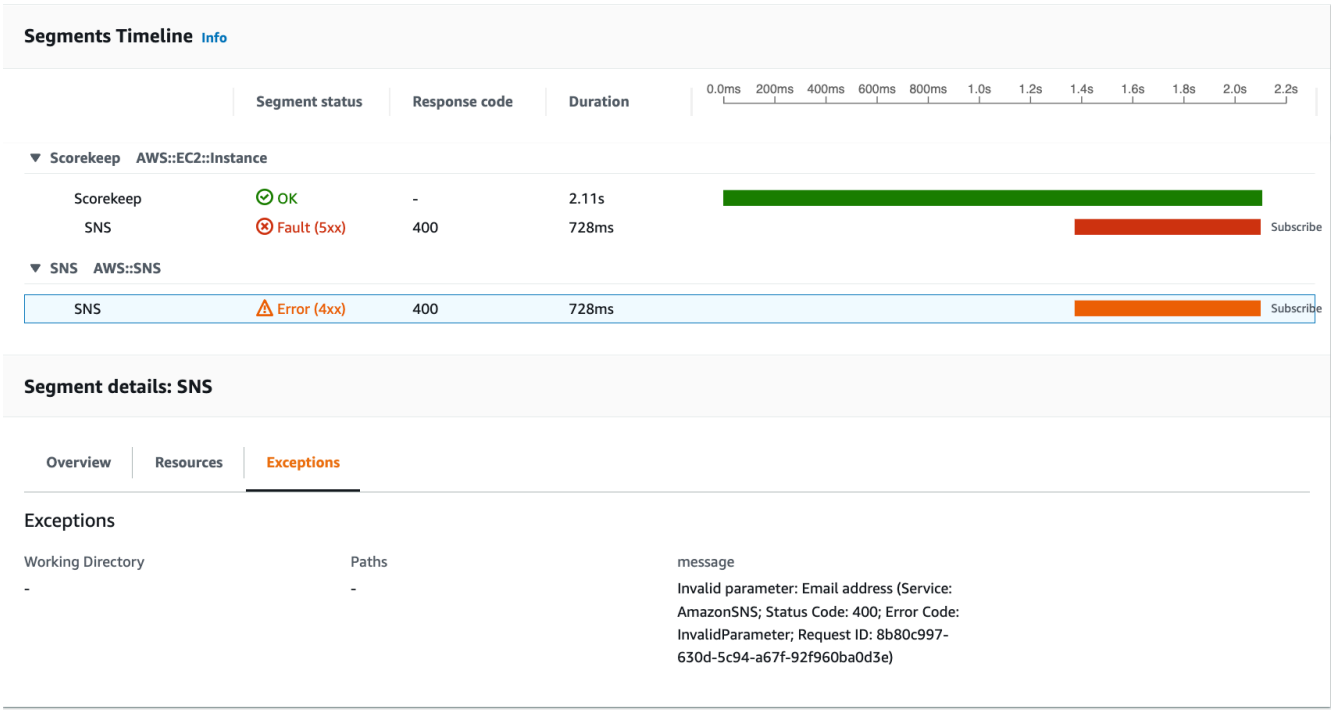

scorekeep-state
DynamoDB Table
scorekeep-game
DynamoDB Table
scorekeep-move
DynamoDB Table
Client
Scorekeep
ECS Container
SNS
SNS
scorekeep-session
DynamoDB Table

Per trovare la causa dell'errore

1. Scegliere il nodo denominato SNS. Il pannello dei dettagli del nodo SNS viene visualizzato sotto la mappa.
2. Scegli Visualizza tracce per accedere alla pagina Traces.
3. Aggiungi la parte inferiore della pagina, scegli la traccia dall'elenco delle tracce. A questo tracciamento non sono associati metodo o URL perché è stato registrato durante l'avvio anziché in risposta a una richiesta in entrata.



4. Scegli il sottosegmento Amazon SNS nella parte inferiore della cronologia dei segmenti e scegli la scheda Eccezioni per il sottosegmento SNS per visualizzare i dettagli dell'eccezione.



La causa indica che l'indirizzo e-mail fornito in una chiamata a `createSubscription` invocata nella classe `WebConfig` non è valido. Nella prossima sezione, lo risolveremo.

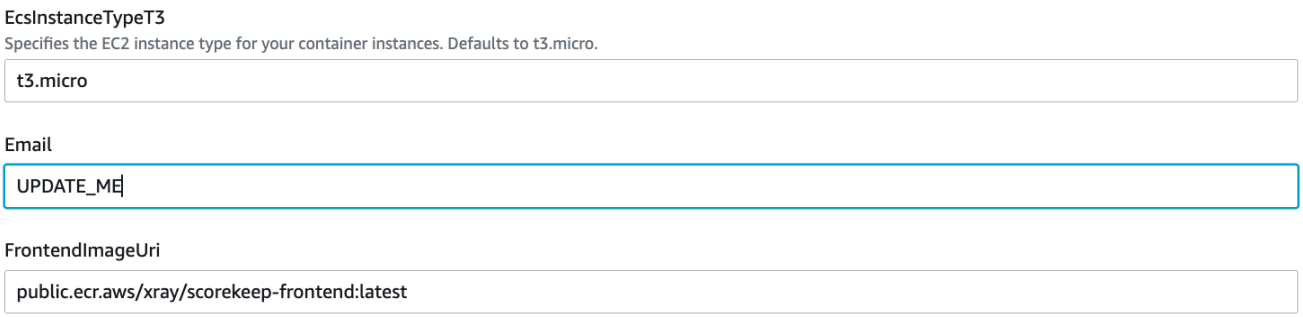
Configurazione delle notifiche Amazon SNS

Scorekeep utilizza Amazon SNS per inviare notifiche quando gli utenti completano una partita. All'avvio, l'applicazione tenta di creare un abbonamento per un indirizzo e-mail definito in un CloudFormation parametro stack. Al momento la chiamata non è riuscita. Configura un'e-mail di notifica per abilitare le notifiche e risolvere gli errori evidenziati nella mappa di traccia.

AWS Management Console

Per configurare le notifiche di Amazon SNS utilizzando AWS Management Console

1. Apri la console CloudFormation
2. Scegli il pulsante di opzione accanto al nome dello **`scorekeep`** stack nell'elenco, quindi scegli Aggiorna.
3. Assicurati che sia selezionata l'opzione Usa modello corrente, quindi fai clic su Avanti nella pagina Aggiorna stack.
4. Individuate il parametro Email nell'elenco e sostituite il valore predefinito con un indirizzo e-mail valido.



5. Scorri fino alla parte inferiore della pagina e scegli Next (Avanti).
6. Scorri fino alla fine della pagina di revisione, seleziona la casella di controllo che conferma che CloudFormation potrebbe creare risorse IAM con nomi personalizzati e scegli Update stack.
7. Lo CloudFormation stack è ora in fase di aggiornamento. Lo stato dello stack rimarrà `UPDATE_IN_PROGRESS` per circa cinque minuti prima di passare a. `UPDATE_COMPLETE` Lo stato verrà aggiornato periodicamente oppure puoi aggiornare la pagina.

AWS CLI

Per configurare le notifiche di Amazon SNS utilizzando AWS CLI

1. Accedi alla `xray-scorekeep/cloudformation/` cartella creata in precedenza e apri il `cf-resources.yaml` file in un editor di testo.
2. Trovate il `Default` valore all'interno del parametro Email e modificatelo da *`UPDATE_ME`* a un indirizzo e-mail valido.

```
Parameters:
  Email:
    Type: String
    Default: UPDATE_ME # <- change to a valid abc@def.xyz email address
```

3. Dalla `cloudformation` cartella, aggiorna lo CloudFormation stack con il seguente comando: AWS CLI

```
aws cloudformation update-stack --stack-name scorekeep --capabilities
 "CAPABILITY_NAMED_IAM" --template-body file://cf-resources.yaml
```

4. Attendi che lo stato dello CloudFormation stack sia `UPDATE_COMPLETE` raggiunto, operazione che richiederà alcuni minuti. Usa il seguente AWS CLI comando per controllare lo stato:

```
aws cloudformation describe-stacks --stack-name scorekeep --query
 "Stacks[0].StackStatus"
```

Quando l'aggiornamento viene completato, Scorekeep si riavvia e crea una sottoscrizione per l'argomento SNS. Controlla la tua e-mail e conferma che la sottoscrizione per visualizzare gli aggiornamenti al completamento di un gioco. Apri la mappa di tracciamento per verificare che le chiamate a SNS non abbiano più esito negativo.

Esplorare l'applicazione di esempio

L'applicazione di esempio è un'API Web HTTP in Java configurata per utilizzare X-Ray SDK for Java. Quando distribuisci l'applicazione con il CloudFormation modello, crea le tabelle DynamoDB, Amazon ECS Cluster e altri servizi necessari per eseguire Scorekeep su ECS. Un file di definizione delle attività per ECS viene creato tramite. CloudFormation Questo file definisce le immagini dei contenitori utilizzate per attività in un cluster ECS. Queste immagini sono ottenute dall'ECR pubblico ufficiale di X-Ray. L'immagine del contenitore dell'API scorekeep ha l'API compilata con Gradle. L'immagine del contenitore del frontend Scorekeep serve il frontend utilizzando il server proxy nginx. Questo server indirizza le richieste verso percorsi che iniziano con /api all'API.

Per analizzare le richieste HTTP in ingresso, l'applicazione aggiunge il `TracingFilter` fornito dall'SDK.

Example WebConfigsrc/main/java/scorekeep/ .java - filtro servlet

```
import javax.servlet.Filter;
import com.amazonaws.xray.javax.servlet.AWSXRayServletFilter;
...

@Configuration
public class WebConfig {

  @Bean
  public Filter TracingFilter() {
    return new AWSXRayServletFilter("Scorekeep");
  }
```

```
...
```

Questo filtro invia i dati di tracciamento di tutte le richieste in entrata elaborate dall'applicazione, inclusi URL della richiesta, metodo, stato della risposta, ora di inizio e ora di fine.

L'applicazione effettua anche chiamate downstream a DynamoDB utilizzando. AWS SDK for Java Per strumentare queste chiamate, l'applicazione utilizza semplicemente i sottomoduli AWS relativi all'SDK come dipendenze e l'X-Ray SDK for Java monitora automaticamente tutti i client SDK. AWS

L'applicazione crea il codice sorgente su istanza con `Gradle Docker Image` e il file `Docker` per eseguire il `Scorekeep API Dockerfile` file JAR eseguibile che Gradle genera al suo interno. `ENTRYPOINT`

Example uso di Docker per creare tramite Gradle Docker Image

```
docker run --rm -v /PATH/TO/SCOREKEEP_REPO/home/gradle/project -w /home/gradle/project
 gradle:4.3 gradle build
```

Example PUNTO DI INGRESSO DEL FILE DOCKER

```
ENTRYPOINT [ "sh", "-c", "java -Dserver.port=5000 -jar scorekeep-api-1.0.0.jar" ]
```

Il file `build.gradle` attiva lo scaricamento dei sottomoduli SDK da Maven durante la compilazione dichiarandoli come dipendenze.

Example build.gradle -- dipendenze

```
...
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")
    }
```

```
}
```

I sottomoduli core, AWS SDK e AWS SDK Instrumentor sono tutto ciò che serve per strumentare automaticamente tutte le chiamate downstream effettuate con l'SDK. AWS

Per inoltrare i dati grezzi del segmento all'API X-Ray, è necessario il demone X-Ray per ascoltare il traffico sulla porta UDP 2000. A tal fine, l'applicazione fa funzionare il demone X-Ray in un contenitore distribuito insieme all'applicazione Scorekeep su ECS come contenitore secondario. Per ulteriori informazioni, consulta l'argomento relativo al demone X-Ray.

Example Definizione del contenitore X-Ray Daemon in una definizione di attività ECS

```
...
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
      ...
```

L'X-Ray SDK for Java fornisce una classe denominata `AWSXRay` che fornisce il registratore globale, `TracingHandler` che puoi usare per strumentare il tuo codice. Puoi configurare la registrazione globale per personalizzare il `AWSXRayServletFilter` che crea i segmenti relativi alle chiamate HTTP in entrata. L'esempio include un blocco statico nella classe `WebConfig` che permette di configurare la registrazione globale con i plug-in e le regole di campionamento.

Example WebConfigsrc/main/java/scorekeep/ .java - registratore

```
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());
    ...

  }
}
```

Questo esempio usa il compilatore per caricare le regole di campionamento da un file denominato `sampling-rules.json`. Le regole di campionamento determinano la frequenza con cui l'SDK registra i segmenti relativi alle richieste in entrata.

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
  }
}
```

Le regole di campionamento definiscono quattro regole di campionamento personalizzate e la regola di default. Per ogni richiesta in entrata, l'SDK valuta le regole nell'ordine in cui sono definite. L'SDK si applica la prima regola che corrisponde al metodo, al percorso e al nome del servizio della richiesta. Per Scorekeep, la prima regola cattura tutte le richieste POST (chiamate di creazione delle risorse) applicando un obiettivo fisso di una richiesta al secondo e una percentuale di 1,0, o il 100%, delle richieste dopo il raggiungimento dell'obiettivo fisso.

Le altre tre regole personalizzate applicano una percentuale del 5% delle chiamate, senza alcun obiettivo fisso per sessioni, giochi e letture dello stato (richieste GET). Questo riduce al minimo il numero di tracciamenti per le chiamate periodiche che il front-end invoca automaticamente ogni pochi secondi per garantire l'aggiornamento del contenuto. Per tutte le altre richieste, il file definisce un obiettivo di default di una richiesta al secondo e una percentuale del 10% delle chiamate successive.

L'applicazione di esempio mostra anche come utilizzare le funzionalità avanzate come l'analisi manuale del client SDK, la creazione di sottosegmenti aggiuntivi e le chiamate HTTP in uscita. Per ulteriori informazioni, consulta AWS X-Ray applicazione di esempio.

Opzionale: policy con privilegio minimo

I contenitori Scorekeep ECS accedono alle risorse utilizzando policy di accesso complete, come e. `AmazonSNSFullAccess` `AmazonDynamoDBFullAccess` L'utilizzo di politiche di accesso completo non è la migliore pratica per le applicazioni di produzione. L'esempio seguente aggiorna la policy IAM di DynamoDB per migliorare la sicurezza dell'applicazione. Per saperne di più sulle best practice di sicurezza nelle policy IAM, consulta Identity and access management for AWS X-Ray.

Example Definizione ECS del modello cf-resources.yaml TaskRole

```
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"
```

Per aggiornare la policy, è necessario innanzitutto identificare l'ARN delle risorse DynamoDB. Quindi si utilizza l'ARN in una policy IAM personalizzata. Infine, applichi tale policy al profilo della tua istanza.

Per identificare l'ARN della tua risorsa DynamoDB:

1. Aprire la console DynamoDB.
2. Scegli Tabelle dalla barra di navigazione a sinistra.
3. Scegli una delle opzioni `scorekeep-*` per visualizzare la pagina dei dettagli della tabella.

4. Nella scheda Panoramica, scegli Informazioni aggiuntive per espandere la sezione e visualizzare Amazon Resource Name (ARN). Copia questo valore.
5. Inserisci l'ARN nella seguente politica IAM, sostituendo `AWS_ACCOUNT_ID` i valori `AWS_REGION` and con la regione e l'ID dell'account specifici. Questa nuova politica consente solo le azioni specificate, anziché la `AmazonDynamoDBFullAccess` politica che consente qualsiasi azione.

   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-*"
           }
       ]
   }
   ```

   Le tabelle create dall'applicazione seguono una convenzione di denominazione coerente. È possibile utilizzare il `scorekeep-*` formato per indicare tutte le tabelle Scorekeep.

Cambia la tua politica IAM

1. Apri il ruolo dell'attività Scorekeep (ScorekeepRole) dalla console IAM.
2. Seleziona la casella di controllo accanto alla **`AmazonDynamoDBFullAccess`** politica e scegli Rimuovi per rimuovere questa politica.
3. Scegli Aggiungi autorizzazioni, quindi Allega politiche e infine Crea politica.
4. Scegli la scheda JSON e incolla la politica creata sopra.
5. Scegli Avanti: Tag nella parte inferiore della pagina.

6. Scegli Avanti: revisione nella parte inferiore della pagina.
7. Per Nome, assegna un nome alla politica.
8. Scegli Crea politica nella parte inferiore della pagina.
9. Allega la politica appena creata al `scorekeepRole` ruolo. Potrebbero essere necessari alcuni minuti prima che la policy allegata abbia effetto.

Se hai allegato la nuova policy al `scorekeepRole` ruolo, devi scollegarla prima di eliminare lo CloudFormation stack, poiché questa politica allegata bloccherà l'eliminazione dello stack. La policy può essere rimossa automaticamente eliminando la policy.

Rimuovi la tua policy IAM personalizzata

1. Apri la console IAM.
2. Scegli Policies dalla barra di navigazione a sinistra.
3. Cerca il nome della politica personalizzata che hai creato in precedenza in questa sezione e scegli il pulsante di opzione accanto al nome della politica per evidenziarlo.
4. Scegli il menu a discesa Azioni, quindi scegli Elimina.
5. Digita il nome della politica personalizzata, quindi scegli Elimina per confermare l'eliminazione. Ciò scollegherà automaticamente la politica dal `scorekeepRole` ruolo.

Eliminazione

Segui questi passaggi per eliminare le risorse dell'applicazione Scorekeep:

> **Note**
>
> Se hai creato e allegato politiche personalizzate utilizzando la sezione precedente di questo tutorial, devi rimuovere la politica `scorekeepRole` prima di eliminare lo stack. CloudFormation

AWS Management Console

Eliminare l'applicazione di esempio utilizzando il AWS Management Console

1. Apri la console CloudFormation

2. Scegli il pulsante di opzione accanto al nome `scorekeep` dello stack nell'elenco, quindi scegli Elimina.
3. Lo CloudFormation stack viene ora eliminato. Lo stato dello stack rimarrà valido `DELETE_IN_PROGRESS` per alcuni minuti fino all'eliminazione di tutte le risorse. Lo stato verrà aggiornato periodicamente oppure puoi aggiornare la pagina.

AWS CLI

Eliminare l'applicazione di esempio utilizzando il AWS CLI

1. Immettete il seguente AWS CLI comando per eliminare lo CloudFormation stack:

```
aws cloudformation delete-stack --stack-name scorekeep
```

2. Attendi che lo CloudFormation stack non esista più, l'operazione richiederà circa cinque minuti. Usa il seguente AWS CLI comando per controllare lo stato:

```
aws cloudformation describe-stacks --stack-name scorekeep --query
 "Stacks[0].StackStatus"
```

Passaggi successivi

Scopri di più su X-Ray nel prossimo capitolo,. AWS X-Ray concetti

Per strumentare la tua app, scopri di più sull'X-Ray SDK for Java o su uno degli altri SDK X-Ray:

- X-Ray SDK per Java — AWS X-Ray SDK per Java
- X-Ray SDK per Node.js — AWS X-Ray SDK per Node.js
- X-Ray SDK per.NET — AWS X-Ray SDK per.NET

Per eseguire il demone X-Ray localmente o su, vedere. AWSAWS X-Ray demone

Per contribuire all'applicazione di esempio su GitHub, vedere. eb-java-scorekeep

Strumentazione manuale dei client SDK AWS

L'X-Ray SDK for Java strumenta automaticamente tutti i client SDK quando includi il AWS sottomodulo SDK Instrumentor nelle AWS dipendenze di compilazione.

Puoi disabilitare l'analisi automatica del client rimuovendo il sottomodulo Instrumentor. In questo modo puoi analizzare manualmente alcuni client escludendone altri, oppure utilizzare diversi gestori del tracciamento su diversi client.

Per illustrare il supporto per la strumentazione di client AWS SDK specifici, l'applicazione passa un gestore di tracciamento come gestore di richieste nel modello utente, di gioco e di `AmazonDynamoDBClientBuilder` sessione. Questa modifica al codice indica all'SDK di strumentare tutte le chiamate a DynamoDB utilizzando tali client.

Example **`src/main/java/scorekeep/SessionModel.java`**— Strumentazione manuale del client SDK AWS

```
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);
```

Se rimuovi il sottomodulo AWS SDK Instrumentor dalle dipendenze del progetto, nella mappa di traccia vengono visualizzati solo i client SDK con strumentazione AWS manuale.

Creazione di sottosegmenti aggiuntivi

Nella classe del modello utente, l'applicazione crea manualmente dei sottosegmenti per raggruppare tutte le chiamate a valle effettuate all'interno della funzione `saveUser` e aggiunge i metadati.

Example **`src/main/java/scorekeep/UserModel.java`** - Sottosegmenti personalizzati

```
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();
    }
  }
```

Registrazione di annotazioni, metadati e ID utente

Nella classe game model, l'applicazione registra `Game` gli oggetti in un blocco di metadati ogni volta che salva un gioco in DynamoDB. Separatamente, l'applicazione memorizza gli ID delle partite nelle annotazioni per l'uso con espressioni filtro.

Example **`src/main/java/scorekeep/GameModel.java`**— Annotazioni e metadati

```
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();
    }
  }
```

Nel controllo delle mosse, l'applicazione memorizza gli ID utente con `setUser`. Gli ID utente vengono memorizzati in un campo separato su segmenti e sono indicizzati per l'uso nelle ricerche.

Example MoveControllersrc/main/java/scorekeep/ .java — ID utente

```
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);
  }
```

Analisi delle chiamate HTTP in uscita

La classe user factory mostra come l'applicazione utilizza la versione X-Ray SDK per Java per `HTTPClientBuilder` strumentare le chiamate HTTP in uscita.

Example **src/main/java/scorekeep/UserFactory.java**— Strumentazione HttpClient

```
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();
    }
  }
```

Se al momento utilizzi `org.apache.http.impl.client.HttpClientBuilder`, è possibile sostituire l'istruzione di importazione della classe con una che includa `com.amazonaws.xray.proxies.apache.http.HttpClientBuilder`.

Analizzare le chiamate a un database PostgresSQL

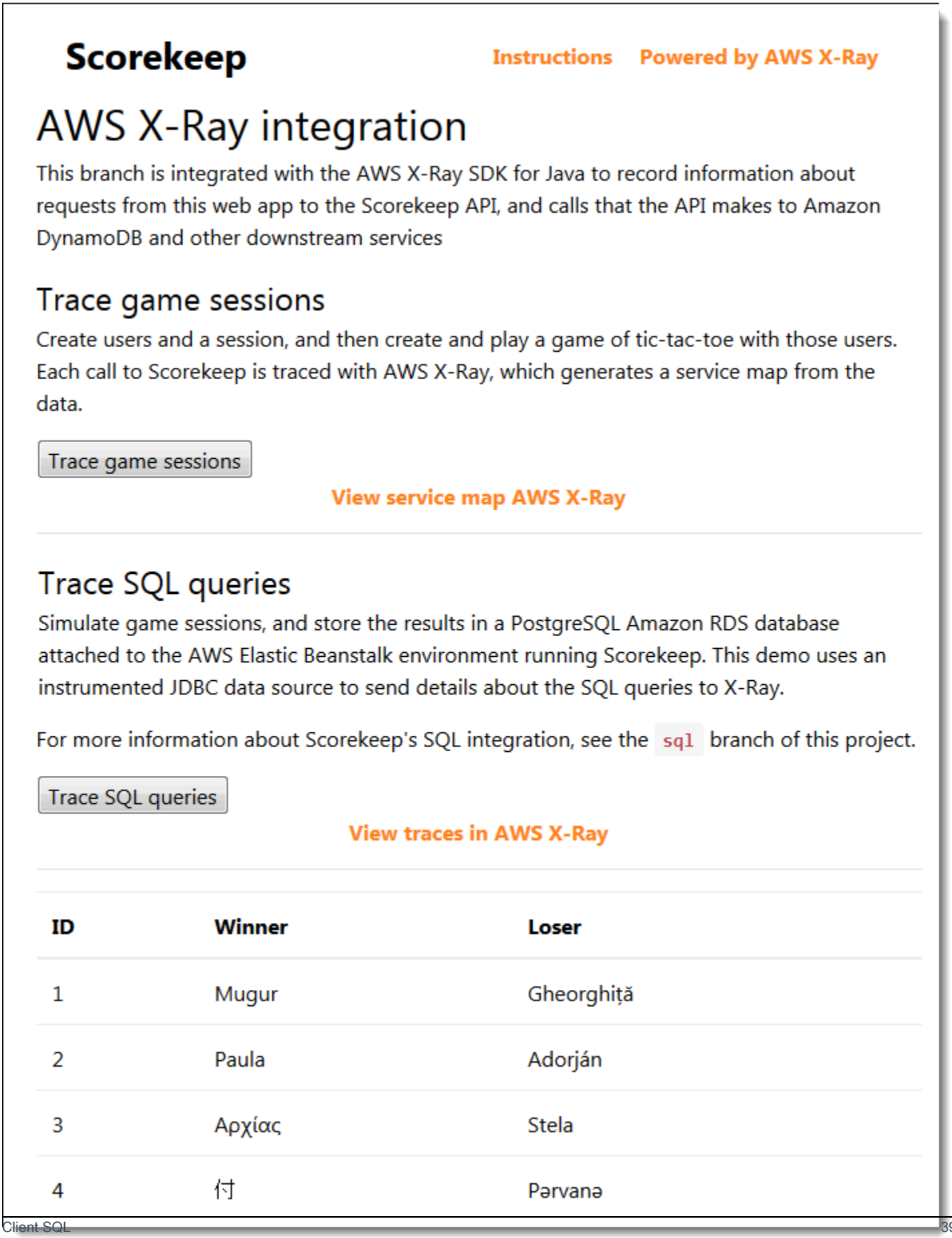
La`application-pgsql.properties`il file aggiunge il tracciamento X-Ray PostgreSQL per all'origine dati creata in`RdsWebConfig.java`.

Example **`application-pgsql.properties`**— Strumenti database PostgreSQL

```
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
```

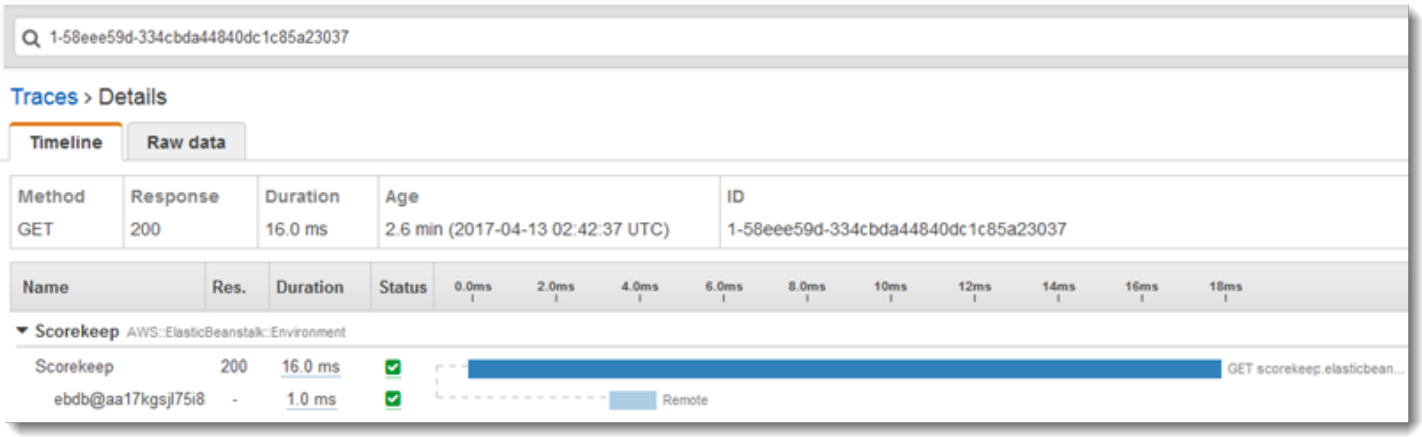
> **Note**
>
> Per ulteriori informazioni su come aggiungere un database PostgreSQL all'ambiente dell'applicazione, consulta Configurazione dei database con Elastic Beanstalk nella Guida per lo sviluppatore di AWS Elastic Beanstalk.

Pagina dimostrativa X-Ray nel`xray`branch include una demo che utilizza l'origine dati analizzata per generare tracciamenti che mostrano le informazioni sulle query SQL generate. Passare alla `./#/xray`percorso nell'applicazione in esecuzione o scegliAlimentato daAWS X-Raynella barra di navigazione per visualizzare la pagina dimostrativa.


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

Scegli Trace SQL queries (Traccia query SQL) per simulare le sessioni di gioco e memorizzare i risultati nel database collegato. Quindi, scegliVisualizzazione dei tracciamenti inAWSX-Rayper visualizzare un elenco filtrato di tracciamenti relativi all'API`/api/history`itinerario.

Scegli uno dei tracciamenti nell'elenco per visualizzare la sequenza temporale, inclusa la query SQL.



AWS Lambda Funzioni di strumentazione

Scorekeep utilizza due AWS Lambda funzioni. La prima è una funzione Node.js del ramo `lambda` che genera nomi casuali per i nuovi utenti. Quando un utente crea una sessione senza immettere un nome, l'applicazione chiama una funzione denominata `random-name` con il AWS SDK for Java. L'X-Ray SDK for Java registra le informazioni sulla chiamata a Lambda in un sottosegmento come qualsiasi altra chiamata effettuata con un client SDK strumentato. AWS

> **Note**
>
> L'esecuzione della funzione `random-name` Lambda richiede la creazione di risorse aggiuntive all'esterno dell'ambiente Elastic Beanstalk. Per ulteriori informazioni e istruzioni, consulta il file readme: AWS Lambda Integration.

La seconda funzione, `scorekeep-worker`, è una funzione Python che viene eseguita indipendentemente dall'API Scorekeep. Quando una partita termina, l'API scrive l'ID della sessione e l'ID della partita su una coda SQS. La funzione worker legge gli elementi dalla coda e chiama l'API Scorekeep per creare record completi di ogni sessione di gioco da archiviare in Amazon S3.

Scorekeep include AWS CloudFormation modelli e script per creare entrambe le funzioni. Poiché è necessario raggruppare X-Ray SDK con il codice della funzione, i modelli creano le funzioni

senza alcun codice. Quando distribuisci Scorekeep, un file di configurazione incluso nella cartella `.ebextensions` crea un pacchetto di codice sorgente che include l'SDK e aggiorna il codice della funzione e la configurazione con AWS Command Line Interface.

Funzioni

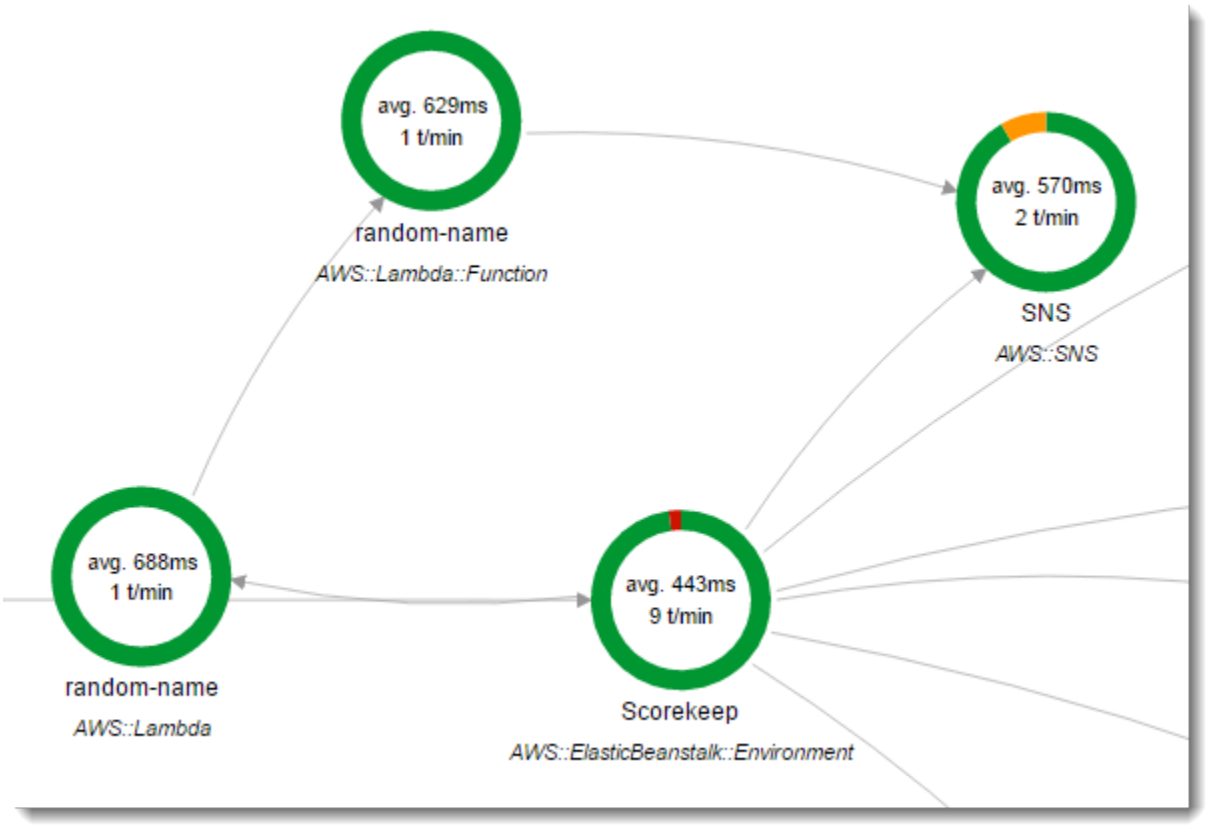
- Nome casuale
- Worker

Nome casuale

Scorekeep richiama la funzione di generazione del nome casuale quando un utente avvia una sessione di gioco senza aver eseguito l'accesso oppure senza aver specificato un nome utente. Quando Lambda elabora la chiamata `arandom-name`, legge l'intestazione di tracciamento, che contiene l'ID di traccia e la decisione di campionamento scritta da X-Ray SDK for Java.

Per ogni richiesta campionata, Lambda esegue il demone X-Ray e scrive due segmenti. Il primo segmento registra informazioni sulla chiamata a Lambda che richiama la funzione. Questo segmento contiene le stesse informazioni del sottosegmento registrato da Scorekeep, ma dal punto di vista Lambda. Il secondo segmento rappresenta l'attività svolta dalla funzione.

Lambda passa il segmento di funzione a X-Ray SDK tramite il contesto della funzione. Quando si strumenta una funzione Lambda, non si utilizza l'SDK per creare un segmento per le richieste in arrivo. Lambda fornisce il segmento e tu usi l'SDK per strumentare i client e scrivere sottosegmenti.



La funzione `random-name` è implementata in Node.js Utilizza l'SDK per Node.js per JavaScript inviare notifiche con Amazon SNS e l'SDK X-Ray per Node.js per strumentare il client SDK. AWS Per scrivere annotazioni, la funzione crea una sottosegmento personalizzato con `AWSXRay.captureFunc` e scrive le annotazioni nella funzione analizzata. In Lambda, non puoi scrivere annotazioni direttamente sul segmento di funzione, ma solo su un sottosegmento che crei.

Example **`function/index.js`** - Funzione Lambda nome casuale

```
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;
```

Questa funzione viene creata automaticamente quando distribuisci l'applicazione di esempio su Elastic Beanstalk. Il `xray` ramo include uno script per creare una funzione Lambda vuota. I file di configurazione nella `.ebextensions` cartella creano il pacchetto di funzioni con `npm install` durante la distribuzione, quindi aggiornano la funzione Lambda con la CLI AWS .

Worker

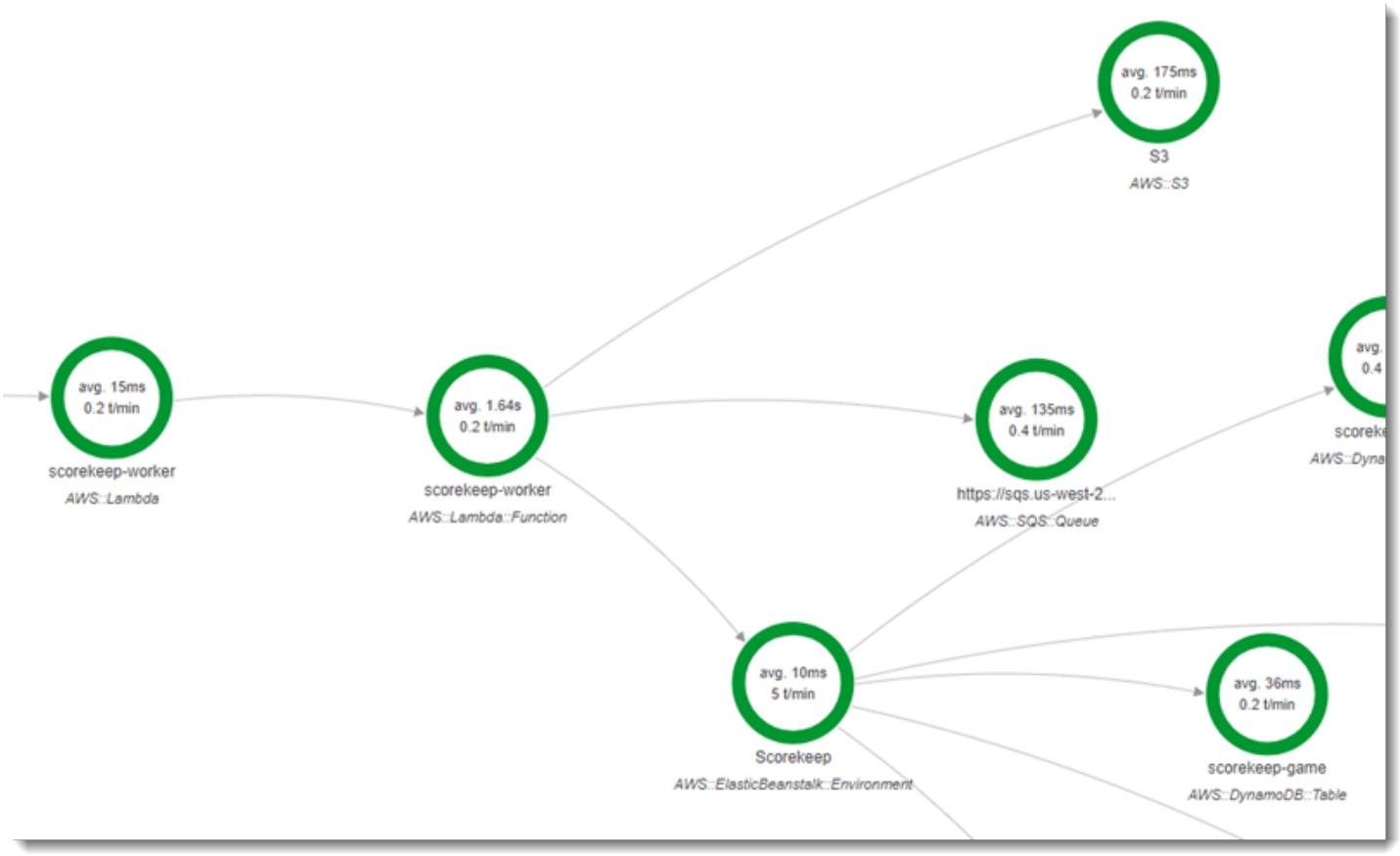
La funzione worker analizzata viene fornito nel suo ramo, `xray-worker`, poiché non può essere eseguita a meno che non si creino prima la funzione worker e le relative risorse. Per le istruzioni, consulta il readme del ramo.

La funzione viene attivata da un evento Amazon CloudWatch Events in bundle ogni 5 minuti. Quando viene eseguita, la funzione estrae un elemento da una coda Amazon SQS gestita da Scorekeep. Ogni messaggio contiene informazioni su una partita completata.

Il worker preleva il record della partita e i documenti dalle altre tabelle a cui il record della partita fa riferimento. Ad esempio, il record di gioco in DynamoDB include un elenco di mosse eseguite durante

il gioco. L'elenco non contiene le mosse, ma gli ID delle mosse che sono memorizzate in una tabella separata.

Anche le sessioni e gli stati sono memorizzati come riferimenti. In questo modo si evita che le voci nella tabella delle partite diventino troppo grandi ma sono necessarie chiamate aggiuntive per recuperare tutte le informazioni sulla partita. L'operatore dereferenzia tutte queste voci e crea un record completo del gioco come documento singolo in Amazon S3. Se desideri eseguire analisi sui dati, puoi eseguire query su di essi direttamente in Amazon S3 con Amazon Athena senza eseguire migrazioni di dati impegnative in lettura per estrarre i dati da DynamoDB.



La funzione worker dispone del tracciamento attivo abilitato nella sua configurazione in AWS Lambda. A differenza della funzione random name, l'operatore non riceve una richiesta da un'applicazione strumentata, quindi AWS Lambda non riceve un'intestazione di tracciamento. Con il tracciamento attivo, Lambda crea l'ID di traccia e prende decisioni di campionamento.

L'X-Ray SDK per Python si trova solo poche righe all'inizio della funzione che importa l'SDK ed esegue la sua `patch_all` funzione per applicare patch ai client HTTP che utilizza per chiamare Amazon SQS AWS SDK for Python (Boto) e Amazon S3. Quando il worker richiama l'API Scorekeep, l'SDK aggiunge l'intestazione di tracciamento alla richiesta per tracciare le chiamate tramite l'API.

Example **_lambda/scorekeep-worker/scorekeep-worker.py** -- Funzione worker Lambda

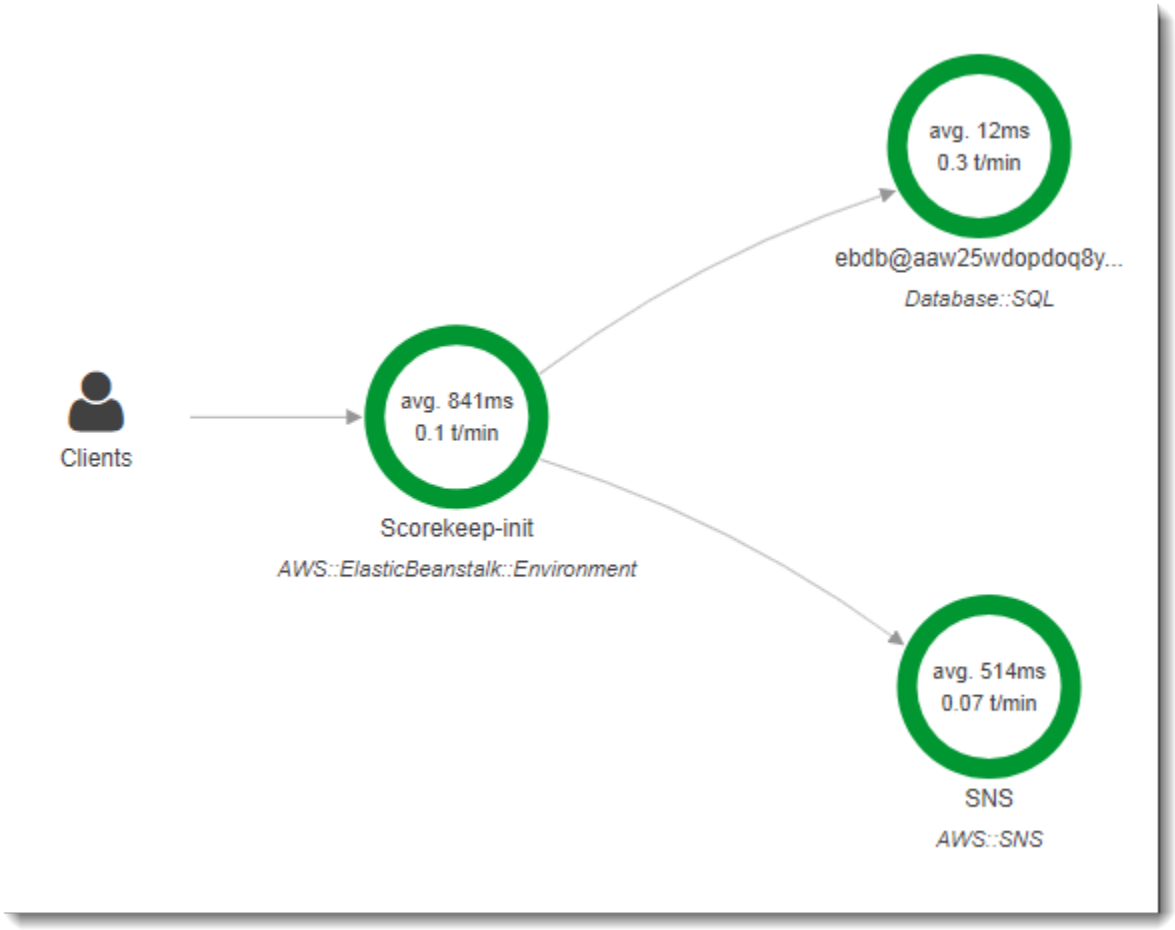
```
import os
import boto3
import json
import requests
import time
from aws_xray_sdk.core import xray_recorder
from aws_xray_sdk.core import patch_all

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
    )
   ...
```

Analisi del codice di avvio

L'SDK X-Ray per Java crea automaticamente dei segmenti per le richieste in entrata. Se una richiesta rientra nell'ambito, puoi utilizzare i client analizzati e registrare i sottosegmenti senza problemi. Se provi ad utilizzare un client analizzato nel codice di avvio, tuttavia, riceverai una SegmentNotFoundException.

Il codice di avvio viene eseguito al di fuori della richiesta standard/del flusso di risposta di un'applicazione web, quindi è necessario creare manualmente i segmenti per analizzarlo. Scorekeep

mostra l'analisi del codice di avvio nel suo file `WebConfig`. Durante l'avvio, Scorekeep effettua una chiamata a un database SQL e a.



Il valore di timeout predefinito per`WebConfig`class crea una sottoscrizione Amazon SNS per le notifiche. Per fornire un segmento su cui l'SDK X-Ray possa scrivere quando viene invocato il client Amazon SNS, Scorekeep chiama`beginSegment`e`endSegment`sul registratore globale.

Example **`src/main/java/scorekeep/WebConfig.java`**— StrumentataAWSSDK client nel codice di avvio

```
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();
```

Nello stato`RdsWebConfig`, che Scorekeep usa quando viene connesso un database Amazon RDS, la configurazione crea inoltre un segmento per il client SQL che viene utilizzato da Hibernate quando si applica lo schema del database durante l'avvio.

Example **`src/main/java/scorekeep/RdsWebConfig.java`**— Controllo client database SQL nel codice di avvio

```
@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` viene eseguito automaticamente e utilizza un client SQL. Poiché il client è analizzato, Scorekeep deve sostituire l'impostazione predefinita e fornire un segmento che l'SDK possa utilizzare quando il client viene richiamato.

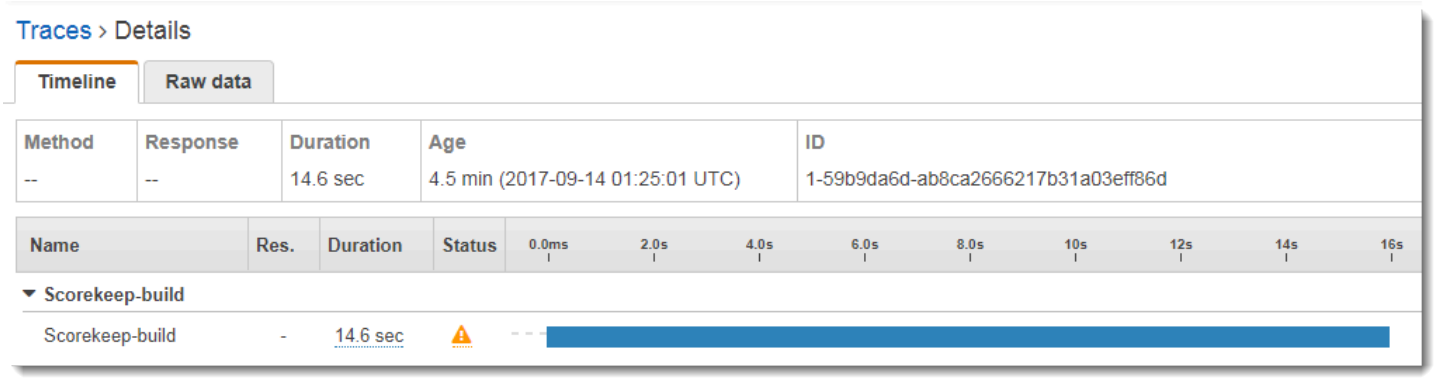
Analisi degli script

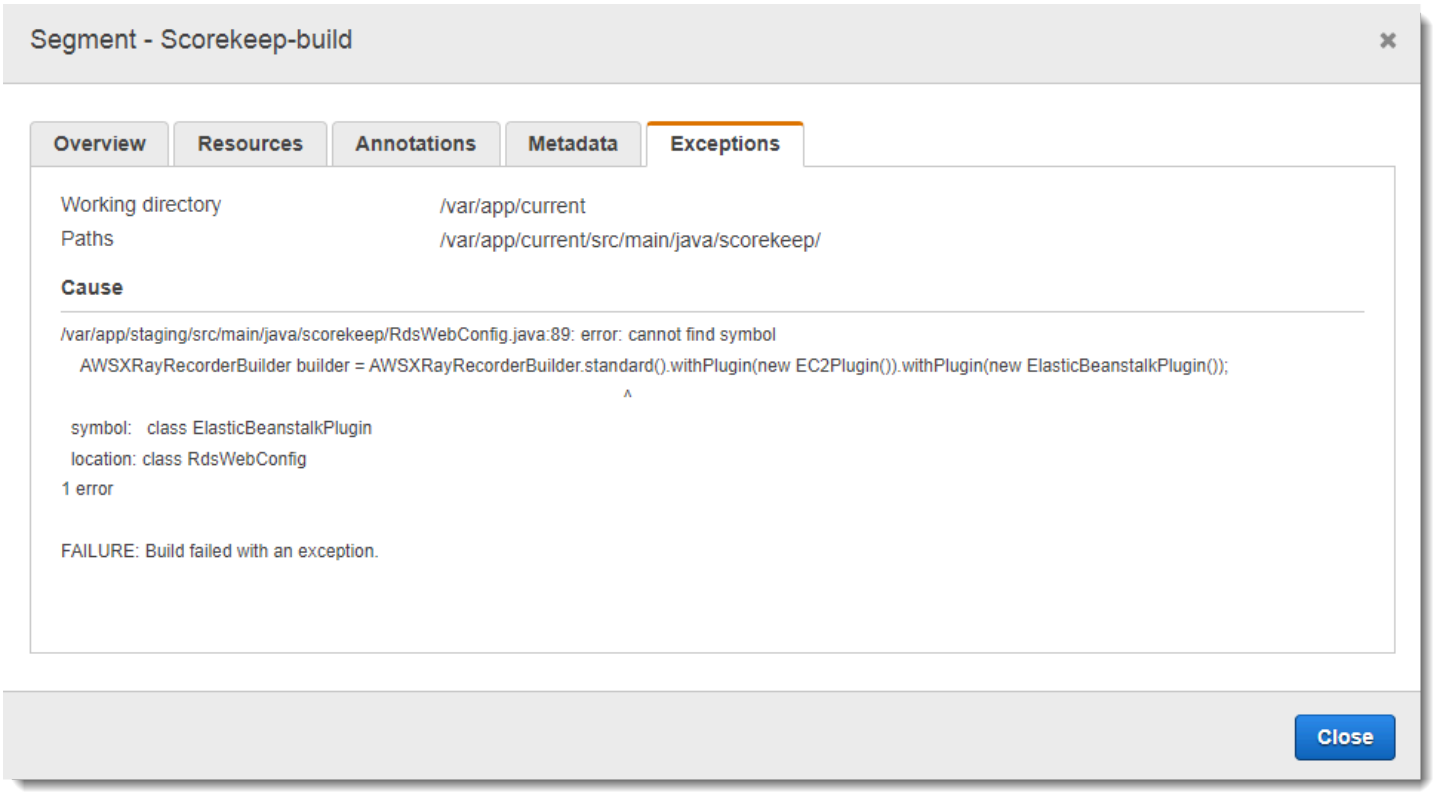
Puoi anche analizzare del codice che non fa parte della tua applicazione. Quando il demone X-Ray è in esecuzione, inoltrerà tutti i segmenti che riceve a X-Ray, anche se non sono generati dall'SDK X-Ray. Scorekeep usa i propri script per analizzare il processo di build che compila l'applicazione durante la distribuzione.

Example **bin/build.sh**— Script di compilazione con strumenti

```
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 e xray_success.py sono semplici script Python che costruiscono oggetti segmenti, li convertono in documenti in formato JSON e li inviano al daemon su UDP. Se la build di Gradle fallisce, puoi trovare il messaggio di errore facendo clic sul nodo scorekeep-build nella mappa di traccia della console X-Ray.





Analisi di un client di app Web

Nella [`xray-cognito`](https://github.com/aws-samples/aws-xray-scorekeep/tree/xray-cognito)filiale, Scorekeep utilizza Amazon Cognito per consentire agli utenti di creare un account e di accedere con esso per recuperare le informazioni utente da un pool di utenti Amazon Cognito. Quando un utente accede, Scorekeep utilizza un pool di identità Amazon Cognito per ottenere credenziali AWS temporanee da utilizzare con. AWS SDK for JavaScript

Il pool di identità viene configurato per garantire agli utenti che accedono l'accesso in scrittura ai dati di tracciamento verso AWS X-Ray. L'applicazione web utilizza queste credenziali per registrare l'ID utente dell'utente cha ha eseguito l'accesso, il percorso del browser e la vista del client delle chiamate all'API di Scorekeep.

La maggior parte del lavoro è effettuato in una classe denominata `xray`. Questa classe di servizio fornisce metodi per generare gli identificatori richiesti, creare segmenti in corso, finalizzare segmenti e inviare documenti di segmento all'API X-Ray.

Example **[`public/xray.js`](https://github.com/aws-samples/aws-xray-scorekeep/blob/xray-cognito/public/xray.js)**— Registrazione e caricamento di segmenti

```
...
  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);
      }
```

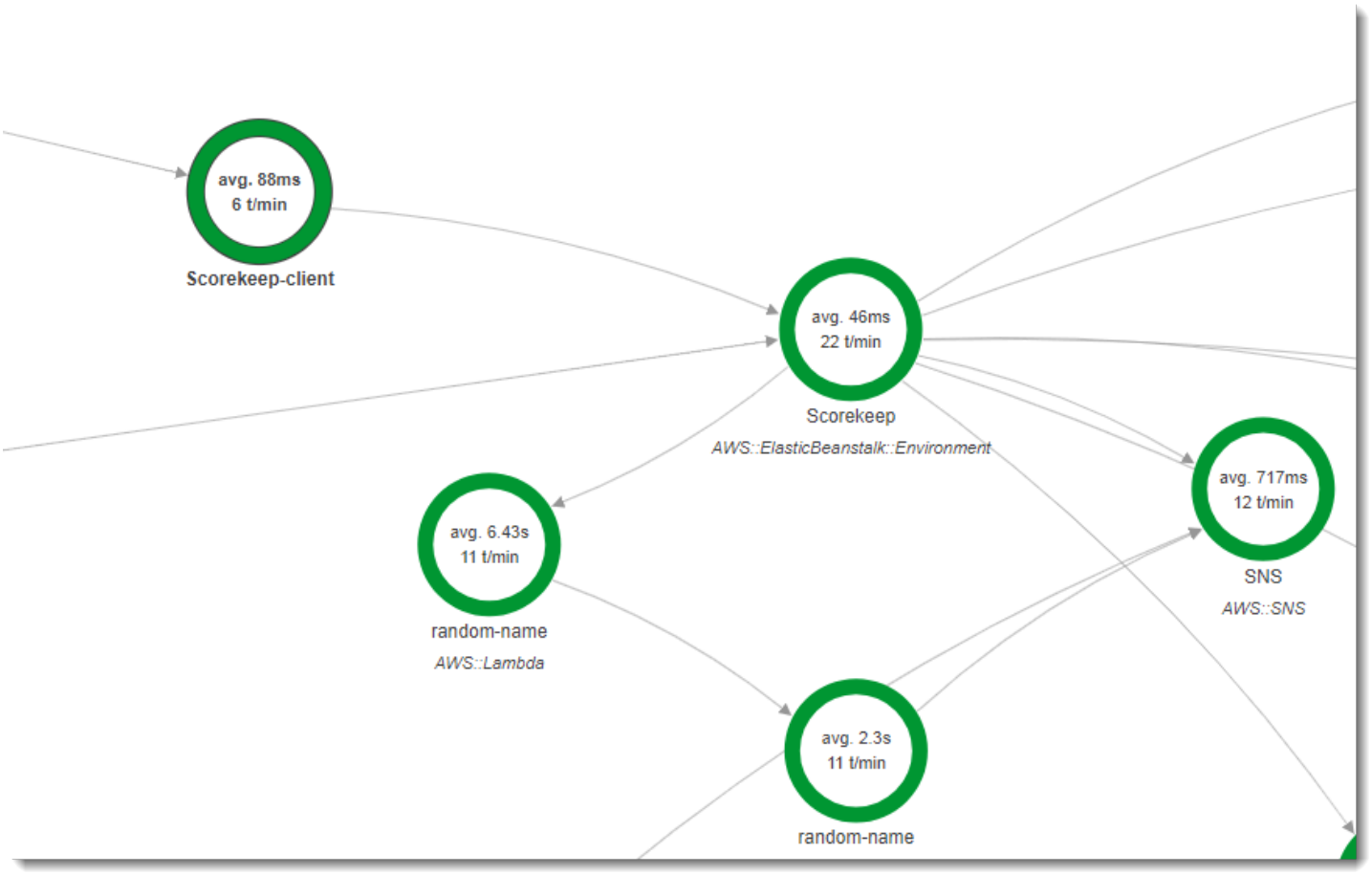
```
    })
  }
```

Questi metodi sono chiamati nell'intestazione e nelle funzioni `transformResponse` nei servizi della risorsa che l'app web Scorekeep utilizza per chiamare l'API. Per includere il segmento client nella stessa traccia del segmento generato dall'API, l'app Web deve includere l'ID di traccia e l'ID del segmento in un'intestazione di tracciamento (`X-Amzn-Trace-Id`) leggibile dall'SDK X-Ray. Quando l'applicazione Java con strumentazione riceve una richiesta con questa intestazione, l'X-Ray SDK for Java utilizza lo stesso ID di traccia e rende il segmento del client dell'app Web il principale del segmento.

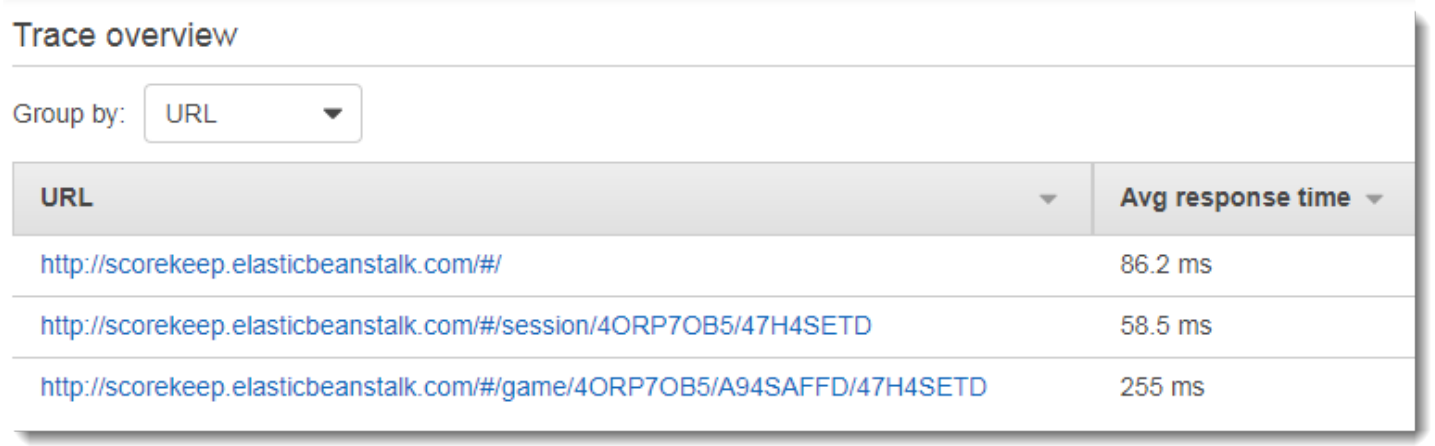
Example **`public/app/services.js`**— Registrazione di segmenti per chiamate di risorse angolari e scrittura di intestazioni di tracciamento

```
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);
      },
    },
...
```

La mappa di traccia risultante include un nodo per il client dell'app Web.



I tracciamenti che includono segmenti dall'app web mostrano l'URL che l'utente visualizza all'interno del browser (percorso che inizia con `/#/`). Senza l'analisi del client, ricevi solo l'URL dell'API della risorsa che l'app web chiama (percorsi che iniziano con `/api/`).



Utilizzo dei client analizzati nei thread worker

Scorekeep utilizza un thread worker per pubblicare una notifica su Amazon SNS quando un utente vince una partita. La pubblicazione della notifica richiede più tempo del resto delle operazioni di richiesta sommate e non ha effetto sul client o sull'utente. Pertanto, eseguire l'attività in modo asincrono è un buon modo per migliorare i tempi di risposta.

Tuttavia, l'SDK X-Ray per Java non sa quale segmento era attivo quando il thread è stato creato. Di conseguenza, quando utilizzi il client AWS SDK for Java analizzato all'interno del thread, questo genera un' eccezione `SegmentNotFoundException` arrestando in modo anomalo il thread.

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
...
```

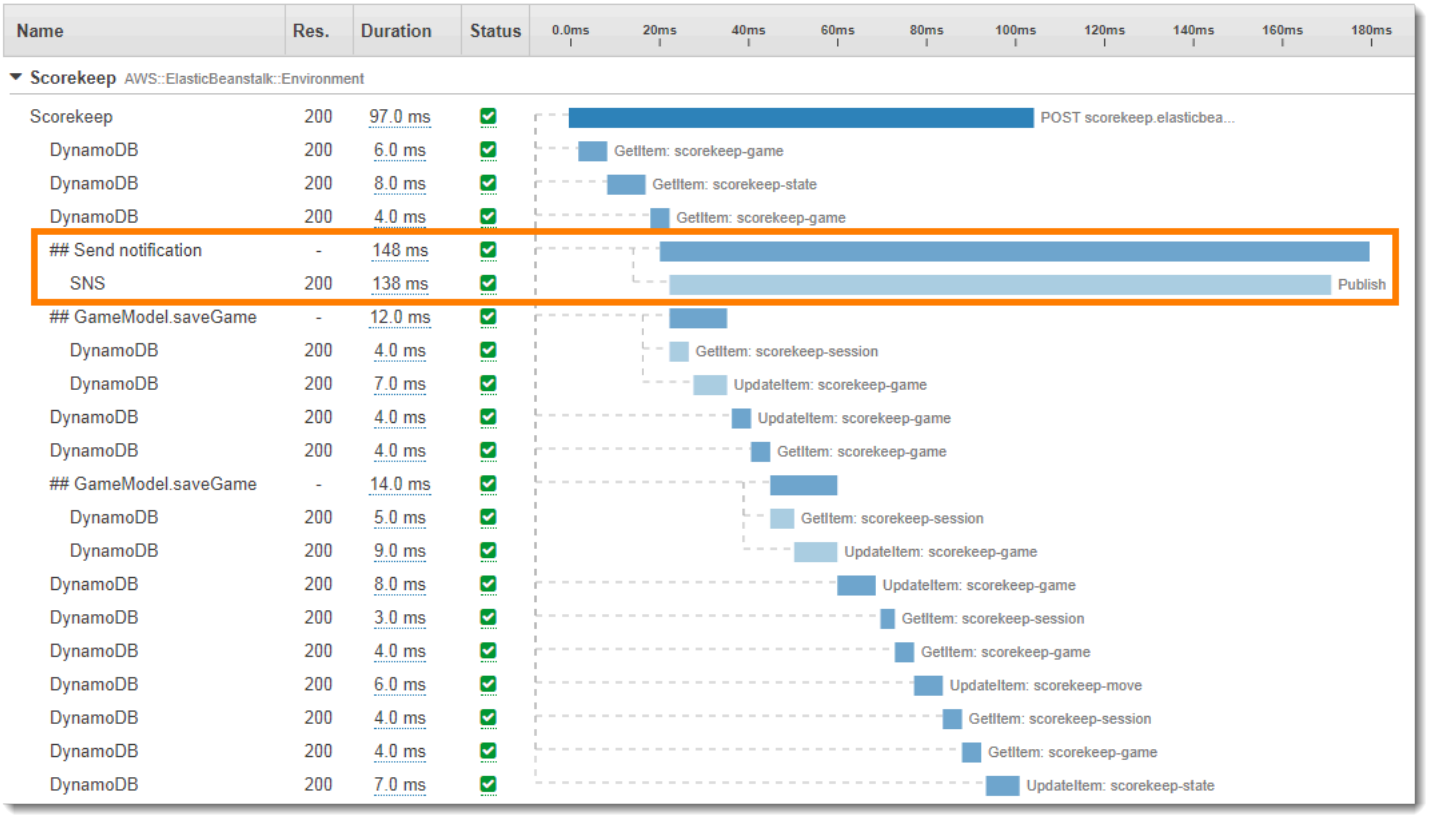
Per risolvere questo problema, l'applicazione utilizza`GetTraceEntity`per ottenere un riferimento al segmento nel thread principale e`Entity.run()`per eseguire in modo sicuro il codice del thread di lavoro con accesso al contesto del segmento.

Example **`src/main/java/scorekeep/MoveFactory.java`**— Passaggio del contesto di tracciamento a un thread worker

```
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();
        }
    }
```

Poiché la richiesta è ora risolta prima della chiamata ad Amazon SNS, l'applicazione crea un sottosegmento distinto per il thread. In questo modo si impedisce che l'SDK X-Ray chiuda il segmento prima che venga memorizzata la risposta da Amazon SNS. Se non è aperto nessun sottosegmento quando Scorekeep conclude la richiesta, la risposta da Amazon SNS potrebbe andare persa.



Per ulteriori informazioni sul multithreading, consulta [Passaggio del contesto del segmento tra thread in un'applicazione multithread].

Risoluzione dei problemi AWS X-Ray

Questo argomento elenca gli errori e i problemi più comuni che potresti riscontrare durante l'utilizzo dell'API X-Ray, della console o degli SDK. Se scopri un problema che non è elencato qui di seguito, puoi utilizzare il pulsante Feedback in questa pagina per segnalarlo.

Sections

- Mappa di tracciamento a raggi X e pagine con i dettagli della traccia
- X-Ray SDK per Java
- X-Ray SDK per Node.js
- Il demone X-Ray

Mappa di tracciamento a raggi X e pagine con i dettagli della traccia

Le seguenti sezioni possono essere utili in caso di problemi con l'utilizzo della mappa di tracciamento a raggi X e della pagina dei dettagli della traccia:

Non vedo tutti i miei registri CloudWatch

La modalità di configurazione dei log in modo che vengano visualizzati nella mappa di tracciamento a raggi X e nelle pagine dei dettagli della traccia dipende dal servizio.

- I log del gateway API vengono visualizzati se la registrazione è attivata in API Gateway.

Non tutti i nodi della mappa dei servizi supportano la visualizzazione dei log associati. Visualizza i log per i seguenti tipi di nodi:

- Contesto Lambda
- Funzione Lambda
- Fase API Gateway
- Cluster Amazon ECS
- Istanza Amazon ECS
- Servizio Amazon ECS
- Attività di Amazon ECS

- Cluster Amazon EKS
- Spazio dei nomi Amazon EKS
- Nodo Amazon EKS
- Pod Amazon EKS
- Servizio Amazon EKS

Non vedo tutti i miei allarmi sulla mappa di tracciamento a raggi X

La mappa di tracciamento a raggi X mostra solo l'icona di avviso per un nodo se gli allarmi associati a quel nodo sono nello stato ALARM.

La mappa di traccia associa gli allarmi ai nodi utilizzando la seguente logica:

- Se il nodo rappresenta un AWS servizio, tutti gli allarmi con lo spazio dei nomi associato a quel servizio sono associati al nodo. Ad esempio, un nodo di tipo C `AWS::Kinesis` è collegato a tutti gli allarmi basati sulle metriche del namespace. CloudWatch `AWS/Kinesis`
- Se il nodo rappresenta una AWS risorsa, gli allarmi su quella risorsa specifica sono collegati. Ad esempio, un nodo di tipo `AWS::DynamoDB::Table` con il nome «MyTable» è collegato a tutti gli allarmi basati su una metrica con lo spazio dei nomi `AWS/DynamoDB` e con la dimensione impostata su. `TableName` `MyTable`
- Se il nodo è di tipo sconosciuto, identificato da un bordo tratteggiato attorno al nome, a tale nodo non vengono associati allarmi.

Non vedo alcune AWS risorse sulla mappa di tracciamento

Non tutte le AWS risorse sono rappresentate da un nodo dedicato. Alcuni AWS servizi sono rappresentati da un singolo nodo per tutte le richieste al servizio. I seguenti tipi di risorse vengono visualizzati con un nodo per risorsa:

- `AWS::DynamoDB::Table`
- `AWS::Lambda::Function`

  Le funzioni Lambda sono rappresentate da due nodi, uno per il contenitore Lambda e uno per la funzione. Questo aiuta a identificare i problemi di avvio a freddo con le funzioni Lambda. I nodi del container Lambda sono associati ad allarmi e pannelli di controllo allo stesso modo dei nodi funzione Lambda.

- `AWS::ApiGateway::Stage`
- `AWS::SQS::Queue`
- `AWS::SNS::Topic`

Ci sono troppi nodi sulla mappa di traccia

Utilizza i gruppi X-Ray per suddividere la mappa in più mappe. Per ulteriori informazioni, consulta Utilizzo delle espressioni filtro con i gruppi.

X-Ray SDK per Java

Errore: eccezione nel thread «Thread-1" com.amazonaws.xray.exceptions. SegmentNotFoundException: Impossibile iniziare il sottosegmento denominato 'AmazonSNS': il segmento non può essere trovato.

Questo errore indica che l'SDK X-Ray ha tentato di registrare una chiamata in uscita verso AWS, ma non è riuscito a trovare un segmento aperto. Ciò può verificarsi nelle seguenti situazioni:

- Un filtro servlet non è configurato: l'SDK X-Ray crea segmenti per le richieste in entrata con un filtro denominato. `AWSXRayServletFilter` Configura un filtro servlet per analizzare le richieste in entrata.
- Stai utilizzando client strumentati al di fuori del codice servlet: se utilizzi un client con strumenti per effettuare chiamate nel codice di avvio o in altro codice che non viene eseguito in risposta a una richiesta in arrivo, devi creare un segmento manualmente. Per esempi, consulta Analisi del codice di avvio.
- Stai usando client strumentati nei thread di lavoro: quando crei un nuovo thread, il registratore X-Ray perde il riferimento al segmento aperto. È possibile utilizzare i `setTraceEntity`metodi `getTraceEntity`and per ottenere un riferimento al segmento o al sottosegmento corrente (`Entity`) e passarlo nuovamente al registratore all'interno del thread. Consulta Utilizzo dei client analizzati nei thread worker per un esempio.

X-Ray SDK per Node.js

Problema: CLS non funziona con Sequelize

Passa lo spazio dei nomi X-Ray SDK per Node.js a Sequelize con il metodo. `cls`

```
var AWSXRay = require('aws-xray-sdk');
const Sequelize = require('sequelize');
Sequelize.cls = AWSXRay.getNamespace();
const sequelize = new Sequelize(...);
```

Problema: CLS non funziona con Bluebird

Utilizza `cls-bluebird` per fare in modo che Bluebird funzioni con CLS.

```
var AWSXRay = require('aws-xray-sdk');
var Promise = require('bluebird');
var clsBluebird = require('cls-bluebird');
clsBluebird(AWSXRay.getNamespace());
```

Il demone X-Ray

Problema: il daemon sta utilizzando delle credenziali errate

Il demone utilizza l'SDK per caricare le AWS credenziali. Se si utilizzano più metodi di fornitura delle credenziali, viene utilizzato il metodo con la massima precedenza. Per ulteriori informazioni, consulta Esecuzione del daemon.

Sicurezza in AWS X-Ray

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS te e te. Il modello di responsabilità condivisa descrive questo come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gira Servizi AWS su Cloud AWS. AWS fornisce inoltre servizi che è possibile utilizzare in modo sicuro. L'efficacia della nostra sicurezza è regolarmente testata e verificata da revisori di terze parti come parte dei programmi di conformitàAWS. Per ulteriori informazioni sui programmi di conformità che si applicano a X-Ray, vedere Servizi AWS Scope by Compliance Program.
- Sicurezza nel cloud: la tua responsabilità è determinata da Servizio AWS ciò che utilizzi. L'utente è anche responsabile per altri fattori, tra cui la riservatezza dei dati, i requisiti dell'azienda e leggi e normative applicabili.

Questa documentazione ti aiuterà a capire come applicare il modello di responsabilità condivisa quando usi X-Ray. I seguenti argomenti mostrano come configurare X-Ray per soddisfare gli obiettivi di sicurezza e conformità. Imparerai anche a utilizzarne altri Servizi AWS che possono aiutarti a monitorare e proteggere le tue risorse X-Ray.

Argomenti

- Protezione dei dati in AWS X-Ray
- Gestione delle identità e degli accessi per AWS X-Ray
- Convalida della conformità per AWS X-Ray
- Resilienza in AWS X-Ray
- Sicurezza dell'infrastruttura in AWS X-Ray

Protezione dei dati in AWS X-Ray

AWS X-Ray esegue sempre la crittografia delle tracce e dei relativi dati memorizzati su disco. Quando è necessario controllare e disabilitare le chiavi di crittografia per la conformità o i requisiti

interni, è possibile configurare X-Ray per utilizzare una chiave AWS Key Management Service (AWS KMS) per crittografare i dati.

X-Ray fornisce un Chiave gestita da AWS nome`aws/xray`. Utilizza questa chiave se desideri controllare l'utilizzo delle chiavi in AWS CloudTrail e non hai bisogno di gestire la chiave stessa. Quando devi gestire l'accesso alla chiave o configurare la rotazione delle chiavi, puoi creare una chiave gestita dal cliente.

Quando si modificano le impostazioni di crittografia, X-Ray impiega del tempo a generare e propagare le chiavi dati. Anche se la nuova chiave è in corso di elaborazione, X-Ray può crittografare i dati con una combinazione di nuove e vecchie impostazioni. I dati esistenti non vengono crittografati nuovamente quando si modificano le impostazioni crittografiche.

> **Note**
>
> AWS KMSviene addebitato quando X-Ray utilizza una chiave KMS per crittografare o decrittografare i dati di traccia.
>
> - Crittografia predefinita: gratuita.
> - Chiave gestita da AWS— Paga per l'uso delle chiavi.
> - chiave gestita dal cliente: paga per l'archiviazione e l'utilizzo delle chiavi.
>
> Per ulteriori informazioni, consulta la sezione AWS Key Management ServicePrezzi.

> **Note**
>
> Le notifiche di X-Ray Insights inviano eventi ad AmazonEventBridge, che attualmente non supporta le chiavi gestite dai clienti. Per ulteriori informazioni, consulta la sezione relativa alla sezione relativa alla EventBridge.

È necessario disporre dell'accesso a livello utente a una chiave gestita dal cliente per configurare X-Ray per utilizzarla e quindi visualizzare le tracce crittografate. Per ulteriori informazioni, consulta Autorizzazioni utente per la crittografia.

CloudWatch console

Per configurare X-Ray per l'utilizzo di una chiave KMS per la crittografia tramite la console CloudWatch

1. Accedi AWS Management Console e apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Scegli Impostazioni nel riquadro di navigazione a sinistra.
3. Scegli Visualizza impostazioni in Crittografia nella sezione Tracce radiografiche.
4. Scegli Modifica nella sezione Configurazione della crittografia.
5. Scegli Usa una chiave KMS.
6. Scegliere una chiave dal menu a discesa:
   - aws/xray — Usa il. Chiave gestita da AWS
   - alias chiave: utilizza una chiave gestita dal cliente nel tuo account.
   - Inserisci manualmente l'ARN della chiave: utilizza una chiave gestita dal cliente in un diverso account. Inserire l'ARN (Amazon Resource Name) completo della chiave nel campo visualizzato.
7. Scegli Aggiorna crittografia.

X-Ray console

Per configurare X-Ray per utilizzare una chiave KMS per la crittografia utilizzando la console X-Ray

1. Apri la console X-Ray.
2. Scegliere Encryption (Crittografia).
3. Scegli Usa una chiave KMS.
4. Scegliere una chiave dal menu a discesa:
   - aws/xray — Usa il. Chiave gestita da AWS
   - alias chiave: utilizza una chiave gestita dal cliente nel tuo account.
   - Inserisci manualmente l'ARN della chiave: utilizza una chiave gestita dal cliente in un diverso account. Inserire l'ARN (Amazon Resource Name) completo della chiave nel campo visualizzato.

5. Seleziona Apply (Applica).

> **Note**
>
> X-Ray non supporta le chiavi KMS asimmetriche.

Se X-Ray non è in grado di accedere alla chiave di crittografia, interrompe la memorizzazione dei dati. Questo può accadere se l'utente perde l'accesso alla chiave KMS o se disabiliti una chiave attualmente in uso. Quando ciò accade, X-Ray mostra una notifica nella barra di navigazione.

Per configurare le impostazioni di crittografia con l'API X-Ray, consultaConfigurazione delle impostazioni di campionamento e di crittografia tramite l'API AWS X-Ray.

Gestione delle identità e degli accessi per AWS X-Ray

AWS Identity and Access Management (IAM) è uno strumento Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle risorse. AWS Gli amministratori IAM controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (dispone delle autorizzazioni) a utilizzare le risorse X-Ray. IAM è uno strumento Servizio AWS che puoi utilizzare senza costi aggiuntivi.

Argomenti

- Destinatari
- Autenticazione con identità
- Gestione dell'accesso con policy
- Come AWS X-Ray funziona con IAM
- AWS X-Ray esempi di politiche basate sull'identità
- Risoluzione dei problemi di identità e accesso in AWS X-Ray

Destinatari

La modalità di utilizzo di AWS Identity and Access Management (IAM) varia a seconda del lavoro svolto in X-Ray.

Utente del servizio: se si utilizza il servizio X-Ray per svolgere il proprio lavoro, l'amministratore fornisce le credenziali e le autorizzazioni necessarie. Man mano che utilizzi più funzionalità X-Ray per svolgere il tuo lavoro, potresti aver bisogno di autorizzazioni aggiuntive. La comprensione della gestione dell'accesso ti consente di richiedere le autorizzazioni corrette all'amministratore. Se non è possibile accedere a una funzionalità di X-Ray, vedere. Risoluzione dei problemi di identità e accesso in AWS X-Ray

Amministratore del servizio: se sei responsabile delle risorse X-Ray della tua azienda, probabilmente hai pieno accesso a X-Ray. Il tuo compito è determinare a quali funzionalità e risorse X-Ray devono accedere gli utenti del servizio. Devi inviare le richieste all'amministratore IAM per cambiare le autorizzazioni degli utenti del servizio. Esamina le informazioni contenute in questa pagina per comprendere i concetti di base relativi a IAM. Per saperne di più su come la tua azienda può utilizzare IAM con X-Ray, consulta. Come AWS X-Ray funziona con IAM

Amministratore IAM: se sei un amministratore IAM, potresti voler saperne di più su come scrivere policy per gestire l'accesso a X-Ray. Per visualizzare esempi di policy basate sull'identità X-Ray che puoi utilizzare in IAM, consulta. AWS X-Ray esempi di politiche basate sull'identità

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. Devi essere autenticato (aver effettuato l' Utente root dell'account AWS accesso AWS) come utente IAM o assumendo un ruolo IAM.

Puoi accedere AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAM Identity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali di Google o Facebook sono esempi di identità federate. Se accedi come identità federata, l'amministratore ha configurato in precedenza la federazione delle identità utilizzando i ruoli IAM. Quando accedi AWS utilizzando la federazione, assumi indirettamente un ruolo.

A seconda del tipo di utente, puoi accedere al AWS Management Console o al portale di AWS accesso. Per ulteriori informazioni sull'accesso a AWS, vedi Come accedere al tuo Account AWS nella Guida per l'Accedi ad AWS utente.

Se accedi a AWS livello di codice, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando (CLI) per firmare crittograficamente le tue richieste utilizzando le tue credenziali. Se non utilizzi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sull'utilizzo del

metodo consigliato per firmare autonomamente le richieste, consulta Signing AWS API request nella IAM User Guide.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, ti AWS consiglia di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza del tuo account. Per ulteriori informazioni, consulta Autenticazione a più fattori nella Guida per l'utente di AWS IAM Identity Center e Utilizzo dell'autenticazione a più fattori (MFA) in AWS nella Guida per l'utente di IAM.

Account AWS utente root

Quando si crea un account Account AWS, si inizia con un'identità di accesso che ha accesso completo a tutte Servizi AWS le risorse dell'account. Questa identità è denominata utente Account AWS root ed è accessibile effettuando l'accesso con l'indirizzo e-mail e la password utilizzati per creare l'account. Si consiglia vivamente di non utilizzare l'utente root per le attività quotidiane. Conserva le credenziali dell'utente root e utilizzarle per eseguire le operazioni che solo l'utente root può eseguire. Per un elenco completo delle attività che richiedono l'accesso come utente root, consulta la sezione Attività che richiedono le credenziali dell'utente root nella Guida per l'utente di IAM.

Utenti e gruppi IAM

Un utente IAM è un'identità interna Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Ove possibile, consigliamo di fare affidamento a credenziali temporanee invece di creare utenti IAM con credenziali a lungo termine come le password e le chiavi di accesso. Tuttavia, per casi d'uso specifici che richiedono credenziali a lungo termine con utenti IAM, si consiglia di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta la pagina Rotazione periodica delle chiavi di accesso per casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente di IAM.

Un gruppo IAM è un'identità che specifica un insieme di utenti IAM. Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, è possibile avere un gruppo denominato Amministratori IAM e concedere a tale gruppo le autorizzazioni per amministrare le risorse IAM.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali

temporanee. Per ulteriori informazioni, consulta Quando creare un utente IAM (invece di un ruolo) nella Guida per l'utente di IAM.

Ruoli IAM

Un ruolo IAM è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche. È simile a un utente IAM, ma non è associato a una persona specifica. Puoi assumere temporaneamente un ruolo IAM in AWS Management Console cambiando ruolo. Puoi assumere un ruolo chiamando un'operazione AWS CLI o AWS API o utilizzando un URL personalizzato. Per ulteriori informazioni sui metodi per l'utilizzo dei ruoli, consulta Utilizzo di ruoli IAM nella Guida per l'utente di IAM.

I ruoli IAM con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per ulteriori informazioni sulla federazione dei ruoli, consulta Creazione di un ruolo per un provider di identità di terza parte nella Guida per l'utente di IAM. Se utilizzi IAM Identity Center, configura un set di autorizzazioni. IAM Identity Center mette in correlazione il set di autorizzazioni con un ruolo in IAM per controllare a cosa possono accedere le identità dopo l'autenticazione. Per ulteriori informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .
- Autorizzazioni utente IAM temporanee: un utente IAM o un ruolo può assumere un ruolo IAM per ottenere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso multi-account: è possibile utilizzare un ruolo IAM per permettere a un utente (un principale affidabile) con un account diverso di accedere alle risorse nell'account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per informazioni sulle differenze tra ruoli e policy basate su risorse per l'accesso multi-account, consulta Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente di IAM.
- Accesso a più servizi: alcuni Servizi AWS utilizzano le funzionalità di altri Servizi AWS. Ad esempio, quando effettui una chiamata in un servizio, è comune che tale servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.

  - Sessioni di accesso diretto (FAS): quando utilizzi un utente o un ruolo IAM per eseguire azioni AWS, sei considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra azione in un servizio diverso. FAS utilizza le autorizzazioni del principale che chiama an Servizio AWS, combinate con la richiesta Servizio AWS per effettuare richieste ai servizi downstream. Le richieste FAS vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le operazioni. Per i dettagli delle policy relative alle richieste FAS, consulta la pagina Forward access sessions.
  - Ruolo di servizio: un ruolo di servizio è un ruolo IAM assunto da un servizio per eseguire operazioni per conto dell'utente. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione Creazione di un ruolo per delegare le autorizzazioni a un Servizio AWS nella Guida per l'utente di IAM.
  - Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non modificarle.
- Applicazioni in esecuzione su Amazon EC2: puoi utilizzare un ruolo IAM per gestire le credenziali temporanee per le applicazioni in esecuzione su un'istanza EC2 e che AWS CLI effettuano richieste API. AWS Ciò è preferibile all'archiviazione delle chiavi di accesso nell'istanza EC2. Per assegnare un AWS ruolo a un'istanza EC2 e renderlo disponibile per tutte le sue applicazioni, crei un profilo di istanza collegato all'istanza. Un profilo dell'istanza contiene il ruolo e consente ai programmi in esecuzione sull'istanza EC2 di ottenere le credenziali temporanee. Per ulteriori informazioni, consulta Utilizzo di un ruolo IAM per concedere autorizzazioni ad applicazioni in esecuzione su istanze di Amazon EC2 nella Guida per l'utente di IAM.

Per informazioni sull'utilizzo dei ruoli IAM, consulta Quando creare un ruolo IAM (invece di un utente) nella Guida per l'utente di IAM.

Gestione dell'accesso con policy

Puoi controllare l'accesso AWS creando policy e collegandole a AWS identità o risorse. Una policy è un oggetto AWS che, se associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La

maggior parte delle politiche viene archiviata AWS come documenti JSON. Per ulteriori informazioni sulla struttura e sui contenuti dei documenti delle policy JSON, consulta Panoramica delle policy JSON nella Guida per l'utente di IAM.

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti l'autorizzazione a eseguire azioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM. Successivamente l'amministratore può aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Le policy IAM definiscono le autorizzazioni relative a un'operazione, a prescindere dal metodo utilizzato per eseguirla. Ad esempio, supponiamo di disporre di una policy che consente l'azione `iam:GetRole`. Un utente con tale policy può ottenere informazioni sul ruolo dall' AWS Management Console AWS CLI, dall'o dall' AWS API.

Policy basate su identità

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruolo IAM). Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta Creazione di policy IAM nella Guida per l'utente di IAM.

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono incorporate direttamente in un singolo utente, gruppo o ruolo. Le politiche gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli nel tuo Account AWS. Le politiche gestite includono politiche AWS gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una policy gestita o una policy inline, consulta Scelta fra policy gestite e policy inline nella Guida per l'utente di IAM.

Policy basate su risorse

Le policy basate su risorse sono documenti di policy JSON che è possibile allegare a una risorsa. Gli esempi più comuni di policy basate su risorse sono le policy di attendibilità dei ruoli IAM e le policy dei bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarle per controllare l'accesso a una risorsa specifica. Quando è allegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non puoi utilizzare le policy AWS gestite di IAM in una policy basata sulle risorse.

Liste di controllo degli accessi (ACL)

Le liste di controllo degli accessi (ACL) controllano quali principali (membri, utenti o ruoli dell'account) hanno le autorizzazioni per accedere a una risorsa. Le ACL sono simili alle policy basate su risorse, sebbene non utilizzino il formato del documento di policy JSON.

Amazon S3 e Amazon VPC sono esempi di servizi che supportano gli ACL. AWS WAF Per maggiori informazioni sulle ACL, consulta Panoramica delle liste di controllo degli accessi (ACL) nella Guida per gli sviluppatori di Amazon Simple Storage Service.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite delle autorizzazioni è una funzione avanzata nella quale si imposta il numero massimo di autorizzazioni che una policy basata su identità può concedere a un'entità IAM (utente o ruolo IAM). È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal` sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità IAM nella Guida per l'utente di IAM.
- Politiche di controllo dei servizi (SCP): le SCP sono politiche JSON che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in. AWS Organizations AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più Account AWS di proprietà dell'azienda. Se abiliti tutte le funzionalità in un'organizzazione, puoi applicare le policy di controllo dei servizi (SCP) a uno o tutti i tuoi account. L'SCP limita le autorizzazioni per le entità negli account dei membri, inclusa ciascuna. Utente root dell'account AWS Per ulteriori informazioni su organizzazioni e policy SCP, consulta la pagina sulle Policy di controllo dei servizi nella Guida per l'utente di AWS Organizations .
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire

da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta Policy di sessione nella Guida per l'utente di IAM.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per scoprire come si AWS determina se consentire una richiesta quando sono coinvolti più tipi di policy, consulta Logica di valutazione delle policy nella IAM User Guide.

Come AWS X-Ray funziona con IAM

Prima di utilizzare IAM per gestire l'accesso a X-Ray, è necessario comprendere quali funzionalità IAM sono disponibili per l'uso con X-Ray. Per avere una visione di alto livello del funzionamento di X-Ray e di Servizi AWS altro tipo con IAM, Servizi AWS consulta That Work with IAM nella IAM User Guide.

Puoi usare AWS Identity and Access Management (IAM) per concedere le autorizzazioni X-Ray agli utenti e alle risorse di calcolo del tuo account. IAM controlla l'accesso al servizio X-Ray a livello di API per applicare le autorizzazioni in modo uniforme, indipendentemente dal client (console, AWS SDK) impiegato dagli utenti. AWS CLI

Per utilizzare la console X-Ray per visualizzare mappe e segmenti di tracciamento, sono necessarie solo le autorizzazioni di lettura. Per abilitare l'accesso alla console, aggiungi la policy `AWSXrayReadOnlyAccess` gestita al tuo utente IAM.

Per lo sviluppo e il test locali, crea un ruolo IAM con autorizzazioni di lettura e scrittura. Assumi il ruolo e archivia le credenziali temporanee per il ruolo. È possibile utilizzare queste credenziali con il demone X-Ray, con e con AWS CLI l'SDK. AWS Per ulteriori informazioni, vedere Utilizzo di credenziali di sicurezza temporanee con. AWS CLI

Per distribuire la tua app strumentata su AWS, crea un ruolo IAM con autorizzazioni di scrittura e assegnalo alle risorse che eseguono l'applicazione. `AWSXRayDaemonWriteAccess`include l'autorizzazione a caricare tracce e alcune autorizzazioni di lettura per supportare l'uso delle regole di campionamento.

Le policy in lettura e in scrittura non includono l'autorizzazione per configurare le impostazioni delle chiavi di cifratura e le regole di campionatura. Utilizza `AWSXrayFullAccess` per accedere a queste impostazioni o aggiungi le API di configurazione in una policy personalizzata. Per la crittografia e la

decodifica con una chiave gestita personalizzata che crei, hai anche bisogno dell'autorizzazione per l'utilizzo della chiave.

Argomenti

- Policy basate sull'identità X-Ray
- Policy basate sulle risorse X-Ray
- Autorizzazione basata su tag X-Ray
- Esecuzione locale dell'applicazione
- Esecuzione dell'applicazione in AWS
- Autorizzazioni utente per la crittografia

Policy basate sull'identità X-Ray

Con le policy basate su identità di IAM, è possibile specificare quali operazioni e risorse sono consentite o respinte, nonché le condizioni in base alle quali le operazioni sono consentite o respinte. X-Ray supporta azioni, risorse e chiavi di condizione specifiche. Per informazioni su tutti gli elementi utilizzati in una policy JSON, consulta Documentazione di riferimento degli elementi delle policy JSON IAM nella Guida per l'utente IAM.

Azioni

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire azioni su quali risorse, e in quali condizioni.

L'elemento `Action` di una policy JSON descrive le azioni che è possibile utilizzare per consentire o negare l'accesso a una policy. Le azioni politiche in genere hanno lo stesso nome dell'operazione AWS API associata. Ci sono alcune eccezioni, ad esempio le azioni di sola autorizzazione che non hanno un'operazione API corrispondente. Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includi le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

Le azioni politiche in X-Ray utilizzano il seguente prefisso prima dell'azione:. `xray:` Ad esempio, per concedere a qualcuno l'autorizzazione a recuperare i dettagli delle risorse di gruppo con l'operazione dell'API `GetGroup` X-Ray, includi `xray:GetGroup` l'azione nella sua politica. Le istruzioni della policy devono includere un elemento `Action` o `NotAction`. X-Ray definisce il proprio set di azioni che descrivono le attività che è possibile eseguire con questo servizio.

Per specificare più operazioni in una sola istruzione, separa ciascuna di esse con una virgola come mostrato di seguito:

```
"Action": [
      "xray:action1",
      "xray:action2"
```

È possibile specificare più operazioni tramite caratteri jolly (*). Ad esempio, per specificare tutte le operazioni che iniziano con la parola `Get`, includi la seguente operazione:

```
"Action": "xray:Get*"
```

Per visualizzare un elenco di azioni X-Ray, consulta Actions Defined by AWS X-Ray nella IAM User Guide.

Risorse

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento JSON `Resource` della policy specifica l'oggetto o gli oggetti ai quali si applica l'azione. Le istruzioni devono includere un elemento `Resource` o un elemento `NotResource`. Come best practice, specifica una risorsa utilizzando il suo nome della risorsa Amazon (ARN). Puoi eseguire questa operazione per azioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

Puoi controllare l'accesso alle risorse utilizzando una policy IAM. Per le operazioni che supportano le autorizzazioni a livello di risorsa, si utilizza un Amazon Resource Name (ARN) per identificare la risorsa a cui si applica la policy.

Tutte le azioni X-Ray possono essere utilizzate in una policy IAM per concedere o negare agli utenti il permesso di utilizzare quell'azione. Tuttavia, non tutte le azioni X-Ray supportano le autorizzazioni a livello di risorsa, che consentono di specificare le risorse su cui è possibile eseguire un'azione.

Per le operazioni che non supportano le autorizzazioni a livello di risorsa, è necessario utilizzare "*" come risorsa.

Le seguenti azioni X-Ray supportano le autorizzazioni a livello di risorsa:

- `CreateGroup`
- `GetGroup`
- `UpdateGroup`
- `DeleteGroup`
- `CreateSamplingRule`
- `UpdateSamplingRule`
- `DeleteSamplingRule`

Di seguito è riportato un esempio di policy di autorizzazione basata su identità per un'operazione `CreateGroup`. L'esempio mostra l'uso di un ARN relativo al nome di gruppo `local-users` con l'ID univoco come carattere jolly. L'ID univoco viene generato quando il gruppo viene creato, quindi non può essere previsto nella policy in anticipo. Quando utilizzi `GetGroup`, `UpdateGroup` oppure `DeleteGroup`, è possibile definirlo come un carattere jolly o l'ARN esatto, incluso l'ID.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:CreateGroup"
            ],
            "Resource": [
                "arn:aws:xray:eu-west-1:123456789012:group/local-users/*"
            ]
        }
    ]
}
```

Di seguito è riportato un esempio di policy di autorizzazione basata su identità per un'operazione `CreateSamplingRule`.

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:CreateSamplingRule"
            ],
            "Resource": [
                "arn:aws:xray:eu-west-1:123456789012:sampling-rule/base-scorekeep"
            ]
        }
    ]
}
```

> **Note**
>
> L'ARN di una regola di campionamento è definito in base al nome. A differenza degli ARN dei gruppi, le regole di campionamento non hanno un ID generato in modo univoco.

Per visualizzare un elenco dei tipi di risorse X-Ray e dei relativi ARN, consulta Resources Defined by AWS X-Ray nella IAM User Guide. Per informazioni sulle operazioni con cui è possibile specificare l'ARN di ogni risorsa, consulta Operazioni definite da AWS X-Ray.

Chiavi di condizione

X-Ray non fornisce chiavi di condizione specifiche del servizio, ma supporta l'utilizzo di alcune chiavi di condizione globali. Per visualizzare tutte le chiavi di condizione AWS globali, consulta AWS Global Condition Context Keys nella IAM User Guide.

Esempi

Per visualizzare esempi di policy basate sull'identità a raggi X, vedere. AWS X-Ray esempi di politiche basate sull'identità

Policy basate sulle risorse X-Ray

X-Ray supporta politiche basate sulle risorse per l' Servizio AWS integrazione attuale e futura, come il tracciamento attivo di Amazon SNS. Le policy basate sulle risorse X-Ray possono essere aggiornate da altri AWS Management Console utenti o tramite l'SDK o la CLI. AWS Ad esempio, la console Amazon SNS tenta di configurare automaticamente una policy basata sulle risorse per l'invio di tracce

a X-Ray. Il seguente documento sulla policy fornisce un esempio di configurazione manuale delle policy basate sulle risorse X-Ray.

Example Esempio di policy basata su risorse X-Ray per il tracciamento attivo di Amazon SNS

Questo documento di policy di esempio specifica le autorizzazioni necessarie ad Amazon SNS per inviare dati di traccia a X-Ray:

```
{
    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"
          }
        }
      }
    ]
  }
```

Utilizza la CLI per creare una policy basata sulle risorse che conceda ad Amazon SNS le autorizzazioni per inviare dati di traccia a X-Ray:

```
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" } } } ] }'
```

Per utilizzare questi esempi, sostituisci *partition*, *regionaccount-id*, e *topic-name*con la AWS partizione, la regione, l'ID dell'account e il nome dell'argomento Amazon SNS specifici. Per autorizzare tutti gli argomenti di Amazon SNS a inviare dati di traccia a X-Ray, sostituisci il nome dell'argomento con. *

Autorizzazione basata su tag X-Ray

È possibile allegare tag ai gruppi di raggi X o alle regole di campionamento oppure passare i tag in una richiesta a X-Ray. Per controllare l'accesso basato su tag, fornisci informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `xray:ResourceTag/`*key-name*, `aws:RequestTag/`*key-name* o `aws:TagKeys`. Per ulteriori informazioni sull'etichettatura delle risorse X-Ray, vedere. Tagging delle regole e dei gruppi di campionamento di X-Ray

Per visualizzare una policy basata sulle identità di esempio per limitare l'accesso a una risorsa basata su tag su tale risorsa, consulta Gestione dell'accesso ai gruppi X-Ray e alle regole di campionamento basate sui tag.

Esecuzione locale dell'applicazione

L'applicazione strumentata invia i dati di traccia al demone X-Ray. Il daemon memorizza i documenti segmentati e li carica sul servizio X-Ray in batch. Il demone necessita di autorizzazioni di scrittura per caricare dati di traccia e telemetria sul servizio X-Ray.

Quando esegui il daemon localmente, crea un ruolo IAM, assumi il ruolo e archivia le credenziali temporanee nelle variabili di ambiente o in un file denominato all'interno di una cartella denominata nella cartella utente. `credentials .aws` Per ulteriori informazioni, consulta Utilizzo di credenziali di sicurezza temporanee con. 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}
```

Se hai già configurato le credenziali da utilizzare con l' AWS SDK oppure AWS CLI il demone può utilizzarle. Se sono disponibili più profili, il daemon usa il profilo di default.

Esecuzione dell'applicazione in AWS

Quando esegui l'applicazione AWS, usa un ruolo per concedere l'autorizzazione all'istanza Amazon EC2 o alla funzione Lambda che esegue il demone.

- Amazon Elastic Compute Cloud (Amazon EC2): crea un ruolo IAM e collegalo all'istanza EC2 come profilo dell'istanza.
- Amazon Elastic Container Service (Amazon ECS): crea un ruolo IAM e collegalo alle istanze del contenitore come ruolo IAM dell'istanza di contenitore.
- AWS Elastic Beanstalk (Elastic Beanstalk) — Elastic Beanstalk include le autorizzazioni X-Ray nel profilo di istanza predefinito. Puoi utilizzare l'impostazione predefinita del profilo dell'istanza o aggiungere autorizzazioni di scrittura a un profilo di istanze personalizzato.
- AWS Lambda (Lambda) — Aggiungi i permessi di scrittura al ruolo di esecuzione della funzione.

Per creare un ruolo da utilizzare con X-Ray

1. Apri la console IAM.
2. Scegli Ruoli.
3. Scegli Crea nuovo ruolo.
4. Per Role Name (Nome ruolo) digitare **`xray-application`**. Selezionare Next Step (Fase successiva).
5. Come Role Type (Tipo ruolo), scegliere Amazon EC2.
6. Allega la seguente policy gestita per consentire all'applicazione di accedere a Servizi AWS:

   - AWSXRayDaemonWriteAccess— Fornisce al demone X-Ray il permesso di caricare dati di traccia.

   Se l'applicazione utilizza l' AWS SDK per accedere ad altri servizi, aggiungete politiche che consentano l'accesso a tali servizi.
7. Selezionare Next Step (Fase successiva).
8. Selezionare Crea ruolo.

Autorizzazioni utente per la crittografia

X-Ray crittografa tutti i dati di traccia e, per impostazione predefinita, è possibile configurarlo per utilizzare una chiave gestita dall'utente. Se scegli una chiave gestita AWS Key Management Service dal cliente, devi assicurarti che la politica di accesso della chiave consenta di concedere l'autorizzazione a X-Ray per utilizzarla per la crittografia. Anche gli altri utenti del tuo account devono accedere alla chiave per visualizzare i dati di traccia crittografati nella console X-Ray.

Per una chiave gestita dal cliente, configura la chiave con una politica di accesso che consenta le seguenti azioni:

- L'utente che configura la chiave in X-Ray è autorizzato a `kms:CreateGrant` chiamare e. `kms:DescribeKey`
- Gli utenti che accedono ai dati di tracciamento crittografati hanno l'autorizzazione di chiamare `kms:Decrypt`.

Quando aggiungi un utente al gruppo Key users nella sezione di configurazione delle chiavi della console IAM, dispone dell'autorizzazione per entrambe queste operazioni. L'autorizzazione deve essere impostata solo sulla policy chiave, quindi non è necessaria alcuna AWS KMS autorizzazione per utenti, gruppi o ruoli. Per ulteriori informazioni, consulta Using Key Policies nella AWS KMS Developer Guide.

Per la crittografia predefinita, o se si sceglie la CMK AWS gestita (`aws/xray`), l'autorizzazione si basa su chi ha accesso alle API X-Ray. Tutti gli utenti con accesso a `PutEncryptionConfig`, inclusi in `AWSXrayFullAccess`, possono modificare la configurazione della crittografia. Per impedire a un utente di modificare la chiave crittografica, non assegnargli l'autorizzazione ad utilizzare `PutEncryptionConfig`.

AWS X-Ray esempi di politiche basate sull'identità

Per impostazione predefinita, gli utenti e i ruoli non dispongono dell'autorizzazione per creare o modificare risorse X-Ray. Inoltre, non possono eseguire attività utilizzando l' AWS API AWS Management Console AWS CLI, o. Un amministratore deve creare le policy IAM che concedono a utenti e ruoli l'autorizzazione per eseguire operazioni API specifiche sulle risorse specificate di cui hanno bisogno. L'amministratore devi quindi collegare queste policy a utenti o gruppi che richiedono tali autorizzazioni.

Per informazioni su come creare una policy basata su identità IAM utilizzando questi documenti di policy JSON di esempio, consulta Creazione di policy nella scheda JSON nella Guida per l'utente IAM.

Argomenti

- Best practice delle policy
- Uso della console X-Ray
- Consentire agli utenti di visualizzare le loro autorizzazioni
- Gestione dell'accesso ai gruppi X-Ray e alle regole di campionamento basate sui tag
- Policy gestite da IAM per X-Ray
- Aggiornamenti X-Ray alle AWS policy gestite
- Specificare una risorsa all'interno di una policy IAM

Best practice delle policy

Le policy basate sull'identità determinano se qualcuno può creare, accedere o eliminare le risorse X-Ray nel tuo account. Queste operazioni possono comportare costi aggiuntivi per l' Account AWS. Quando crei o modifichi policy basate su identità, segui queste linee guida e raccomandazioni:

- Inizia con le policy AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le politiche gestite che concedono le autorizzazioni per molti casi d'uso comuni.AWS Sono disponibili nel tuo. Account AWS Ti consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai AWS clienti specifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta Policy gestite da AWS o Policy gestite da AWS per le funzioni dei processi nella Guida per l'utente IAM.
- Applica le autorizzazioni con privilegi minimi: quando imposti le autorizzazioni con le policy IAM, concedi solo le autorizzazioni richieste per eseguire un'attività. Puoi farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo di IAM per applicare le autorizzazioni, consulta Policy e autorizzazioni in IAM nella Guida per l'utente di IAM.
- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso: per limitare l'accesso a operazioni e risorse puoi aggiungere una condizione alle tue policy. Ad esempio, è possibile scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. Puoi anche utilizzare le condizioni per concedere l'accesso alle azioni del servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per

ulteriori informazioni, consulta la sezione Elementi delle policy JSON di IAM: condizione nella Guida per l'utente di IAM.

- Utilizzo di IAM Access Analyzer per convalidare le policy IAM e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano alla sintassi della policy IAM (JSON) e alle best practice di IAM. IAM Access Analyzer offre oltre 100 controlli delle policy e consigli utili per creare policy sicure e funzionali. Per ulteriori informazioni, consulta Convalida delle policy per IAM Access Analyzer nella Guida per l'utente di IAM.
- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede utenti IAM o un utente root nel Account AWS tuo, attiva l'MFA per una maggiore sicurezza. Per richiedere la MFA quando vengono chiamate le operazioni API, aggiungi le condizioni MFA alle policy. Per ulteriori informazioni, consulta Configurazione dell'accesso alle API protetto con MFA nella Guida per l'utente di IAM.

Per maggiori informazioni sulle best practice in IAM, consulta Best practice di sicurezza in IAM nella Guida per l'utente di IAM.

Uso della console X-Ray

Per accedere alla AWS X-Ray console, è necessario disporre di un set minimo di autorizzazioni. Queste autorizzazioni devono consentire di elencare e visualizzare i dettagli sulle risorse X-Ray presenti nel tuo. Account AWS Se crei una policy basata sull'identità più restrittiva rispetto alle autorizzazioni minime richieste, la console non funzionerà nel modo previsto per le entità (utenti o ruoli) associate a tale policy.

Per garantire che tali entità possano ancora utilizzare la console X-Ray, allega la policy `AWSXRayReadOnlyAccess` AWS gestita alle entità. Questa policy è descritta più dettagliatamente nelle policy gestite da IAM per X-Ray. Per ulteriori informazioni, consulta Aggiunta di autorizzazioni a un utente nella Guida per l'utente IAM.

Non è necessario consentire le autorizzazioni minime della console per gli utenti che effettuano chiamate solo verso AWS CLI o l' AWS API. Al contrario, puoi accedere solo alle operazioni che soddisfano l'operazione API che stai cercando di eseguire.

Consentire agli utenti di visualizzare le loro autorizzazioni

Questo esempio mostra in che modo è possibile creare una policy che consente agli utenti IAM di visualizzare le policy inline e gestite che sono allegate alla relativa identità utente. Questa politica

include le autorizzazioni per completare questa azione sulla console o utilizzando l'API o a livello di codice. AWS CLI 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": "*"
        }
    ]
}
```

Gestione dell'accesso ai gruppi X-Ray e alle regole di campionamento basate sui tag

È possibile utilizzare le condizioni della policy basata sull'identità per controllare l'accesso ai gruppi X-Ray e le regole di campionamento basate sui tag. La seguente politica di esempio potrebbe essere utilizzata per negare a un ruolo utente le autorizzazioni per creare, eliminare o aggiornare gruppi con i tag o. `stage:prod` `stage:preprod` Per ulteriori informazioni sull'etichettatura delle regole e dei

gruppi di campionamento a raggi X, vedere. [Tagging delle regole e dei gruppi di campionamento di X-Ray]

Per negare a un utente l'accesso per creare, aggiornare o eliminare un gruppo con un tag `stage:prod` oppure `stage:preprod` assegnare all'utente un ruolo con una politica simile alla seguente.

```
{
    "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"
                    ]
                }
            }
        }
    ]
}
```

Per negare la creazione di una regola di campionamento, usa `aws:RequestTag` per indicare i tag che non possono essere passati come parte di una richiesta di creazione. Per negare l'aggiornamento o l'eliminazione di una regola di campionamento, usa `aws:ResourceTag` per negare le azioni basate sui tag presenti su tali risorse.

```
{
    "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"
                    ]
                }
            }
        }
    ]
}
```

Puoi allegare queste politiche (o combinarle in un'unica politica, quindi allegare la politica) agli utenti del tuo account. Affinché l'utente possa apportare modifiche a un gruppo o a una regola di campionamento, il gruppo o la regola di campionamento non devono essere etichettati `stage=prepod` o. `stage=prod` La chiave di tag di condizione `Stage` corrisponde a `Stage` e `stage` perché i nomi delle chiavi di condizione non effettuano la distinzione tra maiuscole e minuscole. Per ulteriori informazioni sul blocco condition, consulta IAM JSON Policy Elements: Condition nella IAM User Guide.

Un utente con un ruolo a cui è associata la seguente policy non può aggiungere il tag `role:admin` alle risorse e non può rimuovere i tag da una risorsa a cui è `role:admin` associato.

```
{
    "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"
                }
            }
        }
    ]
}
```

Policy gestite da IAM per X-Ray

Per semplificare la concessione delle autorizzazioni, IAM supporta le policy gestite per ogni servizio. Un servizio può aggiornare queste politiche gestite con nuove autorizzazioni quando rilascia nuove API. AWS X-Ray fornisce policy gestite per casi d'uso di sola lettura, sola scrittura e amministratore.

- `AWSXrayReadOnlyAccess`— Autorizzazioni di lettura per l'utilizzo della console X-Ray AWS o dell'SDK per ottenere dati di traccia AWS CLI, mappe di traccia, approfondimenti e configurazione X-Ray dall'API X-Ray. Include Observability Access Manager (OAM) `oam:ListSinks` e `oam:ListAttachedSinks` autorizzazioni per consentire alla console di visualizzare le tracce condivise dagli account di origine come parte dell'osservabilità tra account. CloudWatch Le azioni `BatchGetTraceSummaryById` e `GetDistinctTraceGraphs` API non sono pensate per essere richiamate dal codice e non sono incluse negli SDK e. 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`— Autorizzazioni di scrittura per l'utilizzo del demone X-Ray AWS o SDK per caricare AWS CLI documenti di segmento e telemetria nell'API X-Ray. Include le autorizzazioni di lettura per accedere alle regole di campionamento e i report sui risultati del campionamento.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:PutTraceSegments",
                "xray:PutTelemetryRecords",
                "xray:GetSamplingRules",
```

```
                "xray:GetSamplingTargets",
                "xray:GetSamplingStatisticSummaries"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
}
```

- `AWSXrayCrossAccountSharingConfiguration`— Concede le autorizzazioni per creare, gestire e visualizzare i collegamenti di Observability Access Manager per la condivisione di risorse X-Ray tra account. Utilizzato per consentire l'osservabilità CloudWatch tra account di origine e account di monitoraggio.

```
{
    "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`— Autorizzazione a utilizzare tutte le API X-Ray, incluse le autorizzazioni di lettura, le autorizzazioni di scrittura e l'autorizzazione a configurare le impostazioni delle chiavi di crittografia e le regole di campionamento. Include Observability Access Manager (OAM) `oam:ListSinks` e `oam:ListAttachedSinks` autorizzazioni per consentire alla console di visualizzare le tracce condivise dagli account di origine come parte dell'osservabilità tra account. CloudWatch

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:*",
                "oam:ListSinks"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "oam:ListAttachedLinks"
            ],
            "Resource": "arn:aws:oam:*:*:sink/*"
        }
    ]
}
```

Per aggiungere una policy gestita a un utente, gruppo o ruolo IAM

1. Apri la console IAM.
2. Aprire il ruolo associato al profilo dell'istanza, a un utente IAM o a un gruppo IAM.

3. In Permissions (Autorizzazioni), collegare la policy gestita.

Aggiornamenti X-Ray alle AWS policy gestite

Visualizza i dettagli sugli aggiornamenti delle policy AWS gestite per X-Ray da quando questo servizio ha iniziato a tracciare queste modifiche. Per ricevere avvisi automatici sulle modifiche a questa pagina, iscriviti al feed RSS nella pagina della cronologia dei documenti X-Ray.

Modifica	Descrizione	Data
Policy gestite da IAM per X-Ray: sono state aggiunte policy nuove `AWSXrayCrossAccountSharingConfiguration` `AWSXrayReadOnlyAccess` e `AWSXrayFullAccess` aggiornate.	X-Ray ha aggiunto le autorizzazioni di Observability Access Manager (OAM) `oam:ListSinks` e `oam:ListAttachedSinks` a queste policy per consentire alla console di visualizzare le tracce condivise dagli account di origine come parte dell'osservabilità tra account. CloudWatch	27 novembre 2022
Policy gestite da IAM per X-Ray: aggiornamento della policy. `AWSXrayReadOnlyAccess`	X-Ray ha aggiunto un'azione API,. `ListResourcePolicies`	15 novembre 2022
Utilizzo della console X-Ray: aggiornamento della policy `AWSXrayReadOnlyAccess`	X-Ray ha aggiunto due nuove azioni API e. `BatchGetTraceSummaryById` `GetDistinctTraceGraphs` Queste azioni non sono destinate a essere richiamate dal codice. Pertanto, queste	11 novembre 2022

Modifica	Descrizione	Data
	azioni API non sono incluse negli AWS SDK AWS CLI and.	

Specificare una risorsa all'interno di una policy IAM

Puoi controllare l'accesso alle risorse utilizzando una policy IAM. Per le operazioni che supportano le autorizzazioni a livello di risorsa, si utilizza un Amazon Resource Name (ARN) per identificare la risorsa a cui si applica la policy.

Tutte le azioni X-Ray possono essere utilizzate in una policy IAM per concedere o negare agli utenti il permesso di utilizzare quell'azione. Tuttavia, non tutte le azioni X-Ray supportano le autorizzazioni a livello di risorsa, che consentono di specificare le risorse su cui è possibile eseguire un'azione.

Per le operazioni che non supportano le autorizzazioni a livello di risorsa, è necessario utilizzare "`*`" come risorsa.

Le seguenti azioni X-Ray supportano le autorizzazioni a livello di risorsa:

- `CreateGroup`
- `GetGroup`
- `UpdateGroup`
- `DeleteGroup`
- `CreateSamplingRule`
- `UpdateSamplingRule`
- `DeleteSamplingRule`

Di seguito è riportato un esempio di policy di autorizzazione basata su identità per un'operazione `CreateGroup`. L'esempio mostra l'uso di un ARN relativo al nome di gruppo `local-users` con l'ID univoco come carattere jolly. L'ID univoco viene generato quando il gruppo viene creato, quindi non può essere previsto nella policy in anticipo. Quando utilizzi `GetGroup`, `UpdateGroup` oppure `DeleteGroup`, è possibile definirlo come un carattere jolly o l'ARN esatto, incluso l'ID.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:CreateGroup"
            ],
            "Resource": [
                "arn:aws:xray:eu-west-1:123456789012:group/local-users/*"
            ]
        }
    ]
}
```

Di seguito è riportato un esempio di policy di autorizzazione basata su identità per un'operazione `CreateSamplingRule`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "xray:CreateSamplingRule"
            ],
            "Resource": [
                "arn:aws:xray:eu-west-1:123456789012:sampling-rule/base-scorekeep"
            ]
        }
    ]
}
```

> **Note**
>
> L'ARN di una regola di campionamento è definito in base al nome. A differenza degli ARN dei gruppi, le regole di campionamento non hanno un ID generato in modo univoco.

Risoluzione dei problemi di identità e accesso in AWS X-Ray

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con X-Ray e IAM.

Argomenti

- Non sono autorizzato a eseguire un'azione in X-Ray
- Non sono autorizzato a eseguire iam:PassRole
- Sono un amministratore e voglio consentire ad altri di accedere a X-Ray
- Voglio consentire l'accesso a persone esterne al mioAccount AWSper accedere alle mie risorse radiologiche

Non sono autorizzato a eseguire un'azione in X-Ray

Se la AWS Management Console indica che non sei autorizzato a eseguire un'operazione, devi contattare l'amministratore per ricevere assistenza. L'amministratore è colui che ti ha fornito le credenziali di accesso.

L'errore di esempio seguente si verifica quando`mateojackson`l'utente tenta di utilizzare la console per visualizzare i dettagli su una regola di campionamento ma non ha`xray:GetSamplingRules`autorizzazioni.

```
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 questo caso, Mateo chiede al suo amministratore di aggiornare le sue policy per poter accedere alla risorsa di esempio relativa alla regola mediante l'operazione `xray:GetSamplingRules`.

Non sono autorizzato a eseguire iam:PassRole

Se ricevi un errore che indica che non sei autorizzato a eseguire`iam:PassRole`azione, le tue politiche devono essere aggiornate per consentirti di trasferire un ruolo a X-Ray.

Alcuni Servizi AWS consentono di passare un ruolo esistente a tale servizio, invece di creare un nuovo ruolo di servizio o un ruolo collegato ai servizi. Per eseguire questa operazione, è necessario disporre delle autorizzazioni per passare il ruolo al servizio.

L'errore di esempio seguente si verifica quando un utente IAM denominato`marymajor`tenta di utilizzare la console per eseguire un'azione in X-Ray. Tuttavia, l'operazione richiede che il servizio disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone delle autorizzazioni per passare il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In questo caso, le policy di Mary devono essere aggiornate per poter eseguire l'operazione `iam:PassRole`.

Per ulteriore assistenza con l'accesso, contatta l'amministratore AWS. L'amministratore è colui che ti ha fornito le credenziali di accesso.

Sono un amministratore e voglio consentire ad altri di accedere a X-Ray

Per consentire ad altri di accedere a X-Ray, devi creare un'entità IAM (utente o ruolo) per la persona o l'applicazione che necessita di accesso. Tale utente o applicazione utilizzerà le credenziali dell'entità per accedere ad AWS. È quindi necessario allegare una policy all'entità che conceda loro le autorizzazioni corrette in X-Ray.

Per iniziare immediatamente, consulta Creazione dei primi utenti e gruppi delegati IAM nella Guida per l'utente di IAM.

Voglio consentire l'accesso a persone esterne al mioAccount AWSper accedere alle mie risorse radiologiche

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per servizi che supportano policy basate su risorse o liste di controllo accessi (ACL), utilizza tali policy per concedere alle persone l'accesso alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se X-Ray supporta queste funzionalità, consultaCome AWS X-Ray funziona con IAM.
- Per informazioni su come garantire l'accesso alle risorse negli Account AWS che possiedi, consulta Fornire l'accesso a un utente IAM in un altro Account AWS in tuo possesso nella Guida per l'utente di IAM.
- Per informazioni su come fornire l'accesso alle risorse ad Account AWS di terze parti, consulta Fornire l'accesso agli Account AWS di proprietà di terze parti nella Guida per l'utente di IAM.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta Fornire l'accesso a utenti autenticati esternamente (Federazione delle identità) nella Guida per l'utente di IAM.

- Per informazioni sulle differenze tra l'utilizzo di ruoli e policy basate su risorse per l'accesso multi-account, consultare Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente di IAM.

Registrazione e monitoraggio in AWS X-Ray

Il monitoraggio è importante per garantire l'affidabilità, la disponibilità e le prestazioni delle soluzioni AWS. È necessario raccogliere i dati sul monitoraggio da tutte le parti del proprioAWSsoluzione per consentire un debug più facile di eventuali guasti in più punti.AWSfornisce diversi strumenti per il monitoraggio delle risorse X-Ray e la risposta a potenziali incidenti.

AWS CloudTrailLog di

AWS X-RayIntegrazione di conAWS CloudTrailper registrare le operazioni API eseguite da un utente, un ruolo o unAWSservizio in X-Ray. Puoi utilizzare CloudTrail per monitorare le richieste API X-Ray in tempo reale e memorizzare i log in Amazon S3, Amazon CloudWatch Logs e Amazon CloudWatch Events. Per ulteriori informazioni, consultare Registrazione delle chiamate all'API X-Ray con AWS CloudTrail.

Tracciamento di AWS Config

AWS X-RayIntegrazione di conAWS Configper registrare le modifiche alla configurazione apportate alle risorse crittografiche X-Ray. É possibile utilizzareAWS Configper eseguire l'inventario delle risorse crittografiche X-Ray, verificare la cronologia di configurazione X-Ray e inviare notifiche in base alle modifiche delle risorse. Per ulteriori informazioni, consultare Tracciamento delle modifiche alla configurazione della crittografia a X-Ray conAWS Config.

Monitoraggio Amazon CloudWatch

Puoi utilizzare l'SDK X-Ray per Java per pubblicare metriche Amazon CloudWatch non campionate di Amazon CloudWatch dai segmenti X-Ray raccolti. Questi parametri sono derivati dall'ora di inizio e di fine del segmento e dai flag di errore, di malfunzionamento e di stato trottled. Utilizzare questi parametri di tracciamento per esporre tentativi e problemi di dipendenza all'interno di sottosegmenti. Per ulteriori informazioni, consultare AWS X-Ray metriche per X-Ray SDK for Java.

Convalida della conformità per AWS X-Ray

Per sapere se un Servizio AWS programma rientra nell'ambito di specifici programmi di conformità, consulta Servizi AWS la sezione Scope by Compliance Program Servizi AWS e scegli il programma di conformità che ti interessa. Per informazioni generali, consulta Programmi di AWS conformità Programmi di .

È possibile scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta Scaricamento dei report in AWS Artifact .

La vostra responsabilità di conformità durante l'utilizzo Servizi AWS è determinata dalla sensibilità dei dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. AWS fornisce le seguenti risorse per contribuire alla conformità:

- Guide introduttive su sicurezza e conformità: queste guide all'implementazione illustrano considerazioni sull'architettura e forniscono i passaggi per l'implementazione di ambienti di base incentrati sulla AWS sicurezza e la conformità.
- Progettazione per la sicurezza e la conformità HIPAA su Amazon Web Services: questo white paper descrive in che modo le aziende possono utilizzare AWS per creare applicazioni idonee all'HIPAA.

  > **Note**
  >
  > Non Servizi AWS tutte sono idonee all'HIPAA. Per ulteriori informazioni, consulta la sezione Riferimenti sui servizi conformi ai requisiti HIPAA.

- AWS Risorse per la per la conformità: questa raccolta di cartelle di lavoro e guide potrebbe essere valida per il tuo settore e la tua località.
- AWS Guide alla conformità dei clienti: comprendi il modello di responsabilità condivisa attraverso la lente della conformità. Le guide riassumono le migliori pratiche per la protezione Servizi AWS e mappano le linee guida per i controlli di sicurezza su più framework (tra cui il National Institute of Standards and Technology (NIST), il Payment Card Industry Security Standards Council (PCI) e l'International Organization for Standardization (ISO)).
- Evaluating Resources with Rules nella AWS Config Developer Guide: il AWS Config servizio valuta la conformità delle configurazioni delle risorse alle pratiche interne, alle linee guida e alle normative del settore.

- AWS Security Hub— Ciò Servizio AWS fornisce una visione completa dello stato di sicurezza interno. AWS La Centrale di sicurezza utilizza i controlli di sicurezza per valutare le risorse AWS e verificare la conformità agli standard e alle best practice del settore della sicurezza. Per un elenco dei servizi e dei controlli supportati, consulta la pagina Documentazione di riferimento sui controlli della Centrale di sicurezza.
- AWS Audit Manager— Ciò Servizio AWS consente di verificare continuamente AWS l'utilizzo per semplificare la gestione dei rischi e la conformità alle normative e agli standard di settore.

Resilienza in AWS X-Ray

L'infrastruttura globale di AWS è progettata attorno a Regioni AWS e zone di disponibilità. Regioni AWS fornisce più zone di disponibilità fisicamente separate e isolate che sono connesse tramite reti altamente ridondanti, a bassa latenza e velocità effettiva elevata. Con le zone di disponibilità, è possibile progettare e gestire le applicazioni e database che eseguono il failover automatico tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, fault tolerant e scalabili rispetto alle infrastrutture a data center singolo o multiplo.

Per ulteriori informazioni sulle Regioni AWS e le zone di disponibilità, consulta Infrastruttura globale di AWS.

Sicurezza dell'infrastruttura in AWS X-Ray

Come servizio gestito, AWS X-Ray è protetto dalla sicurezza di rete globale AWS. Per informazioni sui servizi di sicurezza AWS e su come AWS protegge l'infrastruttura, consulta la pagina Sicurezza del cloud AWS. Per progettare l'ambiente AWS utilizzando le best practice per la sicurezza dell'infrastruttura, consulta la pagina Protezione dell'infrastruttura nel Pilastro della sicurezza di AWS Well-Architected Framework.

UsiAWSchiamate API pubblicate per accedere a X-Ray attraverso la rete. I clienti devono supportare quanto segue:

- Transport Layer Security (TLS). È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Suite di cifratura con Perfect Forward Secrecy (PFS), ad esempio Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale IAM. In alternativa, è possibile utilizzare AWS Security Token Service (AWS STS) per generare le credenziali di sicurezza temporanee per sottoscrivere le richieste.

Utilizzo di AWS X-Ray con endpoint VPC

Se utilizzi Amazon Virtual Private Cloud (Amazon VPC) per ospitare ilAWSrisorse, puoi stabilire una connessione privata tra il tuo VPC e X-Ray. Ciò consente alle risorse del tuo Amazon VPC di comunicare con il servizio X-Ray senza passare attraverso la rete Internet pubblica.

Amazon VPC è unServizio AWSche puoi usare per lanciareAWSrisorse in una rete virtuale definita dall'utente. Con un VPC, detieni il controllo delle impostazioni della rete, come l'intervallo di indirizzi IP, le sottoreti, le tabelle di routing e i gateway di rete. Per connettere il tuo VPC a X-Ray, definisci uninterfaccia: endpoint VPC. L'endpoint fornisce una connettività affidabile e scalabile a X-Ray senza richiedere un gateway Internet, un'istanza NAT (Network Address Translation) o una connessione VPN. Per ulteriori informazioni, consulta Che cos'è Amazon VPC? nella Guida per l'utente Amazon VPC.

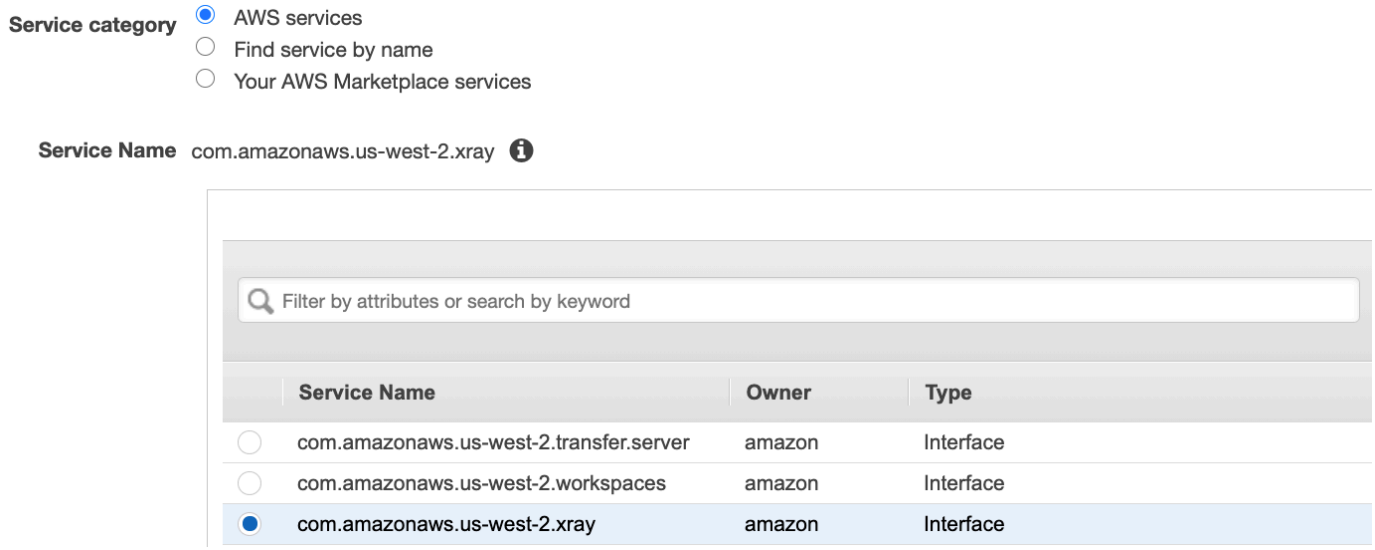
Gli endpoint VPC dell'interfaccia sono alimentati daAWS PrivateLink, unAWStecnologia che consente la comunicazione privata traServizi AWSutilizzando un'interfaccia di rete elastica con indirizzi IP privati. Per ulteriori informazioni, consulta ilNuovo —AWS PrivateLinkperServizi AWSpost sul blog eGuida introduttivanelGuida per l'utente di Amazon VPC.

Per assicurarti di poter creare un endpoint VPC per X-Ray nel tuo paese presceltoRegione AWS, vediRegioni supportate.

Creazione di un endpoint VPC per X-Ray

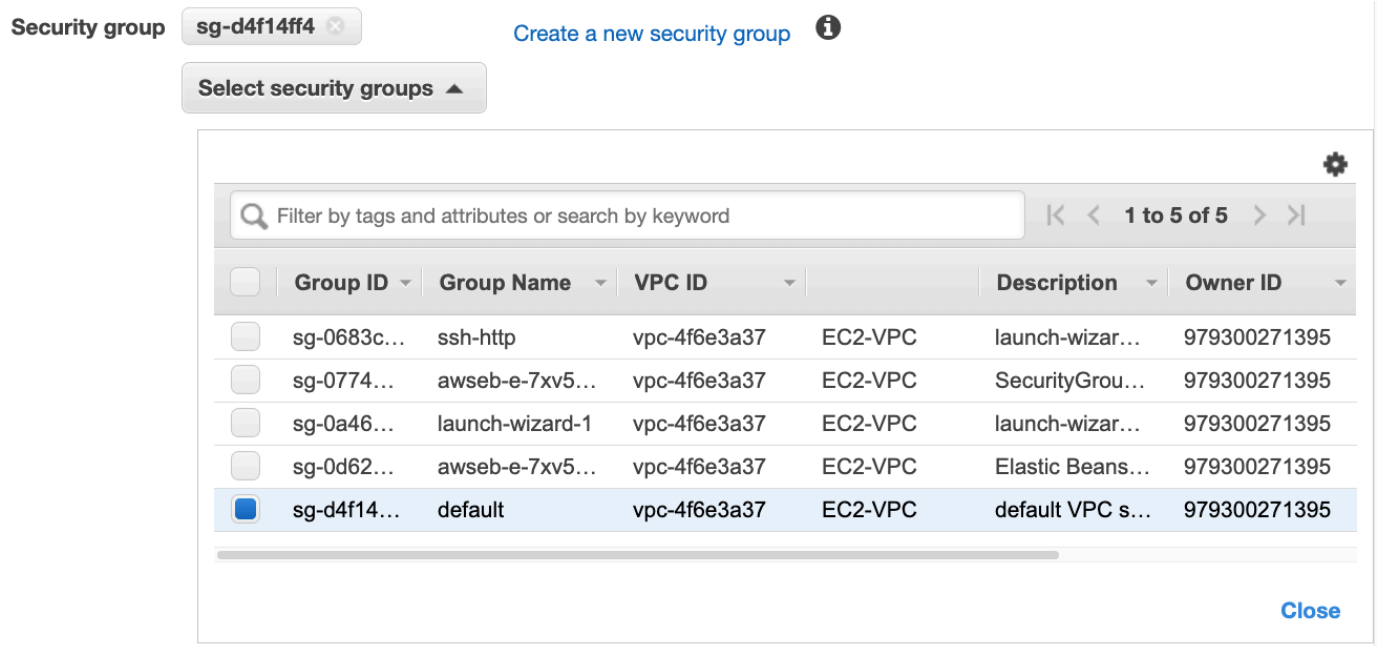
Per iniziare a utilizzare X-Ray con il tuo VPC, crea un endpoint VPC di interfaccia per X-Ray.

1. Accedi alla console Amazon VPC all'indirizzo https://console.aws.amazon.com/vpc/.
2. Passa aEndpointall'interno del pannello di navigazione e scegliCrea Endpoint.
3. Cerca e seleziona il nome delAWS X-Rayservizio:`com.amazonaws.`*`region`*`.xray`.



4. Seleziona il VPC che desideri, quindi seleziona una sottorete nel tuo VPC per utilizzare l'endpoint dell'interfaccia. Un'interfaccia di rete endpoint viene creata nella sottorete selezionata. Puoi specificare più sottoreti in zone di disponibilità differenti (come supportato dal servizio) per garantire che l'endpoint di interfaccia sia resiliente a errori della zona di disponibilità. In tal caso, viene creata un'interfaccia di rete in ogni sottorete specificata.

5. (Facoltativo) Il DNS privato è abilitato per impostazione predefinita per l'endpoint, in modo da poter effettuare richieste a X-Ray utilizzando il suo nome host DNS predefinito. Puoi scegliere di disabilitarlo.
6. Specifica i gruppi di sicurezza da associare all'interfaccia di rete dell'endpoint.



7. (Facoltativo) Specificate una politica personalizzata per controllare le autorizzazioni di accesso al servizio X-Ray. Per impostazione predefinita, è consentito l'accesso completo.

Controllo dell'accesso all'endpoint X-Ray VPC

Una policy endpoint VPC è una policy della risorsa IAM che viene collegata a un endpoint durante la creazione o la modifica dell'endpoint. Se non colleghi una policy durante la creazione di un endpoint, Amazon VPC collega una policy predefinita che consente l'accesso completo al servizio. Una policy endpoint non esclude né sostituisce policy dell'utente IAM o policy specifiche del servizio. Si tratta di una policy separata per controllare l'accesso dall'endpoint al servizio specificato. Le policy endpoint devono essere scritte in formato JSON. Per ulteriori informazioni, consultare Controllo degli accessi ai servizi con endpoint VPC nella Guida per l'utente di Amazon VPC.

La policy degli endpoint VPC ti consente di controllare le autorizzazioni per varie azioni a raggi X. Ad esempio, è possibile creare una policy che consenta soloPutTraceSegmente negare tutte le altre azioni. Ciò limita i carichi di lavoro e i servizi nel VPC a inviare solo i dati di traccia a X-Ray e a negare qualsiasi altra azione come il recupero dei dati, la modifica della configurazione di crittografia o la creazione/aggiornamento di gruppi.

Di seguito è riportato un esempio di policy sugli endpoint per X-Ray. Questa policy consente agli utenti che si connettono a X-Ray tramite VPC di inviare i dati dei segmenti a X-Ray e impedisce loro di eseguire altre azioni a raggi X.

```
{"Statement": [
    {"Sid": "Allow PutTraceSegments",
      "Principal": "*",
      "Action": [
        "xray:PutTraceSegments"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Per modificare la policy degli endpoint VPC per X-Ray

1. Accedere alla console Amazon VPC all'indirizzo https://console.aws.amazon.com/vpc/.
2. Nel pannello di navigazione, seleziona Endpoints (Endpoint).
3. Se non hai ancora creato l'endpoint per X-Ray, segui i passaggi indicatiCreazione di un endpoint VPC per X-Ray.
4. Seleziona ilcom.amazonaws.***regione***.radiografiaendpoint, quindi scegliPoliticascheda.
5. Scegli Edit Policy (Modifica policy), quindi apporta le modifiche.

Regioni supportate

X-Ray attualmente supporta gli endpoint VPC nei seguentiRegioni AWS:

- Stati Uniti orientali (Ohio)
- Stati Uniti orientali (Virginia settentrionale)
- Stati Uniti occidentali (California settentrionale)
- Stati Uniti occidentali (Oregon)
- Africa (Città del Capo)
- Asia Pacific (Hong Kong)
- Asia Pacific (Mumbai)
- Asia Pacific (Osaka)
- Asia Pacific (Seul)
- Asia Pacifico (Singapore)

- Asia Pacifico (Sydney)
- Asia Pacifico (Tokyo)
- Canada (Centrale)
- Europa (Francoforte)
- Europa (Irlanda)
- Europa (Londra)
- Europa (Milano)
- Europa (Parigi)
- Europa (Stoccolma)
- Medio Oriente (Bahrein)
- Sud America (São Paulo)
- AWS GovCloud(Stati Uniti orientali)
- AWS GovCloud(Stati Uniti occidentali)

AWS X-Ray API

L'API X-Ray fornisce l'accesso a tutte le funzionalità X-Ray tramite l' AWS SDK o direttamente tramite HTTPS AWS Command Line Interface. L'X-Ray API Reference documenta i parametri di input per ogni azione API e i campi e i tipi di dati che restituiscono.

Puoi utilizzare l' AWS SDK per sviluppare programmi che utilizzano l'API X-Ray. La console X-Ray e il daemon X-Ray utilizzano entrambi l'SDK AWS per comunicare con X-Ray. L' AWS SDK per ogni lingua ha un documento di riferimento per le classi e i metodi che mappano le azioni e i tipi dell'API X-Ray.

AWS Riferimenti SDK

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

AWS Command Line Interface È uno strumento da riga di comando che utilizza l'SDK per Python per chiamare le AWS API. Quando impari per la prima volta a conoscere un' AWS API, questa AWS CLI fornisce un modo semplice per esplorare i parametri disponibili e visualizzare l'output del servizio in formato JSON o testo.

Vedi AWS CLI Command Reference per i dettagli sui `aws xray` sottocomandi.

Argomenti

- Usando ilAWS X-RayAPI conAWSCLI
- Invio dei dati di tracciamento a AWS X-Ray
- Acquisizione di dati da AWS X-Ray
- Configurazione delle impostazioni di campionamento e di crittografia tramite l'API AWS X-Ray
- Utilizzo delle regole di campionamento con l'API X-Ray
- AWS X-Ray segmentare i documenti

Usando ilAWS X-RayAPI conAWSCLI

LaAWSLa CLI consente di accedere direttamente al servizio X-Ray e di utilizzare le stesse API utilizzate dalla console X-Ray per recuperare il grafico del servizio e i dati grezzi delle tracce. L'applicazione di esempio include script che mostrano come utilizzare queste API conAWSCLIP.

Prerequisiti

Questo tutorial utilizza l'applicazione di esempio Scorekeep e gli script in essa inclusi per generare dei dati di tracciamento e una mappa del servizio. Segui le istruzioni nel tutorial sulle nozioni di base per avviare l'applicazione.

Questo tutorial utilizza ilAWS CLIper mostrare l'uso di base dell'API X-Ray. LaAWSCLIP,disponibile per Windows, Linux e OS-X, fornisce l'accesso da riga di comando alle API pubbliche per tuttiServizi AWS.

> **Note**
>
> È necessario verificare che AWS CLI sia configurata nella stessa regione in cui è stata creata l'applicazione di esempio Scorekeep.

Gli script incluso per testare l'applicazione di esempio utilizzano `cURL` per inviare il traffico verso l'API e `jq` per analizzare l'output. Puoi scaricare l'eseguibile di `jq` da stedolan.github.io e l'eseguibile di `curl` da https://curl.haxx.se/download.html. La maggior parte delle installazioni Linux e OS X include cURL.

Generare i dati di tracciamento

Mentre il gioco è in corso, l'applicazione web genera continuamente traffico verso l'API ogni pochi secondi, ma genera solo un tipo di richiesta. Utilizza lo script `test-api.sh` per eseguire scenari completi e generare dati di tracciamento più eterogeni mentre testi l'API.

Per usare lo script **`test-api.sh`**

1. Apri la console Elastic Beanstalk.
2. Vai alconsole di gestioneper il tuo ambiente.
3. Copiare l'URL di ambiente dall'intestazione della pagina.
4. Aprire `bin/test-api.sh` e sostituire il valore per l'API con l'URL del proprio ambiente.

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

5. Eseguire lo script per generare il traffico verso l'API.

```
~/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
```

Usa l'API X-Ray

LaAWSLa CLI fornisce comandi per tutte le azioni API fornite da X-Ray, tra cuiGetServiceGrapheGetTraceSummaries. Per ulteriori informazioni su tutte le operazioni supportate e i tipi di dati che utilizzano, consulta la Guida di riferimento delle API di AWS X-Ray.

Example bin/service-graph.sh

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

Lo script recupera un grafo del servizio relativo agli ultimi 10 minuti.

```
~/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'
```

Lo script recupera l'URL dei tracciamenti generati tra uno e due minuti fa.

```
~/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[*]'
```

Lo script recupera i tracciamenti completi generati tra uno e due minuti fa.

```
~/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
    }
...
```

Pulizia

Termina il tuo ambiente Elastic Beanstalk per chiudere le istanze Amazon EC2, le tabelle DynamoDB e altre risorse.

Per terminare l'ambiente Elastic Beanstalk

1. Apri la console Elastic Beanstalk.
2. Passa alconsole di gestioneper il tuo ambiente.
3. Scegli Actions (Azioni).
4. Scegliere Terminate Environment (Termina ambiente).
5. Scegliere Terminate (Termina).

I dati di traccia vengono eliminati automaticamente da X-Ray dopo 30 giorni.

Invio dei dati di tracciamento a AWS X-Ray

È possibile inviare dati di traccia a X-Ray sotto forma di documenti di segmento. Un documento di segmento è una stringa in formato JSON che contiene le informazioni sull'attività eseguita dalla tua applicazione per il soddisfacimento di una richiesta. La tua applicazione può memorizzare i dati sull'attività che svolge in segmenti, oppure le attività che utilizzano i servizi e le risorse a valle possono creare dei sottosegmenti.

I segmenti registrano le informazioni sull'attività svolta dalla tua applicazione. Un segmento, come minimo, memorizza il tempo speso per un'attività, un nome e due identificativi. L'ID di tracciamento monitora la richiesta durante il suo passaggio tra i servizi. L'ID segmento monitora l'attività svolta per la richiesta proveniente da uno specifico servizio.

Example Segmento completo minimale

```
{
  "name" : "Scorekeep",
```

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

Quando ricevi una richiesta, puoi inviare un segmento in esecuzione come segnaposto fino al completamento della richiesta.

Example Segmento in esecuzione

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

È possibile inviare segmenti a X-Ray direttamente, `PutTraceSegments`con o tramite il demone X-Ray.

La maggior parte delle applicazioni richiama altri servizi o accede a risorse con l'SDK. AWS Registra le informazioni sulle chiamate downstream in sottosegmenti. X-Ray utilizza i sottosegmenti per identificare i servizi a valle che non inviano segmenti e creare le relative voci nel grafico dei servizi.

Un sottosegmento può essere incorporato in un documento di segmento completo oppure può essere inviato in modo separato. Invia i sottosegmenti separatamente per tracciare in modo asincrono le chiamate downstream per richieste di lunga durata o per evitare di superare la dimensione massima del documento del segmento (64 kB).

Example Segmento secondario

Un sottosegmento presenta un `type` valorizzato a `subsegment` e un `parent_id` che identifica il segmento padre.

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

```
  “type” : “subsegment”,
  “parent_id” : “70de5b6f19ff9a0b”
}
```

Per ulteriori informazioni sui campi e i valori che puoi includere in segmenti e sottosegmenti, consulta AWS X-Ray segmentare i documenti.

Sections

- Generazione degli ID di tracciamento
- Usando PutTraceSegments
- Invio di documenti di segmento al demone X-Ray

Generazione degli ID di tracciamento

Per inviare dati a X-Ray, è necessario generare un ID di traccia univoco per ogni richiesta.

Formato ID di traccia X-Ray

Una radiografia `trace_id` è composta da tre numeri separati da trattini. Ad esempio, `1-58406520-a006649127e371903a2de979`. Questo include:

- Il numero di versione, che è. `1`
- L'ora della richiesta originale in Unix epoch time utilizzando 8 cifre esadecimali.

  Ad esempio, il fuso orario PST delle 10:00 del 1° dicembre 2016 è espresso in secondi o in cifre esadecimali. `1480615200` `58406520`
- Un identificatore globale unico a 96 bit per la traccia in 24 cifre esadecimali.

> **Note**
>
> X-Ray ora supporta gli ID di traccia creati utilizzando OpenTelemetry e qualsiasi altro framework conforme alla specifica W3C Trace Context. Un ID di traccia W3C deve essere formattato nel formato X-Ray Trace ID quando viene inviato a X-Ray. Ad esempio, l'ID di traccia W3C `4efaaf4d1e8720b39541901950019ee5` deve essere formattato come `1-4efaaf4d-1e8720b39541901950019ee5` quando si invia a X-Ray. Gli ID di traccia X-Ray includono il timestamp originale della richiesta in Unix epoch time, ma questo non è necessario quando si inviano gli ID di traccia W3C in formato X-Ray.

È possibile scrivere uno script per generare ID di traccia X-Ray per i test. Qui di seguito sono riportati due esempi.

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"
```

Vedi l'applicazione di esempio Scorekeep per gli script che creano ID di traccia e inviano segmenti al demone X-Ray.

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

Usando PutTraceSegments

Puoi caricare i documenti di segmento tramite l'API `PutTraceSegments`. L'API dispone di un singolo parametro, `TraceSegmentDocuments`, che richiede un elenco di documenti di segmento in formato JSON.

Con l'AWS CLI, usa il `aws xray put-trace-segments` comando per inviare documenti di segmento direttamente a X-Ray.

```
$ 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 e Windows PowerShell hanno requisiti diversi per la citazione e l'escape delle virgolette nelle stringhe JSON. Per i dettagli, consulta Utilizzo delle virgolette nelle stringhe nella Guida per l'utente della AWS CLI .

L'output elenca tutti i segmenti per i quali l'elaborazione non è riuscita. Ad esempio, se la data nell'ID di tracciamento è troppo remota nel passato, verrà visualizzato un errore come il seguente.

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

Puoi passare più documenti di segmento insieme, separati da spazi.

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

Invio di documenti di segmento al demone X-Ray

Invece di inviare documenti di segmento all'API X-Ray, puoi inviare segmenti e sottosegmenti al demone X-Ray, che li memorizzerà nel buffer e li caricherà sull'API X-Ray in batch. L'X-Ray SDK invia i documenti dei segmenti al demone per evitare di effettuare chiamate dirette. AWS

> **Note**
>
> Per istruzioni sull'esecuzione del daemon, consulta Esegui il daemon X-Ray.

Invia il segmento in formato JSON tramite la porta UDP 2000, anteponendo l'intestazione per il daemon, `{"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"}
```

Su Linux, puoi inviare documenti di segmento al daemon tramite un terminale Bash. Salva l'intestazione e il documento di segmento in un file di testo e reindirizzalo verso `/dev/udp` con `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"}
```

Controlla il registro del demone per verificare che abbia inviato il segmento a X-Ray.

```
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)
```

Acquisizione di dati da AWS X-Ray

AWS X-Ray elabora i dati di traccia che gli invii per generare tracce complete, riepiloghi di traccia e grafici di servizio in JSON. Puoi recuperare i dati generati direttamente dall'API con la AWS CLI.

Sections

- Recupero del grafico del servizio
- Recupero del grafico del servizio dal gruppo
- Recupero dei tracciamenti
- Recupero e definizione dell’analisi delle cause principali

Recupero del grafico del servizio

Puoi utilizzare l'API `GetServiceGraph` per recuperare il grafico del servizio in formato JSON. L'API richiede un'ora di inizio e di fine, che puoi calcolare in un terminale Linux con il comando `date`.

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

`date +%s` converte una data in secondi. Utilizza questo numero come ora finale e sottrai da esso il tempo necessario per ottenere un'ora di inizio.

Example Script per recuperare un grafico del servizio relativo agli ultimi 10 minuti

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

L'esempio seguente mostra un grafico di servizio con 4 nodi, tra cui un nodo client, un'istanza EC2, una tabella DynamoDB e un argomento Amazon SNS.

Example GetServiceGraph uscita

```
{
    "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
                }
            ]
        }
    ]
}
```

Recupero del grafico del servizio dal gruppo

Per effettuare la chiamata a un grafico del servizio in base al contenuto di un gruppo, includi `groupName` o `groupARN`. L'esempio seguente mostra una chiamata del grafico del servizio a un gruppo denominato Example1.

Example Script per recuperare un grafico del servizio in base al nome del gruppo Example1

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

Recupero dei tracciamenti

Puoi utilizzare l'API `GetTraceSummaries` per ottenere un elenco di riepiloghi dei tracciamenti. I riepiloghi dei tracciamenti includono informazioni che è puoi utilizzare per identificare i tracciamenti da scaricare in modo completo, tra cui annotazioni, informazioni su richiesta e risposta e ID.

Ci sono due flag `TimeRangeType` disponibili quando si chiama `aws xray get-trace-summaries`:

- TraceId— La `GetTraceSummaries` ricerca predefinita utilizza l'ora TraceID e restituisce le tracce iniziate all'interno dell'intervallo calcolato. `[start_time, end_time)` Questo intervallo di timestamp viene calcolato in base alla codifica del timestamp all'interno di o può essere definito manualmente. TraceId
- Ora dell'evento: per cercare gli eventi man mano che si verificano nel tempo, AWS X-Ray consente di cercare le tracce utilizzando i timestamp degli eventi. L'ora dell'evento restituisce i tracciamenti attivi durante l'intervallo `[start_time, end_time)`, indipendentemente dall'inizio del tracciamento.

Utilizza il comando `aws xray get-trace-summaries` per ottenere un elenco di riepiloghi di tracciamento. I seguenti comandi ottengono un elenco di riepiloghi delle tracce relative a un periodo compreso tra 1 e 2 minuti nel passato utilizzando l'ora predefinita. TraceId

Example Script per ottenere i riepiloghi dei tracciamenti

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

Example GetTraceSummaries uscita

```
{
    "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
```

```
}
```

Utilizza l'ID di tracciamento nell'output per recuperare un tracciamento completo con l'API [BatchGetTraces](#).

Example BatchGetTraces comando

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

Example BatchGetTraces uscita

```
{
    "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": []
}
```

Il tracciamento completo include un documento per ogni segmento, creato a partire da tutti i documenti di segmento con lo stesso ID di tracciamento ricevuti. Questi documenti non rappresentano i dati inviati a X-Ray dall'applicazione. Rappresentano invece i documenti elaborati generati dal servizio X-Ray. X-Ray crea il documento di traccia completo compilando i documenti dei segmenti inviati dall'applicazione e rimuovendo i dati che non sono conformi allo schema del documento del segmento.

X-Ray crea anche segmenti dedotti per le chiamate downstream verso servizi che non inviano segmenti di per sé. Ad esempio, quando si chiama DynamoDB con un client strumentato, l'SDK X-Ray registra un sottosegmento con dettagli sulla chiamata dal suo punto di vista. Tuttavia, DynamoDB non invia un segmento corrispondente. X-Ray utilizza le informazioni nel sottosegmento per creare un segmento dedotto che rappresenti la risorsa DynamoDB nella mappa di traccia e lo aggiunge al documento di traccia.

Per ottenere più tracciamenti tramite le API, hai bisogno di un elenco di ID di tracciamento, che puoi estrarre dall'output di `get-trace-summaries` tramite una query AWS CLI. Reindirizza l'elenco all'input di per ottenere tracce complete `batch-get-traces` per un periodo di tempo specifico.

Example Script per ottenere i tracciamenti completi per un periodo di un minuto

```
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[*]'
```

Recupero e definizione dell’analisi delle cause principali

Dopo aver generato un riepilogo delle tracce con l'GetTraceSummaries API, è possibile riutilizzare i riepiloghi delle tracce parziali nel formato JSON per creare un'espressione di filtro raffinata basata

sulle cause principali. Consulta gli esempi riportati di seguito per una procedura guidata per le fasi di definizione.

Example Esempio GetTraceSummaries di output: sezione relativa al tempo di risposta e alla causa principale

```
{
  "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
        }
      ]
    }
  ]
}
```

Modificando e apportando omissioni all'output precedente, questo JSON può diventare un filtro per le entità di cause principali corrispondenti. Per tutti i campi presenti nel JSON, ogni corrispondenza dei candidati deve essere esatta o il tracciamento non verrà restituito. I campi rimossi diventano valori jolly, un formato compatibile con la struttura di query dell'espressione di filtro.

Example Causa principale del tempo di risposta riformattata

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

Questo JSON viene quindi utilizzato come parte di un'espressione di filtro tramite una chiamata a `rootcause.json = #[{}]`. Consulta il capitolo Espressioni di filtro per ulteriori dettagli sulle query con espressioni di filtro.

Example Esempio di filtro JSON

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

Configurazione delle impostazioni di campionamento e di crittografia tramite l'API AWS X-Ray

AWS X-Ray fornisce le API per configurare le regole di campionamento, le regole dei gruppi e le impostazioni di crittografia.

Sezioni

- Impostazioni di crittografia
- Regole di campionamento
- Gruppi

Impostazioni di crittografia

UtilizzaPutEncryptionConfigper specificare unAWS Key Management Service(AWS KMS) chiave da utilizzare per la crittografia.

> **Note**
>
> La X-Ray non supporta i tasti KMS asimmetrici.

```
$ 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"
    }
}
```

Come ID della chiave, puoi utilizzare un alias (come illustrato nell'esempio), una ID della chiave o di un ARN (Amazon Resource Name).

Utilizza GetEncryptionConfig per recuperare la configurazione corrente. Quando X-Ray termina l'applicazione delle tue impostazioni, lo stato si modifica daUPDATINGaACTIVE.

```
$ aws xray get-encryption-config
```

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

Per interrompere l'utilizzo di una chiave KMS e utilizzare la crittografia di default, impostare il tipo di crittografia a`NONE`.

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

Regole di campionamento

Puoi gestire laregole di campionamentonel tuo account con l'API X-Ray. Per ulteriori informazioni sull'aggiunta e la gestione dei tag, consultaTagging delle regole e dei gruppi di campionamento di X-Ray.

Recupera tutte le regole di campionamento utilizzando l'API `GetSamplingRules`.

```
$ 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
        }
    ]
}
```

La regola di default si applica a tutte le richieste che non soddisfano nessun'altra regola. Si tratta della regola a priorità più bassa e non può essere eliminata. Puoi, tuttavia, modificare la percentuale e la dimensione della riserva utilizzando `UpdateSamplingRule`.

Example Input dell'API**`UpdateSamplingRule`**- 10000-default.json

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

L'esempio seguente utilizza il file precedente come input per modificare la regola di default impostando una percentuale dell'1% senza riserva. I tag sono opzionali. Se si sceglie di aggiungere tag, è necessario un codice tag e i valori dei tag sono facoltativi. Per rimuovere i tag esistenti da una regola di campionamento, utilizzareUntagResource

```
$ 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
        },
```

Crea regole di campionamento aggiuntive con `CreateSamplingRule`. Quando crei una regola, la maggior parte dei campi della regola sono obbligatori. L'esempio seguente crea due regole. Questa prima regola imposta una percentuale fissa per l'applicazione di esempio Scorekeep. Si applica a tutte le richieste elaborate dall'API alle quali non si applicano regole di priorità maggiore.

Example Input dell'API**`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
    }
}
```

Anche la seconda regola si applica a Scorekeep, ma le è assegnata una priorità maggiore ed è più specifica. Questa regola consente di impostare una percentuale di campionamento molto bassa per le richieste di polling. Queste sono le richieste GET effettuate dal client ogni pochi secondi per verificare la presenza di modifiche allo stato del gioco.

Example Input dell'API**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
    }
}
```

I tag sono opzionali. Se si sceglie di aggiungere tag, è necessario un codice tag e i valori dei tag sono facoltativi.

```
$ 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
    }
}
```

Per eliminare una regola di campionamento, utilizza [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
    }
}
```

Gruppi

Puoi utilizzare l'API X-Ray per gestire i gruppi nel tuo account. I gruppi costituiscono una raccolta di tracciamenti definiti da un'espressione di filtro. Puoi utilizzare i gruppi per generare grafi del servizio aggiuntivi e fornire i parametri di Amazon CloudWatch. Consulta .Acquisizione di dati da AWS X-Rayper ulteriori dettagli sull'uso dei parametri e dei grafi del servizio tramite l'API X-Ray. Per ulteriori informazioni sui gruppi, consultaConfigurazione dei gruppi. Per ulteriori informazioni sull'aggiunta e la gestione dei tag, consultaTagging delle regole e dei gruppi di campionamento di X-Ray.

Crea un gruppo con `CreateGroup`. I tag sono opzionali. Se si sceglie di aggiungere tag, è necessario un codice tag e i valori dei tag sono facoltativi.

```
$ 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}"
}
```

Ottieni tutti i gruppi esistenti con `GetGroups`.

```
$ 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"
}
```

Aggiorna un gruppo con `UpdateGroup`. I tag sono opzionali. Se si sceglie di aggiungere tag, è necessario un codice tag e i valori dei tag sono facoltativi. Per rimuovere i tag esistenti da un gruppo, utilizzareUntagResource.

```
$ 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}"
}
```

Elimina un gruppo con `DeleteGroup`.

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

Utilizzo delle regole di campionamento con l'API X-Ray

LaAWS X-RayL'SDK utilizza l'API X-Ray per ottenere le regole di campionamento, i risultati di rialzo del campionamento e le quote. Puoi utilizzare queste API per comprendere meglio il modo in cui funzionano le regole di campionamento, oppure per implementare il campionamento in un linguaggio non supportato dall'SDK X-Ray di.

Inizia a recuperare tutte le regole di campionamento con `GetSamplingRules`.

```
$ 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
        }
    ]
}
```

L'output include la regola di default e regole personalizzate. Se non hai ancora creato nessuna regola di campionamento, consulta Regole di campionamento.

Valuta le regole rispetto alle richieste in entrata in ordine crescente di priorità. Quando una regola viene soddisfatta, utilizza la percentuale fissa e la dimensione della riserva per prendere una decisione sul campionamento. Registra le richieste campionate e ignora (ai fini del tracciamento) le richieste non campionate. Interrompe la valutazione delle regole nel momento un cui viene assunta una decisione sul campionamento.

La dimensione della riserva di una regola è il numero di tracciamenti al secondo da registrare prima di applicare la percentuale fissa. La riserva è conteggiata su tutti i servizi in modo cumulativo, quindi non puoi utilizzarla direttamente. Tuttavia, se è diversa da zero, puoi acquisire un tracciamento al secondo dalla riserva finché X-Ray assegna una quota. Prima di ricevere una quota, registra la prima richiesta ogni secondo e applica la percentuale fissa per ulteriori richieste. La percentuale fissa è un numero decimale tra 0 e 1,00 (100%).

L'esempio seguente mostra una chiamata a `GetSamplingTargets` con dettagli sulle decisioni di campionamento effettuate negli ultimi 10 secondi.

```
$ 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": []
}
```

La risposta di X-Ray include una quota da utilizzare al posto del prelievo dalla riserva. In questo esempio, il servizio ha prelevato 10 tracciamenti dalla riserva durante i 10 secondi, e viene applicata la percentuale fissa del 10% per le altre 100 richieste, generando un totale di 20 richieste campionate. La quota è valida per cinque minuti (indicati dal campo time-live) o finché non viene assegnata una nuova quota. La X-Ray può anche assegnare un intervallo di reporting più esteso rispetto all'impostazione predefinita, anche se non lo ha fatto in questa sede.

> **Note**
>
> La risposta X-Ray prima chiamata potrebbe non includere una quota. Continua a prelevare dalla riserva finché non viene assegnata una quota.

Gli altri due campi della risposta possono indicare dei problemi con l'input. Controlla `LastRuleModification` rispetto all'ultima chiamata di `GetSamplingRules`. Se è più recente, scarica una nuova copia delle regole. `UnprocessedStatistics` può includere errori che indicano che una regola è stata eliminata, che il documento delle statistiche di input era troppo vecchio o che sono presenti dei problemi con le autorizzazioni.

AWS X-Ray segmentare i documenti

Un segmento di tracciamento è una rappresentazione JSON di una richiesta elaborata dalla tua applicazione. Un segmento di traccia registra informazioni sulla richiesta originale, informazioni sul lavoro svolto localmente dall'applicazione e sottosegmenti con informazioni sulle chiamate downstream effettuate dall'applicazione a AWS risorse, API HTTP e database SQL.

Un documento di segmento trasmette informazioni su un segmento a X-Ray. Un documento di segmento può pesare fino a 64 kB e contenere un intero segmento con sottosegmenti, un frammento di segmento che indica che una richiesta è in corso o un singolo sottosegmento inviato separatamente. È possibile inviare documenti segmentati direttamente a X-Ray utilizzando l'`PutTraceSegments`API.

X-Ray compila ed elabora i documenti dei segmenti per generare riepiloghi di tracce interrogabili e tracce complete a cui è possibile accedere utilizzando rispettivamente le API e. `GetTraceSummariesBatchGetTraces` Oltre ai segmenti e ai sottosegmenti inviati a X-Ray, il servizio utilizza le informazioni nei sottosegmenti per generare segmenti dedotti e li aggiunge alla traccia completa. I segmenti dedotti rappresentano i servizi e le risorse a valle nella mappa di traccia.

X-Ray fornisce uno schema JSON per i documenti di segmento. È possibile scaricare lo schema qui: xray-segmentdocument-schema-v 1.0.0. I campi e gli oggetti elencati nello schema sono descritti in modo dettagliato nelle sezioni seguenti.

Un sottoinsieme di campi di segmento viene indicizzato da X-Ray per essere utilizzato con le espressioni di filtro. Ad esempio, se imposti il `user` campo su un segmento su un identificatore univoco, puoi cercare segmenti associati a utenti specifici nella console X-Ray o utilizzando l'API. `GetTraceSummaries` Per ulteriori informazioni, consulta Utilizzo delle espressioni di filtro.

Quando strumentate la vostra applicazione con l'SDK X-Ray, l'SDK genera documenti di segmento per voi. Invece di inviare i documenti segmentati direttamente a X-Ray, l'SDK li trasmette tramite una porta UDP locale al demone X-Ray. Per ulteriori informazioni, consulta Invio di documenti di segmento al demone X-Ray.

Sections

- Campi del segmento
- Sottosegmenti
- Dati sulle richieste HTTP
- Annotazioni
- Metadati
- AWS dati relativi alle risorse
- Errori ed eccezioni
- Query SQL

Campi del segmento

Un segmento memorizza le informazioni di tracciamento relative a una richiesta elaborata dalla tua applicazione. Come minimo, un segmento memorizza nome, ID, ora di inizio, ID di tracciamento e ora di fine della richiesta.

Example Segmento completo minimale

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

Di seguito sono elencati i campi obbligatori, o condizionalmente obbligatori, per i segmenti.

> **Note**
>
> I valori devono essere stringhe (lunghe fino a un massimo di 250 caratteri) a meno che non sia indicato altrimenti.

Campi del segmento obbligatori

- `name`— Il nome logico del servizio che ha gestito la richiesta, fino a 200 caratteri. Ad esempio, il nome dell'applicazione o il nome di dominio. I nomi possono contenere lettere in formato Unicode, numeri e spazi e i seguenti simboli: `_`, `.`, `:`, `/`, `%`, `&`, `#`, `=`, `+`, `\`, `-`, `@`
- `id`— Un identificatore a 64 bit per il segmento, unico tra i segmenti della stessa traccia, in 16 cifre esadecimali.
- `trace_id`— Un identificatore univoco che collega tutti i segmenti e i sottosegmenti provenienti da una singola richiesta del client.

  Formato ID di traccia X-Ray

  Una radiografia `trace_id` è composta da tre numeri separati da trattini. Ad esempio, `1-58406520-a006649127e371903a2de979`. Questo include:

  - Il numero di versione, che è. 1
  - L'ora della richiesta originale in Unix epoch time utilizzando 8 cifre esadecimali.

    Ad esempio, il fuso orario PST delle 10:00 del 1° dicembre 2016 è espresso in secondi o in cifre esadecimali. `1480615200` `58406520`
  - Un identificatore a 96 bit univoco a livello globale per la traccia in 24 cifre esadecimali.

> **Note**
>
> X-Ray ora supporta gli ID di traccia creati utilizzando OpenTelemetry e qualsiasi altro framework conforme alla specifica W3C Trace Context. Un ID di traccia W3C deve essere formattato nel formato X-Ray Trace ID quando viene inviato a X-Ray. Ad esempio, l'ID di traccia W3C `4efaaf4d1e8720b39541901950019ee5` deve essere formattato come `1-4efaaf4d-1e8720b39541901950019ee5` quando si invia a X-Ray. Gli ID di traccia X-Ray includono il timestamp originale della richiesta in Unix epoch time, ma questo non è necessario quando si inviano gli ID di traccia W3C in formato X-Ray.

> **Sicurezza dell'ID di tracciamento**
>
> Gli ID di tracciamento sono visibili nelle intestazioni di risposta. Genera gli ID di tracciamento con un algoritmo di generazione dei numeri casuali sicuro per assicurare

che gli aggressori non siano in grado di calcolare futuri ID di tracciamento e di inviare le richieste con tali ID verso la tua applicazione.

- `start_time`— numero che indica l'ora in cui il segmento è stato creato, in secondi a virgola mobile nell'ora dell'epoca. Ad esempio `1480615200.010` o `1.480615200010E9`. Utilizza il numero di cifre decimali di cui hai bisogno. Quando disponibile, è consigliata la risoluzione al microsecondo.
- `end_time`— numero che indica l'ora in cui il segmento è stato chiuso. Ad esempio `1480615200.090` o `1.480615200090E9`. Specifica un `end_time` oppure il valore `in_progress`.
- `in_progress`— booleano, impostato su `true` invece di specificare an `end_time` per registrare che un segmento è iniziato, ma non è completo. Invia la segnalazione di un segmento in elaborazione per memorizzare la richiesta ricevuta quando l'applicazione riceve una richiesta la cui esecuzione richiede molto tempo. Quando viene inviata la risposta, invia il completamento del segmento per sovrascrivere il segmento in elaborazione. Per ogni richiesta, invia un solo segmento completo e uno o nessun segmento in esecuzione.

> **Nomi dei servizi**
>
> I segmenti `name` devono corrispondere al nome di dominio o al nome logico del servizio che genera il segmento. Tuttavia, questo non viene applicato. Qualsiasi applicazione autorizzata `PutTraceSegments`può inviare segmenti con qualsiasi nome.

I seguenti campi dei segmenti sono facoltativi.

Campi del segmento facoltativi

- `service`— Un oggetto con informazioni sull'applicazione.
  - `version`— Una stringa che identifica la versione dell'applicazione che ha fornito la richiesta.
- `user`— Una stringa che identifica l'utente che ha inviato la richiesta.
- `origin`— Il tipo di AWS risorsa che esegue l'applicazione.

  Valori supportati

  - `AWS::EC2::Instance`— Un'istanza Amazon EC2.
  - `AWS::ECS::Container`— Un contenitore Amazon ECS.

  - `AWS::ElasticBeanstalk::Environment`— Un ambiente Elastic Beanstalk.

  Quando, per la tua applicazione, sono validi più valori, utilizza quello più specifico. Ad esempio, un ambiente Docker Elastic Beanstalk Multicontainer esegue l'applicazione su un contenitore Amazon ECS, che a sua volta viene eseguito su un'istanza Amazon EC2. In questo caso, è possibile impostare l'origine a `AWS::ElasticBeanstalk::Environment` poiché l'ambiente è il padre delle altre due risorse.
- `parent_id`— Un ID di sottosegmento che specifichi se la richiesta proviene da un'applicazione strumentata. L'X-Ray SDK aggiunge l'ID del sottosegmento principale all'intestazione di tracciamento per le chiamate HTTP downstream. Nel caso di sottosegmenti nidificati, un sottosegmento può avere come padre un segmento o un sottosegmento.
- `http`— `http`oggetti con informazioni sulla richiesta HTTP originale.
- `aws`— `aws`oggetto con informazioni sulla AWS risorsa su cui l'applicazione ha fornito la richiesta.
- `error`, `throttle`e `fault`, `cause` — campi di errore che indicano che si è verificato un errore e che includono informazioni sull'eccezione che ha causato l'errore.
- `annotations`— `annotations`oggetto con coppie chiave-valore che si desidera che X-Ray indicizzi per la ricerca.
- `metadata`— `metadata`oggetto con eventuali dati aggiuntivi che si desidera memorizzare nel segmento.
- `subsegments`— matrice di `subsegment`oggetti.

Sottosegmenti

È possibile creare sottosegmenti per registrare le chiamate Servizi AWS e le risorse effettuate con l' AWS SDK, le chiamate alle API Web HTTP interne o esterne o le query del database SQL. Puoi anche creare dei sottosegmenti per eseguire il debug o annotare blocchi di codice dell'applicazione. I sottosegmenti possono contenere a loro volta altri sottosegmenti, perciò un sottosegmento personalizzato che memorizza i metadati su una chiamata ad una funzione interna può contenere altri sottosegmenti personalizzati o sottosegmenti relativi alle chiamate a valle.

Un sottosegmento registra una chiamata a valle dal punto di vista del servizio che la chiama. X-Ray utilizza i sottosegmenti per identificare i servizi a valle che non inviano segmenti e creare le relative voci nel grafico dei servizi.

Un sottosegmento può essere incorporato in un intero documento di segmento oppure può essere inviato in modo indipendente. Invia sottosegmenti separatamente per tracciare in modo asincrono

le chiamate a valle in caso di richieste di lunga durata, oppure per evitare di superare la dimensione massima del documento di segmento.

Example Segmento con sottosegmento incorporato

Un sottosegmento indipendente presenta un `type` valorizzato a `subsegment` e un `parent_id` che identifica il segmento padre.

```
{
  "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
```

```
        }
      }
    }
  ]
}
```

Per le richieste di lunga durata, è possibile inviare un segmento in corso per notificare a X-Ray che la richiesta è stata ricevuta e quindi inviare i sottosegmenti separatamente per tracciarli prima di completare la richiesta originale.

Example Segmento in esecuzione

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

Example Sottosegmento indipendente

Un sottosegmento presenta un `type` valorizzato a `subsegment`, un `trace_id` e un `parent_id` che identifica il segmento padre.

```
{
  "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
      }
  }
}
```

Quando la richiesta viene completata, chiudi il segmento inviandolo nuovamente con il campo `end_time` valorizzato. Il segmento completato sovrascrive il segmento in elaborazione.

Puoi anche inviare i sottosegmenti separatamente per richieste completate che hanno attivato flussi di lavoro asincroni. Ad esempio, un'API web potrebbe restituire immediatamente una risposta `OK 200` prima di avviare l'attività richiesta dall'utente. È possibile inviare un segmento completo a X-Ray non appena viene inviata la risposta, seguito da sottosegmenti per il lavoro completato in un secondo momento. Come nel caso dei segmenti, puoi anche inviare un sottosegmento per memorizzare l'inizio del sottosegmento stesso e quindi sovrascriverlo con un sottosegmento completo una volta completata la chiamata a valle.

Di seguito sono elencati i campi obbligatori, o condizionalmente obbligatori, per i sottosegmenti.

> **Note**
>
> I valori devono essere stringhe lunghe fino a un massimo di 250 caratteri, a meno che non sia indicato altrimenti.

Campi del sottosegmento obbligatori

- `id`— Un identificatore a 64 bit per il sottosegmento, unico tra i segmenti della stessa traccia, in 16 cifre esadecimali.
- `name`— Il nome logico del sottosegmento. Nel caso di chiamate a valle, denominare il sottosegmento in base alla risorsa o al servizio chiamati. Nel caso di sottosegmenti personalizzati, denominare il sottosegmento in base al codice che viene analizzato (ad esempio utilizzando un nome di funzione).
- `start_time`— numero che indica l'ora in cui è stato creato il sottosegmento, in secondi a virgola mobile in base all'ora dell'epoca, preciso in millisecondi. Ad esempio `1480615200.010` o `1.480615200010E9`.
- `end_time`— numero che indica l'ora in cui il sottosegmento è stato chiuso. Ad esempio `1480615200.090` o `1.480615200090E9`. Specificare un `end_time` o il valore `in_progress`.

- `in_progress`— valore booleano impostato su `true` anziché specificare an per `end_time` registrare che un sottosegmento è iniziato, ma non è completo. Per ogni richiesta a valle, invia un solo sottosegmento completo e uno o nessun sottosegmento in lavorazione.
- `trace_id`— ID di traccia del segmento principale del sottosegmento. Obbligatorio solo nel caso in cui si invii un sottosegmento in modo separato.

  Formato ID di traccia X-Ray

  Una radiografia `trace_id` è composta da tre numeri separati da trattini. Ad esempio, `1-58406520-a006649127e371903a2de979`. Questo include:

  - Il numero di versione, che è. 1
  - L'ora della richiesta originale in Unix epoch time utilizzando 8 cifre esadecimali.

    Ad esempio, il fuso orario PST delle 10:00 del 1° dicembre 2016 è espresso in secondi o in cifre esadecimali. `1480615200` `58406520`
  - Un identificatore a 96 bit univoco a livello globale per la traccia in 24 cifre esadecimali.

  > **Note**
  >
  > X-Ray ora supporta gli ID di traccia creati utilizzando OpenTelemetry e qualsiasi altro framework conforme alla specifica W3C Trace Context. Un ID di traccia W3C deve essere formattato nel formato X-Ray Trace ID quando viene inviato a X-Ray. Ad esempio, l'ID di traccia W3C `4efaaf4d1e8720b39541901950019ee5` deve essere formattato come `1-4efaaf4d-1e8720b39541901950019ee5` quando si invia a X-Ray. Gli ID di traccia X-Ray includono il timestamp originale della richiesta in Unix epoch time, ma questo non è necessario quando si inviano gli ID di traccia W3C in formato X-Ray.
- `parent_id`— ID del segmento principale del sottosegmento. Obbligatorio solo nel caso in cui si invii un sottosegmento in modo separato. Nel caso di sottosegmenti nidificati, un sottosegmento può avere come padre un segmento o un sottosegmento.
- `type`—`subsegment`. Richiesto solo se si invia un sottosegmento separatamente.

I seguenti campi dei sottosegmenti sono facoltativi.

Campi del sottosegmento facoltativi

- `namespace`— `aws` per le chiamate SDK AWS; `remote` per altre chiamate downstream.

- `http`— `http`oggetto con informazioni su una chiamata HTTP in uscita.
- `aws`— `aws`oggetto con informazioni sulla AWS risorsa downstream chiamata dall'applicazione.
- `error`, `throttle`, `fault`, e `cause` — campi di errore che indicano che si è verificato un errore e che includono informazioni sull'eccezione che ha causato l'errore.
- `annotations`— `annotations`oggetto con coppie chiave-valore che si desidera che X-Ray indicizzi per la ricerca.
- `metadata`— `metadata`oggetto con eventuali dati aggiuntivi che si desidera memorizzare nel segmento.
- `subsegments`— matrice di `subsegment`oggetti.
- `precursor_ids`— matrice di ID di sottosegmento che identifica i sottosegmenti con lo stesso elemento principale completati prima di questo sottosegmento.

Dati sulle richieste HTTP

Utilizza un blocco HTTP per memorizzare i dettagli relativi a una richiesta HTTP che la tua applicazione ha elaborato (in un segmento) oppure che la tua applicazione ha eseguito a valle verso un'API HTTP (in un sottosegmento). La maggior parte dei campi di questo oggetto corrispondono alle informazioni di una richiesta HTTP e della relativa risposta.

`http`

Tutti i campi sono facoltativi.

- `request`— Informazioni su una richiesta.
  - `method`— Il metodo di richiesta. Ad esempio, `GET`.
  - `url`— L'URL completo della richiesta, compilato a partire dal protocollo, dal nome host e dal percorso della richiesta.
  - `user_agent`— La stringa dell'agente utente proveniente dal client del richiedente.
  - `client_ip`— L'indirizzo IP del richiedente. Può essere recuperato dal campo `Source Address` del pacchetto IP o, in caso di richieste inoltrate, da un'intestazione `X-Forwarded-For`.
  - `x_forwarded_for`— (solo segmenti) booleano che indica che `client_ip` è stato letto da un'`X-Forwarded-For`intestazione e non è affidabile in quanto avrebbe potuto essere falsificato.
  - `traced`— (solo sottosegmenti) booleano che indica che la chiamata a valle è verso un altro servizio tracciato. Se questo campo è impostato su`true`, X-Ray considera la traccia interrotta

fino a quando il servizio downstream non carica un segmento con a `parent_id` che corrisponde al sottosegmento che `id` contiene questo blocco.

- `response`— Informazioni su una risposta.
  - `status`— numero intero che indica lo stato HTTP della risposta.
  - `content_length`— numero intero che indica la lunghezza del corpo della risposta in byte.

Quando effettui una chiamata a un'API Web a valle, registra un sottosegmento con informazioni sulla richiesta e sulla risposta HTTP. X-Ray utilizza il sottosegmento per generare un segmento dedotto per l'API remota.

Example Segmento per chiamata HTTP elaborata da un'applicazione in esecuzione su Amazon EC2

```
{
  "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 Sottosegmento per una chiamata HTTP a valle

```
{
  "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 Segmento dedotto per una chiamata HTTP a valle

```
{
  "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
```

```
}
```

Annotazioni

I segmenti e i sottosegmenti possono includere un `annotations` oggetto contenente uno o più campi indicizzati da X-Ray per l'uso con le espressioni di filtro. I campi possono avere valori di tipo stringa, numerico o booleano (senza oggetti o matrici). X-Ray indica fino a 50 annotazioni per traccia.

Example Segmento per chiamata HTTP con annotazioni

```
{
  "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
```

```
    }
  }
```

Per funzionare con i filtri, le chiavi devono essere di tipo alfanumerico. Il trattino basso è ammesso. Altri simboli e gli spazi non sono ammessi.

Metadati

I segmenti e i sottosegmenti possono includere un `metadata` oggetto contenente uno o più campi con valori di qualsiasi tipo, inclusi oggetti e matrici. X-Ray non indicizza i metadati e i valori possono avere qualsiasi dimensione, purché il documento del segmento non superi la dimensione massima (64 kB). Puoi visualizzare i metadati nel documento di segmento completo restituito dall'API `BatchGetTraces`. Le chiavi di campo (`debug`nell'esempio seguente) che iniziano con `AWS.` sono riservate all'uso da parte di SDK e AWS client forniti.

Example Sottosegmento personalizzato con metadati

```
{
  "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 dati relativi alle risorse

Nel caso dei segmenti, l'oggetto `aws` contiene informazioni sulla risorsa su cui è in esecuzione l'applicazione. A una singola risorsa possono essere associati molteplici campi. Ad esempio, un'applicazione in esecuzione in un ambiente Docker multicontainer su Elastic Beanstalk potrebbe contenere informazioni sull'istanza Amazon EC2, sul contenitore Amazon ECS in esecuzione sull'istanza e sull'ambiente Elastic Beanstalk stesso.

`aws` (Segmenti)

Tutti i campi sono facoltativi.

- `account_id`— Se l'applicazione invia segmenti a un altro, registra l'ID dell'account su cui è in esecuzione l'applicazione. Account AWS
- `cloudwatch_logs`— Matrice di oggetti che descrivono un singolo gruppo di CloudWatch log.
  - `log_group`— Il nome del gruppo di CloudWatch log.
  - `arn`— L' CloudWatch ARN del gruppo Log.
- `ec2`— Informazioni su un'istanza Amazon EC2.
  - `instance_id`— L'ID dell'istanza EC2.
  - `instance_size`— Il tipo di istanza EC2.
  - `ami_id`— L'Amazon Machine Image ID.
  - `availability_zone`— La zona di disponibilità in cui è in esecuzione l'istanza.
- `ecs`— Informazioni su un container Amazon ECS.
  - `container`— Il nome host del contenitore.
  - `container_id`— L'ID completo del contenitore.
  - `container_arn`— L'ARN dell'istanza del contenitore.

- `eks`— Informazioni su un cluster Amazon EKS.
  - `pod`— Il nome host del tuo pod EKS.
  - `cluster_name`— Il nome del cluster EKS.
  - `container_id`— L'ID completo del contenitore.
- `elastic_beanstalk`— Informazioni su un ambiente Elastic Beanstalk. Puoi trovare queste informazioni in un file denominato `/var/elasticbeanstalk/xray/environment.conf` sulle più recenti piattaforme Elastic Beanstalk.
  - `environment_name`: il nome dell'ambiente.
  - `version_label`— Il nome della versione dell'applicazione attualmente distribuita sull'istanza che ha fornito la richiesta.
  - `deployment_id`— numero che indica l'ID dell'ultima distribuzione riuscita nell'istanza che ha fornito la richiesta.
- `xray`— Metadati relativi al tipo e alla versione della strumentazione utilizzata.
  - `auto_instrumentation`— Booleano che indica se è stata utilizzata la strumentazione automatica (ad esempio, l'agente Java).
  - `sdk_version`— La versione dell'SDK o dell'agente in uso.
  - `sdk`— Il tipo di SDK.

Example AWS blocco con plugin

```
"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"
   }
}
```

Per i sottosegmenti, registra le informazioni sulle risorse Servizi AWS e sulle risorse a cui l'applicazione accede. X-Ray utilizza queste informazioni per creare segmenti dedotti che rappresentano i servizi a valle nella mappa dei servizi.

`aws` (Sottosegmenti)

Tutti i campi sono facoltativi.

- `operation`— Il nome dell'azione API richiamata contro una risorsa or. Servizio AWS
- `account_id`— Se l'applicazione accede alle risorse di un account diverso o invia segmenti a un account diverso, registra l'ID dell'account proprietario della AWS risorsa a cui l'applicazione ha avuto accesso.
- `region`— Se la risorsa si trova in una regione diversa dall'applicazione, registra la regione. Ad esempio, `us-west-2`.
- `request_id`— Identificatore univoco per la richiesta.
- `queue_url`— Per le operazioni su una coda Amazon SQS, l'URL della coda.
- `table_name`— Per le operazioni su una tabella DynamoDB, il nome della tabella.

Example Sottosegmento per una chiamata a DynamoDB per salvare un elemento

```
{
  "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",
  }
}
```

Errori ed eccezioni

Quando si verifica un errore, puoi memorizzare le informazioni relative all'errore e le eccezioni generate. Memorizza gli errori nei segmenti quando l'applicazione restituisce un errore all'utente, e nei sottosegmenti quando l'errore è restituito da una chiamata a valle.

tipi di errore

Imposta uno o più dei seguenti campi a `true` per indicare che si è verificato un errore. In caso di errori composti è possibile associare molteplici tipi. Ad esempio, un errore `429 Too Many Requests` da una chiamata a valle può far sì che la tua applicazione ritorni un errore `500 Internal Server Error`, nel qual caso possono essere associati tutti e tre i tipi di errore.

- `error`— booleano che indica che si è verificato un errore del client (il codice di stato della risposta era 4XX Client Error).
- `throttle`— booleano che indica che una richiesta è stata limitata (il codice di stato della risposta era 429 Too Many Requests).
- `fault`— booleano che indica che si è verificato un errore del server (il codice di stato della risposta era 5XX Server Error).

Indica la causa dell'errore includendo un oggetto cause nel segmento o nel sottosegmento.

`cause`

A causa può essere un ID di eccezione lungo 16 caratteri o un oggetto con i campi riportati qui di seguito:

- `working_directory`— Il percorso completo della directory di lavoro quando si è verificata l'eccezione.
- `paths`— L'array di percorsi alle librerie o ai moduli in uso quando si è verificata l'eccezione.
- `exceptions`— La matrice di oggetti di eccezione.

Includi informazioni dettagliate sull'errore in uno o più oggetti exception.

`exception`

Tutti i campi sono facoltativi.

- `id`— Un identificatore a 64 bit per l'eccezione, unico tra i segmenti della stessa traccia, in 16 cifre esadecimali.
- `message`— Il messaggio di eccezione.
- `type`— Il tipo di eccezione.
- `remote`— booleano che indica che l'eccezione è stata causata da un errore restituito da un servizio a valle.
- `truncated`— numero intero che indica il numero di stack frame omessi da. `stack`
- `skipped`— numero intero che indica il numero di eccezioni che sono state saltate tra questa eccezione e la relativa eccezione figlia, ovvero l'eccezione che ha causato.
- `cause`— ID dell'eccezione principale dell'eccezione, ovvero l'eccezione che ha causato l'eccezione.
- `stack`— matrice di oggetti StackFrame.

Se disponibili, memorizza le informazioni sullo stack delle chiamate negli oggetti stackFrame.

`stackFrame`

Tutti i campi sono facoltativi.

- `path`— Il percorso relativo al file.
- `line`— La riga del file.
- `label`— Il nome della funzione o del metodo.

Query SQL

Puoi creare o sottosegmenti per le query che la tua applicazione esegue su un database SQL.

`sql`

Tutti i campi sono facoltativi.

- `connection_string`— Per SQL Server o altre connessioni al database che non utilizzano stringhe di connessione URL, registrare la stringa di connessione, escluse le password.
- `url`— Per una connessione al database che utilizza una stringa di connessione URL, registrare l'URL, escluse le password.
- `sanitized_query`— L'interrogazione del database, con tutti i valori forniti dall'utente rimossi o sostituiti da un segnaposto.
- `database_type`— Il nome del motore del database.
- `database_version`— Il numero di versione del motore di database.
- `driver_version`— Il nome e il numero di versione del driver del motore di database utilizzato dall'applicazione.
- `user`— Il nome utente del database.
- `preparation`— `call` se la query utilizzava un`PreparedCall`; `statement` se l'interrogazione utilizzava un`PreparedStatement`.

Example Sottosegmento con una query SQL

```
{
  "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=?;"
  }
}
```

Cronologia dei documenti per AWS X-Ray

La tabella seguente descrive le modifiche importanti alla documentazione per. AWS X-Ray Per ricevere notifiche sugli aggiornamenti di questa documentazione, puoi abbonarti a un feed RSS.

Ultimo aggiornamento della documentazione: 8 febbraio 2023

Modifica	Descrizione	Data
Funzionalità aggiunta	X-Ray ora registra gli eventi relativi ai dati, tra cui `PutTraceSegments` `GetTraceSummaries` , e verso. `BatchGetTraces` AWS CloudTrail X-Ray ora registra anche l'evento di `GetSamplingStatisticSummaries` gestione su. CloudTrail Per ulteriori informazioni, vedere Registrazione delle chiamate dell'API X-Ray con. AWS CloudTrail	7 marzo 2024
Funzionalità aggiunta	X-Ray ora supporta gli ID di traccia creati tramite OpenTelemetry o qualsiasi altro framework conforme alla specifica W3C Trace Context. Per ulteriori informazioni, vedere Invio di dati di traccia a X-Ray.	25 ottobre 2023
Funzionalità aggiunta	Ora puoi configurare il tracciamento attivo di Amazon SNS, che ti consente di tracciare e analizzare le	8 febbraio 2023

	richieste mentre viaggiano tra i tuoi argomenti di Amazon SNS. Per ulteriori informazioni, consulta Amazon SNS e. AWS X-Ray	
Argomento X-Ray SDK aggiornato per Node.js	Sono stati aggiunti dettagli per la strumentazione dei client che utilizzano la versione V3. AWS SDK for JavaScript Per i dettagli, consulta Tracciare le chiamate AWS SDK con l'SDK X-Ray per Node.js.	7 febbraio 2023
Dettagli aggiornati delle policy gestite da IAM	È stata aggiunta l'autorizzazione IAM per l'osservabilità tra account alle `AWSXRayReadOnlyAccess` policy `AWSXrayCrossAccountSharingConfiguration` gestite `AWSXRayFullAccess` e alle policy. Per i dettagli, consulta le politiche gestite da IAM per X-Ray.	7 febbraio 2023
Funzionalità aggiunta	AWS X-Ray ora supporta l'osservabilità tra più account, consentendoti di monitorare e risolvere i problemi delle applicazioni che si estendono su più account all'interno di un unico. Regione AWSPer i dettagli, consulta Tracciamento tra account.	27 novembre 2022

Funzionalità aggiunta	Ora puoi visualizzare le tracce collegate tra i produttori di messaggi, una coda Amazon SQS e i consumatori, fornendo una vista connessa delle tracce inviate da applicazioni basate sugli eventi. Per ulteriori informazioni, consulta la sezione Tracciamento delle applicazioni basate sugli eventi.	20 novembre 2022
Dettagli aggiornati delle policy gestite da IAM	È stata aggiunta l'autorizzazione IAM per l'elenco delle politiche delle risorse nella politica `AWSXRayReadOnlyAccess` gestita. Per i dettagli, consulta le politiche gestite da IAM per X-Ray.	15 novembre 2022
Autorizzazioni aggiornate della console IAM e dettagli delle policy gestite	Il set di autorizzazioni IAM utilizzato dalla console X-Ray è stato aggiornato, insieme alla descrizione `AWSXRayReadOnlyAccess` della policy gestita. Per i dettagli, vedere Uso della console X-Ray.	11 novembre 2022

Aggiunta AWS Distro per Ruby OpenTelemetry	AWS Distro for OpenTelemetry (ADOT) fornisce un unico set di API, librerie e agenti open source per raccogliere tracce e metriche distribuite. ADOT Ruby ti consente di strumentare la tua applicazione Ruby per X-Ray e altri back-end di tracciamento. Per ulteriori informazioni, consulta Distro for Ruby.AWS OpenTelemetry	7 febbraio 2022
Funzionalità aggiunta	È ora possibile visualizzare le tracce e configurare X-Ray dalla CloudWatch console. Per ulteriori informazioni, vedere Console X-Ray.	24 gennaio 2022
RUM integrato CloudWatch	Con AWS X-Ray and CloudWatch RUM, è possibile analizzare ed eseguire il debug del percorso della richiesta a partire dagli utenti finali dell'applicazione tramite servizi AWS gestiti a valle. Per ulteriori informazioni, consulta CloudWatch RUM e. AWS X-Ray	3 dicembre 2021

AWS Distro integrato per OpenTelemetry	The AWS Distro for OpenTelemetry (ADOT) fornisce un unico set di API, librerie e agenti open source per raccogliere tracce e metriche distribuite. ADOT ti consente di strumentare la tua applicazione per X-Ray e altri back-end di tracciamento. Per ulteriori informazioni, consulta Instrumenting your app.	23 settembre 2021
Funzionalità aggiunta	AWS X-Ray ora si integra con Amazon Virtual Private Cloud, consentendo alle risorse del tuo Amazon VPC di comunicare con il servizio X-Ray senza passare attraverso la rete Internet pubblica. Per ulteriori informazioni, consulta Utilizzo AWS X-Ray con endpoint VPC.	20 maggio 2021
Funzionalità aggiunta	AWS X-Ray ora si integra con AWS CloudFormation, consentendoti di fornire e configurare le risorse X-Ray. Per ulteriori informazioni, vedere Creazione di risorse X-Ray con. CloudFormation	6 maggio 2021

Funzionalità aggiunta	AWS X-Ray ora si integra con Amazon EventBrid ge per tracciare gli eventi che vengono EventBridge trasmessi. Ciò fornisce agli utenti una visione più completa del proprio sistema. Per ulteriori informazioni, consulta Amazon EventBridge e AWS X-Ray.	2 marzo 2021
È stato aggiunto un demone a ECR	Il daemon può ora essere scaricato da Amazon ECR. Per ulteriori informazioni, consulta Download del demone.	1 marzo 2021
Funzionalità aggiunta	AWS X-Ray ora supporta le notifiche relative agli approfondimenti su Amazon EventBridge. Ciò ti consente di intraprendere azioni automatic he sugli approfondimenti utilizzando EventBridge. Per ulteriori informazioni, consulta Notifiche Insights.	15 ottobre 2020
Aggiunti demoni scaricabili	AWS X-Ray introduce il demone di supporto per Linux ARM64. Per ulteriori informazi oni, vedere daemonbrazil ws AWS X-Ray	1 ottobre 2020

Funzionalità aggiunta	AWS X-Ray ora supporta l'integrazione attiva con Amazon CloudWatch Synthetics. Ciò consente di visualizzare i dettagli su un nodo client Synthetics Canary, come il tempo di risposta e lo stato. Puoi anche eseguire analisi nella console Analytics sulla base delle informazioni provenienti da un nodo client Synthetics Canary. Per ulteriori informazioni, consulta Debugging CloudWatch synthetics canaries utilizzando X-Ray.	24 settembre 2020
Funzionalità aggiunta	AWS X-Ray ora supporta end-to-end AWS Step Functions i flussi di lavoro di tracciamento per. È possibile visualizzare i componenti della macchina a stati, identificare i punti deboli nelle prestazioni e risolvere le richieste che hanno provocato un errore. Per ulteriori informazioni, vedere e.AWS Step Functions AWS X-Ray	14 settembre 2020

Funzionalità aggiunta	AWS X-Ray introduce approfondimenti per analizzare continuamente i dati di tracciamento nel tuo account per identificare problemi emergenti nelle tue applicazioni. Insights registra gli incidenti e monitora l'impatto degli incidenti fino alla risoluzione. Per ulteriori informazioni, consulta Utilizzo di Insights nella console AWS X-Ray	3 settembre 2020
Funzionalità aggiunta	AWS X-Ray introduce l'agente di strumentazione automatica a Java, che consente ai clienti di raccogliere dati di traccia senza dover modificare l'applicazione esistente basata su Java. È ora possibile tracciare le applicazioni Java Web e basate su servlet con modifiche minime alla configurazione e senza modifiche al codice. Per ulteriori informazioni, vedere AWS X-Ray Auto Instrumentation Agent for Java.	3 settembre 2020

Funzionalità aggiunta	AWS X-Ray ha aggiunto una nuova pagina Gruppi alla console X-Ray per facilitare la creazione e la gestione di gruppi di tracce. Per ulteriori informazioni, vedere Configurazione dei gruppi nella console X-Ray.	24 agosto 2020
Funzionalità aggiunta	AWS X-Ray ora consente di aggiungere tag ai gruppi e alle regole di campionamento. Puoi anche controllare l'accesso ai gruppi e le regole di campionamento in base ai tag. Per ulteriori informazioni, vedere Etichettatura delle regole e dei gruppi di campionamento a raggi X e Gestione dell'accesso ai gruppi di raggi X e alle regole di campionamento basate sui tag.	24 agosto 2020

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.