



このコンテンツは公開から3年以上経過しており内容が古い可能性があります
最新情報については[サービス別資料](#)もしくはサービスのドキュメントをご確認ください

[AWS Black Belt Online Seminar]

Amazon CloudWatch

サービスカットシリーズ

アマゾンウェブサービスジャパン株式会社
ソリューションアーキテクト 三上 卓也
2019/3/26

AWS 公式 Webinar
<https://amzn.to/JPWebinar>



過去資料
<https://amzn.to/JPArchive>



自己紹介

三上 卓也 (みかみ たくや)

ゲームエンターテイメントソリューション部 / ソリューションアーキテクト

普段の業務

主にゲーム企業のお客様を担当し、お客様のAWS活用を様々な形でサポート

好きなAWSサービス

- AWS Management Tools
- Amazon QuickSight



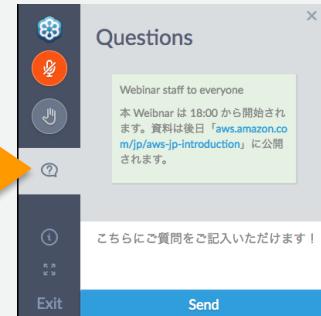
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、Amazon ウェブ サービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- 今後のロードマップに関するご質問はお答えできませんのでご了承下さい

- ① 吹き出しをクリック
- ② 質問を入力
- ③ Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2019年3月26日時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<http://aws.amazon.com>)にてご確認ください。
- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。
- 価格は税抜表記となっています。日本居住者のお客様が東京リージョンを使用する場合、別途消費税をご請求させていただきます。
- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at <http://aws.amazon.com/agreement/>. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

本日のアジェンダ

- Amazon CloudWatch
- CloudWatch Metrics
- CloudWatch Alarms
- CloudWatch Logs
- CloudWatch Logs Insights
- CloudWatch Dashboards
- CloudWatch Events
- 料金
- まとめ

Amazon CloudWatch



Amazon CloudWatch

Amazon CloudWatch はAWS
リソース、アプリケーション、
オンプレミスのモニタリング
サービス



モニタリング



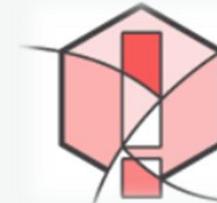
監視の集約



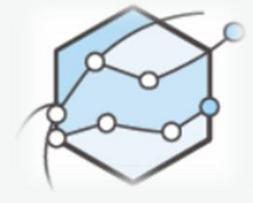
トラブルシュート



ログの分析



自動アクション



運用状況の把握

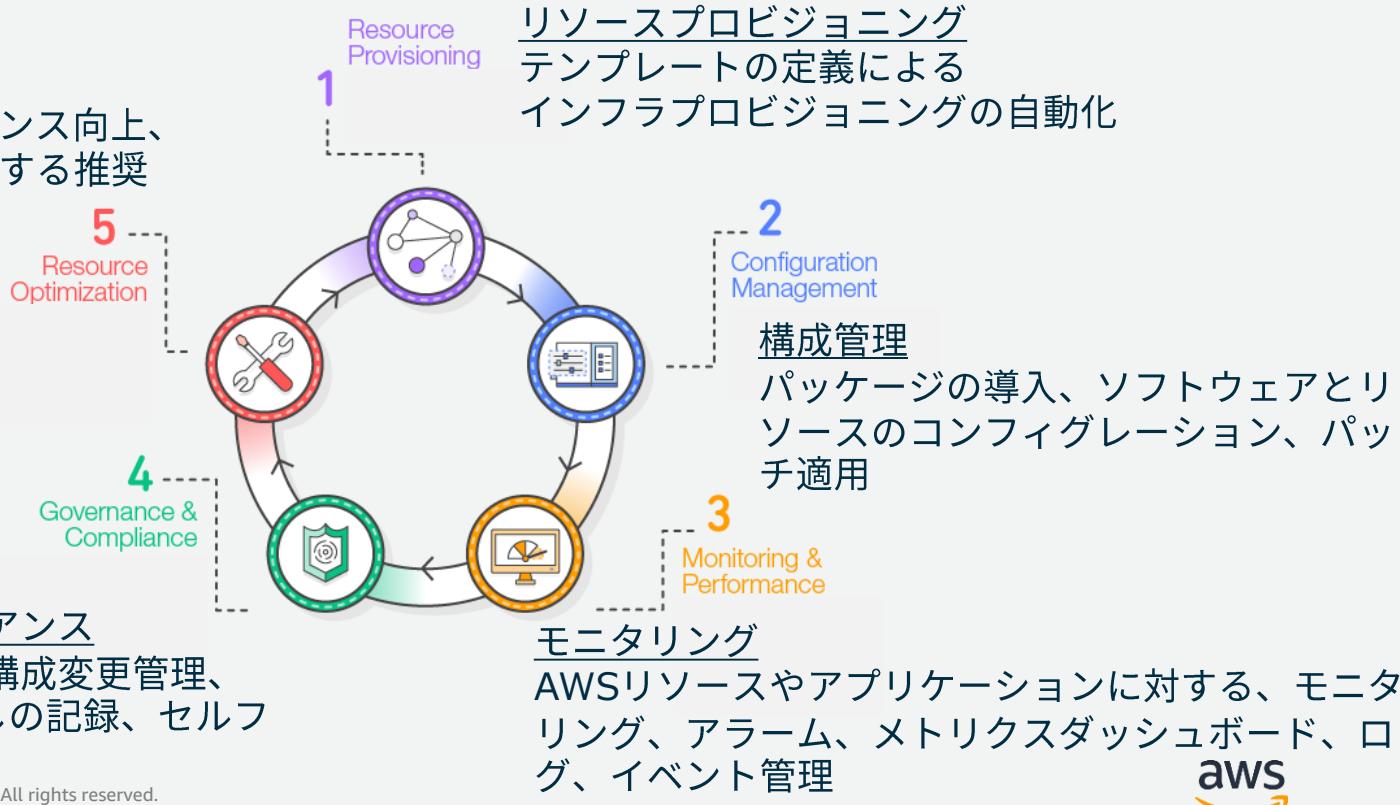
**“Amazon CloudWatch では、1ヶ月で、800兆を
超えるメトリクスの監視、2兆を超えるイベント
のトリガー、50ペタバイトを超えるログの収集を
しています(2018年3月時点)”**

Amazon CloudWatchの位置づけ

AWS Management Tools

リソース最適化

コスト低減、パフォーマンス向上、セキュリティの改善に対する推奨事項の自動提供



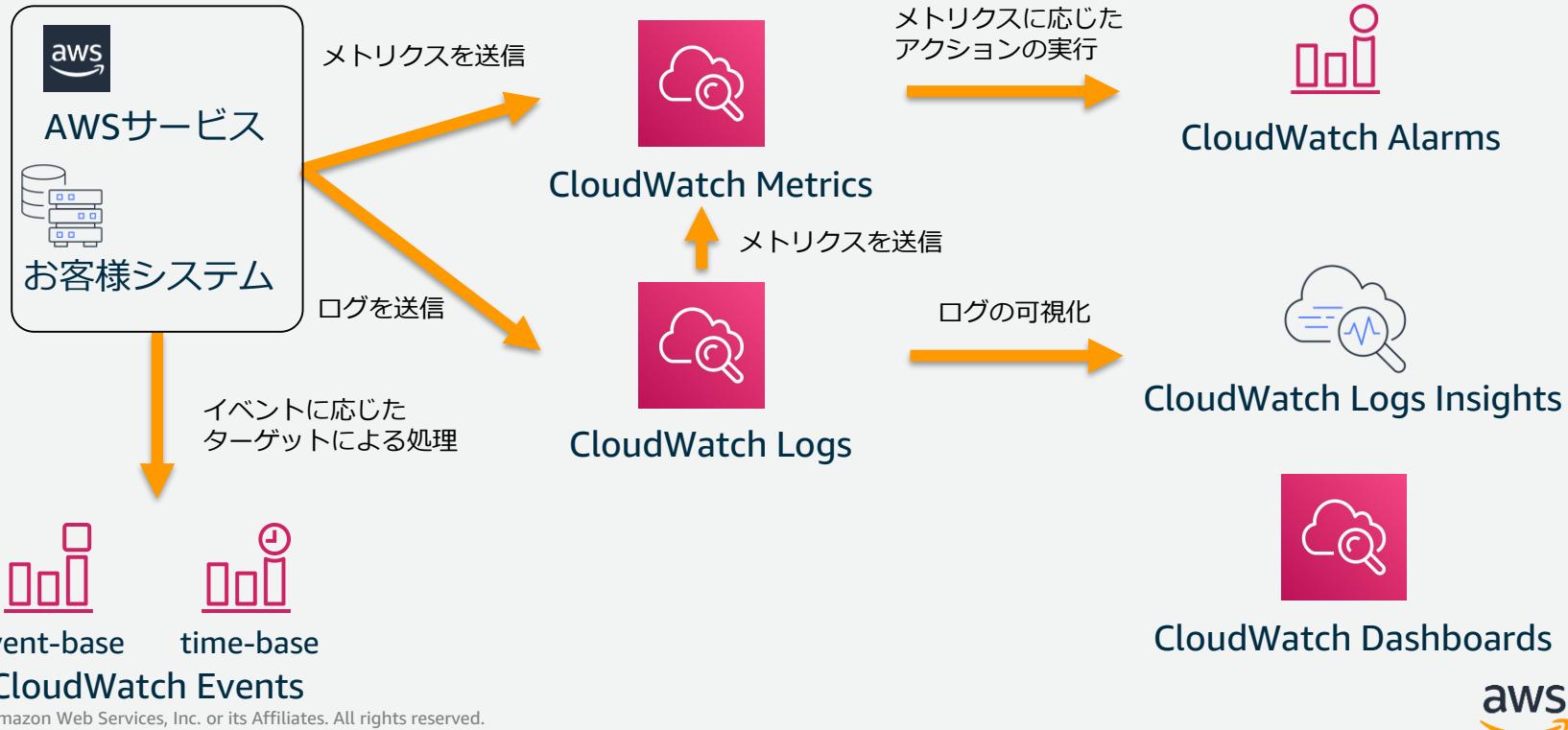
Amazon CloudWatchの位置づけ

AWS Management Tools



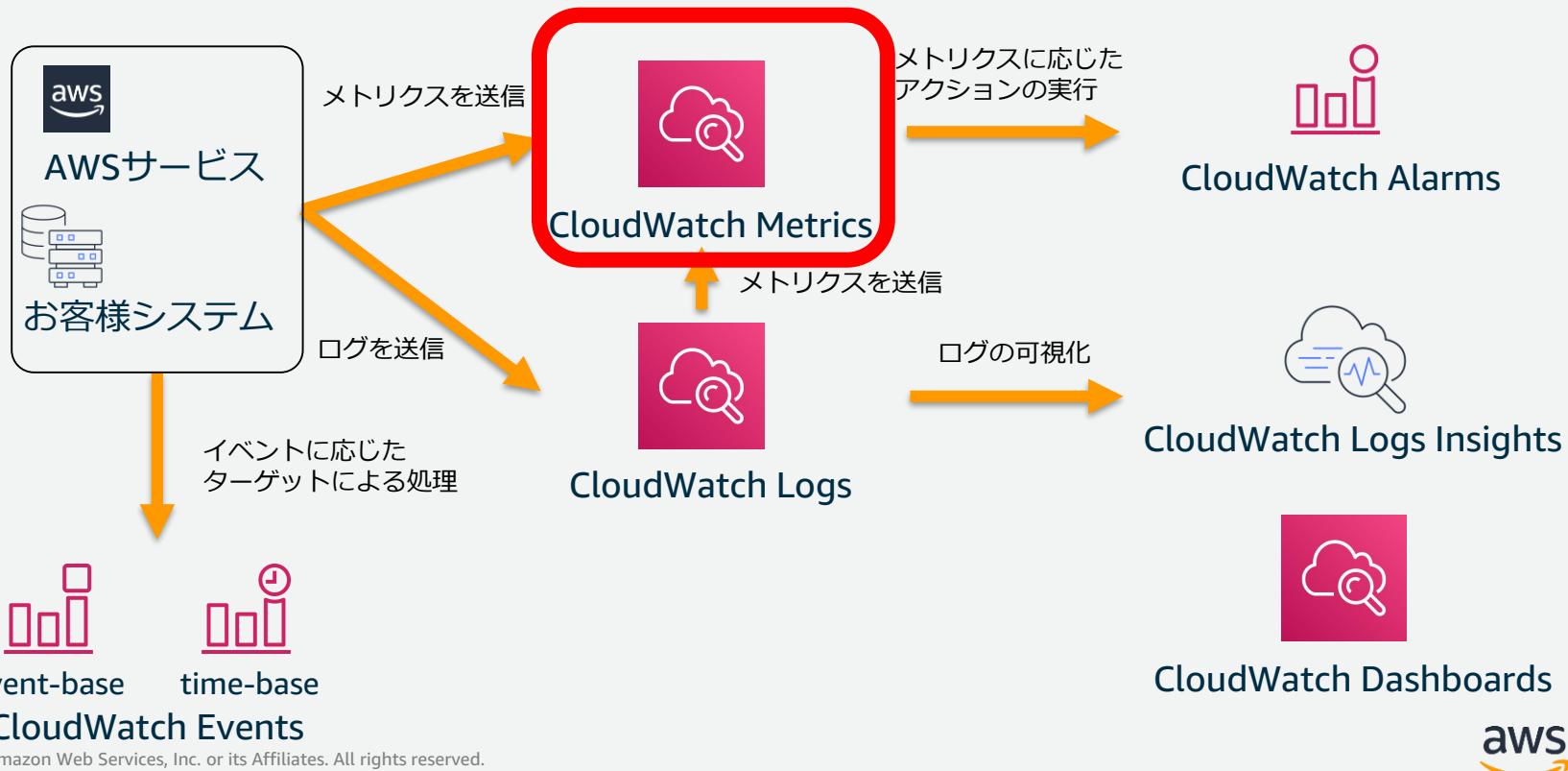
Amazon CloudWatch の概要

CloudWatch はモニタリングに関する様々な機能を提供



CloudWatch Metrics

CloudWatch に発行されたメトリクスを収集し、統計を取得



CloudWatch Metrics

メトリクス

- CloudWatchに発行された時系列のデータポイントのセット
- メトリクス名を持つ
- データポイントはタイプスタンプと測定単位を保持
- メトリクスは作成されたリージョンにのみ存在



名前空間

- CloudWatchメトリクスのコンテナ
- 異なる名前空間のメトリクスは相互に切り離される
- AWSサービスでは AWS/<service>が使用される(例: AWS/EC2)

ディメンション

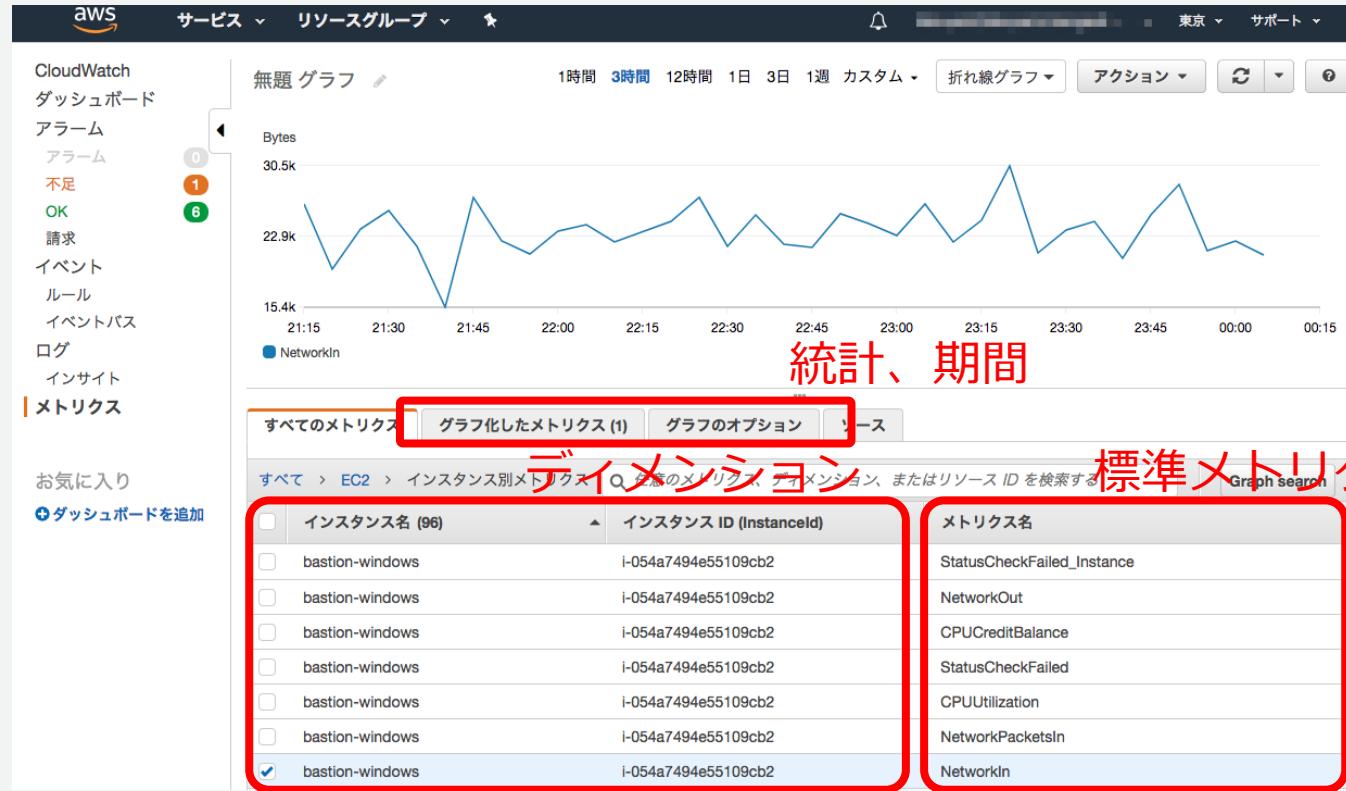
- メトリクスを一意に識別する名前/値のペア(例 : InstanceId=i-12345678)

CloudWatch の利用イメージ



CloudWatch の利用イメージ

ディメンション、メトリクス、統計、期間を選択して任意のグラフを表示



統計、期間の利用イメージ

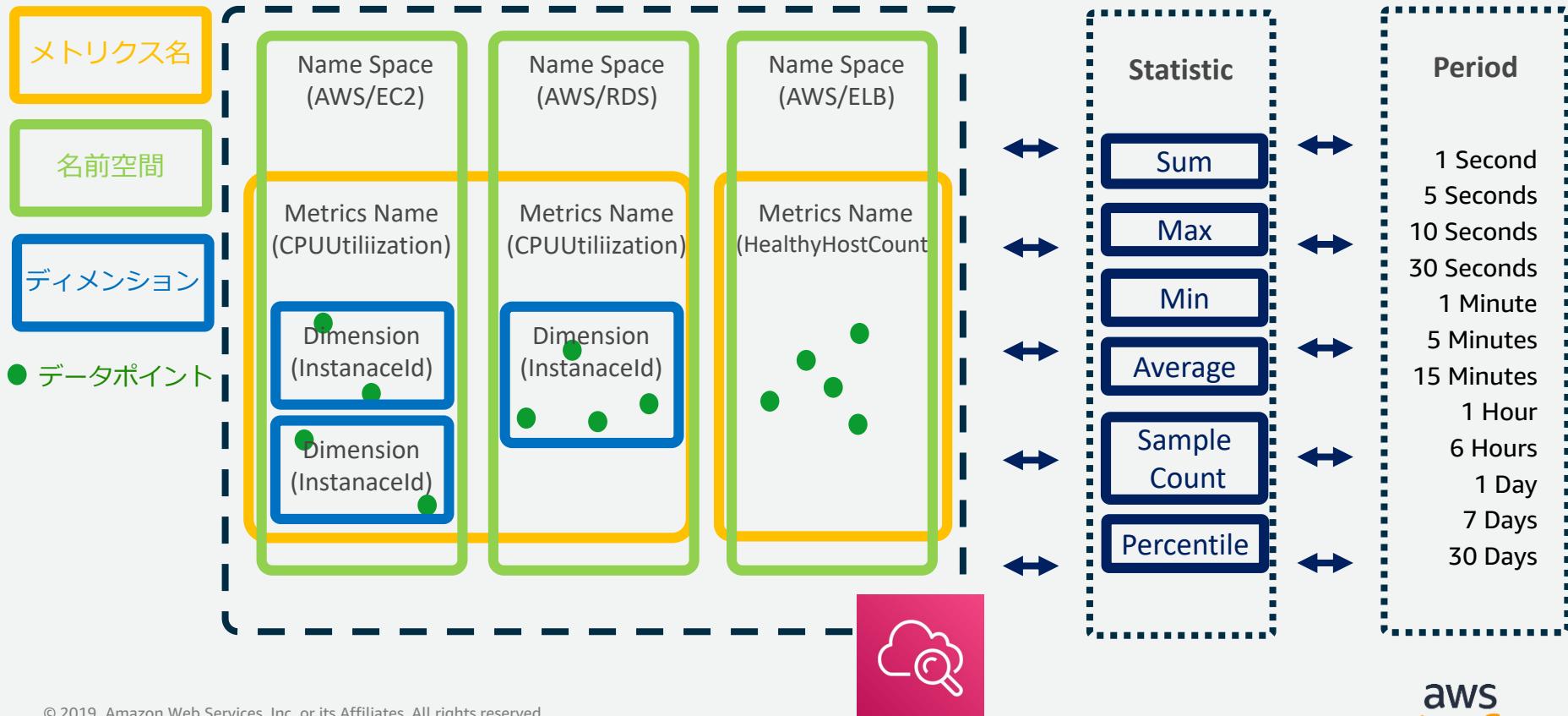


統計、期間の利用イメージ

タイムゾーン、統計情報、統計の粒度を設定し表示



CloudWatch Metrics



CloudWatch のメトリクス値

メトリクスデータの生成

- 基本は1分、カスタムメトリクスの高解像度を利用して最短で1秒
- EC2では基本モニタリングで5分、詳細モニタリングで1分ごとにメトリクスが生成 ※1

メトリクスデータの使用期間

- 粒度で使用期間が決まる
- 1分未満：3時間、1分：15日、5分：63日
- 1時間のデータポイントだと15ヶ月(1年前のイベントとの比較に有用)

	高解像度	標準の解像度		
取得粒度	1分未満	1分～	5分～	1時間～
使用期間	3時間	15分	63日	15ヶ月

※1 メトリクスの取得間隔はサービスやメトリクスにより異なります。詳しくは各サービスのドキュメントをご確認ください。

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/aws-services-cloudwatch-metrics.html

CloudWatch のメトリクス値

CloudWatchで取得される情報は統計情報

- メトリクスデータを指定した期間で集約、統計情報を表示

統計	説明
Minimum	指定された期間に認められた最小値です。この値を用いて、アプリケーションの低ボリュームのアクティビティを判断できます。
Maximum	指定された期間に認められた最大値です。この値を用いて、アプリケーションの高ボリュームのアクティビティを判断できます。
Sum	該当するメトリクスで加算されたすべての合計値です。この統計は、メトリクスの合計ボリュームを判断するのに役立ちます。
Average	指定した期間の Sum/SampleCount の値です。この統計を Minimum および Maximum と比較することで、メトリクスの全容、および平均使用量がどれくらい Minimum と Maximum に近いかを判断できます。この比較は、必要に応じてリソースを増減させるべきかを知るために役立ちます。
SampleCount	統計計算で使用するデータポイントのカウント(数)です。
pNN.NN	指定されたパーセンタイルの値。小数点以下最大 2 衔を使用して、任意のパーセンタイルを指定できます(p95.45など)。パーセンタイル統計は負の値を含むメトリクスに対して使用することはできません。詳細については、「 パーセンタイル 」を参照してください。

Metric Math

CloudWatch メトリクスに数式を使用して、新しいメトリクスを作成

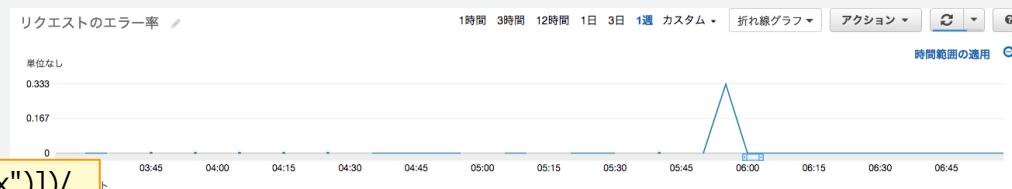
- METRICS関数、基本的な算術関数をはじめとした関数をサポート
- メトリクスに[ID]フィールドを設定し、関数で利用

コマンド	説明	例
AVG	データポイントの平均を表すスカラー	メトリクスの平均値 : AVG(METRICS())
SUM	データポイントの合計値を表すスカラー	メトリクスm1,m2の合計値 : SUM([m1,m2])
METRICS	CloudWatch メトリクスを表す	メトリクスreqall : METRICS("reqall")

etc

ユースケース
全リクエストのうち4XX,5XX
レスポンスの割合を表示

$\text{SUM}([\text{METRICS}(\text{"res4xx"}), \text{METRICS}(\text{"res5xx"})]) / \text{SUM}(\text{METRICS}(\text{"reqall"}))$



メトリクスに[ID]フィールドを指定

Id	ラベル	詳細	統計	期間	Y軸	アクション
calc	4xx/5xx レート	$\text{SUM}([\text{METRICS}(\text{"res4xx"}), \text{METRICS}(\text{"res5xx"})]) / \text{SUM}(\text{METRIC...})$				
res5xx	HTTPCode_ELB_5XX_Count	ApplicationELB * HTTPCode_ELB_5XX_Count * LoadBalancer: app...	平均	5 分		
res4xx	HTTPCode_ELB_4XX_Count	ApplicationELB * HTTPCode_ELB_4XX_Count * LoadBalancer: app...	平均	5 分		
reqall	RequestCount	ApplicationELB * RequestCount * LoadBalancer: app/test-ne-alb/3...	合計	5 分		

アラームを設定可能

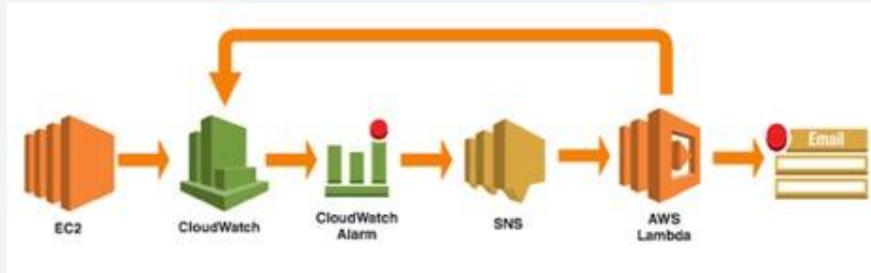
スナップショットグラフ

GetMetricWidgetImage APIによりグラフのPNG画像が取得可能

- 利用方法のサンプルがGitHubリポジトリで公開
- CloudWatch コンソールでメトリクスを表示した際、“Source” タブに表示される JSON 形式のパラメータをコピーして活用可能

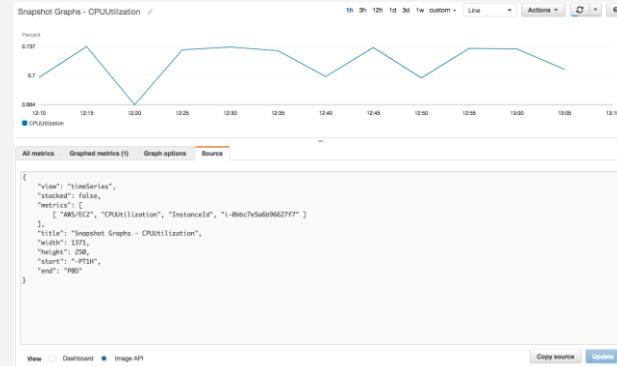
ユースケース

- Wiki、チケットシステム、チャットアプリ等への連携
- CloudWatch Alarmsと組み合わせて異常通知にグラフを添付する



aws-cloudwatch-snapshot-graphs-alert-context
<https://github.com/aws-samples/aws-cloudwatch-snapshot-graphs-alert-context>

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



メトリクスの収集

多くのAWSサービスでメトリクスを標準で発行

サービス	名前空間	サービス	名前空間
Amazon API Gateway	AWS/ApiGateway	AWS IoT	AWS/IoT
AppStream 2.0	AWS/AppStream	AWS IoT Analytics	AWS/IoTAnalytics
AWS Billing and Cost Management	AWS/Billing	AWS IoT Things Graph	AWS/ThingsGraph
Amazon CloudFront	AWS/CloudFront	AWS Key Management Service	AWS/KMS
Amazon CloudSearch	AWS/CloudSearch	Amazon Kinesis Data Analytics	AWS/KinesisAnalytics
Amazon CloudWatch Events	AWS/Events	Amazon Kinesis Data Firehose	AWS/Firehose
Amazon CloudWatch Logs	AWS/Logs	Amazon Kinesis Data Streams	AWS/Kinesis
AWS CodeBuild	AWS/CodeBuild	Amazon Kinesis ビデオストリーム	AWS/KinesisVideo
Amazon Cognito	AWS/Cognito	AWS Lambda	AWS/Lambda
Amazon Connect	AWS/Connect	Amazon Lex	AWS/Lex
AWS Database Migration Service	AWS/DMS	Amazon Machine Learning	AWS/ML
AWS Direct Connect	AWS/DX	Amazon Managed Streaming for Kafka	AWS/Kafka
Amazon DynamoDB	AWS/DynamoDB	Amazon MQ	AWS/AmazonMQ
Amazon EC2	AWS/EC2	Amazon Neptune	AWS/Neptune
Amazon EC2 スポットフリート	AWS/EC2Spot	AWS OpsWorks	AWS/OpsWorks
Amazon EC2 Auto Scaling	AWS/AutoScaling	Amazon Polly	AWS/Polly
AWS Elastic Beanstalk	AWS/ElasticBeanstalk	Amazon Redshift	AWS/Redshift
Amazon Elastic Block Store	AWS/EBS	Amazon Relational Database Service	AWS/RDS
Amazon Elastic Container Service	AWS/ECS	Amazon Route 53	AWS/Route53
Amazon Elastic File System	AWS/EFS	Amazon SageMaker	AWS/SageMaker
Amazon Elastic Inference	AWS/ElasticInference	AWS Shield アドバンスド	AWS/DDoSProtection
Elastic Load Balancing	AWS/ApplicationELB	Amazon Simple Email Service	AWS/SES

etc

詳細はこれら : https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/aws-services-cloudwatch-metrics.html

EC2のメトリクス収集

標準メトリクス(EC2)

- CPUUtilization
- CPUTripBalance
- CPUTripUsage
- CPUSurplusTripBalance
- CPUSurplusCreditsCharged
- DiskReadBytes • DiskReadOps
- DiskWriteBytes • DiskWriteOps
- NetworkOut • NetworkPacketsOut
- NetworkIn • NetworkPacketsIn
- StatusCheckFailed_Instance
- StatusCheckFailed
- StatusCheckFailed_System

カスタムメトリクス

標準メトリクスでは
収集できないメトリクス



Amazon EC2



CloudWatch カスタムメトリクス

標準メトリクス以外の独自メトリクスも監視可能

- AWS CLIの"put-metric-data"もしくは"PutMetricData" APIでデータを登録
- 詳細度が 1秒のデータを含む高解像度なメトリクスを発行可能
→1分未満のアクティビティを迅速に把握

```
$ aws cloudwatch put-metric-data --metric-name RequestLatency  
  --namespace "GetStarted"  
  --timestamp 2014-10-28T12:30:00Z  
  --value 87  
  --unit Milliseconds
```

←単一値の登録

```
$ aws cloudwatch put-metric-data --metric-name RequestLatency  
  --namespace "GetStarted"  
  --timestamp 2014-10-28T12:30:00Z  
  --statistic-value Sum=60,Minimum=15,Maximum=105,SampleCount=5
```

←統計セットの登録

APIコール時のスロットリング対策 (上限緩和申請も検討)

- 単一のput-metric-dataに複数のデータポイントを入れる
- StaticSetを利用する
- リトライ処理をする

CloudWatch で利用できるエージェント/プロトコル

統合CloudWatch エージェント

- 現在の推奨エージェント
- メトリクスとログの両方を単一のエージェントで収集
- クラウドでもオンプレミスでも利用可能
- Linux, Windowsの両方で稼働

以前のCloudWatch Logs エージェント

- EC2 インスタンスのログデータを発行

StatsD のサポート

- StatsD プロトコルを使用して、カスタムメトリクスを取得(Linux,Windows)
- 以下のStatsD 形式をサポート

```
MetricName:value|type|@sample_rate|#tag1: value, tag1...
```

collectd のサポート

- collectd プロトコルを使用してカスタムメトリクスを取得(Linux)
- collectd ソフトウェアを使用して CloudWatch エージェントにメトリクスを送信

procstatプラグイン

- 個別のプロセスからメトリクスを収集
- CloudWatchエージェントで設定

統合CloudWatch エージェント

EC2やオンプレミスサーバにインストールすることでメトリクスを収集

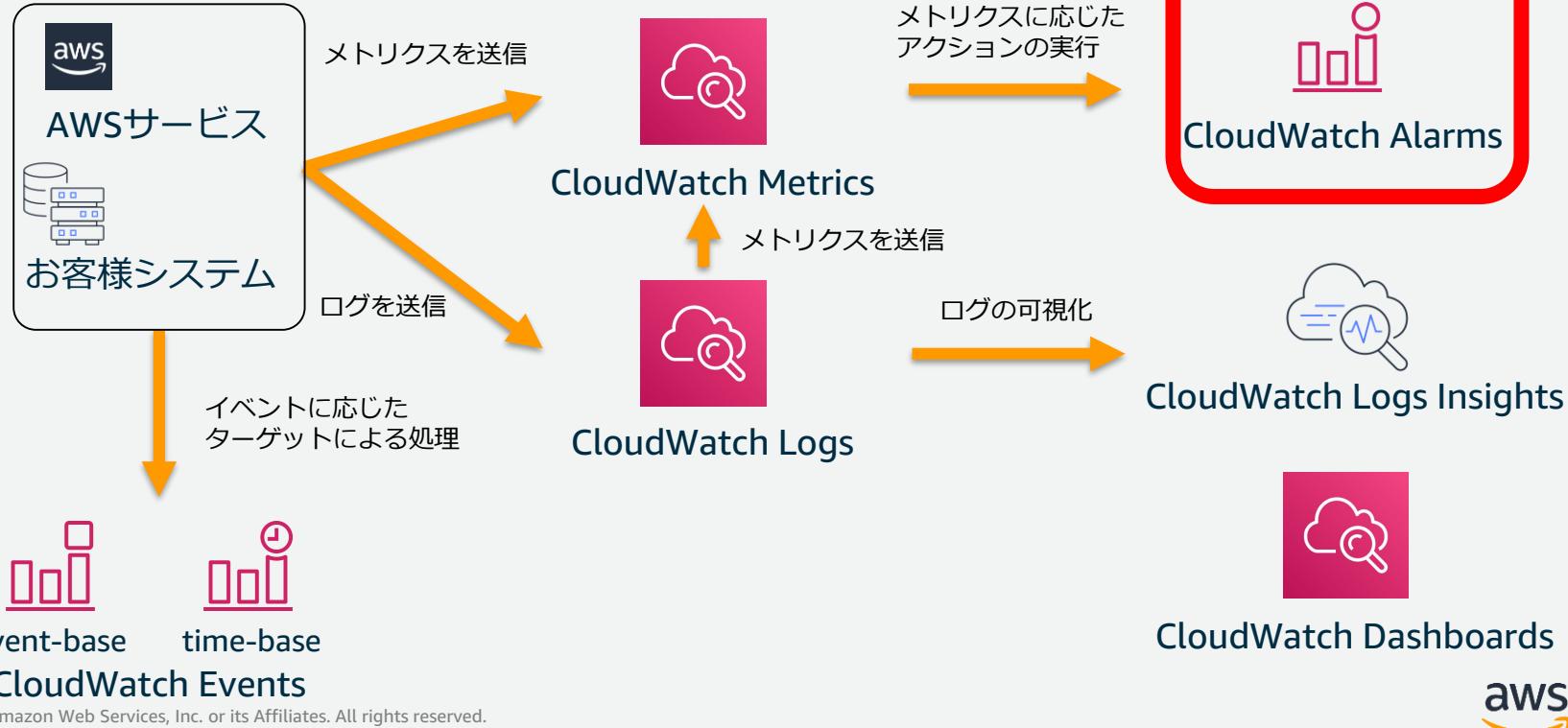
- 手動もしくはウィザードにより設定が可能
- agent、metrics、logs の 3つのセクションを持つ JSON ファイルを設定

```
{  
  "agent": {  
    "metrics_collection_interval": 10,  
    "logfile": "/opt/aws/amazon-cloudwatch-agent/logs/amazon-cloudwatch-agent.log"  
  },  
  "metrics": {  
    "metrics_collected": {  
      "mem": {  
        "measurement": [  
          "mem_used"  
        ],  
        "metrics_collection_interval": 1  
      },  
      "net": {  
        "resources": [  
          ---省略---  
        ]  
      }  
    },  
    "logs": {  
      ---省略---  
      "file_path": "/opt/aws/amazon-cloudwatch-agent/logs/test.log",  
      "log_group_name": "test.log",  
      "log_stream_name": "test.log",  
      "timezone": "Local"  
      ---省略---  
      "log_stream_name": "my_log_stream_name",  
      "force_flush_interval": 15  
    }  
  }  
}
```

- ✓ ファイルの格納パス
- ✓ データの取得頻度
- ✓ 取得対象
- ✓ 名前空間やログストリーム等の CloudWatchに関する情報

CloudWatch Alarms

CloudWatch メトリクスを監視するアラームと関連サービスのアクション



CloudWatch Alarms

- CloudWatch Metricsをモニタリングしてアラームを発行可能
- 条件を指定して、自動アクションを実行が可能
 - 例えば、CPU利用率が80%を超えた時にアラートメールを通知等
- アラームの状態はOK, ALARM, INSUFFICIENT_DATAの3種類

OK

ALARM

INSUFFICIENT_DATA

正常値

- ✓ 定義された閾値を下回っている

異常値

- ✓ 定義された閾値を上回っている

判定不能

- ✓ データ不足のため状態を判定できない
- ✓ CloudWatch特有

CloudWatch Alarms

- CloudWatch Metricsをモニタリングしてアラームを発行可能
- 条件を指定して、自動アクションを実行が可能
 - 例えば、CPU利用率が80%を超えた時にアラートメールを通知等
- アラームの状態はOK, ALARM, INSUFFICIENT_DATAの3種類

OK

ALARM

INSUFFICIENT_DATA

正常値

- ✓ 定義された閾値を下回っている

異常値

- ✓ 定義された閾値を上回っている

判定不能

- ✓ データ不足のため状態を判定できない
- ✓ CloudWatch特有

- データポイントとはCloudWatchに送信される値(CPU値など)
- OK / アラーム時は入力されたデータポイントを基準に状態評価
- INSUFFICIENT_DATA時はCloudWatchにデータポイントの入力が十分に無い状態
→ "INSUFFICIENT_DATA"は必ずしも障害を表すステータスではない

データ欠落時の処理

データ欠落時の処理

- 接続が失われた場合、サーバーがダウンした場合、断片的にのみデータがある場合などの、欠落データポイントの評価方法を指定可能

オプション	評価方法
missing	このアラームは、状態を変更するかどうかを評価する際に、欠落データポイントを考慮に入れません。デフォルト設定。
notBreaching	欠落データポイントはしきい値内として処理
breaching	欠落データポイントはしきい値超過として処理
ignore	現在のアラーム状態が維持

ユースケース

- EC2のCPUUtilizationにbreachingを設定：継続的にデータを報告しているメトリクスの場合は、問題が発生していることを表すため
- DynamoDBのThrottledRequestsにnotBreachingを設定：スロットリング発生時のみデータポイントを生成するため

CloudWatchアラームの設定

AWS サービス EC2 CloudWatch CloudTrail Config S3 編集 バージニア北部 サポート

EC2 ダッシュボード イベント タグ レポート 制限 インスタンス インスタンス スポットリクエスト リザーブドインスタンス イメージ AMI バンドルタスク ELASTIC BLOCK STORE ポリューム スナップショット ネットワーク & セキュリティ セキュリティグループ Elastic IP プレイスマ投注ループ

インスタンスの作成 接続 アクション

タグや属性によるフィルタ、またはキーワードによる検索

Name	インスタンス ID	インスタンスタイプ	アベイラビリティゾーン	インスタンスの状態	ステータスチェック	パブリック IP	プライベート IP アドレス
cwl-sqlserver	i-0cbe78721	m3.medium	us-east-1c	running	2/2 のチェックに合格しました	54.172.19.212	10.0.0.77
TestEc2AmazonLinux	i-0d5e1f1	t2.micro	us-east-1b	running	2/2 のチェックに合格しました	52.1.185.57	172.0.1.135
CloudTrail CWL	i-0f901f9	t2.micro	us-east-1c	running	2/2 のチェックに合格しました	54.88.140.117	10.0.1.104
web-td-001	i-f56c2e22	t2.micro	us-east-1b	running	2/2 のチェックに合格しました	52.4.74.227	172.31.63.209

インスタンス: i-0cbe78721 (cwl-sqlserver) Elastic IP: 54.172.19.212

説明 ステータスチェック モニタリング タグ

CloudWatch アラーム: OK の 1 の 1 アラームの作成

CloudWatch メトリックス: 基本モニタリング。詳細モニタリングを有効化 次のデータを表示: 過去 1 時間

以下は、選択されたリソースの CloudWatch メトリックスです (最大 10)。画面を拡大するには、グラフをクリックします。すべての時刻は協定世界時 (UTC) で表示されています。> すべての CloudWatch メトリックスを表示

CPU 使用率 (%) (パーセント)	ディスク読み取り (Bytes)	ディスク読み取り操作 (操作)	ディスク書き込み (Bytes)
25 20	1 0.75	1 0.75	1 0.75

CloudWatchアラームの設定

AWS メニュー: EC2 CloudWatch CloudTrail Config S3 編集 バージニア北部 サポート

EC2 ダッシュボード イベント タグ レポート 制限 インスタンス インスタンス スポットリクエスト リザーブドインスタンス イメージ AMI バンドルタスク ポリューム スナップショット セキュリティグループ Elastic IP プレイスマッチング

アラームの作成

CloudWatch アラームを使用すると、メトリックスデータがお客様の設定したレベルに達したときに、自動的に通知されます。アラームを編集するには、まず通知先を選択してから、通知を送信するタイミングを設定します。

通知の送信先: 手動でトピック名を入力... キャンセル
受信者: awsAccount@domain.com

アクションを実行: このインスタンスを復元する
このインスタンスを停止する
このインスタンスを終了する

次の時: 平均 / CPU 使用率(%)
状況: >= 80 パーセント
最低発生数: 3 度次の間隔で発生 5 分

CPU 使用率(%) パーセント

アラームの作成

過去 1 時間

アラームの作成

CPU 使用率(%) (パーセント) ディスク読み取り (Bytes) ディスク読み取り操作 (操作) ディスク書き込み (Bytes)

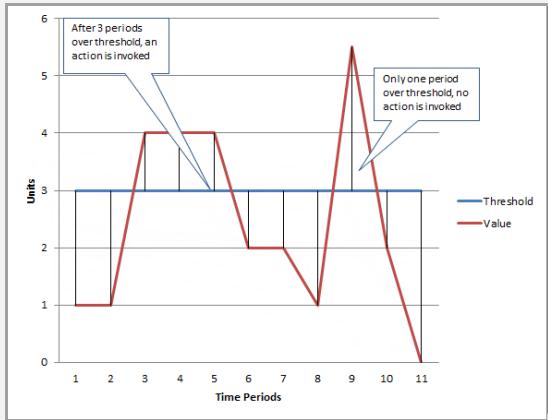
aws

M out of N (N 個中 M 個) のアラーム設定

アラームを発生させるデータポイント数を設定可能

評価期間：アラームの状態を決定するまでに要する直近の期間（データポイント）の数

Datapoints to Alarm : アラームが ALARM 状態に遷移するために超過する必要がある評価期間内のデータポイントの数



※例 : [評価期間] および [Datapoints to Alarm] はどちらも 3

ユースケース

短時間で変化が大きいメトリクスで誤報を抑制

アラーム詳細

アラームの詳細としきい値を指定します。適切なしきい値を設定するには、グラフを参照してください。

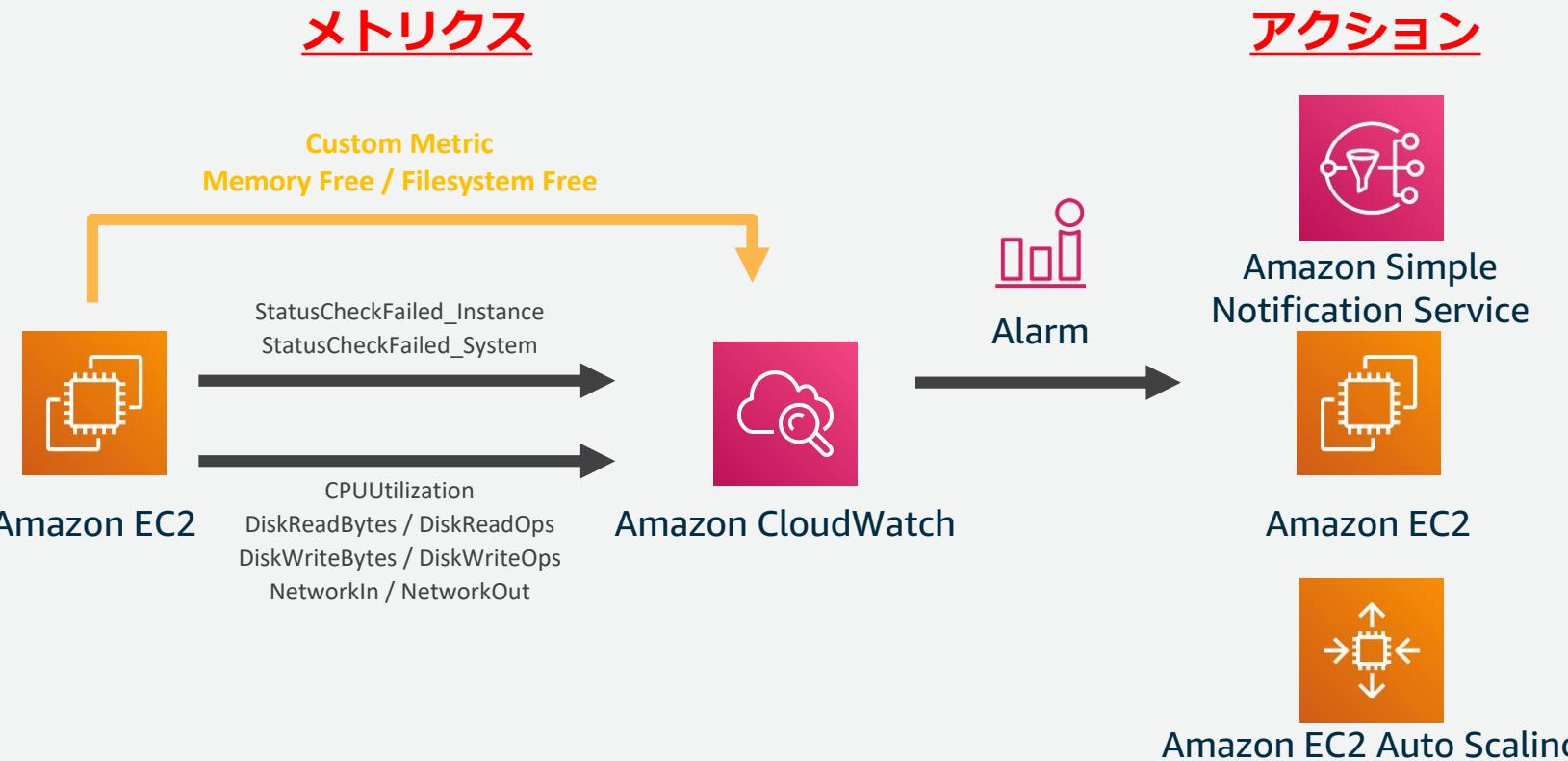
名前:

説明:

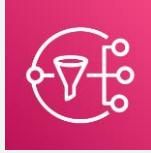
次の時: CPUUtilization (CPUUtilization)
が: 0

期間: / データポイント

CloudWatch Alarmsのアクション機能



ユースケース：アクションの設定例



Amazon Simple
Notification Service

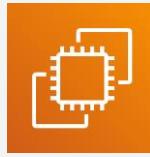
- SNS トピックを追加して、アラームの状態が変わったときにトピックを発行
- サブスクリーブにEメールを選択して、指定アドレスにメール通知
- **サブスクリーブにLambda 関数を選択して、処理を実行**

例：GetMetricWidgetImage APIによるグラフ送付



1. CloudWatch MetricsにEC2からメトリクスを送付
2. 指定したしきい値を超えた時にSNS のアクションを実行
3. SNS をトリガーにAWS Lambdaを実行
4. GetMetricWidgetImage APIを利用して CloudWatch Metricsのグラフを取得する
5. 取得したグラフの画像ファイルを添付してEmail を送信/運用システムに連携

ユースケース：アクションの設定例



Amazon EC2

- EC2 インスタンスを自動的に停止、終了、再起動、または復旧するアラームを作成
- **StatusCheckFailed_System**アラームがトリガーとしたAutoRecovery
- 一定のしきい値に達したときに EC2 インスタンスを自動的に終了



Amazon EC2 Auto Scaling

- AutoScalingのEC2インスタンス台数を増減するアラームを作成
- 負荷に応じてリソースを調整

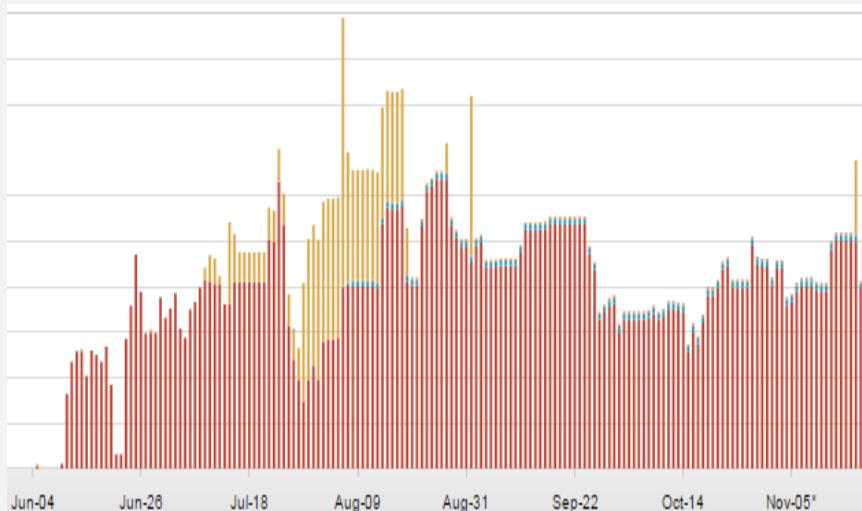
クラウドならではの監視



CloudWatchによるコストの監視

Billingアラーム設定

- ・ 課金状況をCloudWatch監視
- ・ 一定金額を超えるとアラームメール通知が可能
- ・ アラームの設定はVirginiaリージョンから設定



Alarm Threshold

Provide the details and threshold for your alarm. Use the graph on the right to help set the appropriate threshold.

Name: Billing Alarm

Description: AWS Billing Alarm

Whenever charges for: EstimatedCharges

is: \geq USD \$ 100

Actions

Define what actions are taken when your alarm changes state.

Notification

Delete

Whenever this alarm: State is ALARM

Send notification to: Select a notification list ▾ New list Enter list ⓘ

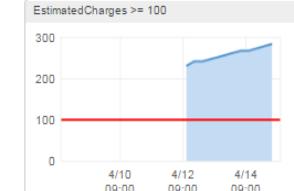
+ Notification

+ AutoScaling Action

+ EC2 Action

Alarm Preview

This alarm will trigger when the blue line goes up to or above the red line



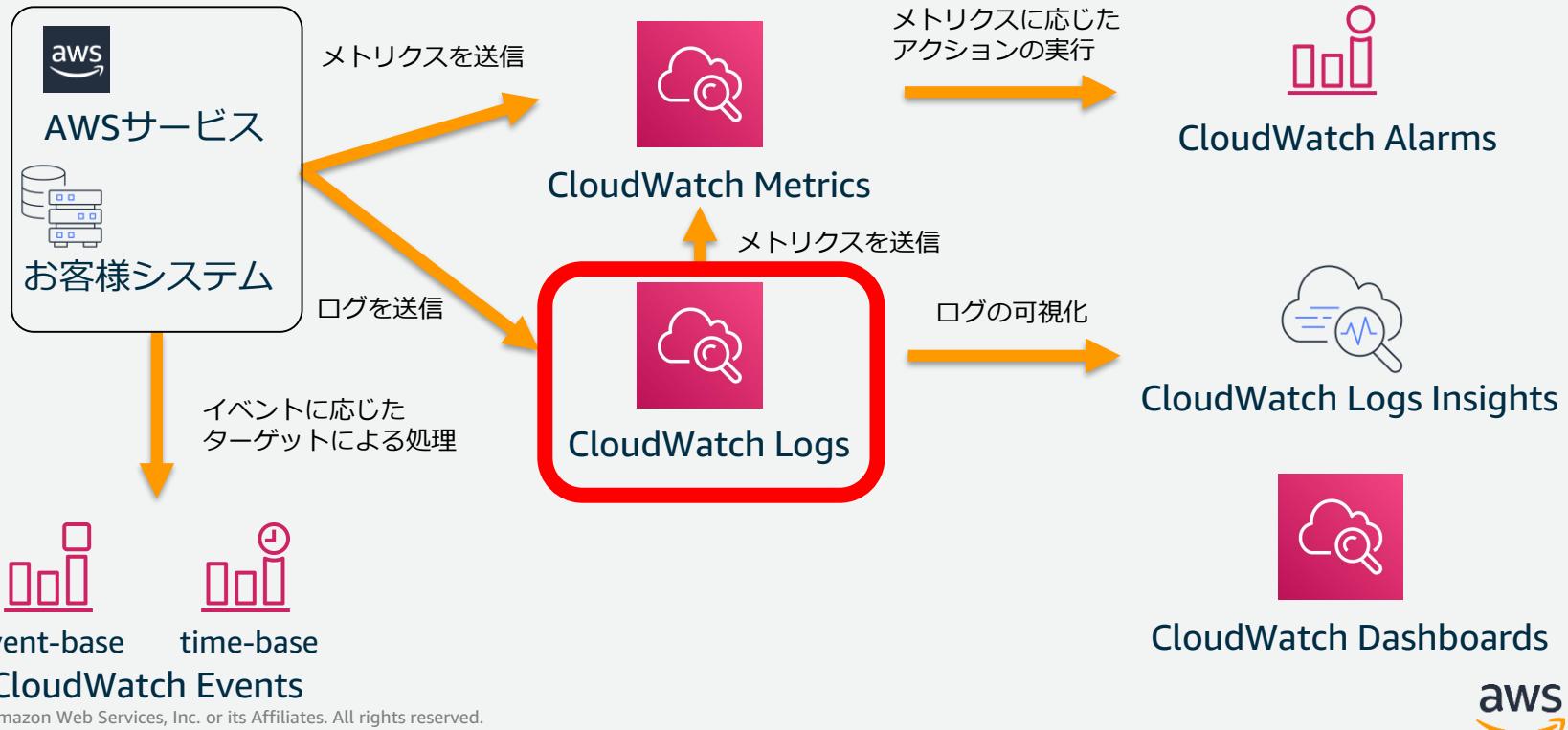
Namespace: AWS/Billing

Currency: USD

Metric Name: EstimatedCharges

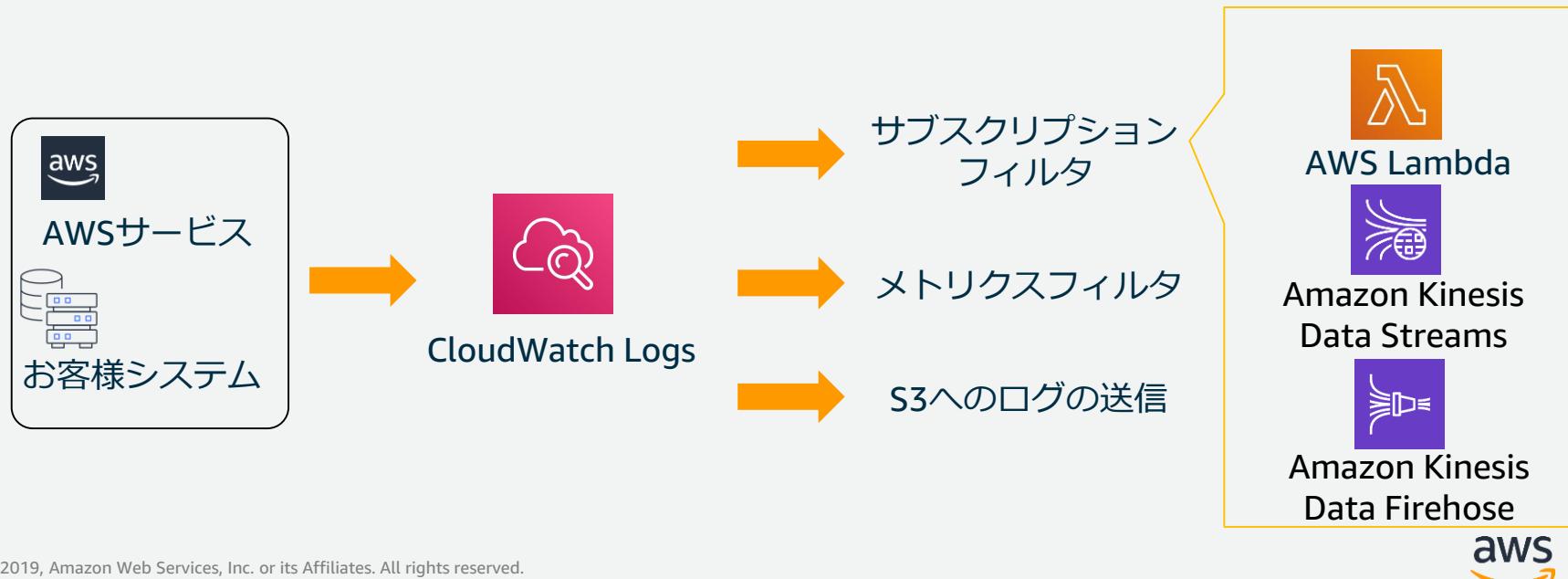
CloudWatch Logs

AWSサービスおよびお客様システムのログファイルの監視、保存、アクセス



CloudWatch Logs とは

- AWSサービスおよび顧客システムのログの監視、保存、アクセスを提供
- エージェント経由でログメッセージをCloudWatchエンドポイントに転送
- ログデータの保存期間を設定可能(1日~永久保存で選択可能)
- Amazon S3へのログのエクスポートが可能

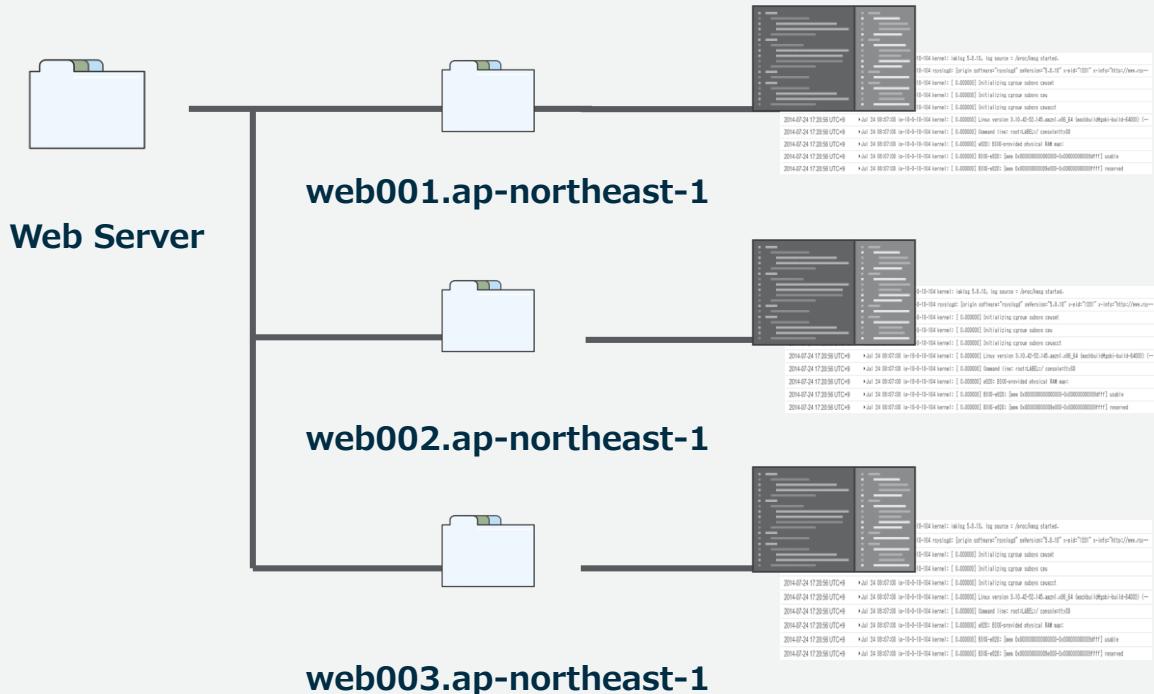


CloudWatch Logsのディレクトリ階層

ロググループ

ログストリーム

ログイベント



ログイベント

- 1つのログエントリ
 - モニタリングしているリソースによって記録されたアクティビティのレコード
 - イベント発生時のタイムスタンプおよび生のイベントメッセージで構成

ログストリーム

- 複数のログイベントで構成
 - モニタリングしているリソースのタイムスタンプ順でイベントを表す

ロググループ

- 複数のログストリームで構成
 - 保持、監視、アクセス制御について同じ設定を共有するログストリームのグループを定義

ログイベントのイメージ

ログ内容はタイムスタンプとログメッセージ（UTF-8）で構成

CloudWatch > ロググループ > stg-log > eni-0044968407da4ed68-all

すべて展開 行 テキスト   

イベントのフィルター		すべて 2019-03-07 (00:22:26)
	時間 (UTC +00:00)	メッセージ
	2019-03-07	
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 10.1.28.199 210.173.160.87 33097 123 17 1 76 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 139.180.207.65 10.1.28.199 123 51551 17 1 76 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 10.1.28.199 139.180.207.65 51551 123 17 1 76 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 68.183.37.224 10.1.28.199 50797 8088 6 1 40 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 210.173.160.87 10.1.28.199 123 33097 17 1 76 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 104.168.204.23 10.1.28.199 36348 8088 6 1 40 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 10.1.28.199 210.173.160.27 123 123 17 8 608 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 210.173.160.27 10.1.28.199 123 123 17 8 608 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 52.1.36.76 10.1.28.199 443 39188 6 44 10197 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 122.215.240.52 10.1.28.199 123 123 17 9 684 1551937715 155193
▶	05:48:35	2 981255050802 eni-0044968407da4ed68 133.243.238.243 10.1.28.199 123 123 17 9 684 1551937715 155193

ログの保存期間

CloudWatch Logsはログを永久保存可能

CloudWatch > Log Groups

Create Metric Filter Actions ▾

Filter: Log Group Name Prefix

Log Groups	Insights	Expire Events After	Metric Filters
/aws-glue/crawlers	Explore	Never Expire	0 filters
/aws/lambda/eslambda	Explore	Never Expire	0 filters
Asystem-production-logs	Explore	Never Expire	1 filter
Asystem-stage-logs	Explore	5 days	1 filter

Never Expire ▾

- Never Expire
- 1 day
- 3 days
- 5 days
- 1 week (7 days)
- 2 weeks (14 days)
- 1 month (30 days)
- 2 months (60 days)
- 3 months (90 days)
- 4 months (120 days)
- 5 months (150 days)
- 6 months (180 days)
- 1 year (365 days)
- 13 months (400 days)
- 18 months (545 days)
- 2 years (731 days)
- 5 years (1827 days)
- 10 years (3653 days)

CloudWatch Logs メトリクスフィルタ (1/2)

- ログデータから特定の文字列のフィルタリングが可能
- 正規表現の利用ができない点に注意

Define Logs Metric Filter

Filter for Log Group: Linux-Sysstem-Logs

You can use metric filters to monitor events in a log group as they are sent to CloudWatch Logs. You can monitor and count specific terms or extract values from log events and associate the results with a metric. [Learn more about pattern syntax.](#)

Filter Pattern
DHCPREQUEST

Show examples

Select Log Data to Test

i-156818e7

Test Pattern

Oct 18 03:01:04 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
Oct 18 03:01:04 ip-172-31-29-54 dhclient[1878]: DHCPACK from 172.31.16.1 (xid=0x15513168)
Oct 18 03:01:06 ip-172-31-29-54 dhclient[1878]: bound to 172.31.29.54 -- renewal in 1890
Oct 18 03:28:16 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 68
Oct 18 03:28:16 ip-172-31-29-54 dhclient[1878]: DHCPACK from 172.31.16.1 (xid=0x15513168)
Oct 18 03:28:18 ip-172-31-29-54 dhclient[1878]: bound to 172.31.29.54 -- renewal in 1585

Results

Found 17 matches out of 50 event(s) in the sample log.

Show test results

Cancel Assign Metric

Results

Found 17 matches out of 50 event(s) in the sample log.

Line Number	Line Content
1	Oct 18 03:01:04 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
4	Oct 18 03:29:16 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
7	Oct 18 03:55:43 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
10	Oct 18 04:20:18 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
13	Oct 18 04:43:37 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
16	Oct 18 05:11:50 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
19	Oct 18 05:39:11 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
22	Oct 18 06:06:23 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
25	Oct 18 06:32:33 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
29	Oct 18 03:01:04 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
32	Oct 18 03:29:16 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
35	Oct 18 03:55:43 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
38	Oct 18 04:20:18 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
41	Oct 18 04:43:37 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
44	Oct 18 05:11:50 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
47	Oct 18 05:39:11 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551
50	Oct 18 06:06:23 ip-172-31-29-54 dhclient[1878]: DHCPREQUEST on eth0 to 172.31.16.1 port 67 (xid=0x1551

CloudWatch Logs メトリクスフィルタ (2/2)

CloudWatch メトリクスに一致したパターン数を記録
→特定文字列のエントリ頻度等によりアラーム作成、SNS連携が可能

メトリクスフィルタの作成とメトリクスの割り当て

ロググループのフィルター: prd-log

ログイベントが定義したパターンと一致すると、指定したメトリクスに記録されます。メトリクスをグラフ表示でき、メトリクスにアラームを設定して通知することもできます。

フィルタの名前: cloudwatchalarm-test

フィルタパターン:

メトリクスの詳細

メトリクス名前空間: LogMetrics

① 新しい名前空間の作成

メトリクス名: flowlog-check

①

メトリクス値: 1

①

デフォルト値: 0

①

- 定義したパターンに一致した時にメトリクスにパブリッシュされる値
- 単純なパターン数を求める時は1を指定

CloudWatch Logs サブスクリプションフィルタ

CloudWatch Logsに集めたログをフィルタパターンに応じてリアルタイムに
Kinesis Data Streams/Kinesis Data Firehose/Lambdaへ転送

- 1つのロググループにつき、1つのサブスクリプションフィルタを設定可能



AWS CLIからのみ設定可能

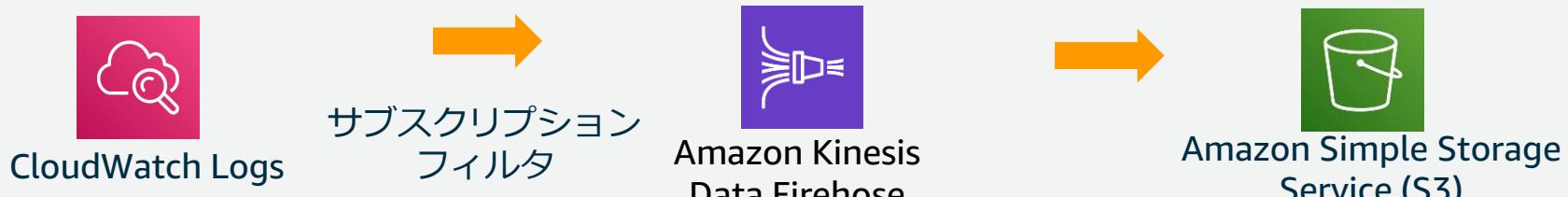
```
aws logs put-subscription-filter ¥
--log-group-name "xxxxxxxx" ¥
--filter-name "xxxxxxxx" ¥
--filter-pattern "{xxxxxxxx = xxxxxxxx}" ¥
--destination-arn "arn:aws:kinesis:ap-northeast-1:123456789012:stream/xxxxxxxx" ¥
--role-arn "arn:aws:iam::123456789012:role/xxxxxxxx"
```

ユースケース：サブスクリプションフィルタの利用例

Lambdaを活用したElasticsearch Serviceへのストリーミング



Kinesis DataFirehoseを活用したS3へのデータ転送



Kinesis DataFirehoseのカスタムプレフィックスによりタイムスタンプ情報をApache Hiveフォーマットに設定可能
myPrefix/year=!{timestamp:yyyy}/month=!{timestamp:MM}/day=!{timestamp:dd}/hour=!{timestamp:HH}/

パスの例
myPrefix/year=2018/month=07/day=06/hour=23/

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/CWL_ES_Stream.html
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/Subscriptions.html
https://docs.aws.amazon.com/ja_jp/firehose/latest/dev/s3-prefixes.html

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



Amazon S3 へのログデータのエクスポート

コンソールかAWS CLIにより利用、タスクの完了には数秒から数時間

ユースケース：ログ調査のために一部データをアドホックに取得

コンソールの使用

- エクスポートするデータの期間
- S3バケットの指定
- バケットプレフィックスの指定



AWS CLIを使用したエクスポート

エクスポートタスクの作成

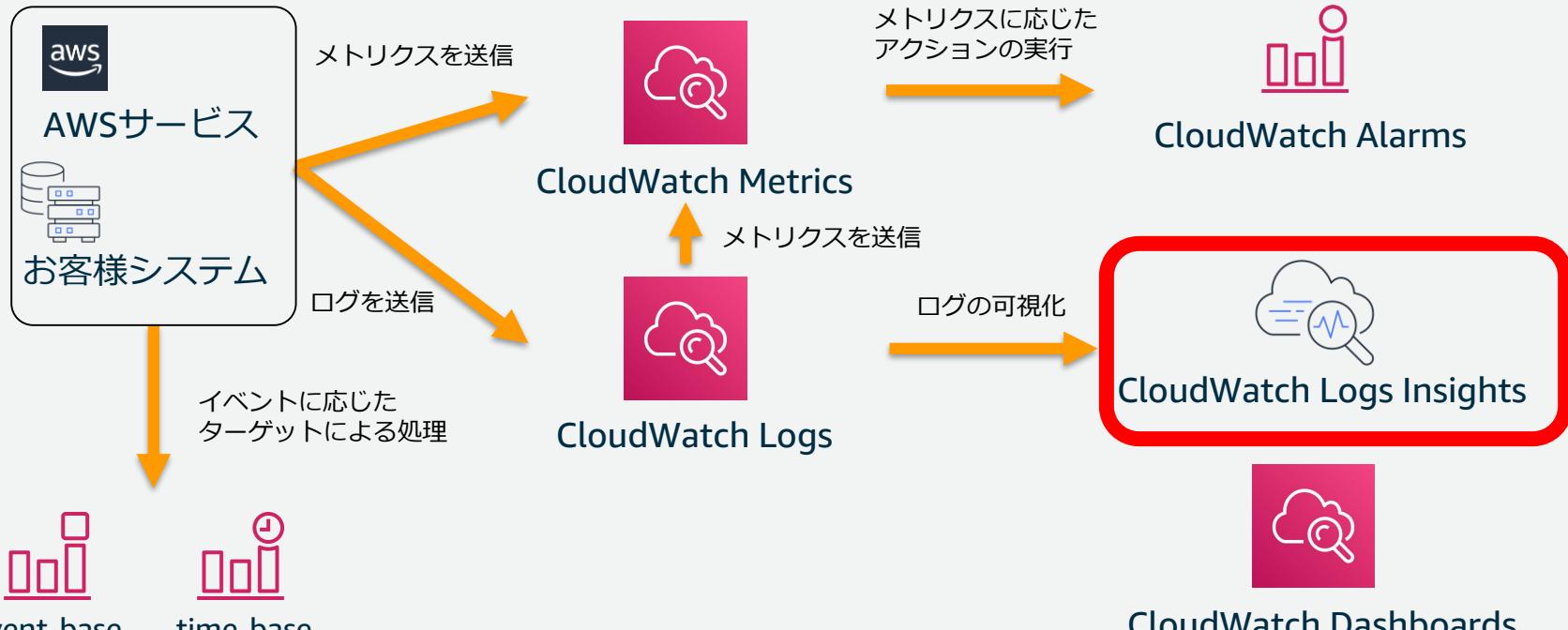
```
aws logs create-export-task --task-name "my-Log-group-09-10-2015" --log-group-name "my-Log-group" --from 1441490400000 --to 1441494000000 --destination "my-exported-logs" --destination-prefix "export-task-output"
```

タスクの作成ステータス確認

```
aws logs describe-export-tasks --task-id "cda45419-90ea-4db5-9833-aade86253e66"
```

CloudWatch Logs Insights

CloudWatch Logs のログデータをインタラクティブに検索して分析



CloudWatch Logs Insightsとは

CloudWatch Logs のログデータをインタラクティブに検索して分析

- 専用のクエリ言語といくつかのシンプルで強力なコマンドを提供
- サンプルクエリ、クエリの自動補完、ログフィールドの検出を提供
- 2018/11/5 以降に CloudWatch Logs に送信されたログデータを検索可能

The screenshot shows the CloudWatch Logs Insights interface. At the top, there's a search bar with the query `stg-log` and a dropdown menu for log groups. Below the search bar is a time range selector from 15分 to カスタム. A red box highlights the search bar and dropdown.

In the center, there's a button labeled "実行するクエリ" (Run Query) and a section titled "クエリ結果" (Query Results). A red box highlights the "クエリ結果" section. It contains a histogram titled "時間の経過に伴うログイベントの分布" (Distribution of log events over time) showing event counts from 07:30 to 08:25. Below the histogram is a table of log events with columns for timestamp and message. The first five events are listed:

#	: @timestamp	: @message
1	2019-03-08 23:26:06.000	2 981255050802 eni-05ec65ff60d063ec7 10.1.62.174 10.1.105.124 80 16816 6 5 533 1552087566 1552087624 ACCEPT OK
2	2019-03-08 23:26:06.000	2 981255050802 eni-05ec65ff60d063ec7 185.254.122.120 10.1.105.124 53283 2273 6 1 40 1552087566 1552087624 REJECT OK
3	2019-03-08 23:26:06.000	2 981255050802 eni-05ec65ff60d063ec7 10.1.105.124 10.1.9.244 54586 80 6 6 455 1552087566 1552087624 ACCEPT OK
4	2019-03-08 23:26:06.000	2 981255050802 eni-05ec65ff60d063ec7 104.248.247.78 10.1.105.124 49937 8088 6 1 40 1552087566 1552087624 REJECT OK
5	2019-03-08 23:26:06.000	2 981255050802 eni-05ec65ff60d063ec7 10.1.62.174 10.1.105.124 80 16852 6 5 533 1552087566 1552087624 ACCEPT OK

To the right, there's a sidebar titled "検出されたログのフィールド" (Detected fields in logs) which lists the following fields with 100% detection rate:

- @logStream
- @message
- @timestamp
- accountId
- end
- interfaceId
- logStatus
- start
- version
- action
- bytes
- dstAddr
- dstPort
- packets

サポートされるログと検出されるフィールド

CloudWatch Logs Insights は様々なタイプのログをサポート

送信されるログごとに3つのフィールドを生成

- @message: 生の未解析のログイベント
- @timestamp: ログイベントが CloudWatch Logs に追加された時間
- @logStream: ログイベントの追加先のログストリームの名前

VPCフローログ、Route53ログ、Lambdaログは自動でフィールドを検出

ログタイプ	検出されるログフィールド
Amazon VPC フローログ	@timestamp、@logStream、@message、accountId、endTime、interfaceId、logStatus、startTime、version、action、bytes、dstAddr、dstPort、packets、protocol、srcAddr、srcPort
Route 53 ログ	@timestamp、@logStream、@message、edgeLocation、hostZoneId、protocol、queryName、queryTimestamp、queryType、resolverIp、responseCode、version
Lambda ログ	@timestamp、@logStream、@message、@requestId、@duration、@billedDuration、@type、@maxMemoryUsed、@memorySize

CloudWatch Logs Insights クエリ構文

6つのクエリコマンドをサポートし多数の関数やオペレーションが用意

コマンド	説明	例
fields	指定したフィールドをログイベントから取得	・ログフィールド @message を取得 fields @message
filter	クエリの結果を 1 つ以上の条件に基づいて フィルタリング	・文字列"error"が含まれるログフィールド@messageを取得 fields @message filter @message like "error"
stats	ログフィールドの値に基づいて集約統計を計算。 sum(), avg(), count(), min(), max()に対応	・文字列"error"が含まれるログフィールド@messageをカウント filter @message like "error" stats count(*)
sort	取得したログイベントをソート	・ログフィールドbytesの降順で@messageを表示 fields @message sort bytes desc
limit	クエリから返されるログイベントの数を制限	・ログフィールドbytesの降順で@messageを10件表示 fields @message sort bytes desc limit 10
parse	ログフィールドからデータを抽出し、1つ以上のエフェメラルフィールドを作成	・ログフィールド @message から、エフェメラルフィールド @user, @latency を抽出し、@user の一意な組み合わせごとに平均レイテンシーを返す parse @message "user=*,latency := *" as @user, @latency stats avg(@latency) by @user

CloudWatch Logs Insights クエリ構文

比較オペレーション(filter コマンドで利用)

- = != < <= > >=

算術オペレーション(filter コマンドとfields コマンドで利用)

- 加算: +, 減算: -, 乗算: *, 除算: /, 指数: ^, 剰余: %

数値オペレーション(filter コマンドとfields コマンドで利用)

- 絶対値 : **abs()**, 上限 : **ceil()**, 下限 : **floor()**, 最大値 : **greatest(a,b,...,z)**, etc.

一般関数(filter コマンドとfields コマンドで利用)

- **ispresent(fieldname)** , **coalesce(fieldname1, fieldname2,...fieldnamex)**

文字列関数(filter コマンドとfields コマンドで利用)

- **isempty(fieldname)**, **concat(string1, string2, ... stringz)**, etc.

日時関数(filter コマンドとfields コマンドで利用)

datefloor(a, period), dateceil(a, period), etc.

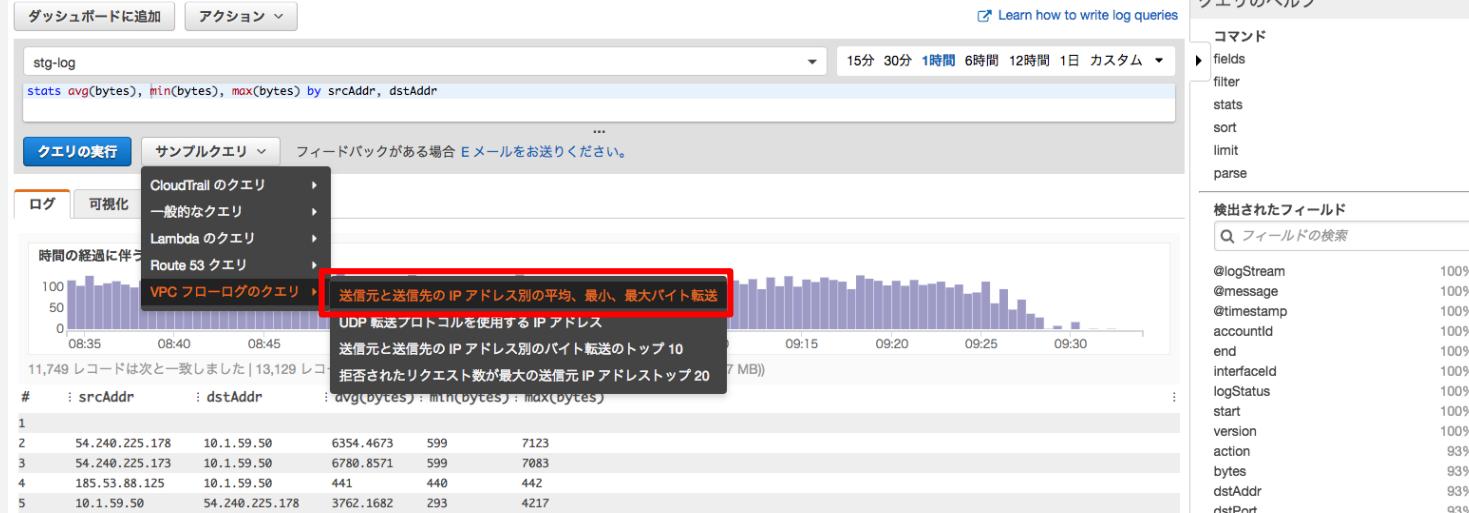
<https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/CloudWatchLogsQuerySyntax.html>
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



サンプルクエリ

CloudTrail / 一般 / Lambda / Route53 / VPCフローログのサンプルクエリを提供

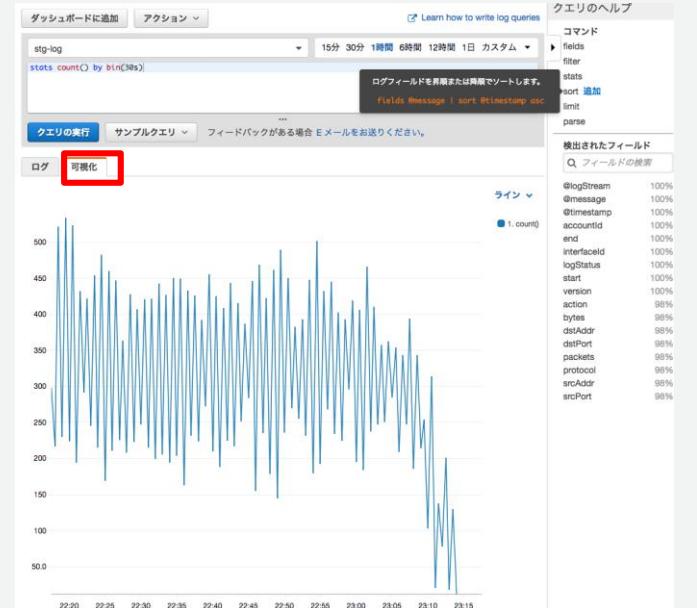
- 例：VPCフローログ向けに「送信元と送信先のIPアドレス別の平均、最小、最大バイト転送」のクエリを提供
- CloudWatch Logsにログを格納していればすぐに利用が可能



Visualization

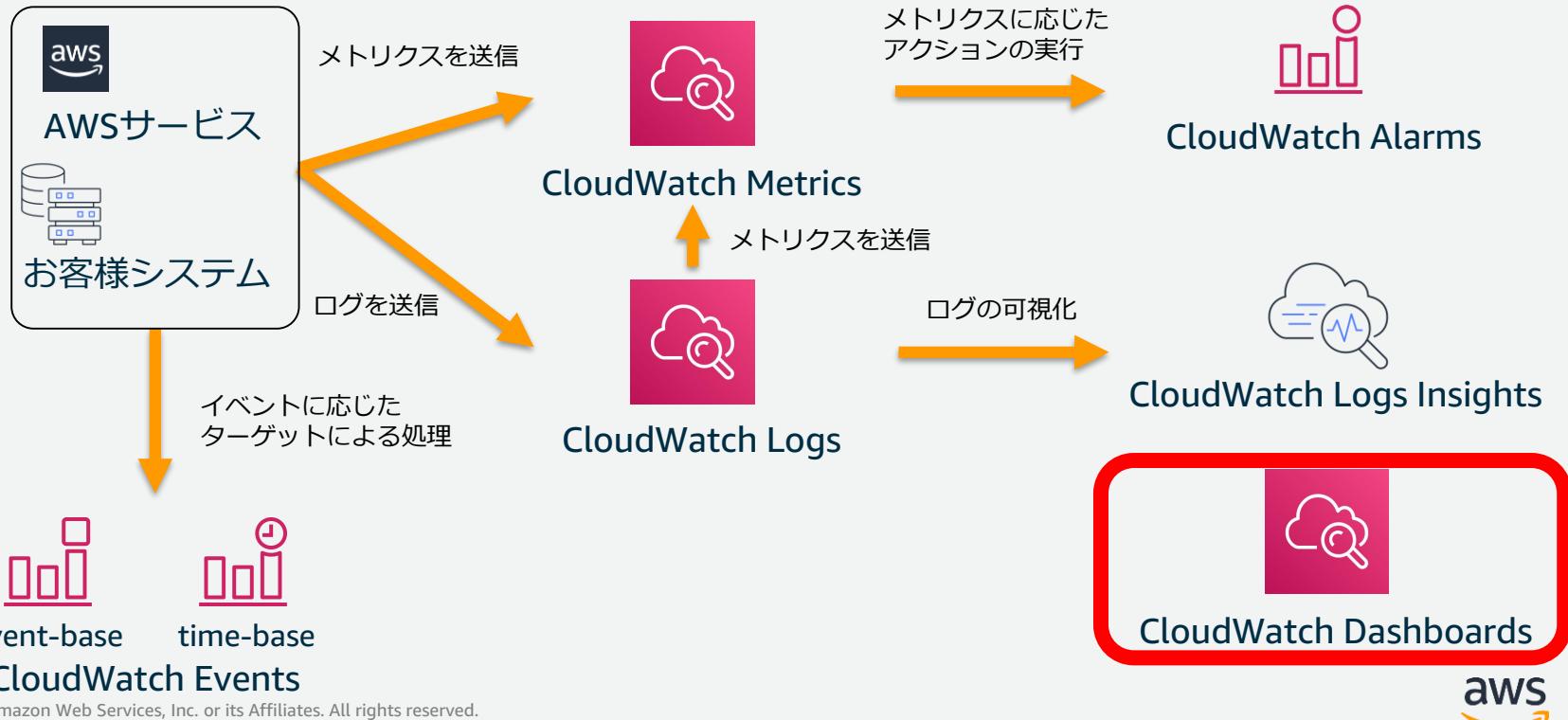
CloudWatch Logs Insights で可視化機能を提供

- 時間軸に沿ってトレンドやパターンを特定・分析
- Visualization の生成に必要な条件
 - ひとつ以上の集約関数 **stats()** 関数を使用
 - グルーピングに 期間の切り上げ **bin()** 関数を使用
- Visualizationをダッシュボードに追加可能



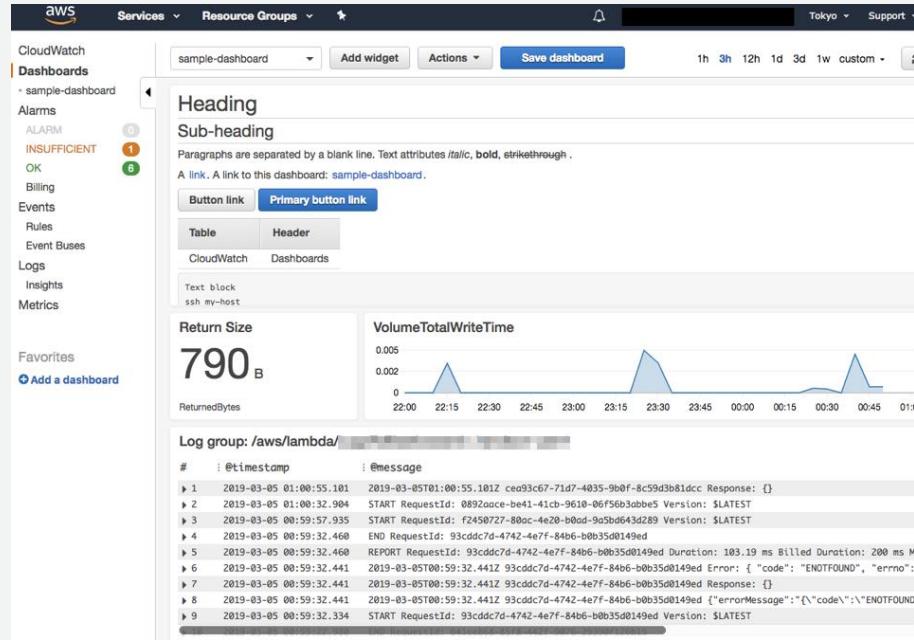
CloudWatch Dashboards

CloudWatch コンソールにあるカスタマイズ可能なホームページ



CloudWatch Dashboards

- CloudWatch コンソールでカスタマイズ可能なダッシュボードを作成
- 異なるリージョンのリソースでも、1つのダッシュボードでモニタリング可能
- 自動更新間隔を指定可能(10s, 1m, 2m, 5m, 15m)



ダッシュボードで表示可能な5つのウィジェット

折れ線グラフ

- 時間経過でメトリクスを比較

スタックエリア

- 時間経過で合計を比較

数値

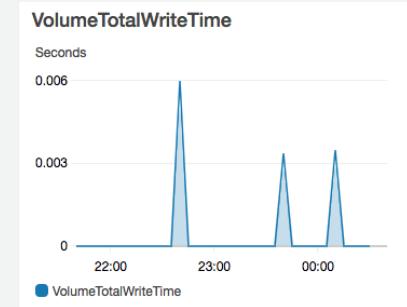
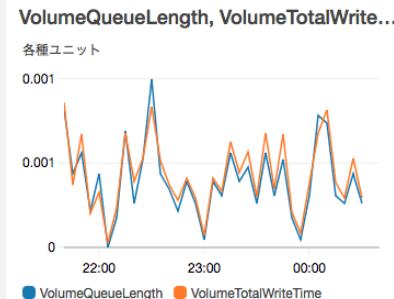
- メトリクスの最新値を表示

テキスト

- マークダウン形式による表示
- ボタンとしてウェブリンクを指定

クエリの結果

- Logs Insightsから結果を表示



Heading

Sub-heading

Paragraphs are separated by a blank line. Text attributes *italic*, **bold**, strikethrough .

A [link](#). A link to this dashboard: [sample-dashboard](#).

[Button link](#) [Primary button link](#)

[Table](#) [Header](#)

[CloudWatch](#) [Dashboards](#)

List syntax:

- CloudWatch
- Dashboards
 - 1. Graphs
 - 2. Text widget

ロググループ: [REDACTED]

```
# : @timestamp :@message
↳ 1 2019-03-09 00:42:13.690 2019-03-09T00:42:13.690Z f8e534cd-21e8-4db1-bcb4-84eb8d411c7 Response: {}
↳ 2 2019-03-09 00:41:57.428 START RequestId: 7eb0d89-1b7d-4e7f-bef-958e98440bf Version: $LATEST
↳ 3 2019-03-09 00:41:43.996 REPORT RequestId: eb609ee-1c7d-4150-8701-001b0001480f Duration: 148.62 ms Billed Duration: 200 ms Memory Size: 128 MB Max Memory Used: 71 MB
↳ 4 2019-03-09 00:41:43.996 END RequestId: eb609ee-1c7d-4150-8701-001b00014802
↳ 5 2019-03-09 00:41:43.995 2019-03-09T00:41:43.995Z eb609ee-1c7d-4150-8701-001b00014802 Error: { "code": "ENOTFOUND", "errno": "ENOTFOUND", "syscall": "getaddrinfo", "hostname": "search-hands-on-use-", "host": "search-hands-on-use-", "port": 443 }
↳ 6 2019-03-09 00:41:43.995 2019-03-09T00:41:43.995Z eb609ee-1c7d-4150-8701-001b00014802 ("errorMessage": "\\"code\\":\\"ENOTFOUND\\",\\"errno\\":\\"ENOTFOUND\\",\\"syscall\\":\\"getaddrinfo\\",\\"hostname\\":\\"search-hands-on-use-",\\"host\\":\\"search-hands-on-use-",\\"port\\":443")
↳ 7 2019-03-09 00:41:43.995 2019-03-09T00:41:43.995Z eb609ee-1c7d-4150-8701-001b00014802 Response: {}
↳ 8 2019-03-09 00:41:42.945 START RequestId: eb609ee-0246-4c67-9bc9-f38209407bf Version: $LATEST
↳ 9 2019-03-09 00:41:37.635 REPORT RequestId: fc3e4681-0246-4c67-9bc9-f38209407bf Duration: 20.67 ms Billed Duration: 100 ms Memory Size: 128 MB Max Memory Used: 71 MB
↳ 10 2019-03-09 00:41:37.635 END RequestId: fc3e4681-0246-4c67-9bc9-f38209407bf
```

ReturnedBytes

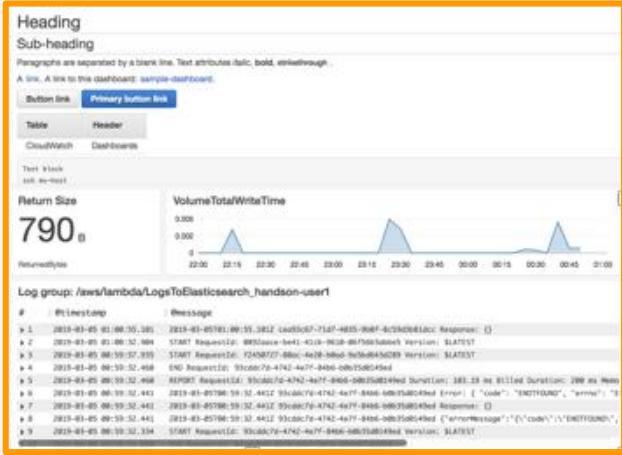
790 B

ReturnedBytes

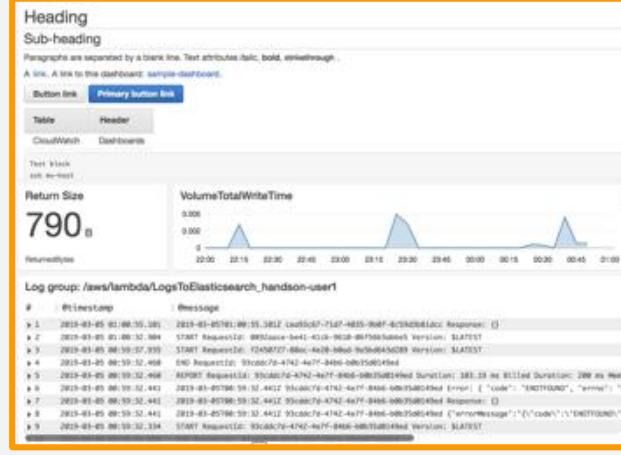
API/CLIの提供

APIおよびCLIによりダッシュボードの作成が可能

既存のダッシュボード



新しいダッシュボード



get-dashboard

```
aws cloudwatch get-dashboard --dashboard-name old-dashboard  
→ダッシュボードのbodyをjsonで取得
```

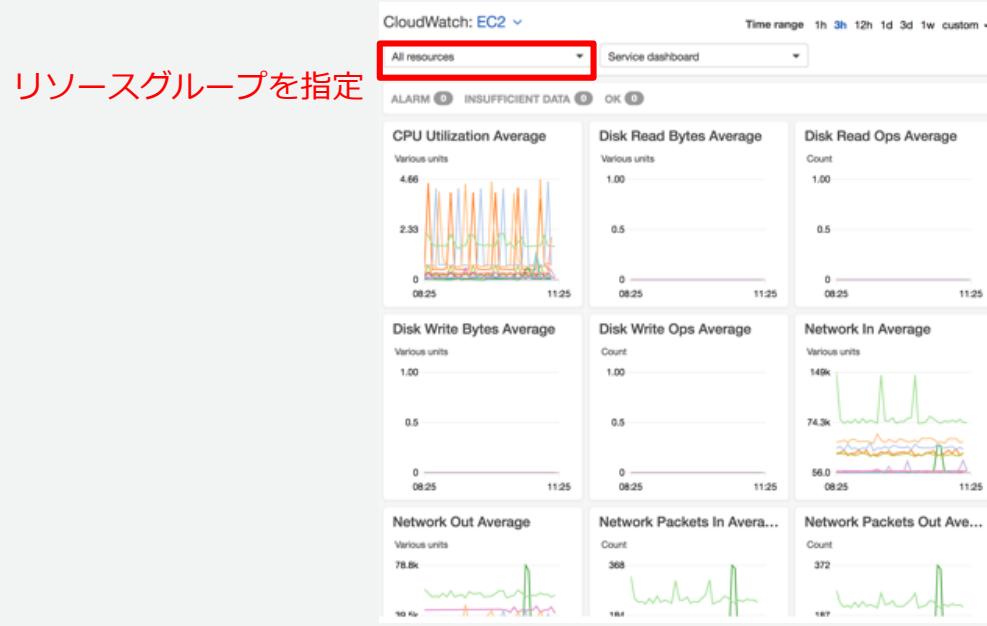
put-dashboard

```
aws cloudwatch put-dashboard --dashboard-name new-dashboard --dashboard-body “作成したbody”  
→新しいダッシュボードを作成
```

Automated dashboard

AWSが推奨するベストプラクティスに基づいたダッシュボードを自動生成

- 各主要サービスごとあるいはサービスをまたいだダッシュボード
- リソースグループを使用してシステムごとにフィルタリング可能



リソースグループを指定

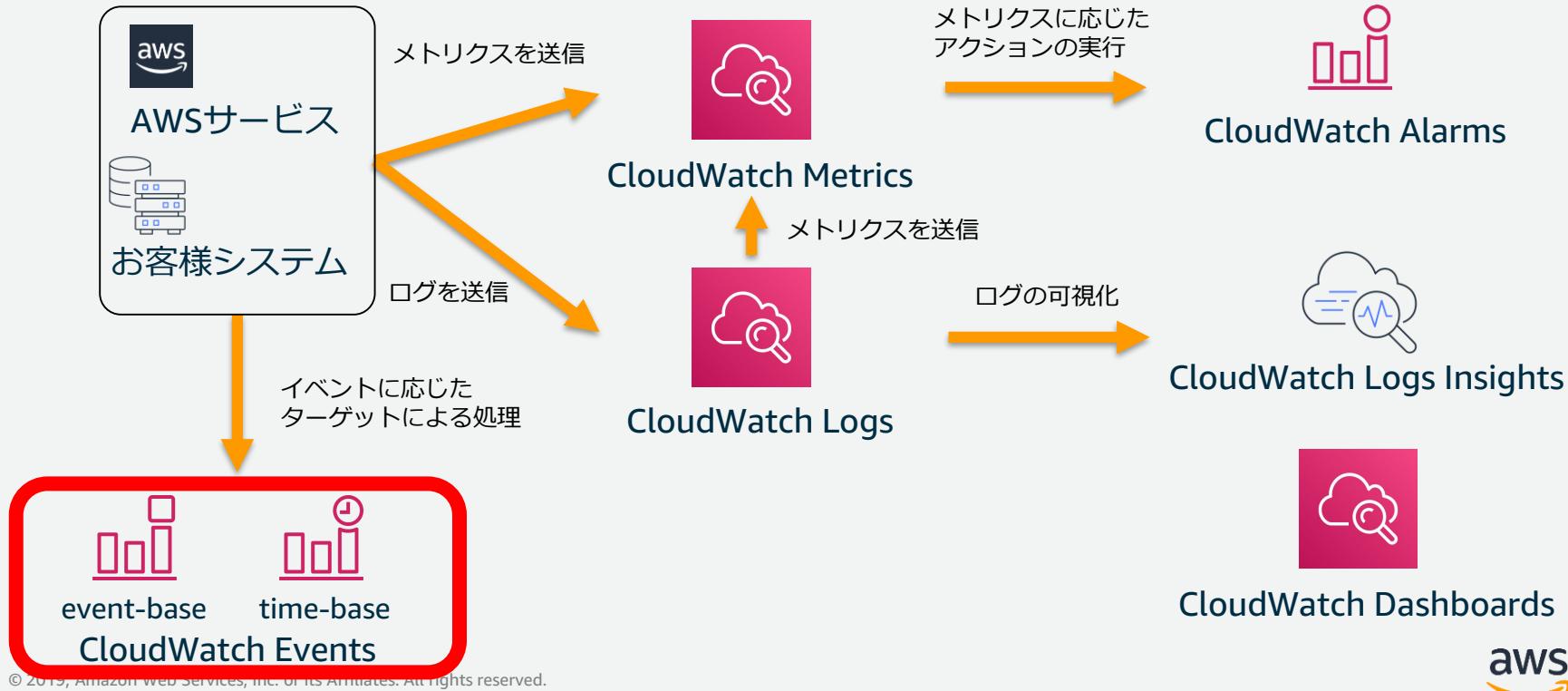
CloudWatch Alarmsとの統合

- 作成したCloudWatch Alarms をダッシュボードに追加が可能
- ALARM状態になるとウィジェットが赤に変わる
- 折れ線グラフ、スタックグラフ、数値で利用可能



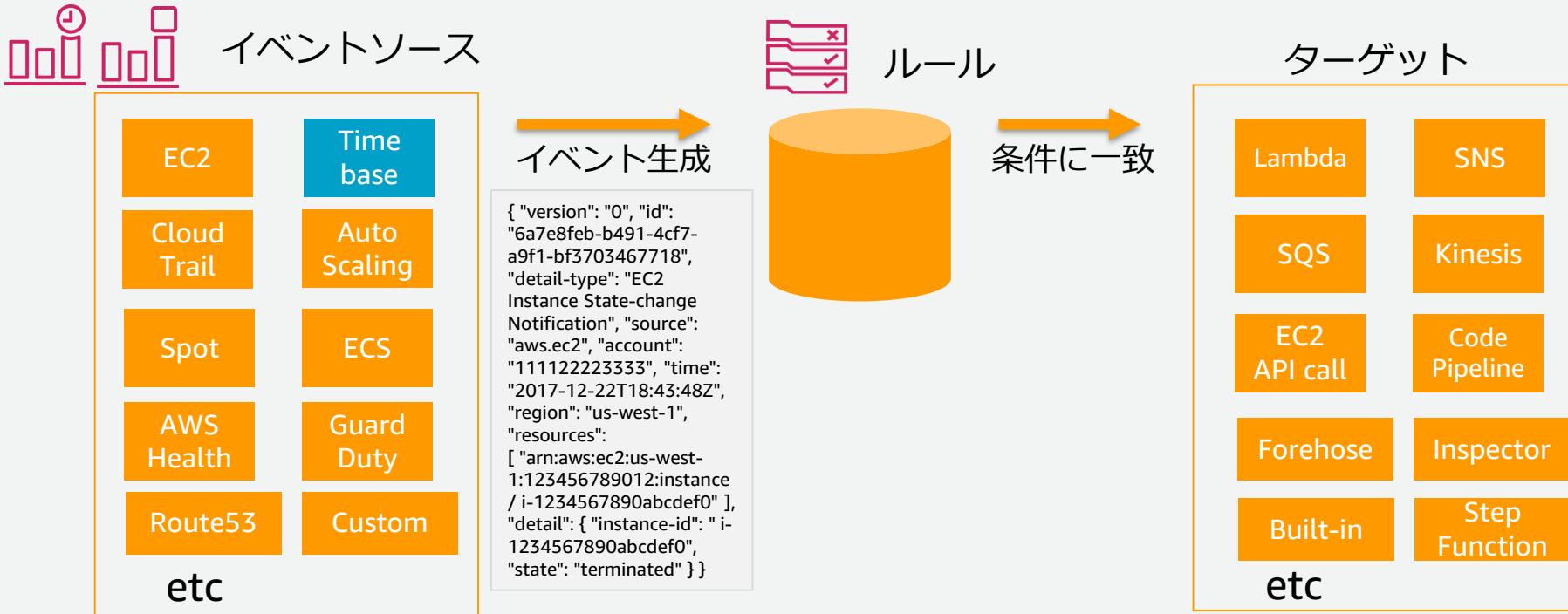
CloudWatch Events

リソース変更のイベントに応答してアクションを実行



CloudWatch Events

- AWS上リソースの変更を示すシステムイベントのストリームを提供
- システムイベントをトリガーとして、ターゲットがイベントを処理



CloudWatch Events (ルールの作成)

Step 1: Create rule

Create rules to invoke Targets based on Events happening in your AWS environment.

Event Source

Build or customize an Event Pattern or set a Schedule to invoke Targets.

Event Pattern ? Schedule ?

Build event pattern to match events by service

Service Name

Select or type to search...

Event Type

Select or type to search...

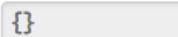
Targets

Select Target to invoke when an event matches your Event Pattern or when schedule is triggered.

Add target*

Event Pattern Preview

Copy to clipboard Edit



イベントソース

ターゲット

イベントパターンを選択することで、条件を指定したルールを作成可能

新しいイベントターゲットのサポート

CloudWatch Events のルールとして新規のターゲットを追加

- AWS Batch ジョブ
- CodeBuild プロジェクト
- CodePipeline
- EBS スナップショットの作成
- EC2 インスタンスの停止/再起動/削除
- ECS タスク
- 他アカウントへのイベントバス
- Kinesis Data Firehose の配信ストリーム
- Inspector アセスメントテンプレート
- Kinesis Data Streams のストリーム
- Lambda 関数
- SNS トピック
- SQS キュー
- SSM Automation
- SSM RunCommand
- Step Functions State Machine

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/events/EventTypes.html#ec2_event_type

作成したルールの管理

ルールの有効化・無効化が可能

Rules

Rules route events from your AWS resources for processing by selected targets. You can create, edit, and delete rules.

The screenshot shows the AWS Rules management interface. At the top, there is a blue button labeled "Create rule" and a dropdown menu labeled "Actions ▾" containing "Edit", "Delete", and "Enable". The "Enable" option is highlighted with a dark gray background. To the right of the dropdown are two small icons: a refresh symbol and a question mark. Below the header, there is a table with columns "Status" and "Description". The table contains three rows of data:

Status	Description
<input checked="" type="radio"/> ●	TerminationPolicyChecker
<input type="radio"/> ○	shootMyInstances
<input checked="" type="radio"/> ●	shootMyInstances-2

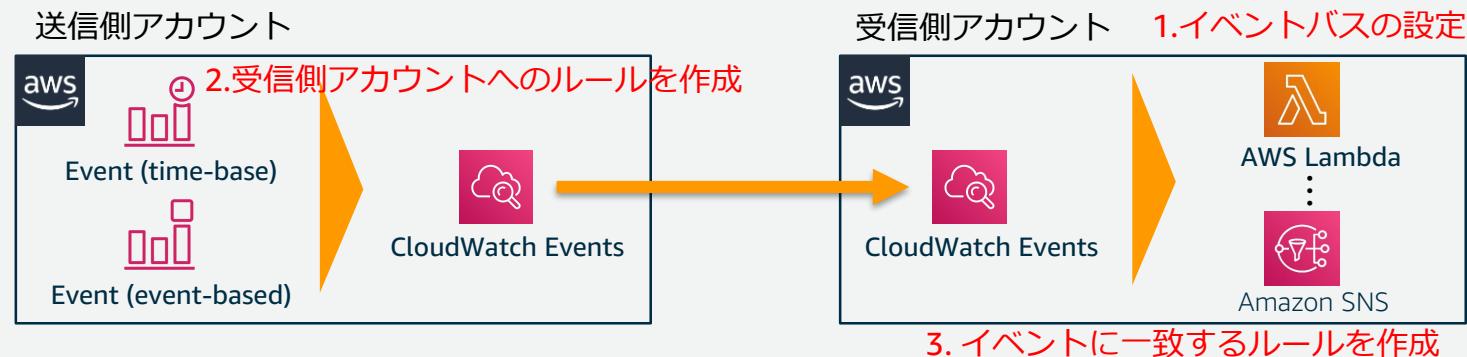
At the bottom right of the table, there is a message: "Viewing 1 to 3 of 3 Rules".

イベントバス

他の AWS アカウントとイベントを送受信するように AWS アカウントを設定

1. 受信側アカウントでイベントの受信を許可するAWSアカウント番号/Organizationsを指定
2. 送信側アカウントで受信側アカウントをターゲットとするルールを作成
3. 受信側アカウントで受信したイベントをイベントソースとするルールを作成

ユースケース：マルチアカウントにおけるリソース管理



ユースケース： Cloudwatch Eventsの利用例

- EC2のRunningイベントをトリガーに指定タグがついていない場合はterminate



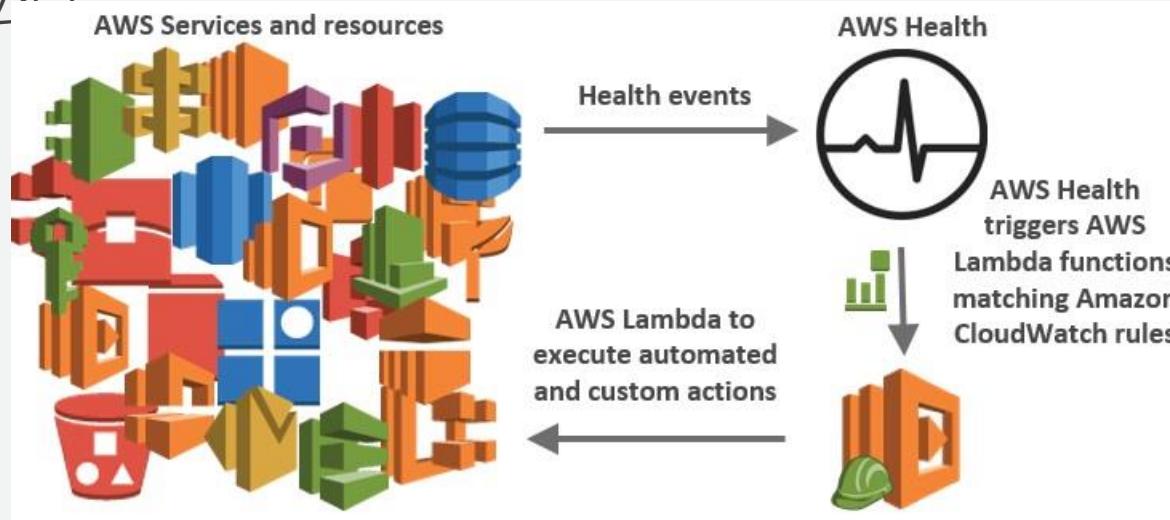
- スケジュール式をトリガーにEBSのスナップショットの取得(定期バックアップ)



Amazon CloudWatch Events での AWS Health イベントのモニタリング

AWS Health は、AWS のリソース、サービス、およびアカウントの状態を可視化

- AWS Healthイベントのステータスの変化を検出し、アクションするツールをGitHubで公開

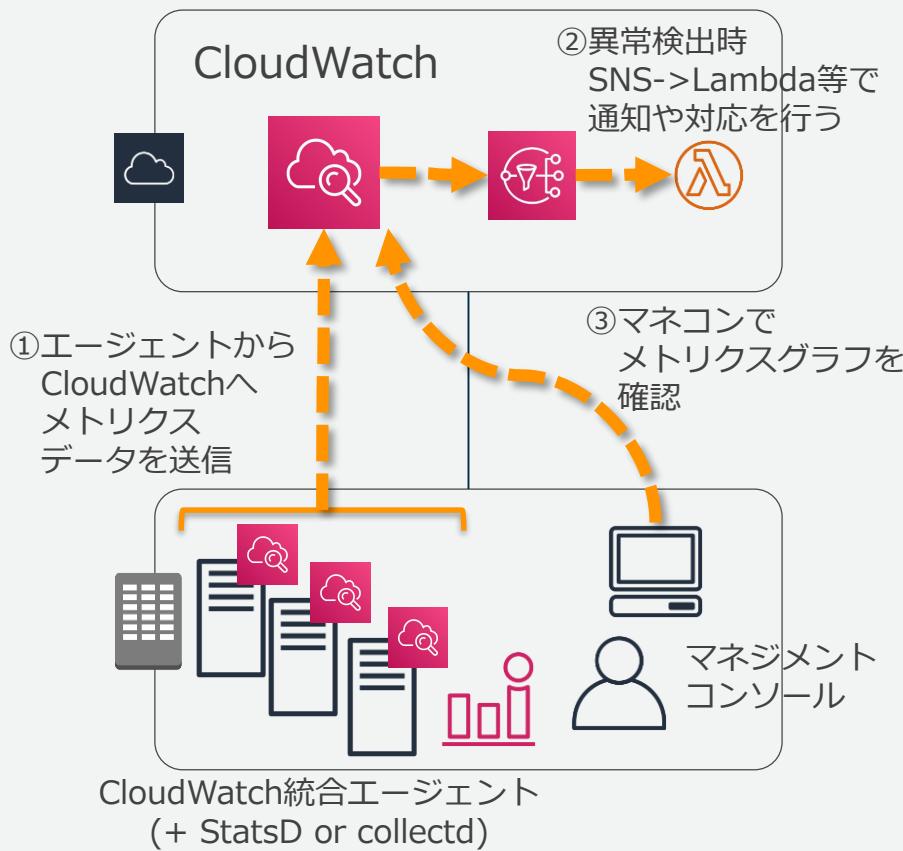


aws-health-tools: <https://github.com/aws/aws-health-tools>

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



Private Link の対応(Metrics / Events / Logs)



やりたいこと

- ・ オンプレミスもしくはプライベートサブネット環境におけるCloudWatch の利用
- ・ サーバ群のOSメトリクスを収集&監視
- ・ 異常時に通知や対応を行う

実現方法

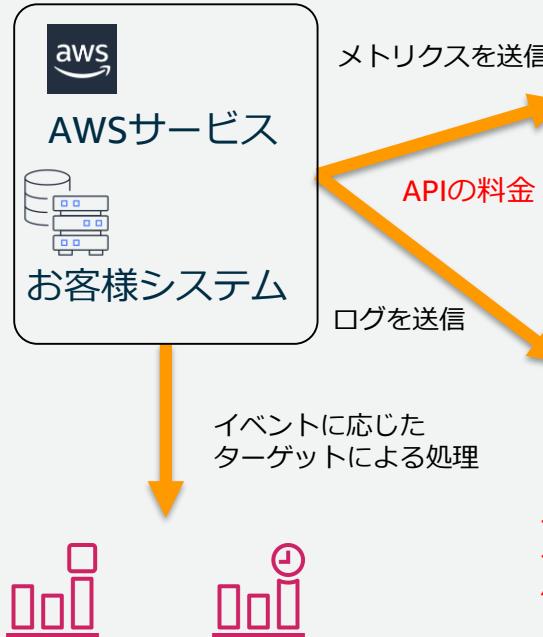
- ・ CloudWatch統合エージェント導入
Windows/Linuxメトリクスを取得可能

備考

- ・ PrivateLinkによる閉域通信が可能
- ・ StatsDやcollectdエージェントを追加してより詳しい情報を収集可能
- ・ カスタムメトリクスの課金に注意

料金

料金の概要



event-base time-base イベントの料金
CloudWatch Events



- アラームのメトリクス料金
CloudWatch Alarms
- スキャンされたデータの料金
CloudWatch Logs Insights
- ダッシュボードの料金
CloudWatch Dashboards

Amazon CloudWatch の料金

CloudWatch Metrics(カスタムメトリクスを含む)

範囲	コスト(月額)
最初の 10,000 メトリクス	0.30 USD
次の 240,000 メトリクス	0.10 USD
次の 750,000 メトリクス	0.05 USD
1,000,000 メトリクスを超える場合	0.02 USD

CloudWatch Events

範囲	コスト(月額)
カスタムイベント 100 万件あたり	1 USD
クロスアカウントイベント 100 万件あたり	1 USD

API

- ひとつの GetMetricData API リクエストで同じメトリクスに対して、最大 5 つの統計をリクエスト可能

対象	コスト(月額)
GetMetricData を使用してリクエストされた 1,000 個のメトリクスあたり	0.01 USD
GetMetricWidgetImage を使用してリクエストされた 1,000 個のメトリクスあたり	0.02 USD
GetMetricStatistics、ListMetrics、PutMetricData、GetDashboard、ListDashboards、PutDashboard、DeleteDashboards のリクエスト 1,000 件あたり	0.01 USD

2019年3月時点の東京リージョンの価格表

<https://aws.amazon.com/jp/cloudwatch/pricing/>

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



Amazon CloudWatch の料金

CloudWatch Logs 及び Insights

- CloudWatch Logs のデータ転送送信 (アウト) は EC2 料金ページの表と同じ料金です。

ログデータ

対象	コスト(月額)
収集(データの取り込み)	0.76 USD / GB
保存(アーカイブ)	0.033 USD / GB
分析(Logs Insightsのクエリ) (スキヤンされたデータ)	0.0076 USD / GB

S3にログを配信(VPCフローログ)

範囲	コスト(月額)
10TB まで	0.38 USD / GB
次の20TB	0.228 USD / GB
次の20TB	0.114 USD / GB
50TB ~	0.076 USD / GB

Vended Logs(VPCとRoute53のログ)

対象	コスト(月額)
データ取り込み 10TBまで	0.76 USD / GB
データ取り込み 次の20TB	0.38 USD / GB
データ取り込み 次の20TB	0.152 USD / GB
データ取り込み 50TB~	0.076 USD / GB
データの保存	0.033 USD / GB

※Vended Logsは、AWS のサービスが発行するログ

Amazon CloudWatch の料金

CloudWatch Dashboard

対象	コスト(月額)
ダッシュボードあたり	3 USD

CloudWatch Alarms

対象	コスト(月額)
標準分解能アラーム (60 秒)	アラームのメトリクスあたり 0.10 USD
高分解能アラーム (10 秒)	アラームのメトリクスあたり 0.30 USD

無料利用枠

サービス	無料利用枠
メトリクス	<ul style="list-style-type: none">基本モニタリングのメトリクス (5 分間隔)詳細モニタリングのメトリクス 10 個 (1 分間隔)100 万の API リクエスト (GetMetricData および GetMetricWidgetImage には適用されません)
ダッシュボード	<ul style="list-style-type: none">毎月最大 50 個のメトリクスに対応するダッシュボード 3 個
アラーム	<ul style="list-style-type: none">10 件のアラームメトリクス (高分解能アラームには適用されません)
ログ	<ul style="list-style-type: none">5 GB データ (取り込み、ストレージのアーカイブ、Logs Insights クエリによってスキャンされたデータ)
イベント	<ul style="list-style-type: none">カスタムイベントを除くすべてのイベントが対象

2019年3月時点の東京リージョンの価格表

<https://aws.amazon.com/jp/cloudwatch/pricing/>

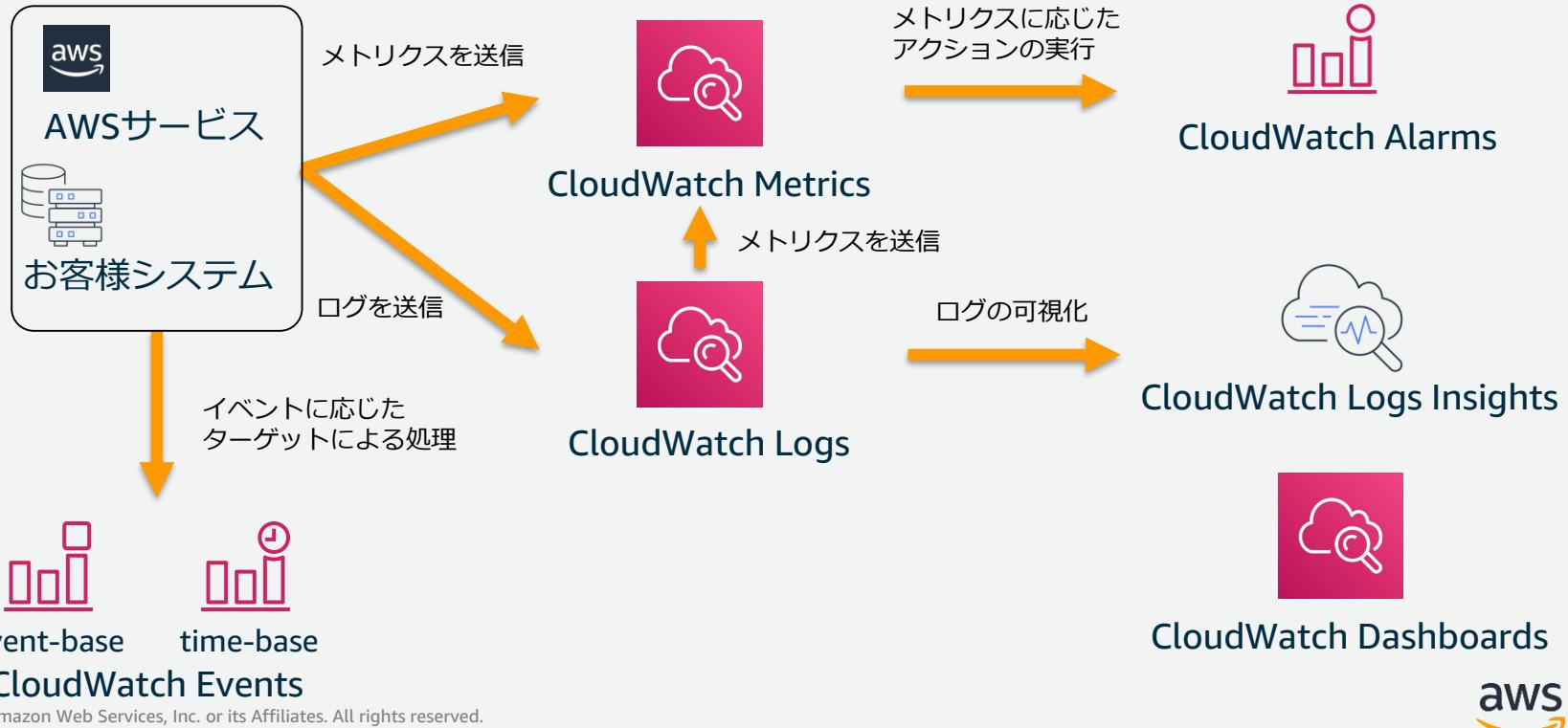
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



まとめ

まとめ

本セミナーでは Amazon CloudWatch が提供する機能を紹介



まとめ

- Amazon CloudWatch はAWSリソース、アプリケーション、オンプレミスのモニタリングサービス
- Metrics / Alarms / Logs / Logs Insights / Dashboards / Eventsを活用することで効率の良い運用をするための機能提供

Q&A

お答えできなかったご質問については

AWS Japan Blog 「<https://aws.amazon.com/jp/blogs/news/>」にて
資料公開と併せて、後日掲載します。

AWS の日本語資料の場所「AWS 資料」で検索

The screenshot shows the homepage of the AWS Cloud Service Utilization Document Collection. The background is dark blue with a light blue network-like graphic. The title 'AWS クラウドサービス活用資料集トップ' is displayed prominently in large white font. Below the title is a detailed explanatory paragraph in Japanese. At the bottom, there are three buttons: 'AWS Webinar お申込 »' (Yellow button), 'AWS 初心者向け »' (White button), and 'サービス別資料 »' (White button). The URL 'https://amzn.to/JPArchive' is overlaid at the bottom of the screenshot.

AWS クラウドサービス活用資料集トップ

アマゾン ウェブ サービス (AWS) は安全なクラウドサービスプラットフォームで、ビジネスのスケールと成長をサポートする処理能力、データベースストレージ、およびその他多種多様な機能を提供します。お客様は必要なサービスを選択し、必要な分だけご利用いただけます。それらを活用するために役立つ日本語資料、動画コンテンツを多数ご提供しております。(本サイトは主に、AWS Webinar で使用した資料およびオンデマンドセミナー情報を掲載しています。)

[AWS Webinar お申込 »](#)

[AWS 初心者向け »](#)

[サービス別資料 »](#)

<https://amzn.to/JPArchive>

ご視聴ありがとうございました

AWS 公式 Webinar
<https://amzn.to/JPWebinar>



過去資料
<https://amzn.to/JPArchive>





このコンテンツは公開から3年以上経過しており内容が古い可能性があります
最新情報については[サービス別資料](#)もしくはサービスのドキュメントをご確認ください

[AWS Black Belt Online Seminar]

Amazon CloudWatch Container Insights で 始めるコンテナモニタリング入門

サービスカットシリーズ

Solutions Architect
水馬 拓也
2019/11/27

AWS 公式 Webinar
<https://amzn.to/JPWebinar>



過去資料
<https://amzn.to/JPArchive>



自己紹介

水馬 拓也 (みずま たくや)

- 所属

アマゾン ウェブ サービス ジャパン 株式会社
技術統括本部 ソリューション アーキテクト



- 好きなサービス



AWS Fargate



AWS Lambda

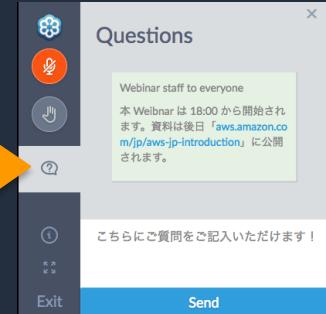
AWS Black Belt Online Seminar とは

「サービス別」「ソリューション別」「業種別」のそれぞれのテーマに分かれて、Amazon ウェブ サービス ジャパン株式会社が主催するオンラインセミナーシリーズです。

質問を投げることができます！

- 書き込んだ質問は、主催者にしか見えません
- いただいたQ&Aをピックアップしてblogに紹介させていただく場合がございます
- 今後のロードマップに関するご質問はお答えできませんのでご了承下さい

- ①吹き出しをクリック
- ②質問を入力
- ③Sendをクリック



Twitter ハッシュタグは以下をご利用ください
#awsblackbelt

内容についての注意点

- 本資料では2019年11月27日時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<http://aws.amazon.com>)にてご確認ください。
- 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます。
- 価格は税抜表記となっています。日本居住者のお客様が東京リージョンを使用する場合、別途消費税をご請求させていただきます。
- AWS does not offer binding price quotes. AWS pricing is publicly available and is subject to change in accordance with the AWS Customer Agreement available at <http://aws.amazon.com/agreement/>. Any pricing information included in this document is provided only as an estimate of usage charges for AWS services based on certain information that you have provided. Monthly charges will be based on your actual use of AWS services, and may vary from the estimates provided.

はじめに

- 想定聴講者:
 - 今後、Production環境でのコンテナ運用を検討されている方
 - コンテナワークロードにおけるモニタリング手法についてお悩みを持たれている方
- 本セッションで学べること:
 - コンテナワークロードのモニタリングにおいて必要な観点
 - Container Insightsを用いたコンテナワークロードのモニタリング方法、及び課題解決の手法

本日の内容

- **前提知識の整理**
 - モニタリングのスコープについて
 - コンテナオーケストレーション、Amazon ECSの主要要素
 - Container Insights登場前のコンテナモニタリング
- **Container Insightsの紹介**
 - Container Insightsの概要
 - 開始方法
 - メトリクスの表示方法
 - 収集されるパフォーマンスログの詳細
 - 具体的なユースケース
 - 料金について

本日の内容

- **前提知識の整理**
 - モニタリングのスコープについて
 - コンテナオーケストレーション、Amazon ECSの主要要素
 - Container Insights登場前のコンテナモニタリング
- **Container Insightsの紹介**
 - Container Insightsの概要
 - 開始方法
 - メトリクスの表示方法
 - 収集されるパフォーマンスログの詳細
 - 具体的なユースケース
 - 料金について

モニタリングのスコープについて

本セッションにおけるモニタリングのスコープ

- ・ モニタリングといっても目的によって意味することが様々
- ・ 本セッションではコンテナワークロードにおけるモニタリングに絞る

障害検知

性能劣化検知

コスト効率

不正アクセス検知

コンプライアンス
準拠

監査対応

ユーザ体験の把握

..etc

コンテナオーケストレーションについて

コンテナオーケストレーションツールについて

- 本セッションでは内容を簡潔にするために、コンテナオーケストレーションツールはAmazon ECSを前提にする



Amazon Elastic Container Service



Amazon Elastic Kubernetes Service



Kubernetes
on
Amazon EC2

Amazon ECSとは？

■ コントロールプレーンとして提供



Amazon ECS

Linux & Windows

{...} ECS CLI



コンテナレベルの
ネットワーク構成



高度なタスク配置
戦略



他のAWSサービスと
の連携



グローバル展開



強力なスケジュールエンジン



オートスケーリング

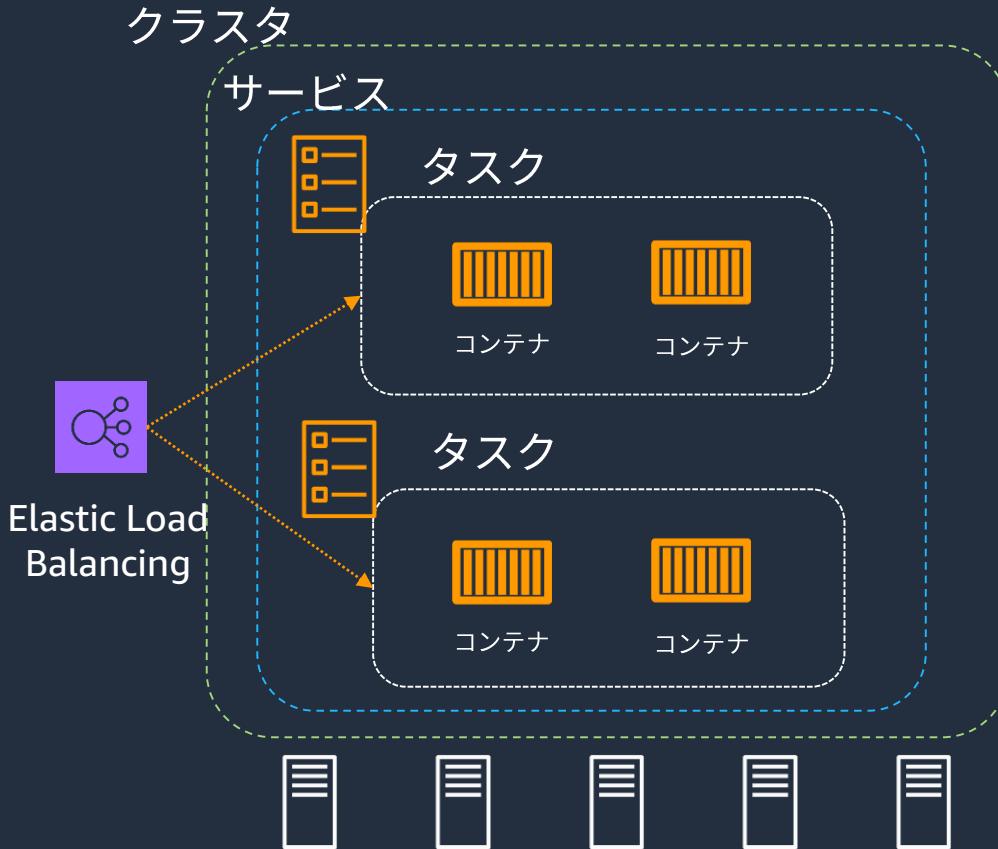


CloudWatch 連携
(ログ/メトリクス/イベント)



ロードバランサー

ECSの構成要素



タスク

- ・コンテナ(群)の実行単位
- ・タスクおよび各コンテナのCPUとメモリ上限を指定し、それを元にスケジュールされる

サービス

- ・指定されたタスク数の維持
- ・ELBとの連携
- ・メトリクスに応じたオートスケール

クラスタ

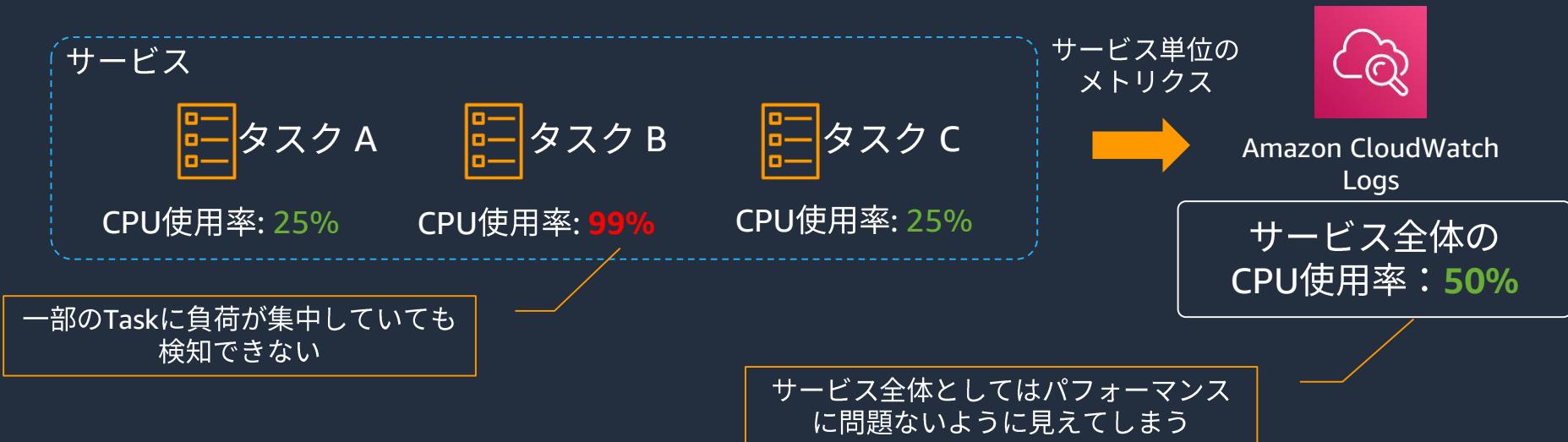
- ・コンテナ実行環境である論理的なグループ

Container Insights登場前の コンテナモニタリング

Container Insights登場前のコンテナモニタリングの例

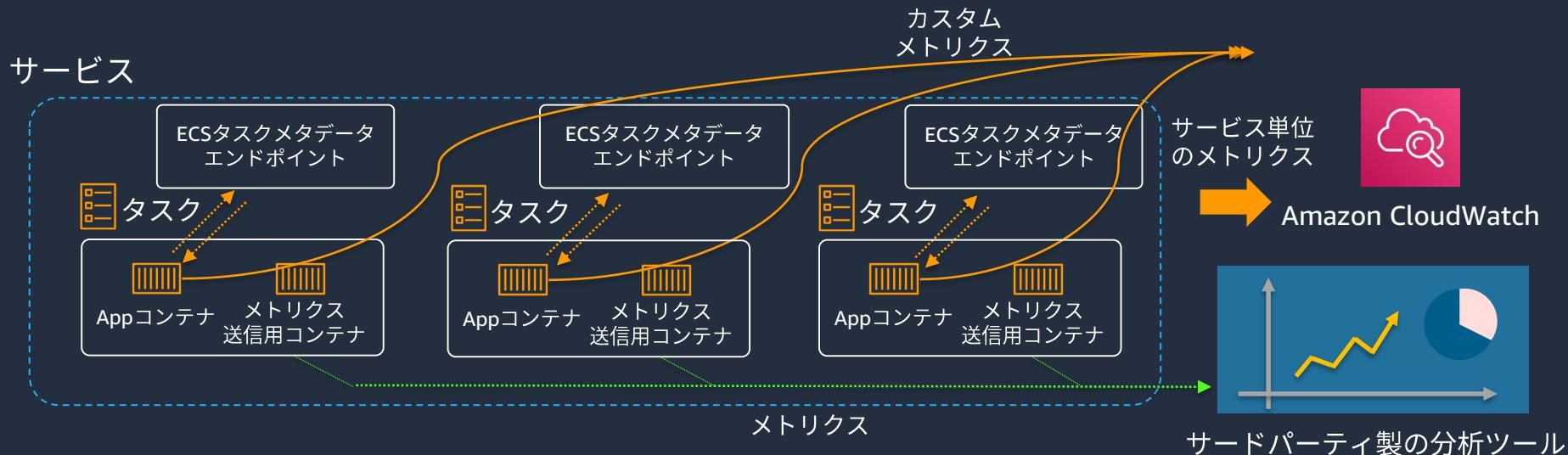
- CloudWatchのデフォルトの設定ではタスク、コンテナ単位のメトリクスが取得できなかった
- タスク、コンテナ単位レベルの問題を検知する際に、より詳細なメトリクス情報が必要となる場合があった

タスク、コンテナ単位レベルの問題の検知が難しい例)



Container Insights登場前のコンテナモニタリングの例

- 各コンテナからECSタスクメタデータエンドポイントを呼び出し、カスタムメトリクスとしてCloudWatchにメトリクスを送信する
- サードパーティ製のメトリクス送信用のコンテナをサイドカーとして配置する



タスクやコンテナレベルのメトリクスを取得するには何かしらの工夫が必要だった

本日の内容

- 前提知識の整理
 - モニタリングのスコープについて
 - コンテナオーケストレーション、Amazon ECSの主要要素
 - Container Insights登場前のコンテナモニタリング
- **Container Insightsの紹介**
 - Container Insightsの概要
 - 開始方法
 - メトリクスの表示方法
 - 収集されるパフォーマンスログの詳細
 - 具体的なユースケース
 - 料金について

Container Insightsの概要

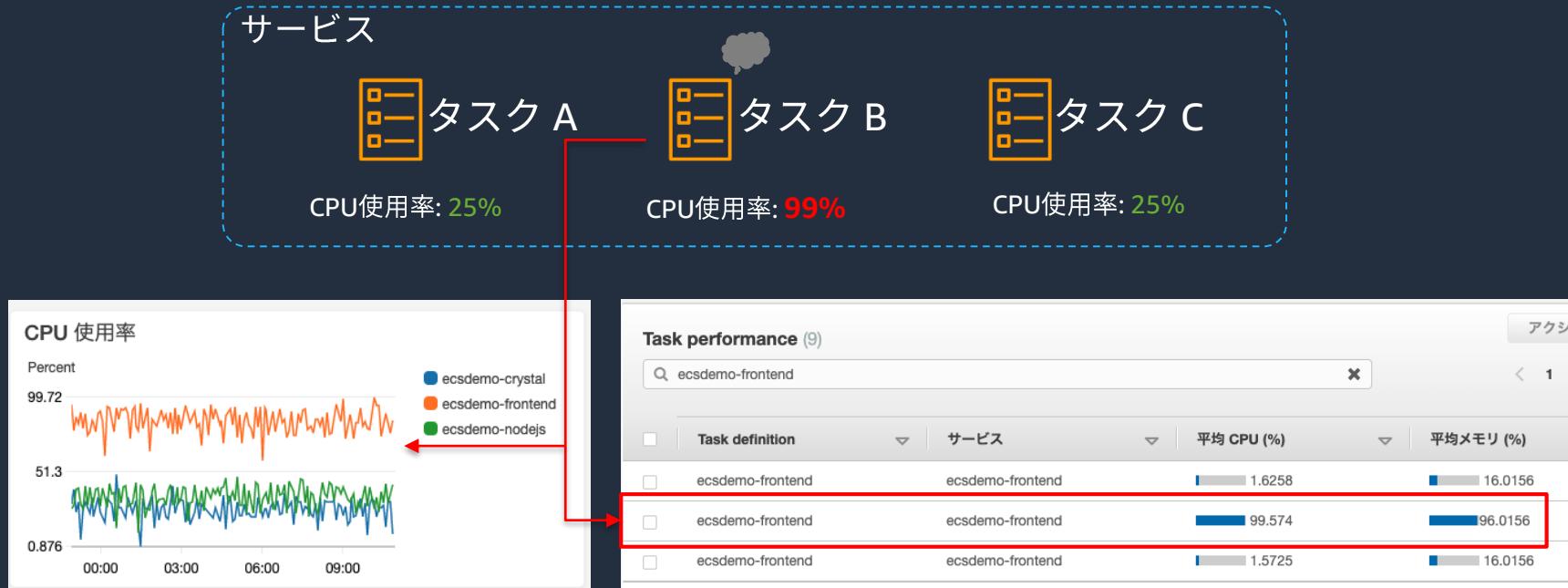
Amazon CloudWatch Container Insightsとは

- コンテナ化されたアプリケーションのメトリクスとログを収集、集計、要約できるCloudWatchの機能の一つ
- CloudWatchにてタスク、コンテナレベルでのモニタリングが可能
- Container Insights が収集するメトリクスは自動的に作成されるダッシュボードに集約され、より鋭い洞察を行うことが可能
- AWSが提供するコンテナオーケストレーションツールであるAmazon ECSや、Amazon EKS、およびAmazon EC2 の Kubernetes プラットフォームでご利用可能

※ 2019/11/27 現在、AWS Batch で Container Insights はサポートされていません。

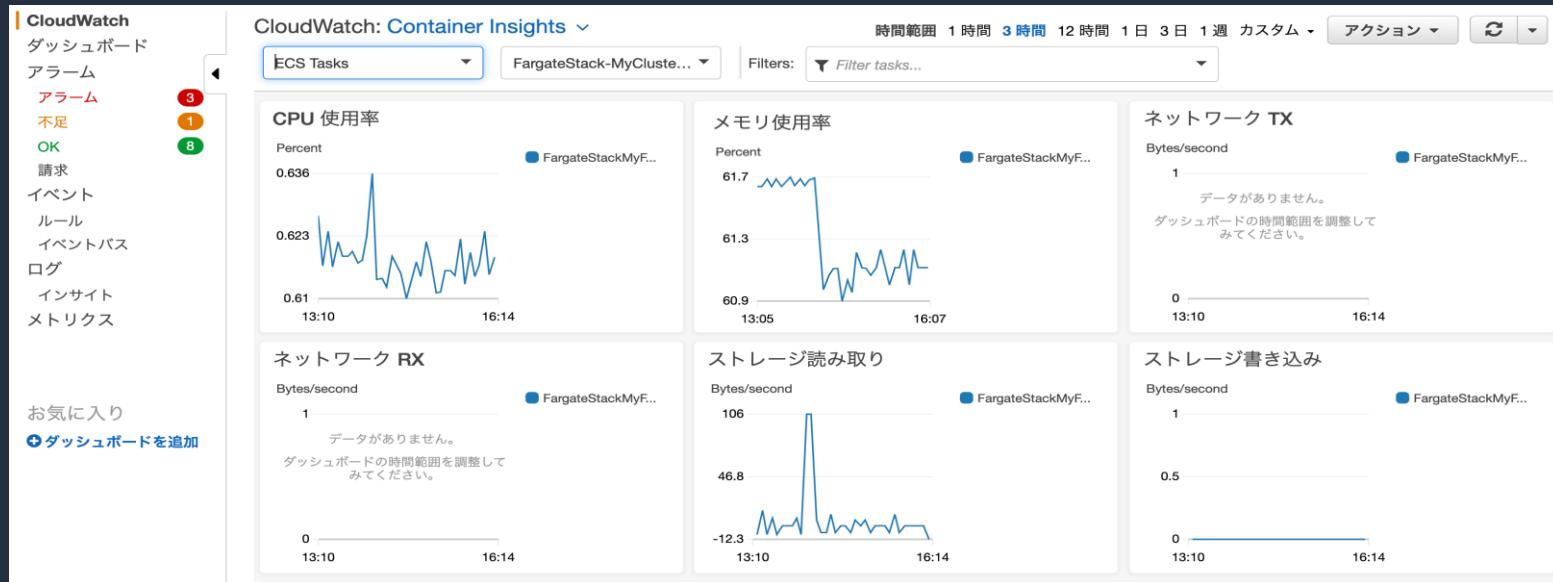
Amazon CloudWatch Container Insightsの概要

- Amazon CloudWatchと統合された、タスクやコンテナレベルでメトリクスやログを取得することが可能



Amazon CloudWatch Container Insightsの概要

- Container Insights が収集するメトリクスは自動的にダッシュボードに集約され、可視化を行うことが可能



Amazon CloudWatch Container Insightsの概要

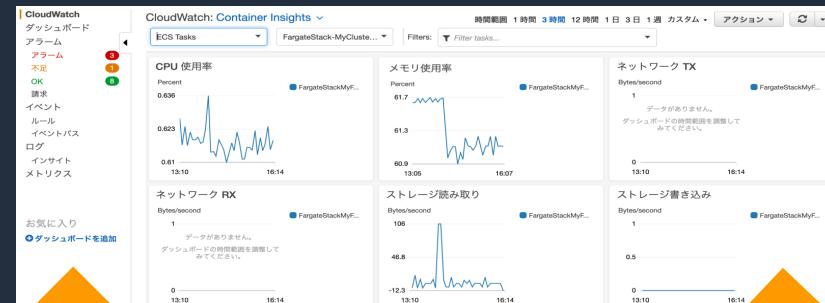
- CloudWatch Logs InsightsやX-Rayとも統合されている
- Container Insightsのダッシュボードを起点により詳細な分析が可能

CloudWatch Logs Insightsの要件の例

- グラフのより詳細な値を見たい
- ログに対し分析クエリを発行したい



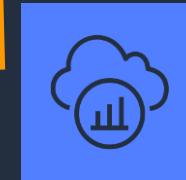
Amazon CloudWatch
Logs Insights



Container Insights

X-Ray を使う要件の例

- タスク間の通信をトレースしたい

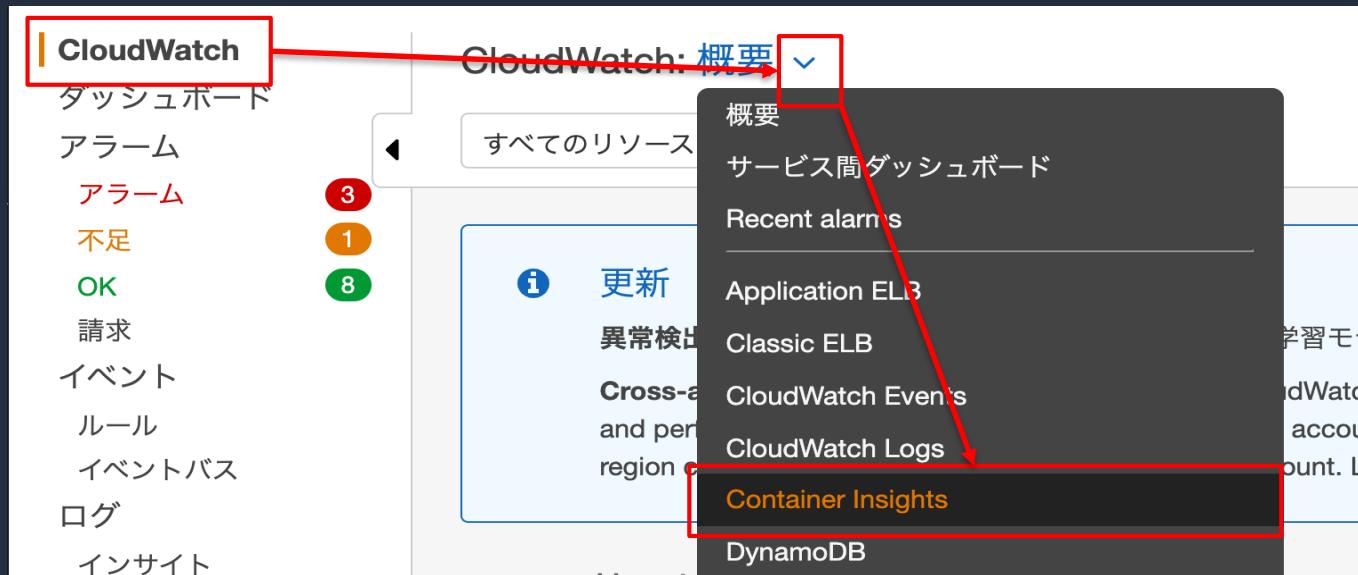


AWS X-Ray

Container Insightsの使用方法

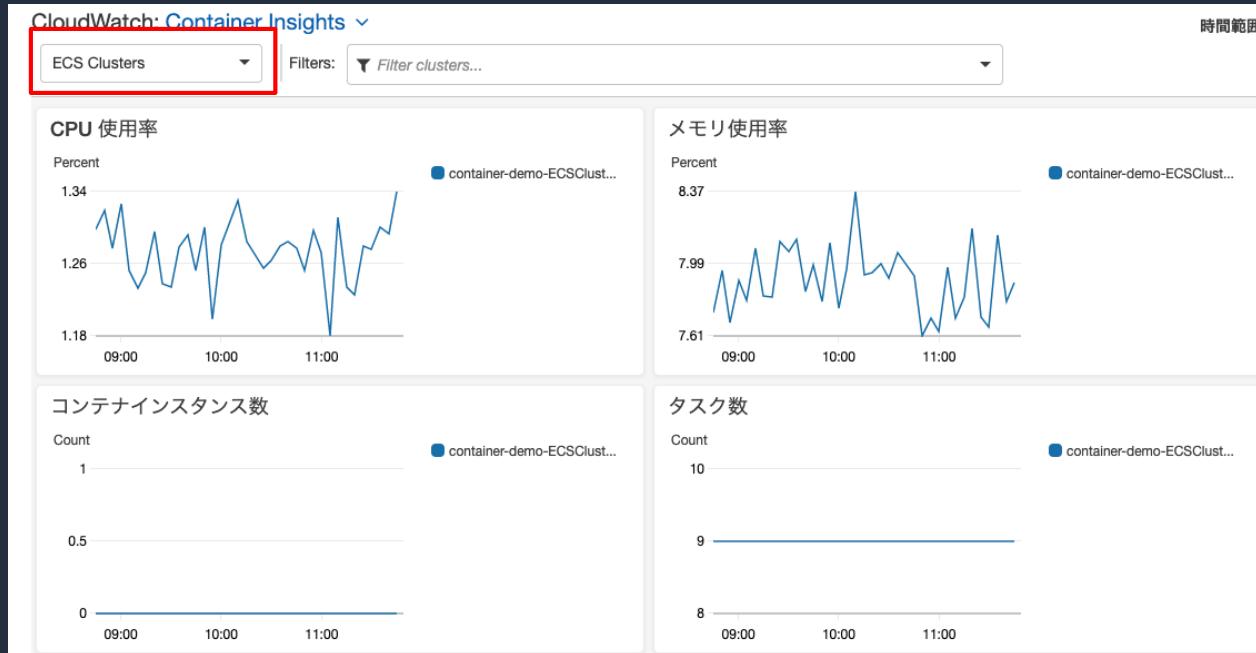
Container Insightsのメトリクスの確認方法 (1)

- CloudWatchのトップ画面を開き「概要」横のプルダウンからの「Container Insights」を選択
- 対象リージョンにContainer Insightsが有効なコンテナが起動していない場合はこの項目が表示されないので注意



Container Insightsのメトリクスの確認方法 (2)

- メトリクスを表示する取得単位を選択する



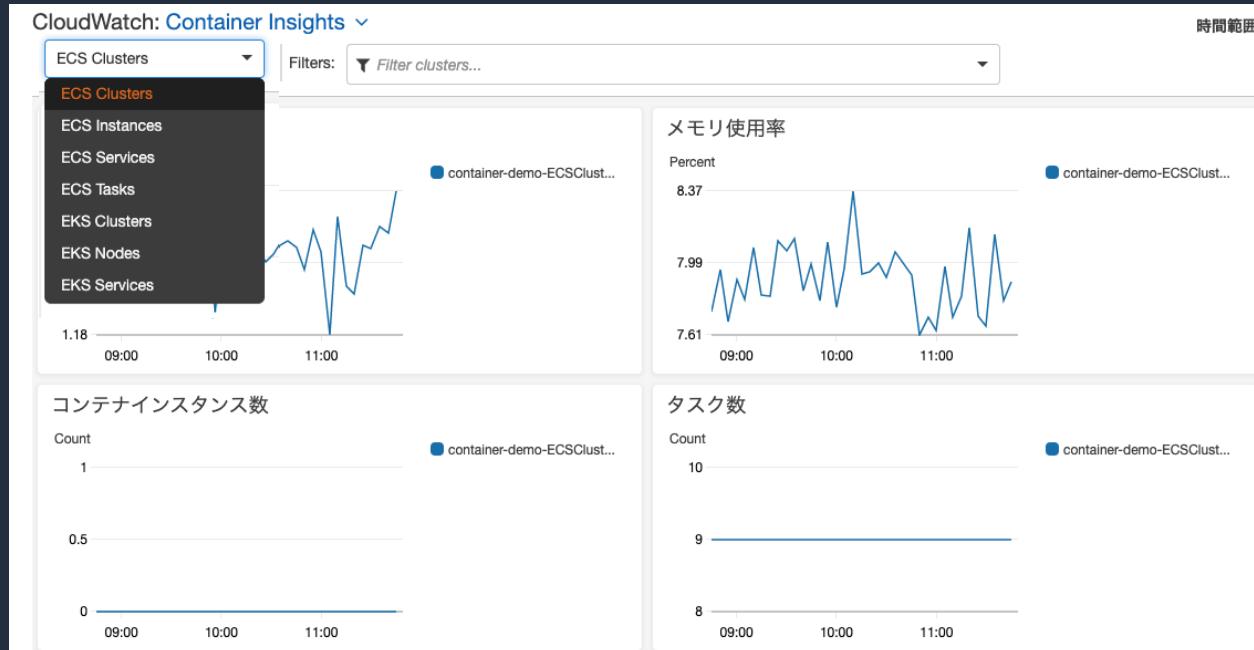
ECSにおける選択可能な単位

- ECS Clusters
- ECS Instances *1
- ECS Services
- ECS Tasks

*1 Fargate 起動タイプでタスクを起動している場合、ECS Instances メトリクスは取得されません

Container Insightsのメトリクスの確認方法 (2)

- メトリクスを表示する取得単位を選択する



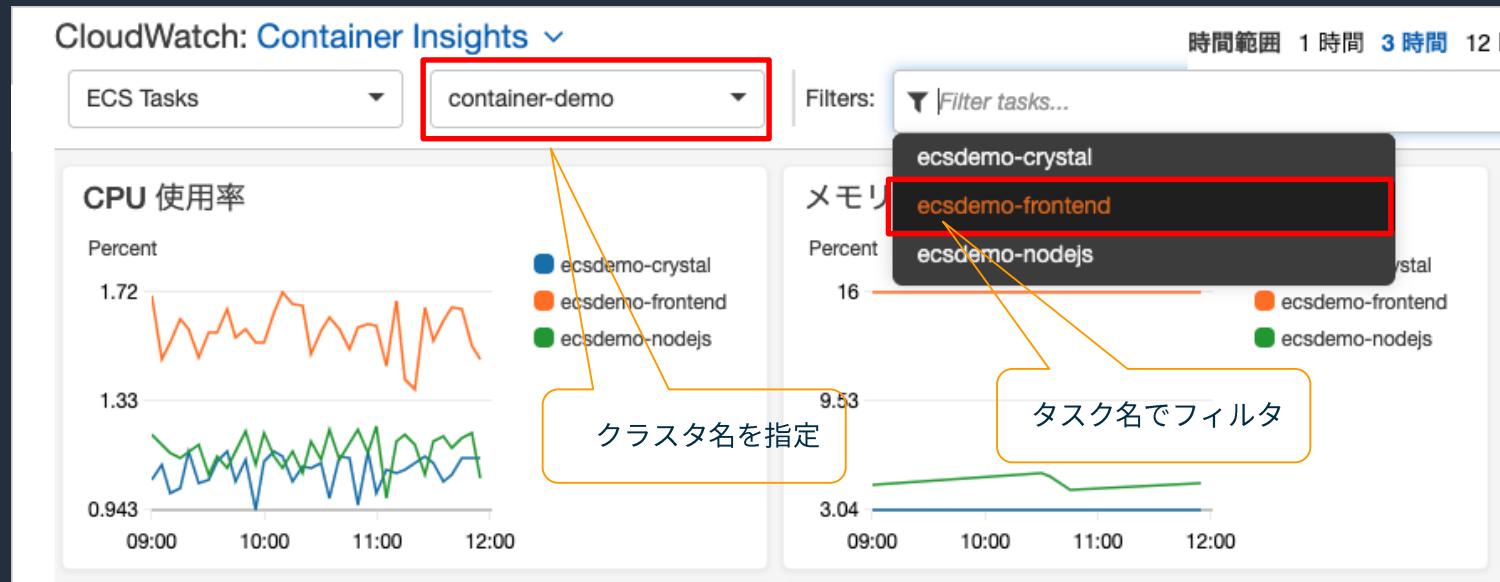
ECSにおける選択可能な単位

- ECS Clusters
- ECS Instances *1
- ECS Services
- ECS Tasks

*1 Fargate 起動タイプでタスクを起動している場合、ECS Instances メトリクスは取得されません

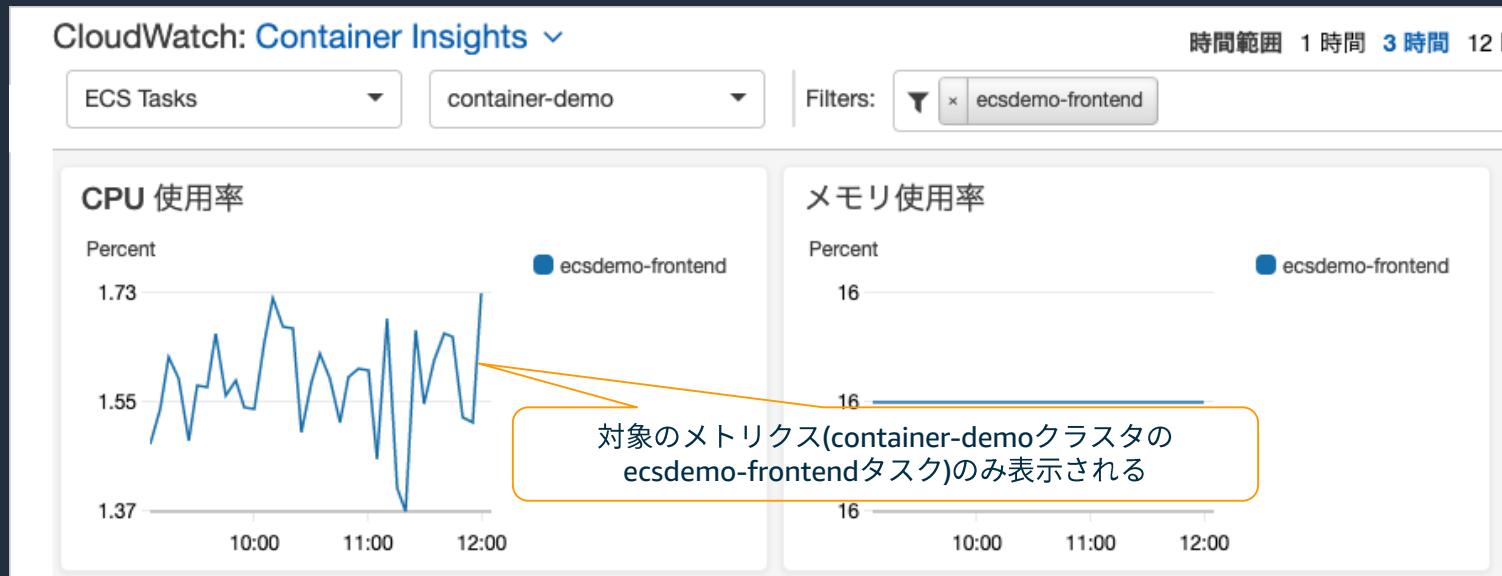
Container Insightsのメトリクスの確認方法 (3)

- 特定のメトリクスのみ表示させるようフィルタをかける事が可能



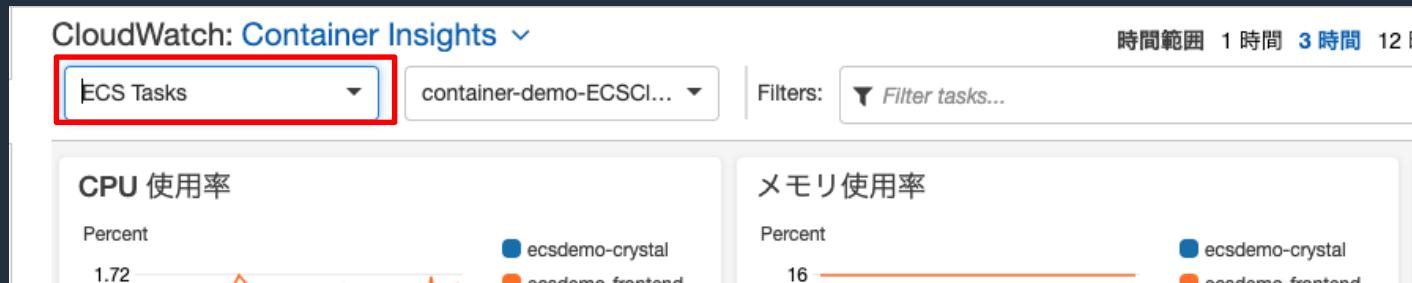
Container Insightsのメトリクスの確認方法 (3)

- 特定のメトリクスのみ表示させるようフィルタをかける事が可能

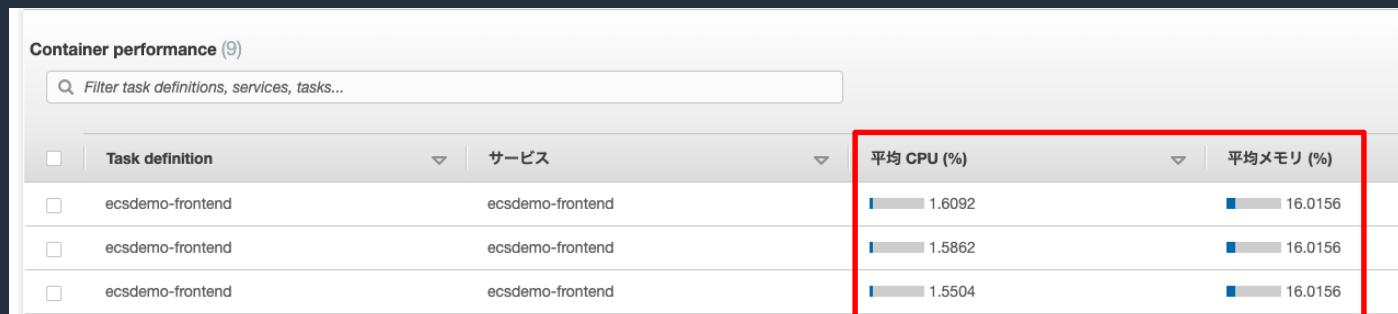


Container Insightsのメトリクスの確認方法 (4)

- コンテナ単位のメトリクスを取得するには「ECS Task」を指定する



：画面下部に移動

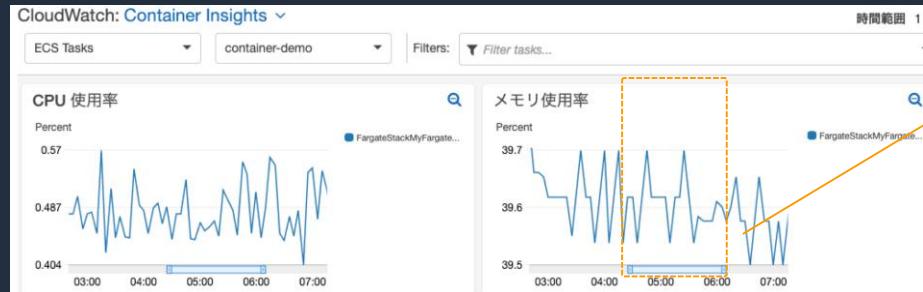


The screenshot shows the 'Container performance' section of the CloudWatch Container Insights interface. It lists 9 items under 'Container performance'. A search bar is present above the table. The table has columns for 'Task definition', 'サービス' (Service), '平均 CPU (%)' (Average CPU %), and '平均メモリ (%)' (Average Memory %). The '平均 CPU (%)' column is highlighted with a red box. The data for the three rows is as follows:

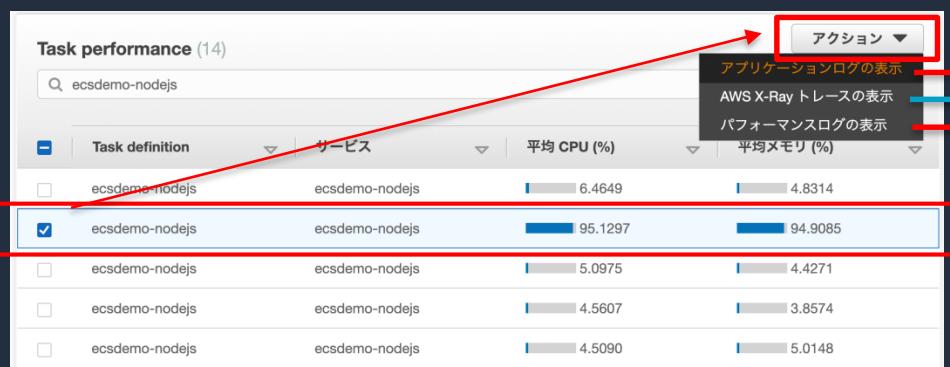
Task definition	サービス	平均 CPU (%)	平均メモリ (%)
ecsdemo-frontend	ecsdemo-frontend	1.6092	16.0156
ecsdemo-frontend	ecsdemo-frontend	1.5862	16.0156
ecsdemo-frontend	ecsdemo-frontend	1.5504	16.0156

タスク単位のドリルダウンを行う

- Task performanceからCloudWatch Logs Insights、X-Rayの画面へ遷移
- タスクやコンテナを選択し「アクション」からより詳細な分析が可能



取得するログの期間を指定可能



Amazon CloudWatch
Logs Insights

- グラフのより詳細な値を確認
- ログに対し分析クエリを発行
- アプリケーションログの確認



AWS X-Ray

- タスク間の通信をトレース



アプリケーションログの表示



Amazon CloudWatch
Logs Insights

- Cloudwatch Logs Insightsの画面に遷移し、対象タスクが\$outputするアプリケーションログやエラーログに対し抽出条件を指定したクエリを発行する事ができる
 - 期間を絞った検索や、統計情報の取得も可能
- 例) アプリケーションログから「Exception」という文字を含むログを集計し、時系列で件数を可視化



対象のタスク、コンテナが指定された状態

クエリの発行

ヒットしたログの数を時系列で表示

ヒットしたログの表示

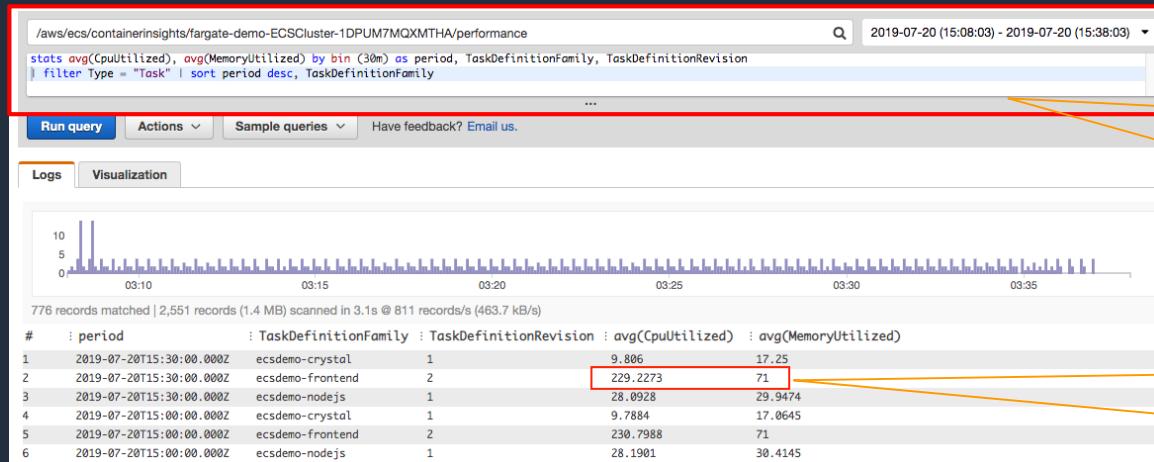


Amazon CloudWatch
Logs Insights

パフォーマンスログの表示

- CloudWatch Logs Insightsの画面に遷移し対象タスクが output するパフォーマンスログに対し抽出条件を指定したクエリを発行する事ができる
- 自動ダッシュボード画面で表示されていたメトリクスをさらにドリルダウンして分析が可能

例) パフォーマンスが悪化しているタスクのボトルネックを調査



調査クエリの発行

パフォーマンスのボトルネックを調査



クエリ構築のためのヘルプ機能



Amazon CloudWatch
Logs Insights

- 一般的なクエリを「サンプルクエリ」という雛形として提供
- 画面右の「クエリのヘルプ」からコマンド構築していくことも可能

The screenshot shows the Amazon CloudWatch Logs Insights interface. On the left, there's a search bar with the URL '/aws/ecs/containerinsights/FargateStac...', a date range selector (2019-11-26 to 2019-11-26), and a query editor containing:

```
filter @message like /Exception/
| stats count(*) as exceptionCount by bin(5m)
| sort exceptionCount desc
```

Below the query editor are two buttons: 'クエリの実行' (Run Query) and 'アクション' (Actions). A dropdown menu is open over the 'サンプルクエリ' (Sample Query) button, which is highlighted with a red box. The dropdown menu lists several types of queries:

- Lambda のクエリ
- VPC フローログのクエリ
- CloudTrail のクエリ
- 一般的なクエリ
- Route 53 クエリ
- AWS AppSync クエリ

For the '一般的なクエリ' option, two results are shown in a tooltip:

- 最近追加された 25 件のログイベント
- 5 分ごとにログに記録される例外の数

To the right of the main interface is a sidebar titled 'クエリのヘルプ' (Query Help) with a '詳細はこちら' (More details) link. The sidebar contains sections for 'コマンド' (Commands) and '検出されたフィールド' (Detected Fields), along with a search bar for fields.

クエリ構築のためのヘルプ機能



Amazon CloudWatch
Logs Insights

- 一般的なクエリを「サンプルクエリ」という雛形として提供
- 画面右の「クエリのヘルプ」からコマンド構築していくことも可能

The screenshot shows the AWS CloudWatch Logs Insights console. On the left, there's a query editor window with a sample query:

```
filter @message like /Exception/  
| stats count(*) as exceptionCount by bin(5m)  
| sort exceptionCount desc
```

Below the editor are tabs for "クエリの実行" (Run Query) and "アクション" (Actions). A dropdown menu is open under "サンプルクエリ" (Sample Query), showing various query types: Lambda のクエリ, VPC フローログのクエリ, CloudTrail のクエリ, 一般的なクエリ, Route 53 クエリ, and AWS AppSync クエリ. The "一般的なクエリ" option is expanded, showing two examples: "最近追加された 25 件のログイベント" and "5 分ごとにログに記録される例外の数".

To the right of the editor, a sidebar titled "クエリのヘルプ" (Query Help) provides documentation for the query language. It includes sections for "コマンド", "fields", and "parse". A red box highlights the "filter 追加" (Add filter) button under the "filter" section. Below it, a list of detected fields is shown, with "CloudWatchMetrics.0.Dime..." partially visible.

分析のためのサンプルクエリ for EKS



Amazon CloudWatch
Logs Insights

コンテナ名ごとの CPU 使用率(パフォーマンスログ)

```
stats pct(container_cpu_usage_total, 50) as CPUpercMedian by kubernetes.container_name  
| filter Type="Container"
```

ノードの平均CPU使用率(パフォーマンスログ)

```
STATS avg(node_cpu_utilization) as avg_node_cpu_utilization by NodeName | SORT  
avg_node_cpu_utilization DESC
```

クラスタノードの障害数(パフォーマンスログ)

```
stats avg(cluster_failed_node_count) as CountOfNodeFailures | filter Type="Cluster" | sort @timestamp  
desc
```

アプリケーションログエラー (アプリケーションログ)

```
Count by container name: stats count() as countoferrors by kubernetes.container_name | filter  
stream="stderr" | sort countoferrors desc
```

■ Container Insights メトリクスの表示

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/Container-Insights-view-metrics.html

AWS X-Ray トレスの表示



AWS X-Ray

- リクエスト単位の情報をトレース可能
 - リクエストのエラー率
 - レスポンスタイムの統計情報 .etc
- Container InsightsでX-Rayのトレース情報の取得するにはアプリケーション用コンテナのサイドカーとしてX-Rayデーモンを実行するコンテナを配置する
例) X-Rayデーモン用コンテナをサイドカーとして配置する



AWS X-Ray コンソールで可視化

AWS X-Ray

Traces 詳細

タイムライン | Raw データ

ID: 1-5dcda055-54a1c088e0ffefad711e40

メソッド レスポンス 所要時間 間隔 ID

GET 200 795 ms 1.3 hr (2019-11-14 23:25:09 UTC) 1-5dcda055-54a1c088e0ffefad711e40

名前 レスポンス 所要時間 ステータス

xray-fargate-a-eb-94232038.ap-northeast-1.ebs.amazonaws.com AWS EBS Container

xray-fargate-a-eb-94232038.ap-northeast-1.ebs 200 795 ms

xray-fargate-b-eb-1332454598.ap-northeast-1.ebs 403 795 ms

xray-fargate-b-eb-1332454598.ap-northeast-1.ebs 403 6.0 ms

参考 AWS X-Ray

https://docs.aws.amazon.com/ja_jp/xray/latest/devguide/aws-xray.html

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



Container Insightsの開始方法

Container Insightsを有効にする～ECS編～

新規のクラスタに対し有効化する

- マネジメントコンソールの「ステップ 2: クラスターの設定」画面にて「Container Insights を有効にする」にチェックを入れる

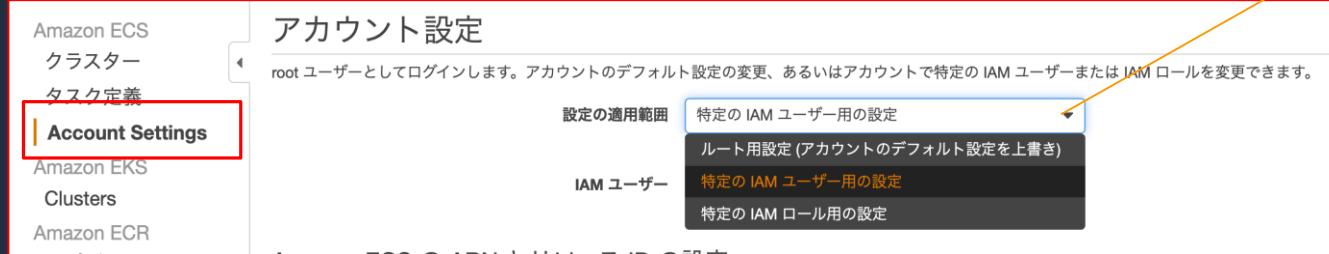


- AWS CloudFormationやAWS Cloud Development Kit(CDK)から作成する場合
 - アカウント単位、IAMロール単位、IAMユーザ単位のいずれかでContainer Insightsがオプトインされている必要がある

Container Insightsを有効にする～ECS編～

- Container Insightsのオプトインについて

- Container Insightsは以下の単位でデフォルトで有効化することが可能



- アカウント全体
- IAMロール
- IAMユーザ

- 画面下部へ移動

リソース	アカウントデフォルト設定を上書き
Container Insights	<input checked="" type="checkbox"/>

適用範囲でContainer Insightsがデフォルトで有効になる

Container Insightsを有効にする～ECS編～

- Container Insightsのオプトインについて

- Container InsightsのオプトインはAWS CLIからも実行可能

アカウント単位でのオプトイン

```
aws ecs put-account-setting-default --name containerInsights --value enabled --region us-east-1
```

ユーザ単位でのオプトイン

```
aws ecs put-account-setting --name containerInsights --value enabled --principal-arn arn:aws:iam::aws_account_id:user/userName --region us-east-1
```

ロール単位でのオプトイン

```
aws ecs put-account-setting --name containerInsights --value enabled --principal-arn arn:aws:iam::aws_account_id:role/roleName --region us-east-1
```

Container Insightsを有効にする～ECS編～

既存のクラスタに対し有効化する

- Container Insights は、既存の Amazon ECS クラスタ及び、新規に作成したクラスターで有効にできる
- 既存のクラスタでContainer Insightsを有効にするには、AWS CLIから以下のコマンドを発行

```
$ aws ecs update-cluster-settings --cluster <myCICluster> --settings  
name=containerInsights,value=enabled
```

※ AWS CLIのバージョンが1.16.200以降が必要

Amazon ECS での Container Insights のセットアップ

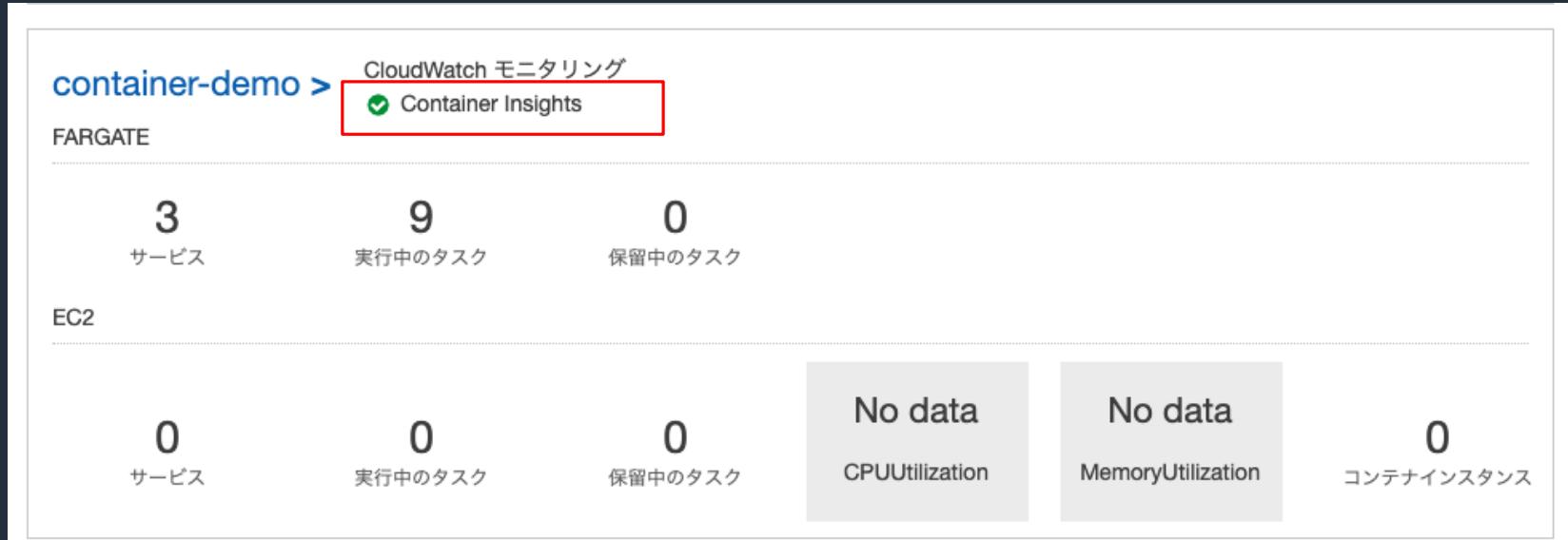
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/deploy-container-insights-ECS.html

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



Container Insightsを有効にする～ECS編～

- Container Insightsが有効かどうかは、ECSのクラスター画面で「Container Insights」に緑色のチェックが入っているかどうかで確認できる



Container Insightsを有効にする～EKS、Kubernetes on EC2編～

Container Insightsを有効にするための前提条件を確認

- EKS / Kubernetes on EC2 で必要な条件
 - Container Insightsが利用可能なリージョンでクラスタが起動している
 - kubectl がインストール、権限設定が完了しており、kubectl apply が実行可能な状態になっている
- Kubernetes on EC2 で必要な条件
 - Kubernetes クラスターで、ロールベースのアクセス制御 (RBAC) が有効になっている
 - kubelet で Webhook 認証モードが有効になっている
 - コンテナランタイムがDockerになっている

Amazon EKS と Kubernetes で Container Insights をセットアップする 前提条件の確認

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/Container-Insights-prerequisites.html

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



Container Insightsを有効にする～EKS、Kubernetes on EC2編～

Container Insightsを有効にする手順

- 必須手順
 - CloudWatch にメトリクスを送信するCloudWatchエージェントをDaemonSetとしてセットアップ
 - CloudWatch Logsにログを送信するDaemonSetとしてFluentDをセットアップ
- オプション手順
 - Amazon EKS コントロールプレーンのログ記録を有効する(コントロールプレーンのログをCloudWatch Logsで監査する必要がある場合)
 - StatsD エンドポイントとして CloudWatch エージェントをセットアップする(StatsDを使用している場合)

【Amazon EKS と Kubernetes で Container Insights をセットアップする】

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/deploy-container-insights-EKS.html

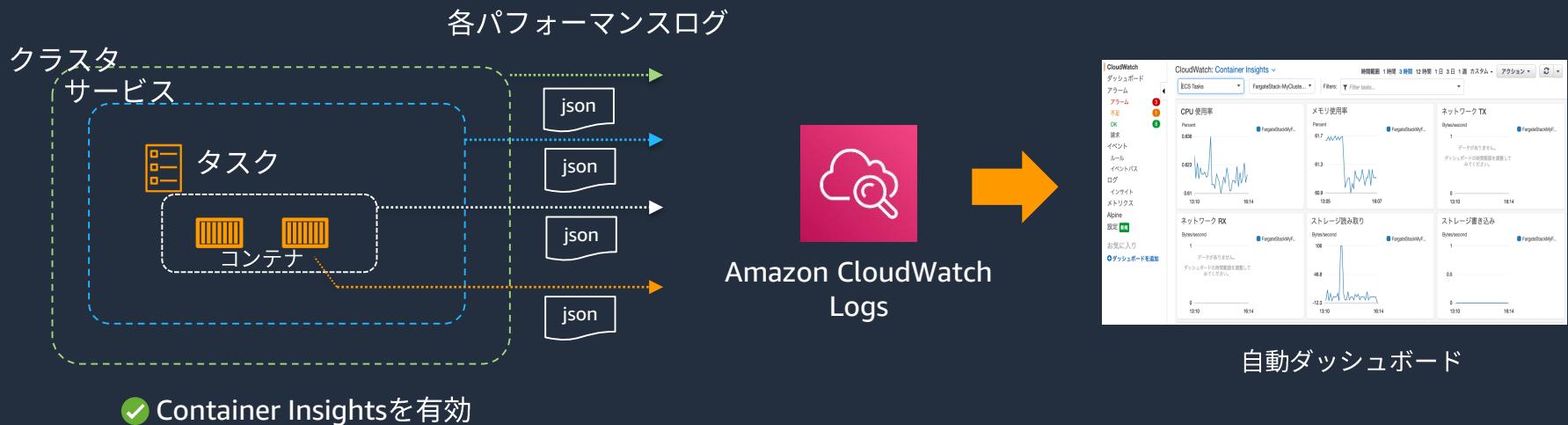
© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



収集されるパフォーマンスログの詳細

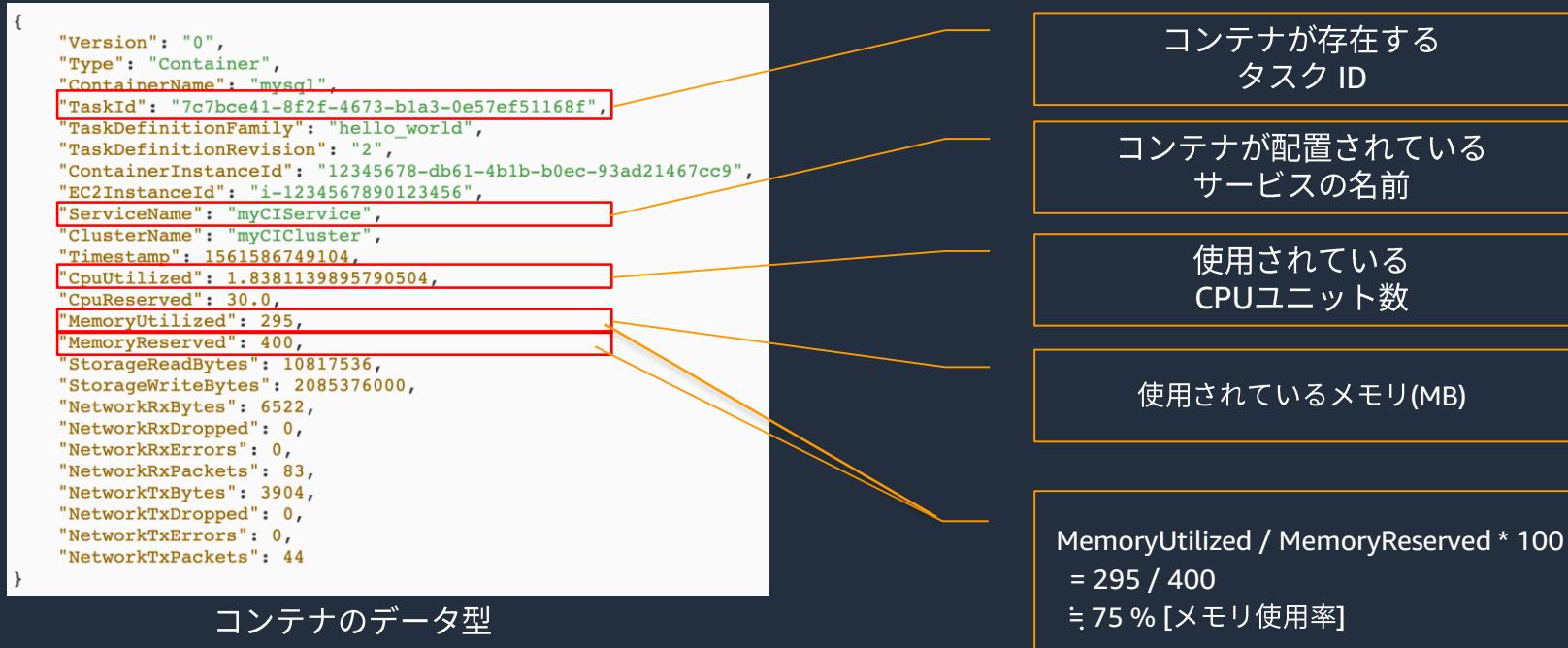
パフォーマンスログとは？

- Container Insightsの機能を有効にするとカスタムメトリクスとしてパフォーマンスログがラスタ単位、サービス単位、タスク単位、コンテナ単位でCloudWatch Logsに出力される
- パフォーマンスログをもとに自動ダッシュボード上で可視化が行われる
- 通常のカスタムメトリクス同様、ロググループやCloudWatch Logs Insightsからクエリで分析することも可能



パフォーマンスログの内容

- パフォーマンスログの各項目のデータ型をドキュメントで公開



■ Amazon ECS の Container Insights パフォーマンスログイベント

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/Container-Insights-reference-performance-logs-ECS.html

■ Amazon EKS の Container Insights パフォーマンスログイベント

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/Container-Insights-reference-performance-logs-EKS.html

パフォーマンスログの確認方法

- パフォーマンスログはCloudWatch Logsのロググループからも確認可能
 - ECSの場合 : /aws/ecs/containerinsights/<クラスタ名>/performance
 - EKSの場合 : /aws/containerInsights/<クラスタ名>/performance

例) container-demo clusterに配置されているcontainer のパフォーマンスログ

CloudWatch > ロググループ > /aws/ecs/containerinsights/container-demo/performance > FargateTelemetry-6938

イベントのフィルター

時間 (UTC +00:00)	メッセージ
2019-11-25	<p>いまのところ古いイベントはありません。再試行。</p> <pre>{"Version": "0", "Type": "Container", "ContainerName": "~internal-ecs-pause", "TaskId": "951aaef28-dc8e-4bb2-ac2b-298ff24f7063", "TaskDefinition": "951aaef28-dc8e-4bb2-ac2b-298ff24f7063", "TaskDefinitionFamily": "ecsdemo-nodejs", "TaskDefinitionRevision": "1", "ServiceName": "ecsdemo-nodejs", "ClusterName": "container-demo", "Timestamp": 1574709240000, "CpuUtilized": 4.148334635963217, "CpuReserved": 0, "MemoryUtilized": 15, "StorageReadBytes": 3420160, "StorageWriteBytes": 0}</pre>

CloudWatch Logs Insightsからのパフォーマンスログの確認方法

- ・ パフォーマンスログ内容は@messageフィールドに保持される
- ・ CpuUtilized項目を取得したい場合は@message.CpuUtilizedで取得が可能

例) タスクのCpuUtilizedとMemoryUtilized項目を取得する

The screenshot shows the CloudWatch Logs Insights interface. The URL bar says '/aws/ecs/containerin...'. The time dropdown is set to '2019-11-01 (00:00)'. The query editor contains the following code:
fields @message.CpuUtilized, @message.MemoryUtilized
| filter hasTaskId and TaskId like /7bd7f800-c3ee-4dct-8301-728f0beb4c5/ and Type = "Task"
The 'クエリの実行' (Run Query) button is highlighted in blue.

The screenshot shows the results of the query. The URL bar is the same as the previous screenshot. The results table shows the following fields:

名前	説明
MemoryUtilized	フィールド
MemoryReserved	フィールド
Timestamp	フィールド
@timestamp	フィールド
CloudWatchMetrics.0.Metrics.0.Name	フィールド
CloudWatchMetrics.0.Metrics.0.Unit	フィールド
CloudWatchMetrics.0.Metrics.1.Name	フィールド
CloudWatchMetrics.0.Metrics.1.Unit	フィールド

The 'ログ' (Log) tab is selected at the bottom.

対象フィールド項目を
補完してくれる

具体的なユースケース

Big Data

ユースケース1.

ECSでタスクに配置するコンテナのリソースを適切な サイズにチューニングしたい

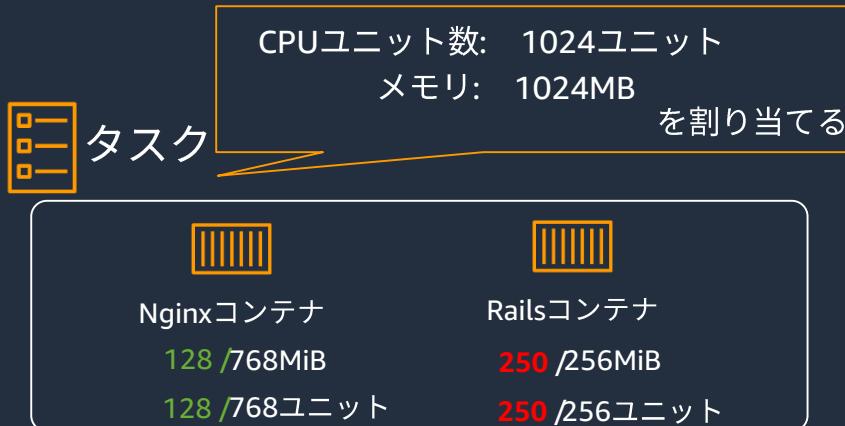


ユースケースの詳細

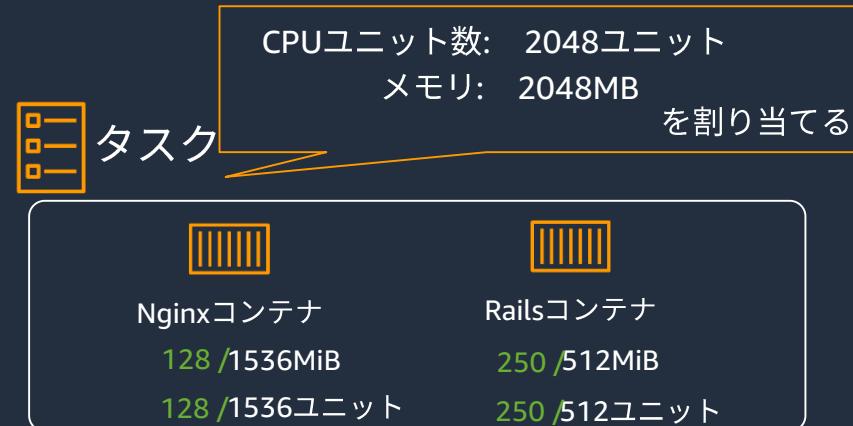
- ・ コンテナごとの適切なリソースの配分は非常に重要
- ・ 適切でないリソース配分はパフォーマンスの劣化や、過剰なリソースの確保、リソース不足によるプロセスの停止につながる可能性がある

例) リソースが適切に配分できていない例

- ・ タスク全体ではリソースに余力があるが、片方のコンテナでリソースが足りていない

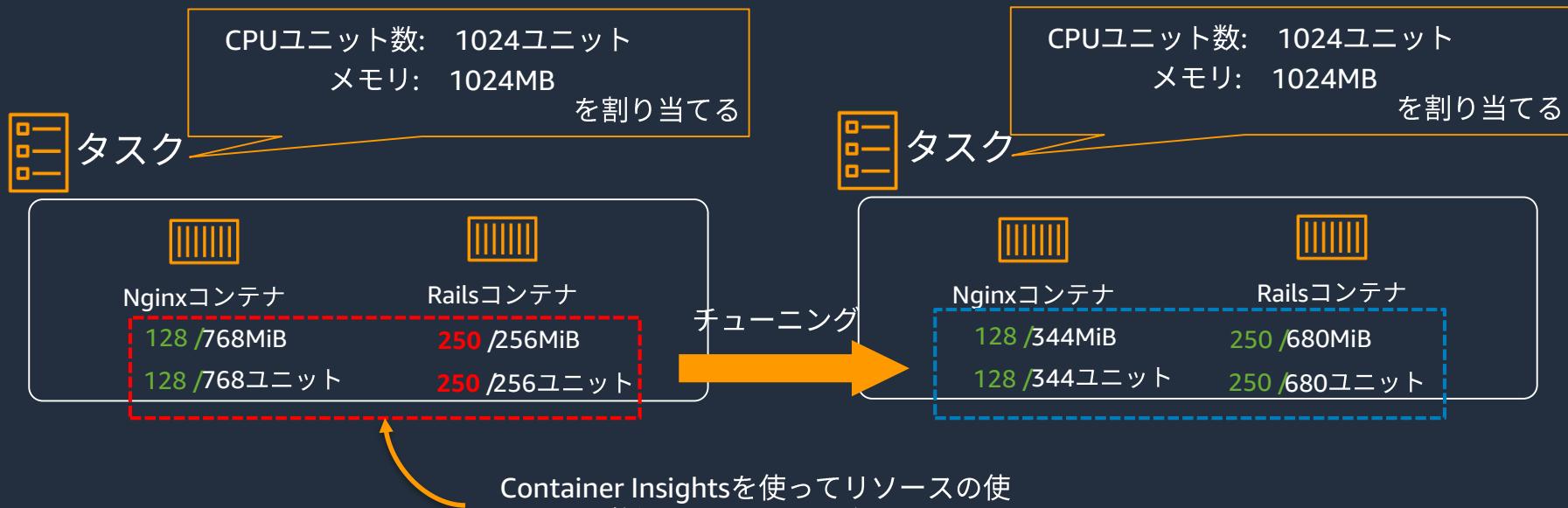


- ・ どちらのコンテナにも余力があるが、タスク全体でリソースを効率的に使えていない状態



Container Insightsを使って適切なリソース配分を行う

- Container Insightsはコンテナ単位のリソースモニタリングが可能であるため、各コンテナに適切なリソース配分が行われているかを確認できる
- 適切なリソース配分により、効率的かつ安全に運用できるリソースを割り当てる



ステップ1. Containerごとのメトリクスを確認する

- Container Insightsの自動ダッシュボードから「ECS Tasks」を選択し、Containerごとのパフォーマンスを確認する



The screenshot shows a table titled 'Container performance' with four columns: Task definition, コンテナ名 (Container Name), 平均 CPU (%), and 平均メモリ (%). It lists two tasks: 'task-demo' with containers 'container-a' and 'container-b'. Container-a has a higher average CPU usage (89.1227%) compared to container-b (9.6542%). Container-a also has a higher average memory usage (85.6548%) compared to container-b (11.6954%).

Task definition	コンテナ名	平均 CPU (%)	平均メモリ (%)
task-demo	container-a	89.1227	85.6548
task-demo	container-b	9.6542	11.6954

片方のコンテナに負荷が偏っていることが判明した！

ステップ.2 Containerリソース配分を変更する

- ECSコンソールのタスク定義からタスク内のコンテナリソース配分を変更する

container-a : メモリ256MB/CPU 128ユニット
container-b : メモリ256MB/CPU 128ユニット

タスクサイズ

タスクサイズにより、タスクの固定サイズを指定できます。Fargate 起動タイプを使用したタスクにはタスクサイズが必須で、EC2 起動タイプではオプションです。タスクサイズが設定されている場合、コンテナレベルのメモリ設定はオプションです。タスクサイズは Windows コンテナではサポートされません。

タスクメモリ (GB) 0.5GB
0.25 vCPU の有効なメモリ範囲: 0.5GB - 2GB。

タスク CPU (vCPU) 0.25 vCPU
0.5 GB メモリの有効な CPU: 0.25 vCPU

コンテナメモリの予約用のタスクメモリの最大割り当て

コンテナへの CPU の最大割り当て

コンテナの定義

コンテナの追加

コンテナ名	イメージ	ハード/ソフトメモリ...	CPU ユニ...	GPU	Inference Accelerat...	基本
container-a	888727018440.dkr.ec...	256/256	128		true	
container-b	888727018440.dkr.ec...	256/256	128		true	



container-a : メモリ384MB/CPU 192ユニット
container-b : メモリ128MB/CPU 64 ユニット

タスクサイズ

タスクサイズにより、タスクの固定サイズを指定できます。Fargate 起動タイプを使用したタスクにはタスクサイズが必須で、EC2 起動タイプではオプションです。タスクサイズが設定されている場合、コンテナレベルのメモリ設定はオプションです。タスクサイズは Windows コンテナではサポートされません。

タスクメモリ (GB) 0.5GB
0.25 vCPU の有効なメモリ範囲: 0.5GB - 2GB。

タスク CPU (vCPU) 0.25 vCPU
0.5 GB メモリの有効な CPU: 0.25 vCPU

コンテナメモリの予約用のタスクメモリの最大割り当て

コンテナへの CPU の最大割り当て

コンテナの定義

コンテナの追加

コンテナ名	イメージ	ハード/ソフトメモリ...	CPU ユニ...	GPU	Inference Accelerat...	基本
container-a	888727018440.dkr.ec...	384/384	192		true	
container-b	888727018440.dkr.ec...	128/128	64		true	

ステップ3. 再びContainerごとのリソースを確認

- あらためて Container Insights からコンテナごとのリソース使用状況を確認する



適切なリソース割り当てが
できたことを確認！

The Container performance table lists tasks and their associated containers, along with their average resource usage:

Task definition	コンテナ名	平均 CPU (%)	平均メモリ (%)
task-demo	container-a	45.3721	50.2441
task-demo	container-b	42.9787	44.1687

Big Data

ユースケース2.

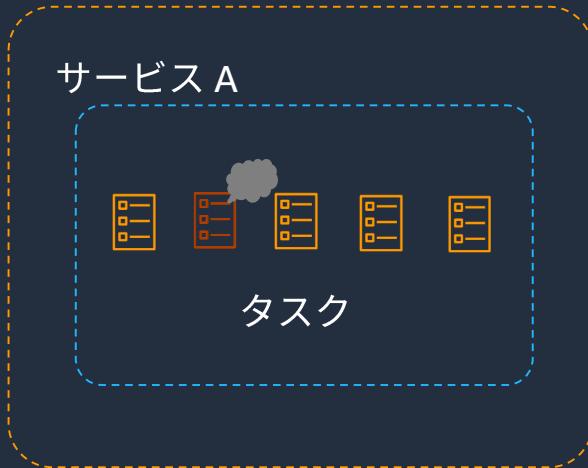
特定のタスクだけで発生している問題の調査を行いたい



ユースケースの詳細

- サービスで起動している複数のタスクのうち、特定のタスクで問題が発生した場合に調査を行う

クラスタ

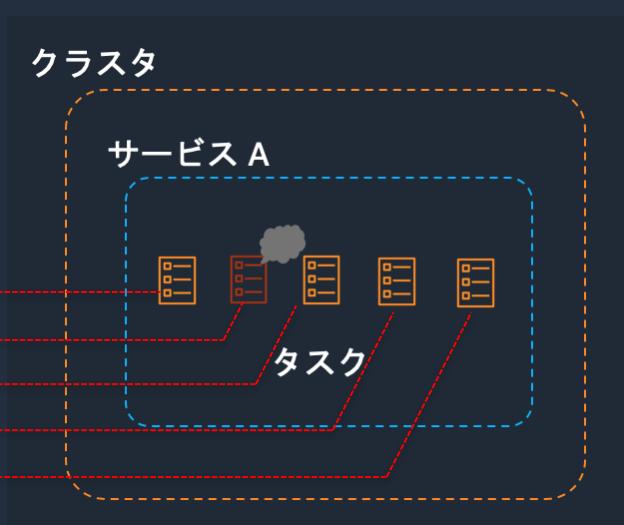
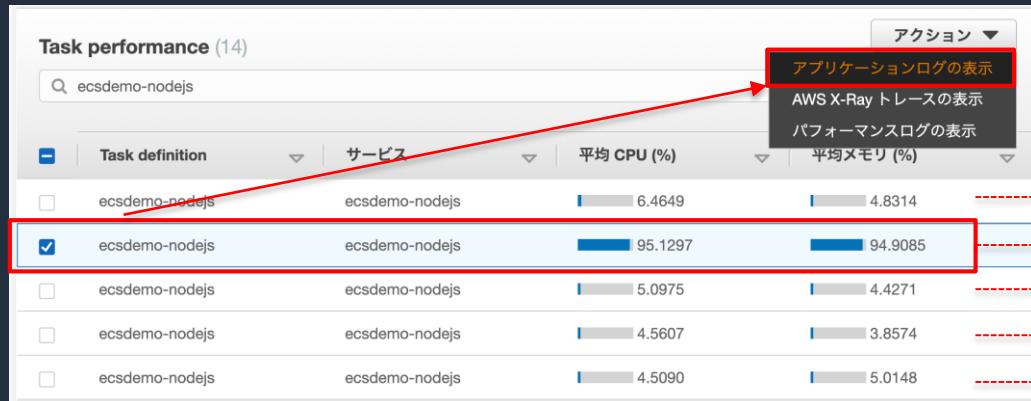


調査例

- 不定期に特定のリクエストのレスポンスが悪化する
- 定期的にタスクの再起動が発生している

ステップ1. タスクごとのパフォーマンスを確認

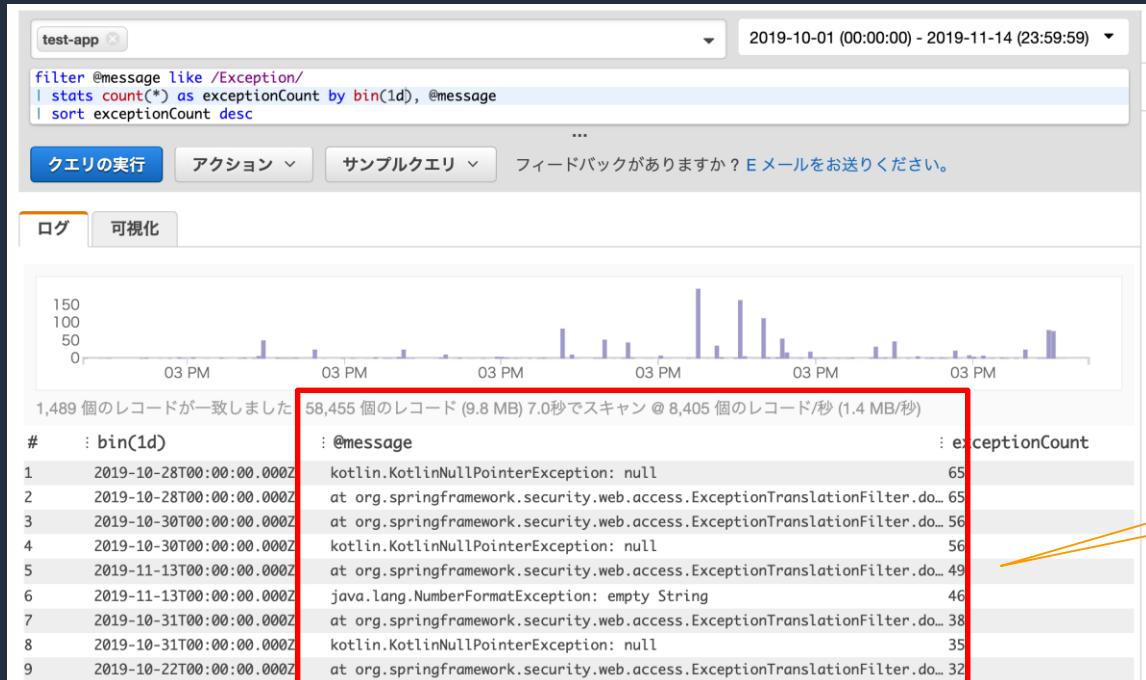
- 特定のタスクのみリソースを著しく消費している
- 問題の特定のため、アプリケーションログを確認する
- 「アクション」>「アプリケーションログの表示」



リソース表示の観点

ステップ2. CPU、メモリ使用率の高いタスクのログを確認

- ・ アプリケーションログから異常が発生していないかを確認
- ・ 異常が発生していない他のタスクとの比較も有効



@messageフィールド
にログの内容が表示
されている

料金について

料金について

Container Insights は以下の項目で課金される

ECSのContainer Insightsの料金の例

前提:

10 個の Amazon EC2 インスタンス、50 個の平均的な実行中のコンテナ、20 個の一意のタスク名、5 個の一意のサービス名をもつ 1 つのコンテナクラスターをモニタリングする場合

CloudWatchカスタムメトリクス費用:

8 個のクラスターメトリクス + (6 個のタスクメトリクス * 20 個の一意のタスク名) + (11 個のサービスメトリクス * 5 個の一意のサービス名) = 183 CloudWatch メトリクス

183 * 0.30USD = 54.90 USD

CloudWatch Logs費用:

Amazon ECS では平均して 1 時間あたりメトリクスごとに 13 KB が取り込まれる
(13 KB/1024/1024) GB * 183 メトリクス * 月平均 730 時間 = 月間 1.66 GB

1.66 GB * 0.50USD = 0.83USD

合計: 月額55.73 USD

■ AWS CloudWatchの料金

<https://aws.amazon.com/jp/cloudwatch/pricing/>

© 2019, Amazon Web Services, Inc. or its Affiliates. All rights reserved.



まとめ

- Container Insightsはコンテナ化されたアプリケーションのメトリクスとログを収集、集計、要約できるCloudWatchの機能の一つ
- Container Insightsの登場により、サードパーティ製のツールや自前のカスタムメトリクスを使用しなくても、タスク、コンテナレベルでのモニタリングが可能になった
- CloudWatch 自動ダッシュボードを起点にタスク、コンテナレベルでのログ分析やパフォーマンスマニタリングなどの統合的な分析が可能

Q&A

ご質問については、
AWS Japan Blog 「<https://aws.amazon.com/jp/blogs/news/>」
にて資料公開と併せて、後日掲載します

AWS の日本語資料の場所「AWS 資料」で検索



The screenshot shows the AWS Japan Language Resources page. At the top, there's a navigation bar with the AWS logo, search bar, and links for "日本担当チームへお問い合わせ", "サポート", "日本語", "アカウント", and "コンソールにサインイン". Below the navigation is a horizontal menu with links for "製品", "ソリューション", "料金", "ドキュメント", "学習", "パートナー", "AWS Marketplace", "その他", and a search icon. The main content area features a large title "AWS クラウドサービス活用資料集トップ" and a descriptive paragraph about the service. At the bottom, there are four call-to-action buttons: "AWS Webinar お申込", "AWS 初心者向け", "業種・ソリューション別資料", and "サービス別資料".

AWS クラウドサービス活用資料集トップ

アマゾン ウェブ サービス (AWS) は安全なクラウドサービスプラットフォームで、ビジネスのスケールと成長をサポートする処理能力、データベースストレージ、およびその他多種多様な機能を提供します。お客様は必要なサービスを選択し、必要な分だけご利用いただけます。それらを活用するために役立つ日本語資料、動画コンテンツを多数ご提供しております。(本サイトは主に、AWS Webinar で使用した資料およびオンデマンドセミナー情報を掲載しています。)

AWS Webinar お申込 »

AWS 初心者向け »

業種・ソリューション別資料 »

サービス別資料 »

<https://amzn.to/JPArchive>

AWS Well-Architected 個別技術相談会

毎週”W-A個別技術相談会”を実施中

- AWSのソリューションアーキテクト(SA)に
対策などを相談することも可能

• 申込みはイベント告知サイトから

(<https://aws.amazon.com/jp/about-aws/events/>)

AWS イベント で[検索]



ご視聴ありがとうございました

AWS 公式 Webinar
<https://amzn.to/JPWebinar>



過去資料
<https://amzn.to/JPArchive>





Amazon CloudWatch RUM

AWS Black Belt Online Seminar

辻林 侑 (Yu Tsujibayashi)

Solutions Architect
2023/04

AWS Black Belt Online Seminarとは

- ・ 「サービス別」「ソリューション別」「業種別」などのテーマに分け、
アマゾン ウェブ サービス ジャパン合同会社が提供するオンラインセミナー
シリーズです
- ・ AWSの技術担当者が、AWSの各サービスやソリューションについてテーマご
とに動画を公開します
- ・ 動画を一時停止・スキップすることで、興味がある分野・項目だけの聴講も
可能、スキマ時間の学習にもお役立ていただけます
- ・ 以下のURLより、過去のセミナー含めた資料などをダウンロードするこ
とができます
 - ・ <https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-service-cut/>
 - ・ <https://www.youtube.com/playlist?list=PLzWGOASvSx6FIwIC2X1nObr1KcMCBBlqY>

内容についての注意点

- ・ 本資料では2023年4月時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<https://aws.amazon.com/>)にてご確認ください
- ・ 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます
- ・ 価格は税抜表記となっています。日本居住者のお客様には別途消費税をご請求させていただきます

自己紹介

名前：辻林 侑 (Tsujibayashi Yu)

所属：アマゾン ウェブ サービス ジャパン
技術統括本部 西日本ソリューショングループ
ソリューションアーキテクト

経歴：国内電機メーカーでシステムエンジニア



好きなAWSサービス：Amazon CloudWatch、Amazon Managed Grafana

本セミナーの対象者

これから AWS を利用した監視設計を担当する方

AWS すでに監視を実施されてる方

AWS での Observability やリアルユーザーモニタリングに興味のある方

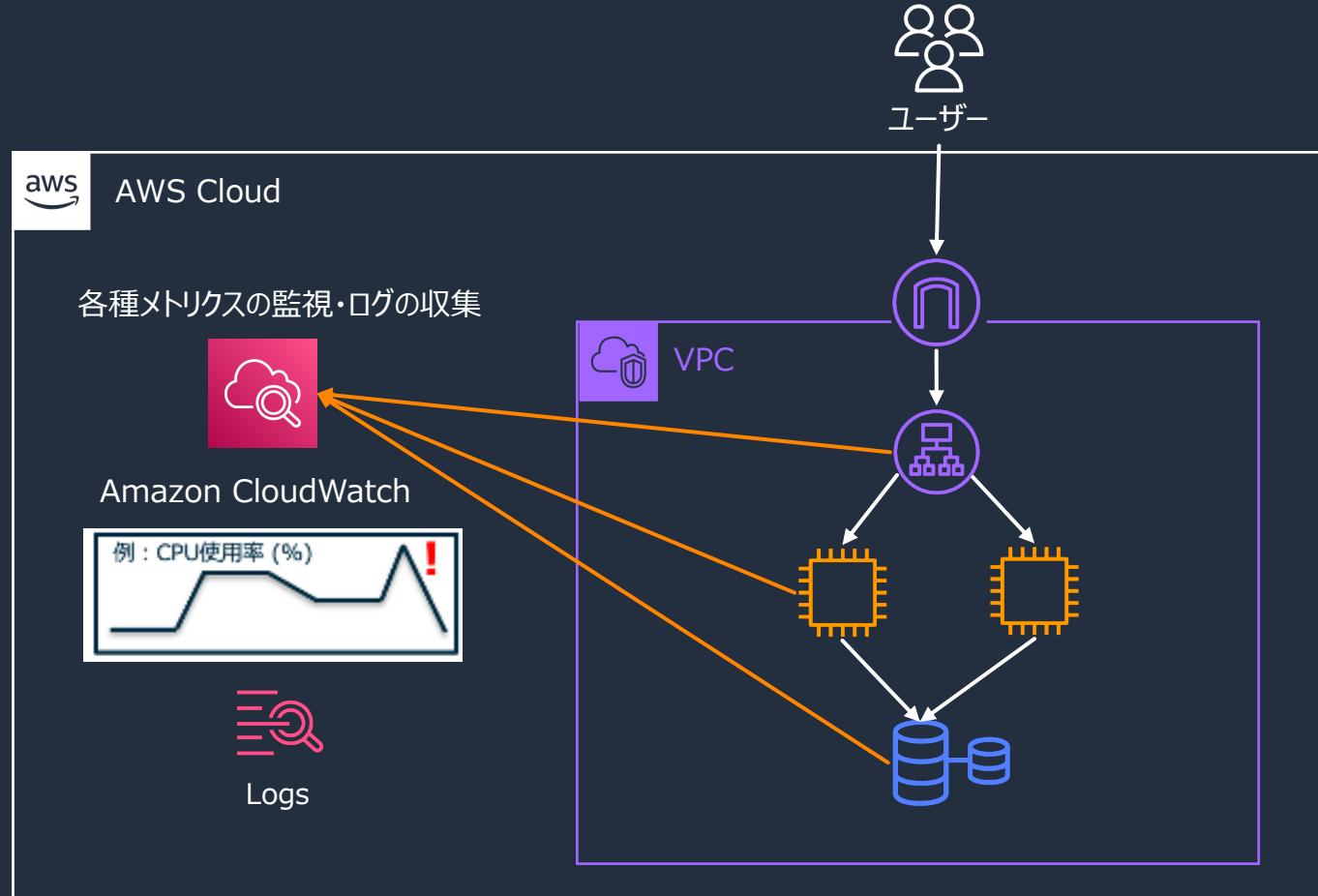
アジェンダ

1. Amazon CloudWatch RUM とは
2. Amazon CloudWatch RUM による Real User Monitoring
3. Amazon CloudWatch RUM のセットアップ
4. Amazon CloudWatch RUM の注意事項、料金
5. まとめ

Amazon CloudWatch RUMとは

はじめに（システムの監視について）

各リソースのメトリクスやログだけでは、ユーザーにどのように影響しているかわからない。
→“**ユーザー視点**”に立った監視も重要



ユーザーエクスペリエンスの監視の重要性



サービスやアプリの健全性をチェックする



アプリケーションを元の状態に回復する



トラブルの原因を調査する



ユーザ行動を分析する



キヤパシティを分析する

ユーザーエンタリーモニタリング

Synthetic Monitoring

※ Active monitoring の一種

- 定定期的に一定間隔で**計測プログラム**がアクセスしてデータを取得することを通じてパフォーマンスを計測する手法(主に製造業で利用される統計的品質管理手法と同じ思想に基づく計測手法)
- 短期のパフォーマンス計測用途に好適
- 一般に「外形監視」と訳される

Real User Monitoring

※ Passive monitoring の一種

- **実ユーザー**アクセスの情報を取得して計測する手法
(実ユーザーによる実ブラウザでの実ロケーションからのアクセスデータを取得・計測サーバーに送信)
- 中長期のユーザー傾向把握用途に好適

ユーザーエンタリーモニタリング

Synthetic Monitoring

※ Active monitoring の一種

- 定定期的に一定間隔で**計測プログラム**がアクセスしてデータを取得することを通じてパフォーマンスを計測する手法(主に製造業で利用される統計的品質管理手法と同じ思想に基づく計測手法)
- 短期のパフォーマンス計測用途に好適
- 一般に「外形監視」と訳される



Amazon CloudWatch Synthetics

Real User Monitoring

※ Passive monitoring の一種

- 実ユーザー**アクセスの情報を取得して計測する手法
(実ユーザーによる実ブラウザでの実ロケーションからのアクセステータデータを取得・計測サーバーに送信)
- 中長期のユーザー傾向把握用途に好適



Amazon CloudWatch RUM

ユーザーエンタリーモニタリング

Synthetic Monitoring

※ Active monitoring の一種

- 定定期的に一定間隔で**計測プログラム**がアクセスしてデータを取得することを通じてパフォーマンスを計測する手法(主に製造業で利用される統計的品質管理手法と同じ思想に基づく計測手法)
- 短期のパフォーマンス計測用途に好適
- 一般に「外形監視」と訳される



Amazon CloudWatch Synthetics

Real User Monitoring

※ Passive monitoring の一種

- 実ユーザー**アクセスの情報を取得して計測する手法
(実ユーザーによる実ブラウザでの実ロケーションからのアクセスデータを取得・計測サーバーに送信)
- 中長期のユーザー傾向把握用途に好適



Amazon CloudWatch RUM

Amazon CloudWatchの全体像

Application Monitoring

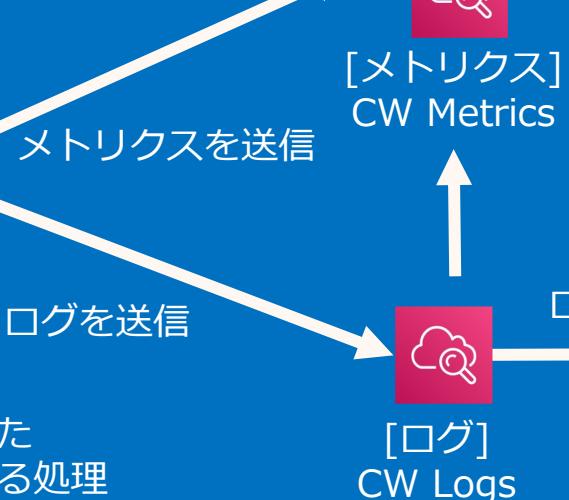
- [外形監視] CW Synthetics
- [リアルユーザー モニタリング]** CW RUM
- [フィーチャーフラグA/Bテスト] CW Evidently

- [インターネット監視] CW Internet Monitor
- [トレース] CW ServiceLens

[ダッシュボードに統合]
CW Dashboard



Infrastructure



[アラーム]
CW Alarms

メトリクスに応じた
アクション



[リアルタイムメトリクス分析]
CW Metrics Insight
[タグベースの視覚化]
CW Metrics Explorer



[メトリクストリーム]
CW Metrics Stream



Amazon Kinesis
Data Firehose

パートナー
サービス、
S3/Redshift



[ログ分析]
CW Logs Insights



[イベント]
Amazon EventBridge/
Amazon EventBridge
Scheduler

Insights



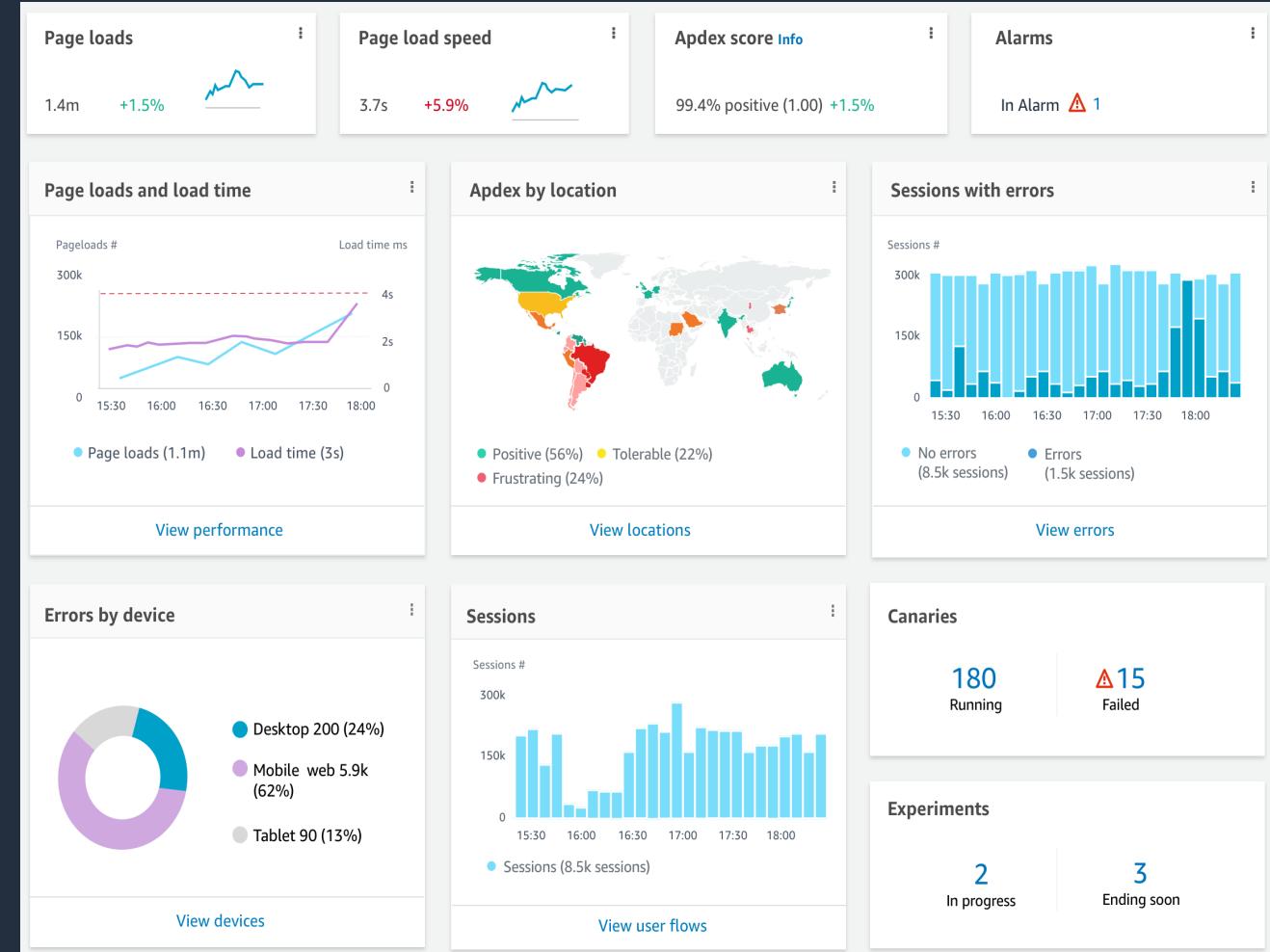
- [構造化ログによるメトリクス]
CW Container Insights / Contributor Insights
- [Lambda拡張機能によるメトリクス]
Lambda Insights
- [アプリケーションコンポーネントのメトリクス]
Application Insights

※CW = CloudWatch

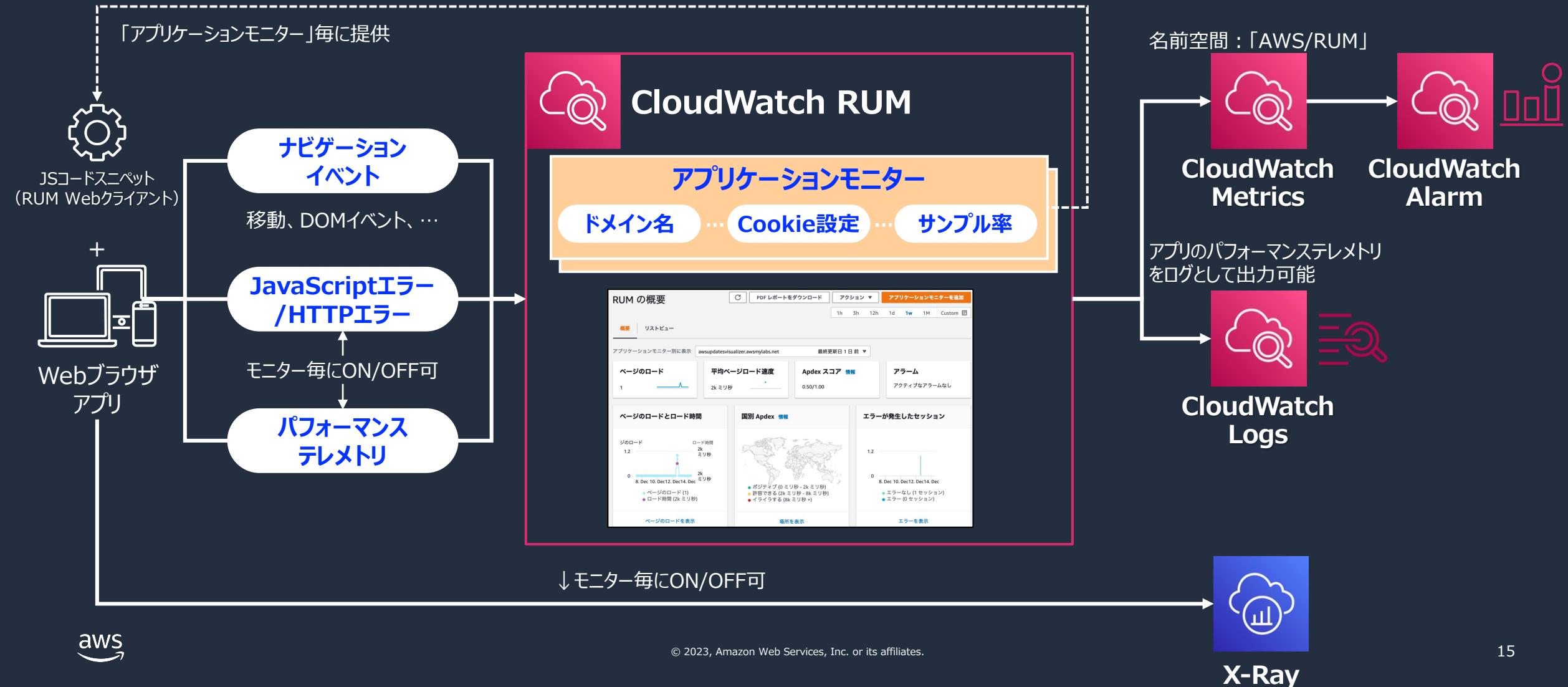
Amazon CloudWatch RUM (Real User Monitoring)

アプリのパフォーマンスに関するクライアントサイドのデータをリアルタイムで取得し、ユーザー体験を最適化

- リアルユーザーのパフォーマンスをモニタし、ブラウザやデバイスの種類、位置、ネットワークの接続性の問題などを把握できる
 - ダッシュボードでページの読み込み順序や JavaScript / HTTP レスポンスのエラーなど、パフォーマンス問題に関する情報を可視化
 - 同じ問題の影響下にあるユーザー セッション数を提示するため、改修の優先順位を付けることが容易
 - Amazon CloudWatch ServiceLens、AWS X-Ray と組み合わせることも可能



Amazon CloudWatch RUM の動作イメージ



Amazon CloudWatch RUMによる Real User Monitoring



Amazon CloudWatch RUM ダッシュボード

Web バイタルデータをはじめ、Webブラウザ統計 や ユーザー挙動の可視化・分析のためのダッシュボードを提供



MyRUMApp

Filter by selecting or typing attributes and values (ex. "b") Select a filter ▾ 1h 3h 12h 1d 1w 1M Custom

Performance Errors Sessions Events Browsers & Devices User journey Configuration

カテゴリ	主な情報	
Performance (パフォーマンス)	ページのロード回数	ページロード時間(「日付・24時間別」グラフあり)
	エラー数	Webバイタル
	リソースリクエスト	経時的なページロードのステップ
	アクセス元の地域/国	
Errors (エラー)	エラー数と時間	エラーが発生したセッション数
Sessions (セッション)	平均セッション長さ	セッションあたりのエラー数
Events (イベント)	イベント数とイベント詳細	
Browser & Devices (ブラウザとデバイス)	アクセスブラウザ種別内訳	ブラウザ別平均ページロード時間
	ブラウザ別のスループット (ページロード数/分)	
User Journey (ユーザージャーニー)	ユーザーパス遷移とドロップオフ数	

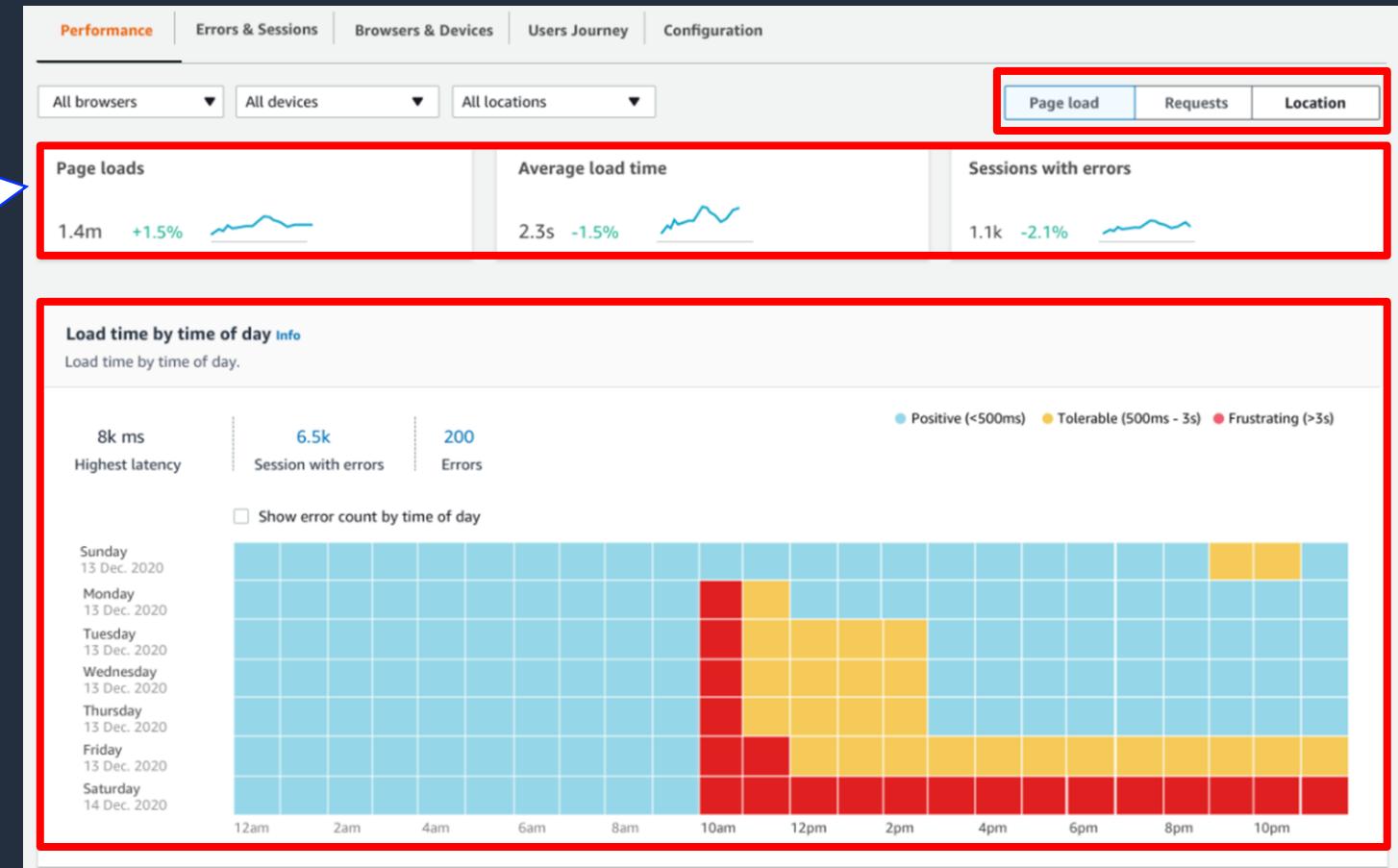
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-apdex.html

© 2023, Amazon Web Services, Inc. or its affiliates.



ダッシュボード - パフォーマンス

アプリケーションモニターが収集したパフォーマンス関連のサマリーを分析・可視化



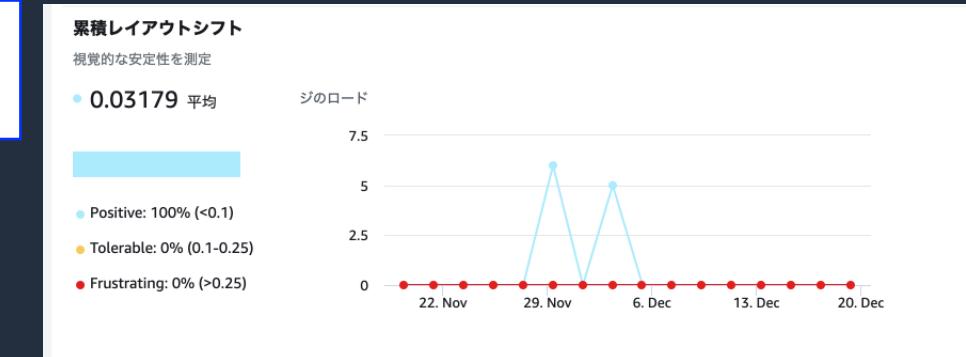
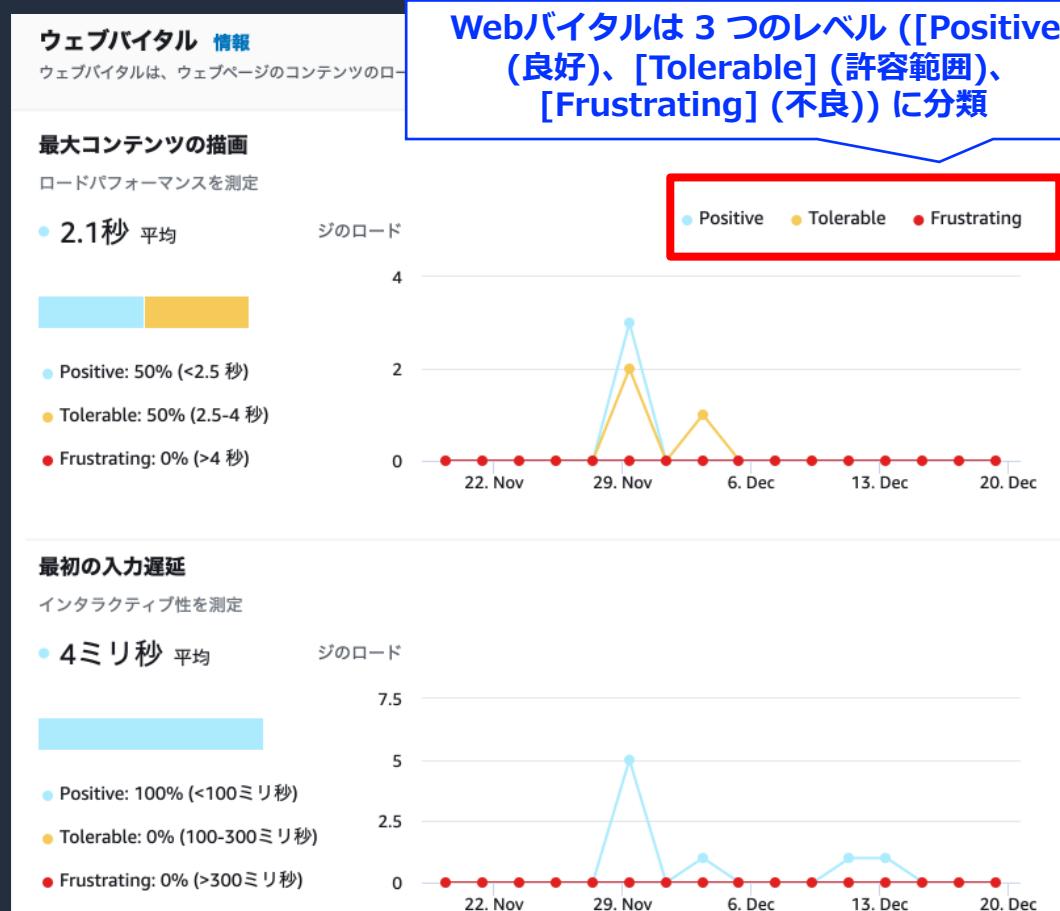
ページの読み込み回数
平均ロード時間、
エラーが発生していた
セッション数のトレンド
グラフ

「ページのロード」
「リクエスト」
「場所」ごとのサマリー

日付・時間別のアプ
リケ
シヨン側ロード時間のレイ
テンシ可視化

ダッシュボード - パフォーマンス (Webバイタル)

最大コンテンツの描画 (Largest Contentful Paint)、最初の入力遅延 (First Input Delay)、累積レイアウトシフト (Cumulative Layout Shift) を表示



ダッシュボード - エラーとセッション

エラー (HTTPエラー/JSエラー) 発生とセッション数・セッションIDとを組み合わせた分析と可視化

エラー

最もよく見られるエラーメッセージ

多くのエラーが発生するデバイス

多くのエラーが発生するブラウザ

エラー

タイプ別のエラインスタンス

エラーの数

500 Internal Server Error

多くのエラーが発生するデバイス

Desktop 6 個のエラー

多くのエラーが発生するブラウザ

Chrome 4 個のエラー

HTTP エラーの数

12:00 12. Mar 12:00 13. Mar 12:00 14. Mar 12:00 15. Mar 12:00 16. Mar 12:00 17. Mar 12:00 18. Mar 12:00

エラー (1)

エラーメッセージ タイプ エラー セッション ユーザー 最初の発生 最後の発生

エラーメッセージ	タイプ	エラー	セッション	ユーザー	最初の発生	最後の発生
500 Internal Server Error	HTTP	6	4	2	今日 2023年3月17日 14:47:35 JST	今日 2023年3月18日 12:41:44 JST

リンクからエラーの詳細を確認できる

Error details: 500 Internal Server Error

Error details

Error message	Type	Error count	Sessions	Users	First occurred	Last occurred
Internal Server Error	HTTP	6	4	2	今日 2023年3月17日 14:47:35 JST	今日 2023年3月18日 12:41:44 JST

Latest raw event

```
{ "event_type": "error", "id": "1234567890", "event_type": "error", "event_id": "8e4b8d4-1c9f-4d3d-984d-04c2bae70556", "event_version": "1.0.0", "log_stream": "2023-3-18T12:00:00Z", "application_id": "000100-0739-4553-909c-b085dd408841", "application_version": "1.0.0", "metadata": { "version": "1.0.0", "browserlanguage": "en-US", "browserName": "Chrome" }, }
```

セッション

平均セッション長

6分 2秒

エラーがないセッション

20.0%

セッションあたりの平均エラー

1.2

セッション

セッション

5

2.5

0

12:00 12. Mar 12:00 13. Mar 12:00 14. Mar 12:00 15. Mar 12:00 16. Mar 12:00 17. Mar 12:00 18. Mar 12:00

セッション (5)

Filter by selecting or typing keys and values (ex. "sessionId=1234")

セッション ID	期間	開始日:	ページ	問題	ブラウザ	デバイス	言語	トレース
215f56d4-6caa-43e5-9af7-efc28f2b53d0	05:24	今日 2023年3月17日 15:20:19 JST	19	2	Firefox v102.0	desktop Mac OS	ja	2
3ac9395d-07fe-4ea8-bb6a-e37cd91c45e3	01:21	今日 2023年3月17日 14:46:37 JST	14	2	Chrome v111.0.0.0	desktop Mac OS	en-US	-
492b48ec-8f2a-413c-8f29-f58ca42aa2d6	22:59	今日 2023年3月18日 12:41:35 JST	6	1	Chrome v111.0.0.0	desktop Mac OS	en-US	1

アプリケーションモニターで X-Ray を有効化にした場合に表示
X-Ray からトレースを確認できる

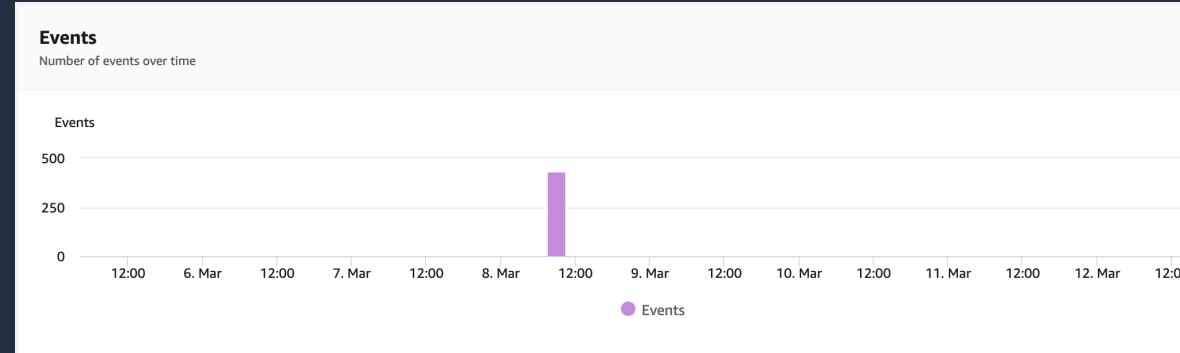
aws

© 2023, Amazon Web Services, Inc. or its affiliates.

20

ダッシュボード - イベント

特定期間のイベント数とイベント詳細の分析と可視化



Events (100+) 情報

Filter by selecting or typing keys and values (ex. "sessionId=1234")

Event	Date	Page ID	Session
▶ com.amazon.rum.page_view_event	3日前 2023年3月8日 14:57:18 JST	/	f33b19af-2363-4a23-aede-9c29c9fcbb35
▶ com.amazon.rum.first_input_delay_event	3日前 2023年3月8日 14:57:18 JST	/PetListAdoptions	f33b19af-2363-4a23-aede-9c29c9fcbb35
▶ my_custom_event	3日前 2023年3月8日 14:57:18 JST	/	f33b19af-2363-4a23-aede-9c29c9fcbb35
▶ com.amazon.rum.performance_navigation_event	3日前 2023年3月8日 14:57:17 JST	/PetListAdoptions	f33b19af-2363-4a23-aede-9c29c9fcbb35
▶ com.amazon.rum.performance_resource_event	3日前 2023年3月8日 14:57:17 JST	/PetListAdoptions	f33b19af-2363-4a23-aede-9c29c9fcbb35
▶ com.amazon.rum.performance_resource_event	3日前 2023年3月8日 14:57:17 JST	/PetListAdoptions	f33b19af-2363-4a23-aede-9c29c9fcbb35
▶ com.amazon.rum.performance_resource_event	3日前 2023年3月8日 14:57:17 JST	/PetListAdoptions	f33b19af-2363-4a23-aede-9c29c9fcbb35
▶ com.amazon.rum.performance_resource_event	3日前 2023年3月8日 14:57:17 JST	/PetListAdoptions	f33b19af-2363-4a23-aede-9c29c9fcbb35

イベントをドリルダウンして、イベントの「Metadata」や「Row event」を確認できる

Event

Date

Page ID

Session

com.amazon.rum.page_view_event

3日前
2023年3月8日 14:57:18 JST

/

f33b19af-2363-4a23-aede-9c29c9fcbb35

Metadata

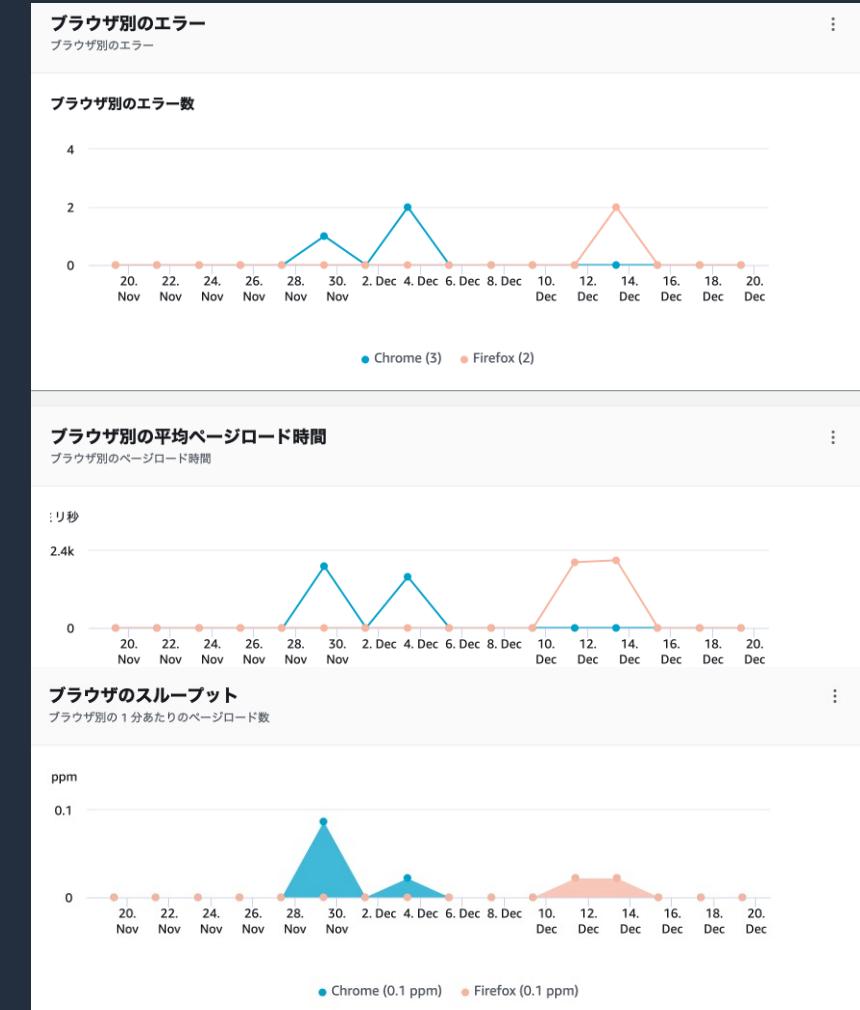
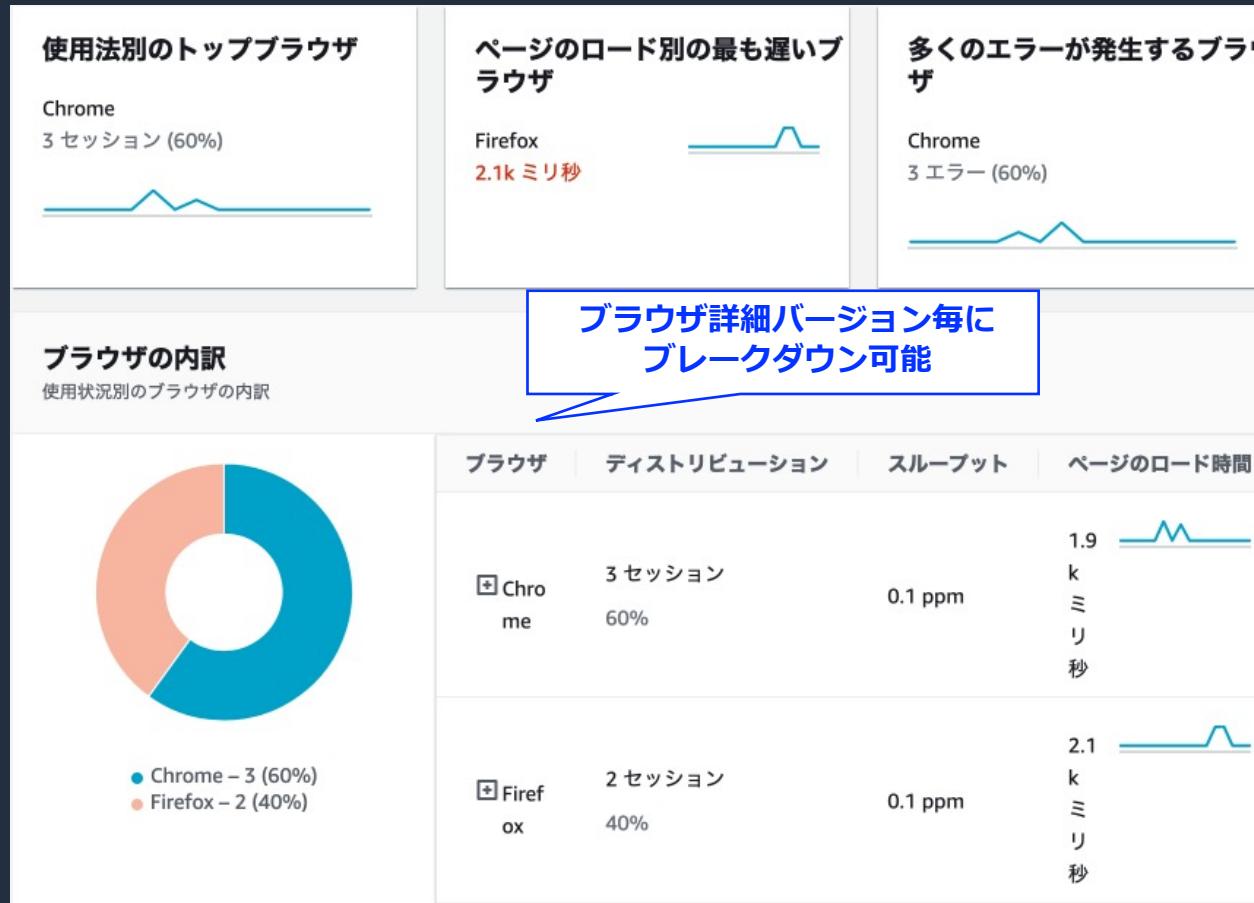
```
version: 1.0.0 | browserLanguage: en-US | browserName: Chrome | browserVersion: 110.0.0 | osName: Mac OS | osVersion: 10.15.7 | deviceType: desktop | platformType: web | pageId: / | parentPageId: /PetListAdoptions | interaction: 11 | title: Home Page - Observability PetAdoptions | domain: servi-petsi-m5zbl39v08ga-965651489.us-east-1.elb.amazonaws.com | countryCode: US | subdivisionCode: VA
```

Raw event

```
{"event_timestamp": 1676255038000, "event_type": "com.amazon.rum.page_view_event", "event_id": "7ce4d40f-67cc-46cf-b53f-dc5925de49c4", "event_version": "1.0.0", "log_stream": "2023-3-8T14", "application_id": "3fbfa02b7-a66e-47f7-a031-13500691d668", "application_version": "1.0.0", "metadata": { "version": "1.0.0", "browserLanguage": "en-US", "browserName": "Chrome", "browserVersion": "110.0.0", "osName": "Mac OS", "osVersion": "10.15.7", "deviceType": "desktop", "platformType": "web", "pageId": "/", "parentPageId": "/PetListAdoptions", "interaction": "11", "title": "Home Page - Observability PetAdoptions", "domain": "servi-petsi-m5zbl39v08ga-965651489.us-east-1.elb.amazonaws.com", "countryCode": "US", "subdivisionCode": "VA" }}
```

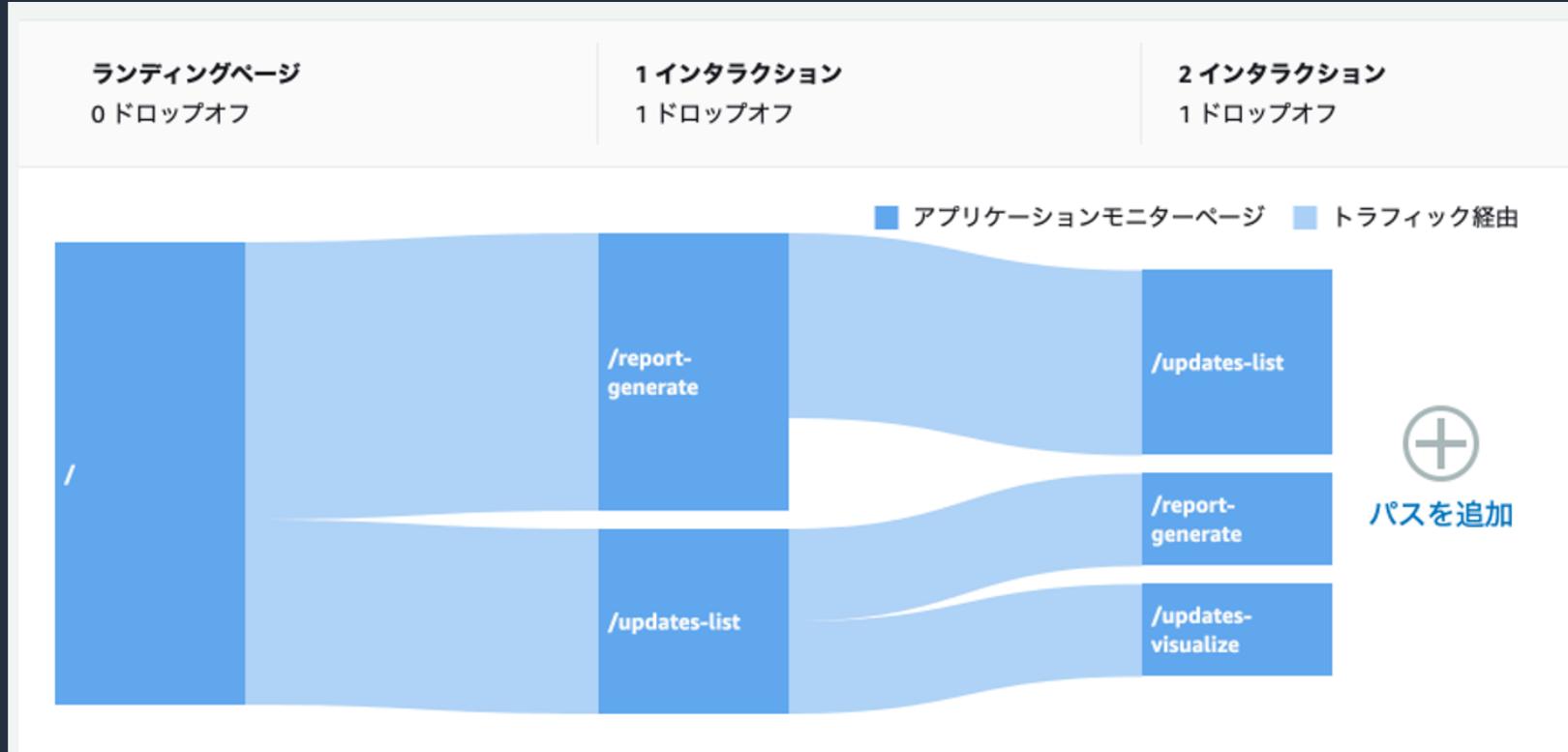
ダッシュボード - ブラウザとデバイス

アプリケーションにアクセスしたブラウザ種類・バージョンごとの情報(エラー数、平均ページロード時間など)を分析と可視化



ダッシュボード - ユーザージャーニー

アプリケーション(Webサイト)内のパス遷移の導線を可視化 (左から右へ)



※アプリケーションモニター作成時にCookie利用を有効に設定する必要がある

CloudWatch RUM で収集できる CloudWatch メトリクス

- “AWS/RUM” 名前空間に自動的に下記メトリクスを発行
- 収集したメトリクスから CloudWatch アラームを設定

#	メトリクス	単位	説明
1	HttpStatusCodeCount	カウント	レスポンスステータスコードによるアプリケーション内の HTTP レスポンスの数
2	JSErrorCount	カウント	取り込まれた JavaScript エラーイベントの数
3	NavigationFrustratedCount	カウント	応答時間が 8000 ミリ秒より大きいナビゲーションイベントの数 (Apdex のしきい値の 4 倍を超過)
4	NavigationSatisfiedCount	カウント	応答時間が 2000ms 以下のナビゲーションイベントの数 (Apdex の目標値)
5	NavigationToleratedCount	カウント	応答時間が 2000 ミリ秒から 8000 ミリ秒の間のナビゲーションイベントの数
6	PerformanceResourceDuration	ミリ秒	リソースイベントの時間
7	PerformanceNavigationDuration	ミリ秒	ナビゲーションイベントの時間
8	RumEventPayloadSize	バイト	CloudWatch RUM によって取り込まれるすべてのイベントのサイズ
9	SessionCount	カウント	アプリケーションモニターによって取り込まれたセッション開始イベントの数
10	WebVitalsCumulativeLayoutShift	なし	累積的レイアウトシフトイベントの値を追跡
11	WebVitalsFirstInputDelay	ミリ秒	最初の入力遅延イベントの値を追跡
12	WebVitalsLargestContentfulPaint	ミリ秒	最大のコンテンツフルペイントイベントの値を追跡

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-metrics.html

カスタムメトリクスと拡張メトリクス

- ・ カスタムメトリクス
 - ユーザー定義のイベント(カスタムイベント)、事前定義のRUM イベント、およびユーザ定義のメタデータ属性(カスタム属性)のデータに基づいてメトリクスを定義
- ・ 拡張メトリクス
 - デフォルトの CloudWatch RUM メトリクスにディメンションを追加し、CloudWatch に送信することで、より詳細なメトリクスを確認できる

The screenshot shows the AWS CloudWatch Metrics configuration interface. The top navigation bar includes tabs for Performance, Errors, Sessions, Events, Browsers and Devices, User Journeys, and Settings. The 'Settings' tab is selected. On the left, a sidebar lists General Settings, Alarms, Tags, and a JavaScript Snippet, with 'RUM Extended Metrics' highlighted by a red box. The main content area is titled 'Extended Metrics (22)' and contains a table of metrics with dimensions. A large red box highlights the table's header and the first few rows. The table columns are: Type, Name, BrowserName, DeviceType, OSName, CountryCode, and PageId. The data in the table includes various browser and device types across different countries.

Type	Name	BrowserName	DeviceType	OSName	CountryCode	PageId
Errors	HttpErrorCount	Edge	desktop	Windows		/housekeeping
Errors	JsErrorCount	Edge	desktop	Windows		/housekeeping
Apdex performance	NavigationFrustratedTransaction	Edge	mobile	Android	United States of America	すべて
Apdex performance	NavigationFrustratedTransaction	Edge	mobile	iOS	United States of America	すべて
Apdex performance	NavigationFrustratedTransaction	Firefox	mobile	Android	United States of America	すべて
Apdex performance	NavigationFrustratedTransaction	Firefox	mobile	iOS	United States of America	すべて
Apdex performance	NavigationSatisfiedTransaction	Chrome	desktop	Mac OS		すべて
Performance	PerformanceNavigationDuration	Chrome	desktop	Mac OS		すべて
Performance	PerformanceResourceDuration	Chrome	desktop	Mac OS		すべて
Sessions	SessionCount	Chrome	desktop	Mac OS		すべて

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-custom-and-extended-metrics.html>



Amazon CloudWatch RUM のセットアップ



Amazon CloudWatch RUM のセットアップ



Webブラウザ
アプリ

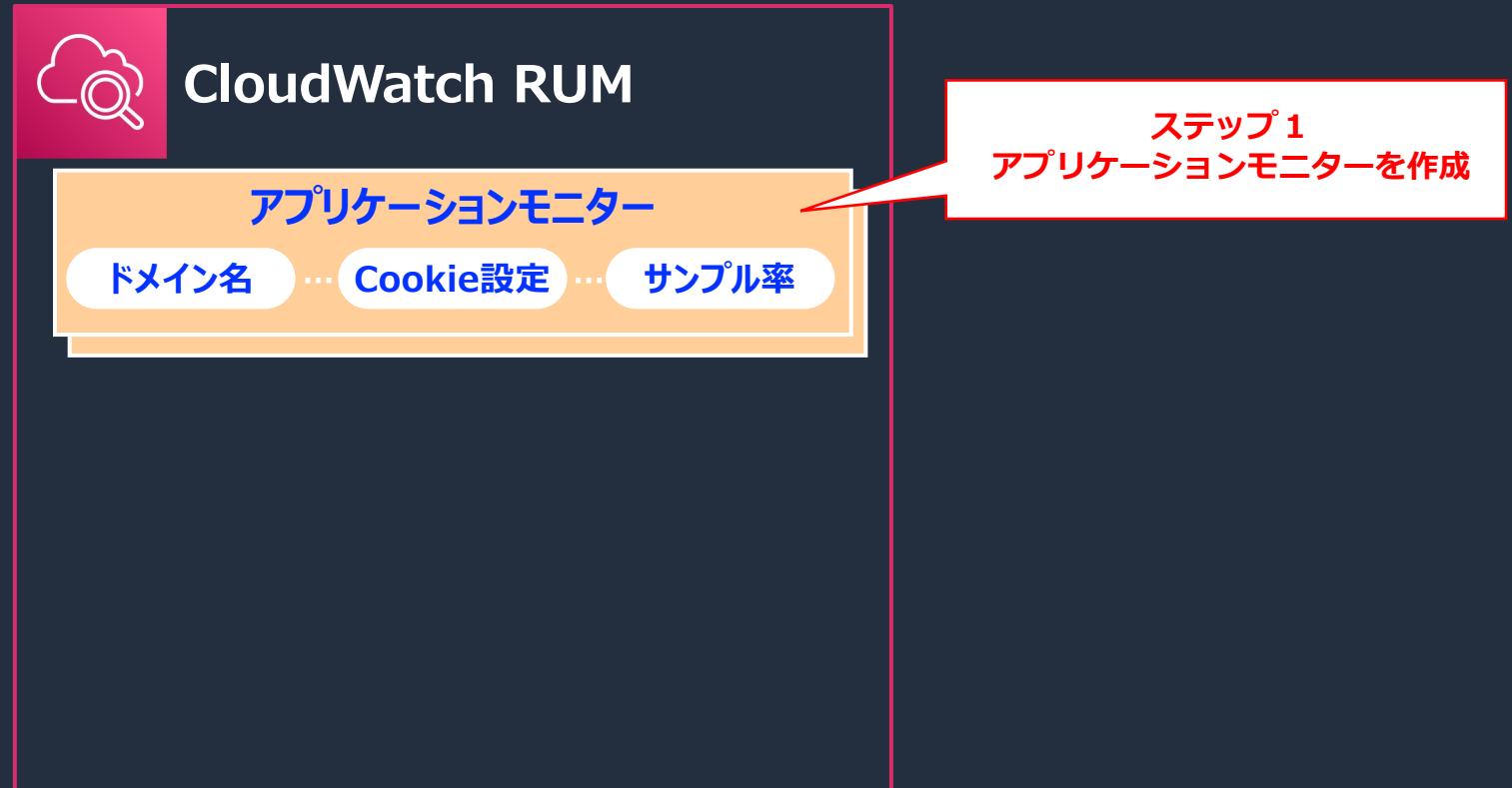


CloudWatch RUM

Amazon CloudWatch RUM のセットアップ



Webブラウザ
アプリ



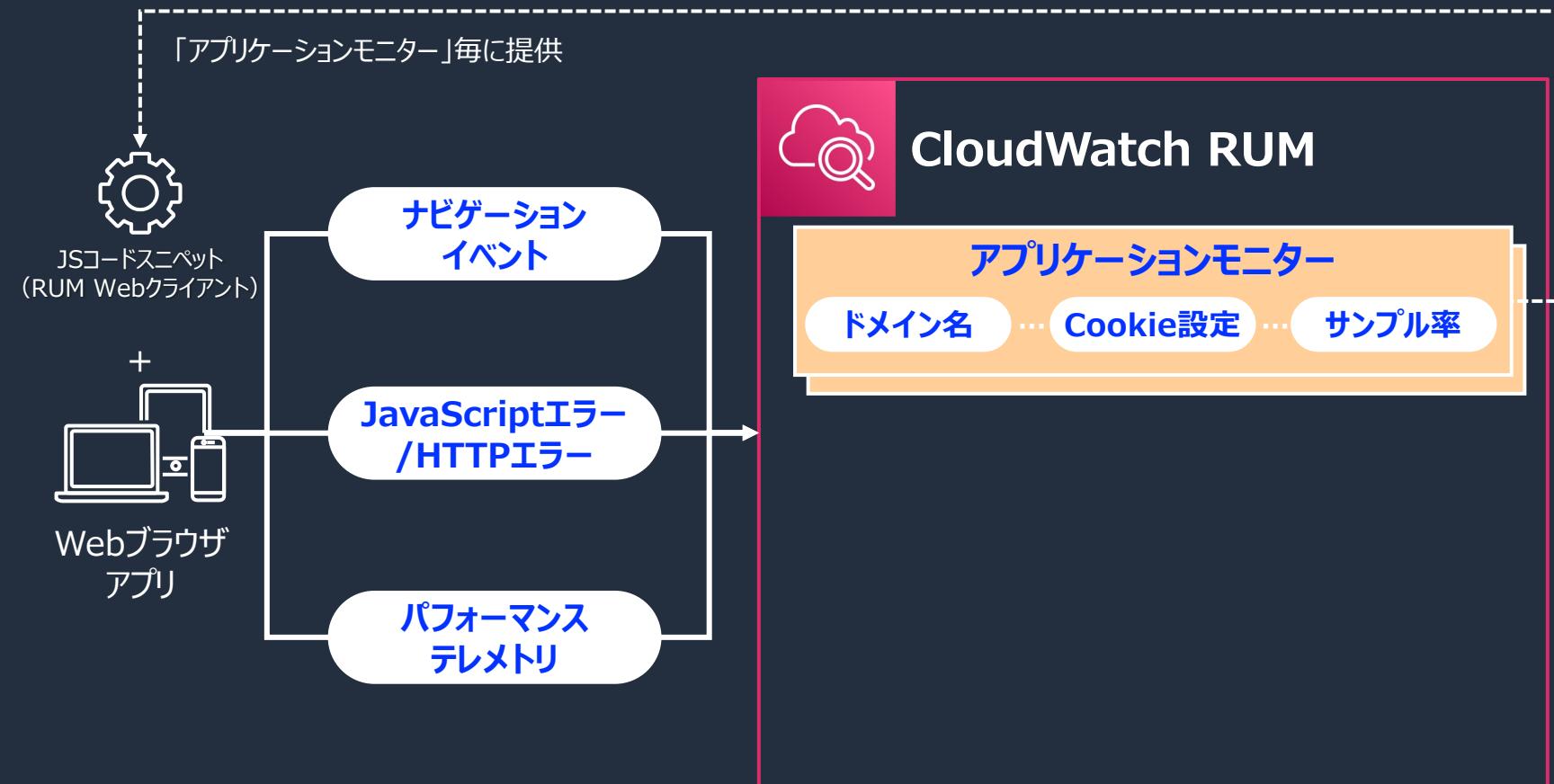
Amazon CloudWatch RUM のセットアップ



Amazon CloudWatch RUM のセットアップ



Amazon CloudWatch RUM のセットアップ



Amazon CloudWatch RUM のセットアップ



セットアップのステップ

- 1 : アプリケーションモニターの作成
- 2 : コードスニペットをアプリケーションに挿入

セットアップのステップ

1 : アプリケーションモニターの作成

2 : コードスニペットをアプリケーションに挿入

1：アプリケーションモニターの作成（1/5）

- 収集対象データの設定

- **パフォーマンスメトリ**

- ページとリソースのロード時間

- **JavaScriptエラー**

- アプリケーションによって発生した未処理の JavaScript エラー

- **HTTPエラー**

- アプリケーションによってスローされた HTTP エラー

- **カスタムイベント**（※CloudWatch RUM Web クライアント ver 1.12.0 以降）

- ユーザ定義のイベント https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-custom-events.html

上記いづれも選択しない場合でも、アプリケーションを使用しているユーザの数をブラウザの種類、デバイスの種類、場所など詳細情報毎に確認できます。

RUM データ収集を設定する

RUM トラッカーにインストールするデータプラグインを選択します。

プラグイン

各プラグインは 1 つ以上のイベントを記録します。イベントには、ユーザーとモニタリング対象のアプリケーション間の（パッシブまたはアクティブ）インタラクションに関する一意の属性のセットが含まれます。

パフォーマンスメトリ

RUM エージェントは、ウェブアプリケーションとそのリソースのロード方法とレンダリング方法に関するタイミングデータを記録します。これにはコアウェブバイタルが含まれます。RUM はこのメトリクスを使用して、アプリケーションのユーザーエクスペリエンスについてのインサイトを提供します。

JavaScript エラー

RUM ウェブクライアントは、ウェブアプリケーションによって挙げられた未処理の例外を記録します。

HTTP エラー

RUM ウェブクライアントは、ウェブアプリケーションによってスローされた HTTP エラーを記録します。

その他のデータ

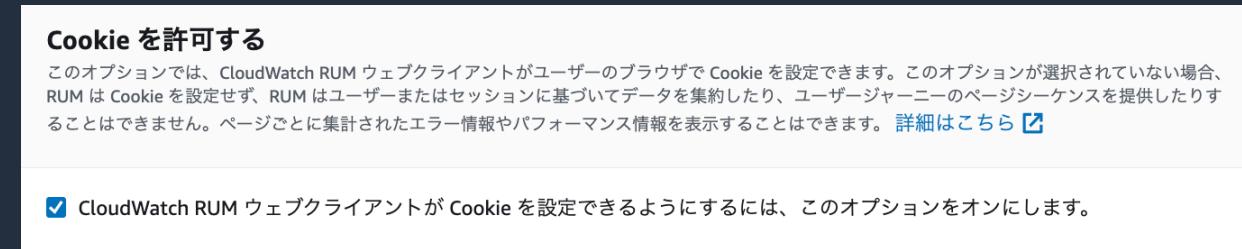
カスタムイベント:

RUM を使用してカスタムイベントを送受信するには、このオプションをチェックします。このオプションが選択されていない場合、RUM は事前定義されたイベントのみを受け入れ、アプリケーションによって他のタイプのイベントを拒否します。このオプションは後で有効になります。

1：アプリケーションモニターの作成（2/5）

- Cookie利用有無の設定

Cookieを許可することにより、
以下のデータを収集、表示できる



- ユーザー ID に基づいて集計されたデータ
 - 一意のユーザー数、個々のエラーが発生しているユーザーの数など
- セッション ID に基づいて集計されたデータ
 - セッションの総数やエラーが発生したセッションの数など
- ユーザージャーニー
 - サンプリングされた各ユーザーセッションに含まれるページ移動のシーケンス

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-privacy.html#CloudWatch-RUM-cookies

1：アプリケーションモニターの作成（3/5）

- サンプリングするセッションの割合の設定

データの収集に使用されるユーザーセッションの割合を入力

- ・デフォルトでは 100 %に設定
- ・値を減らすと、取得されるデータは少なくなるがコスト削減

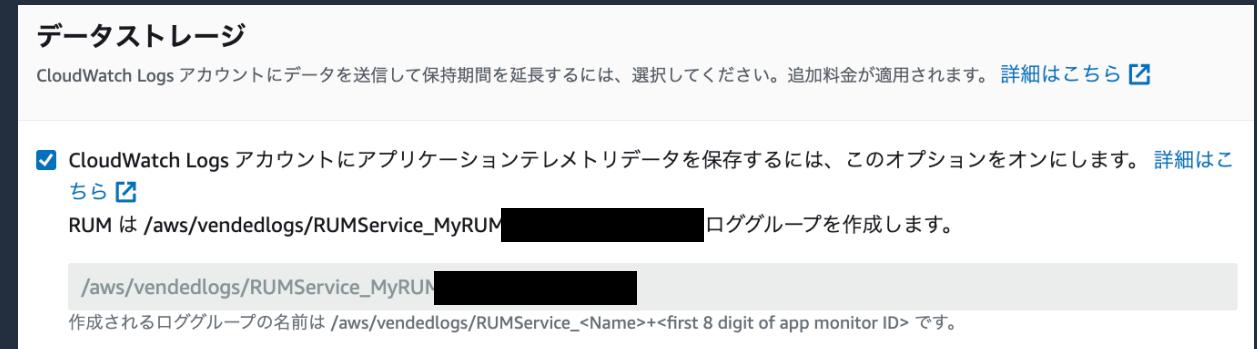


1 : アプリケーションモニターの作成 (4/5)

- データストレージ

CloudWatch RUM 用に収集したエンドユーザーデータは 30 日間保持され、その後削除される

- RUM イベントのコピーを CloudWatch Logs に保存し、そのコピーの保持期間を設定する場合は有効にする
- ログの保持期間は、CloudWatch Logs コンソールで管理



1：アプリケーションモニターの作成（5/5）

- 認証の設定

以下の3つの方法から選択

- CloudWatch RUM用に新規の Amazon Cognito IDプールを作成
 - アプリケーションモニター作成時に一緒に作成される（最小限の労力でセットアップ可能）
- 既存の Amazon Cognito ID プールを利用
- サードパーティの ID プロバイダー



CloudWatch RUM – 認証(Cognito ID プール)

RUM Webクライアントは Cognito ID プールから未認証ユーザー用ロールの一時クレデンシャルを取得してデータを送信

The diagram illustrates the process of obtaining temporary AWS credentials from an unauthenticated role in a Cognito ID Pool. It shows three main components: an **Amazon Cognito ID Pool**, an **AWS Security Token Service (STS)**, and **CloudWatch RUM**.

Amazon Cognito ID Pool (「RUM-Monitor-<region>-<accountId>-<UniqueId>」) contains a red icon representing an unauthenticated role.

ID プールの編集 (Edit ID Pool) screen shows the ID Pool configuration. A red box highlights the "認証されていないロール" (Unauthenticated Role) section, which lists "RUM-Monitor-us-west-2" and "61-Unauth".

AWS Security Token Service (STS) is shown with a red icon representing AssumeRoleWithWebIdentity.

CloudWatch RUM is shown with a pink icon representing PutRumEvents.

The flow of data is as follows:

- ① Cognito ID プールにリクエストし、sts:AssumeRoleWithWebIdentity アクションの呼び出し許可を得る
- ② AssumeRoleWithWebIdentity APIを呼び出し STSから一時的なクレデンシャルを取得 (対応するアプリケーションモニターに対してのみの rum:PutRumEvents アクションが許可)
- ③ PutRumEvents APIでRUMにデータを送信

① Cognito ID プールにリクエストし、sts:AssumeRoleWithWebIdentity アクションの呼び出し許可を得る

アプリケーション
モニター

1対1で対応



RUM Webクライアント
(cwr.js)

② AssumeRoleWithWebIdentity APIを呼び出し
STSから一時的なクレデンシャルを取得
(対応するアプリケーションモニターに対してのみの
rum:PutRumEvents アクションが許可)



AWS Security Token Service (STS)

③ PutRumEvents APIでRUMにデータを送信



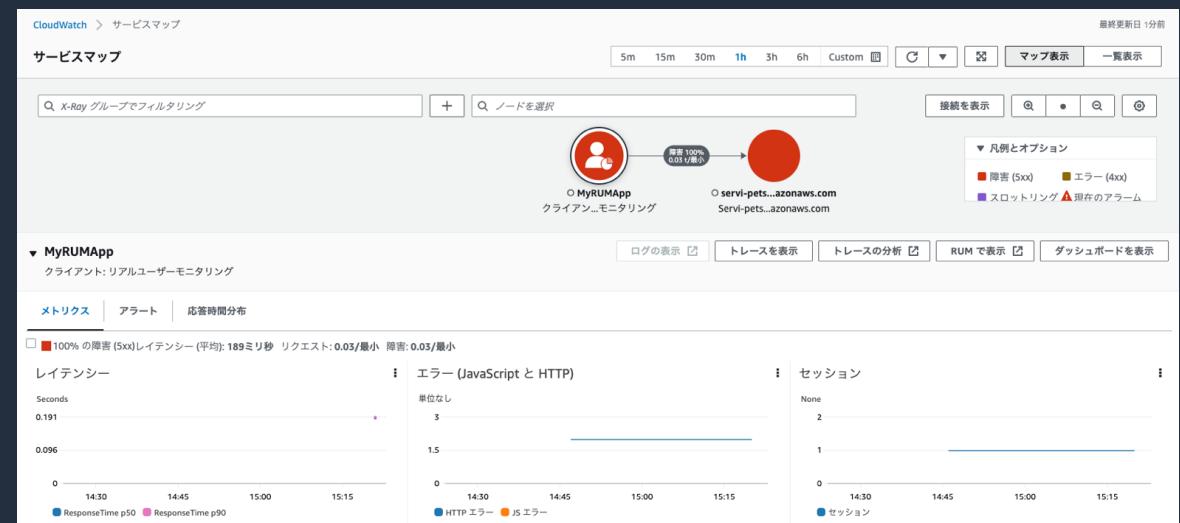
CloudWatch RUM

※補足：認証ユーザーのみがRUMメトリクスを送信する構成を採用する場合は、アプリケーションが明示的にWebクライアントバンドルにsetAwsCredentials関数で一時クレデンシャルをセットする

1：アプリケーションモニターの作成（オプション）

- アクティブトレース

- ユーザーセッションの X-Ray トレースを有効にすると、CloudWatch RUM が HTTP リクエストに X-Ray トレースヘッダーを追加し、HTTP リクエストの X-Ray セグメントを記録
- X-ray および CloudWatch ServiceLens コンソールで、これらのユーザー session のトレースとセグメントを確認できる



セットアップのステップ

1 : アプリケーションモニターの作成

2 : コードスニペットをアプリケーションに挿入

2 : コードスニペットをアプリケーションに挿入 (1/2) - コードスニペットのコピー、ダウンロード

アプリケーションへのRUM Web クライアントのインストール方法として
下記 2つから選択

- JavaScript モジュールとしてインストール
 - TypeScript、JavaScriptのサンプルコードを用意
- HTML への組み込みスクリプトとしてインストール



サンプルコード

サイトから CloudWatch RUM サービスにデータを送信するには、アプリケーションに CloudWatch RUM ウェブクライアントをインストールする必要があります。

TypeScript

TypeScript Watch RUM ウェブクライアントパッケージをインストールします。

JavaScript

JavaScript CloudWatch RUM web

HTML

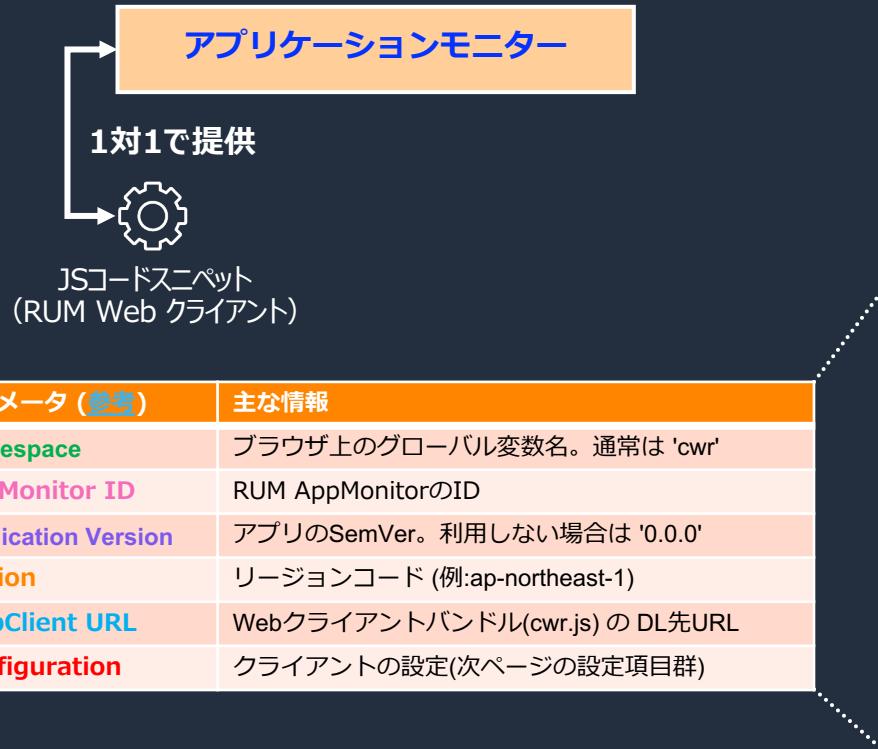
次に、アプリケーションの起動スクリプト内に次のコードを貼り付けます。

```
1 import { AwsRum, AwsRumConfig } from 'aws-rum-web';
2
3 try {
4   const config: AwsRumConfig = {
5     sessionSampleRate: 1,
6     guestRoleArn: "arn:aws:iam:",
7     identityPoolId: "us-east-1:",
8     endpoint: "https://dataplane.rum.us-east-1.amazonaws.com",
9     telecommunications: ["performance", "errors", "http"],
10    allowCookies: true,
11    enableXRay: false
12  };
13
14 const APPLICATION_ID: string = '23b23b25-ba2b-4515-b498-a3fd061831b1';
15 const APPLICATION_VERSION: string = '1.0.0';
16 const APPLICATION_REGION: string = 'us-east-1';
17
18 const awsRum: AwsRum = new AwsRum(
19   APPLICATION_ID,
20   APPLICATION_VERSION,
21   APPLICATION_REGION,
22   config
23 );
24 } catch (error) {
25   // Ignore errors thrown during CloudWatch RUM web client initialization
26 }
```

2 : コードスニペットをアプリケーションに挿入 (2/2)

- アプリケーションへの組み込み

「アプリケーションモニター」の定義毎にJavaScriptコードスニペットが生成
→ アプリケーションに組み込み (コピー&ペースト)



#	パラメータ (参考)	主な情報
1	Namespace	ブラウザ上のグローバル変数名。通常は 'cwr'
2	AppMonitor ID	RUM AppMonitorのID
3	Application Version	アプリのSemVer。利用しない場合は '0.0.0'
4	Region	リージョンコード (例:ap-northeast-1)
5	WebClient URL	Webクライアントバンドル(cwr.js) の DL先URL
6	Configuration	クライアントの設定(次ページの設定項目群)

例) HTMLへの組み込み コードスニペット
(<head>要素に下記を挿入)

```
<script>
(function (n, i, v, r, s, c, x, z) {
  x = window.AwsRumClient = {q: [], n: n, i: i, v: v, r: r, c: c};
  window[n] = function (c, p) {
    x.q.push({c: c, p: p});
  };
  z = document.createElement('script');
  z.async = true;
  z.src = s;
  document.head.insertBefore(z, document.getElementsByTagName('script')[0]);
})('cwr'
  '194a1c89-87d8-41a3-9d1b-5c5cd3dafbd0',
  '1.0.0',
  'us-east-2',
  'https://client.rum.us-east-1.amazonaws.com/1.0.2/cwr.js',
  {
    sessionSampleRate: 1,
    guestRoleArn: "arn:aws:iam::123456789012:role/RUM-Monitor-us-east-2-123456789012-5934510917361-Unauth",
    identityPoolId: "us-east-2:c90ef0ac-e3b8-4d1a-b313-7e73cf21443",
    endpoint: "https://dataplane.rum.us-east-2.amazonaws.com",
    telemetries: ["performance", "errors", "http"],
    allowCookies: true,
    enableXRay: false
  });
</script>
```

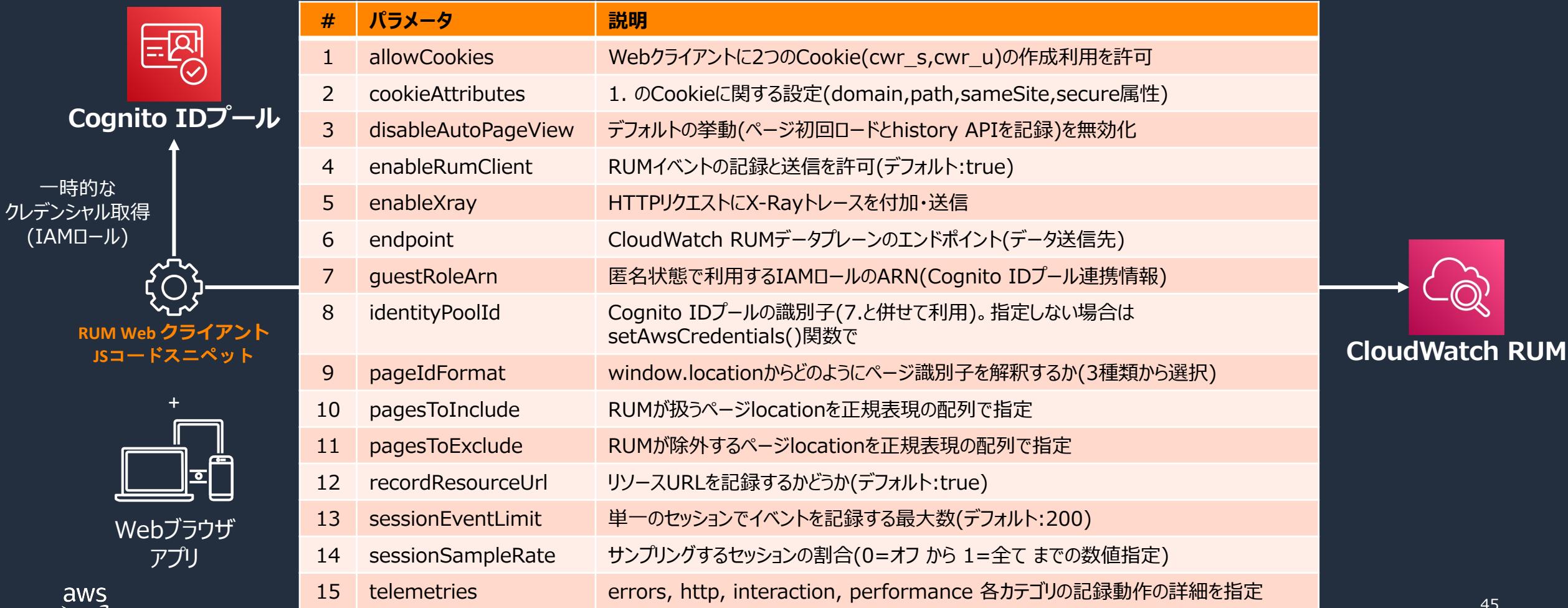
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-get-started-insert-code-snippet.html

CloudWatch RUM Web クライアント

RUM データプレーンに様々なデータを送信するクライアント

Configuration パラメータでその振る舞いを調整可能（下記にパラメータの一部を抜粋）

<https://github.com/aws-observability/aws-rum-web/blob/main/docs/configuration.md>

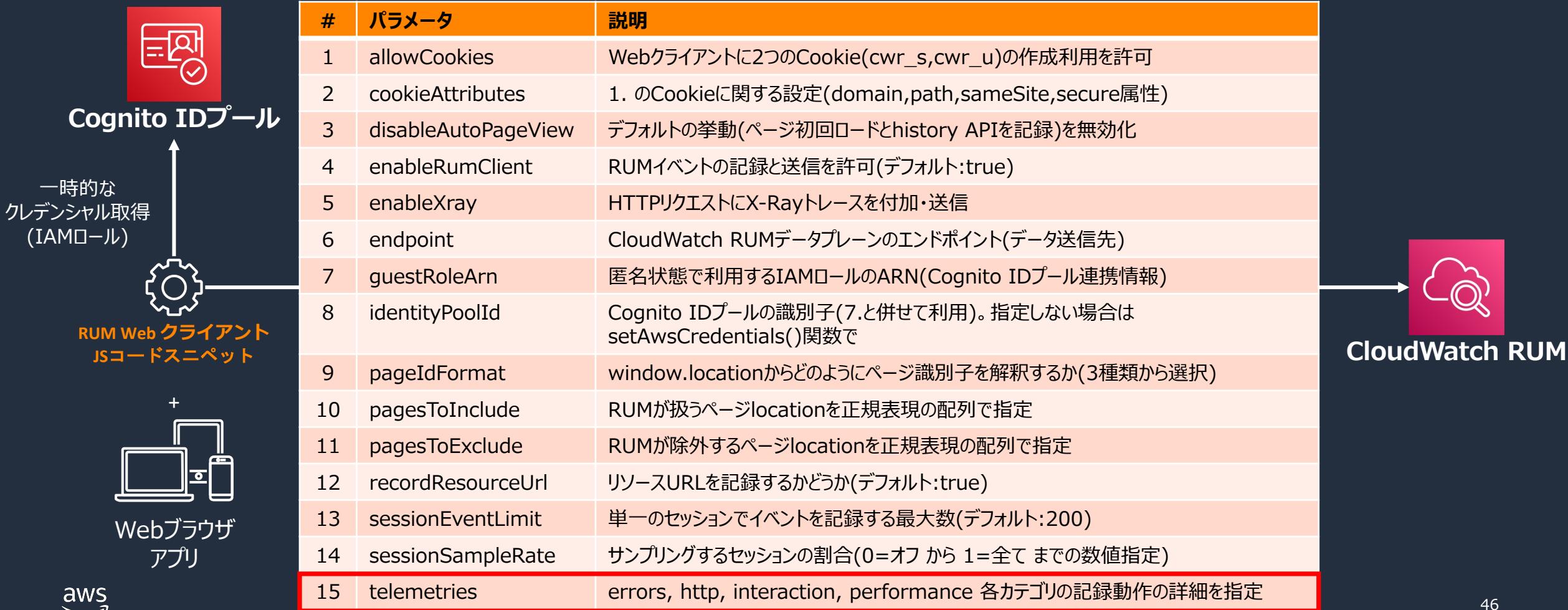


CloudWatch RUM Web クライアント

RUM データプレーンに様々なデータを送信するクライアント

Configuration パラメータでその振る舞いを調整可能（下記にパラメータの一部を抜粋）

<https://github.com/aws-observability/aws-rum-web/blob/main/docs/configuration.md>



CloudWatch RUM Web クライアント - telemetry 設定

4つのカテゴリ(errors, http, interaction, performance) におけるテレメトリデータを送信対象として設定可能

#	telemetry設定カテゴリ	詳細設定パラメータ	説明
1	errors		JavaScriptエラーを記録(送信)することを宣言
2		stackTraceLength	送信するスタックトレースのデータ長(デフォルト:200)
3	http		HTTPエラーを記録(送信)することを宣言
4		urlsToInclude	記録対象として含めるXMLHttpRequest or fetch先URL (デフォルト: [/.*/] ※全て)
5		urlsToExclude	記録対象として除外するXMLHttpRequest or fetch先URL (デフォルト: [] ※なし)
6		stackTraceLength	送信するスタックトレースのデータ長(デフォルト:200)
7		recordAllRequests	成功含む全てのHTTPリクエストを記録かどうか (デフォルト:false ※失敗のみ)
8		addXRayTraceIdHeader	X-Amzn-Trace-Id ヘッダを設定して送信するかどうか(デフォルト:false)
9	interaction		DOMイベントを記録(送信)することを宣言
10		enableMutationObserver	window.load イベントが発生した後に DOM に追加された要素を含む、すべての DOM 要素のイベントを記録するかどうか (デフォルト:false)
11		events	(デフォルト:[] ※対象なし) 例 : [{ event: 'click', elementId: 'mybutton' }] ※特定ボタンのClickイベント 例 : [{ event: 'click', element: document }] ※全Clickイベント
12	performance		ページのパフォーマンスマトリクスを記録(送信)することを宣言
13		eventLimit	ページロード時に読み取るリソース(HTML,CSS,画像, …)の最大数(デフォルト:10)

セットアップのステップ（再掲）

- 1 : アプリケーションモニターの作成
- 2 : コードスニペットをアプリケーションに挿入

Amazon CloudWatch RUM の注意事項、料金

Amazon CloudWatch RUM の制限事項

- 利用可能リージョン (2023/4月時点)

- 米国東部 (バージニア北部)
- 米国東部 (オハイオ)
- 米国西部 (オレゴン)
- 欧州 (フランクフルト)
- 欧州 (ストックホルム)
- 欧州 (アイルランド)
- 欧州 (ロンドン)
- アジアパシフィック (東京)
- アジアパシフィック (シンガポール)
- アジアパシフィック (シドニー)



大阪リージョンは未対応
(2023/4月時点)

- クオータ

クオータ	デフォルト値	引き上げリクエスト可能か
アプリケーションモニター数	アカウントあたり 20 個	Yes
RUM 取り込み率	50 PutRumEvents リクエスト / 秒 (TPS)	Yes

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM.html

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch-RUM-quotas.html

RUM Webクライアントに HTML への組み込みコードスニペットを利用する場合

広告ブロッカーがデフォルトの cwr.js ディストリビューションをブロックすることがある。下記いずれかの方法で回避可能

- Web アプリケーションで CloudWatch RUM Web クライアントをホスト
 - a) cwr.js を Web アプリケーションの assets ディレクトリにコピー
 - b) (a) の cwr.js のコピーを使用するようにコード スニペットを変更
- RUM ウェブ クライアントを JavaScript モジュールとしてインストール

https://github.com/aws-observability/aws-rum-web/blob/main/docs/cdn_installation.md#instrument-the-application



VPC エンドポイント経由でデータを送信する場合

CloudWatch RUM は専用の VPC エンドポイント (PrivateLink) が用意 (VPCエンドポイントポリシーに対応)

サービスカテゴリ AWS サービス サービスを名前で検索 ご使用の AWS Marketplace サービス

サービス名 com.amazonaws.ap-northeast-1.rum i

サービス名	所有者	タイプ
com.amazonaws.ap-northeast-1.rum	amazon	Interface
com.amazonaws.ap-northeast-1.rum-dataplane	amazon	Interface

サービス名	説明
com.amazonaws.<region>.rum	CloudWatch RUMコントロールプレーンのVPCエンドポイント
com.amazonaws.<region>.rum-dataplane	CloudWatch RUMデータプレーンのVPCエンドポイント (=メトリクス送信先)

Amazon CloudWatch RUM の料金

クライアントから送信されるイベント数に基づく従量課金

2023/4月時点での利用料金(全リージョン同一)

Amazon CloudWatch RUM 料金

イベント数

100,000イベント
あたり 1.00 USD/月

※イベント数(*)はサンプル率等の設定に依存

+

(関連サービス利用料)

CloudWatch
(Logs/Metrics/Alarm/…)

+

Cognito

+

X-Ray

+

※イベント例：ページビュー、描画、ナビゲーション、リソースロード、エラー 等
(それぞれが1つのイベントとしてカウント)

<https://aws.amazon.com/jp/cloudwatch/pricing/>



まとめ



まとめ

- Amazon CloudWatch RUMはさまざまな場所、デバイス、プラットフォーム、ブラウザからの洞察と問題を特定するためのダッシュボードを提供
 - JavaScript / HTTP レスポンスのエラーなど、パフォーマンス問題に関する情報を可視化
 - アプリケーションのエラーとクラッシュの診断
 - AWS X-Ray などの他のAWSサービスとも連携

**実際のユーザアクセスに基に計測を行うことで、ユーザ一体験向上に
フォーカスした機能改善が可能**

本資料に関するお問い合わせ・ご感想

技術的な内容に関しましては、有料のAWSサポート窓口へ
お問い合わせください

<https://aws.amazon.com/jp/premiumsupport/>

料金面でのお問い合わせに関しましては、カスタマーサポート窓口へ
お問い合わせください（マネジメントコンソールへのログインが必要です）

<https://console.aws.amazon.com/support/home#/case/create?issueType=customer-service>

具体的な案件に対する構成相談は、後述する個別相談会をご活用ください



ご感想はTwitterへ！ハッシュタグは以下をご利用ください
#awsblackbelt



その他コンテンツのご紹介

ウェビナーなど、AWSのイベントスケジュールをご参照いただけます

<https://aws.amazon.com/jp/events/>

ハンズオンコンテンツ

<https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-hands-on/>

AWS 個別相談会

AWSのソリューションアーキテクトと直接会話いただけます

<https://pages.awscloud.com/JAPAN-event-SP-Weekly-Sales-Consulting-Seminar-2021-reg-event.html>



Thank you!



Amazon CloudWatch Synthetics

AWS Black Belt Online Seminar

高野 翔史

Solutions Architect
2023/03

AWS Black Belt Online Seminarとは

- ・ 「サービス別」「ソリューション別」「業種別」などのテーマに分け、
アマゾン ウェブ サービス ジャパン合同会社が提供するオンラインセミナー
シリーズです
- ・ AWSの技術担当者が、AWSの各サービスやソリューションについてテーマご
とに動画を公開します
- ・ 動画を一時停止・スキップすることで、興味がある分野・項目だけの聴講も
可能、スキマ時間の学習にもお役立ていただけます
- ・ 以下のURLより、過去のセミナー含めた資料などをダウンロードするこ
とができます
 - ・ <https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-service-cut/>
 - ・ <https://www.youtube.com/playlist?list=PLzWGOASvSx6FIwIC2X1nObr1KcMCBBlqY>

内容についての注意点

- ・ 本資料では 2023 年 03 月時点のサービス内容および価格についてご説明しています。最新の情報は AWS 公式ウェブサイト (<https://aws.amazon.com/>) にてご確認ください
- ・ 資料作成には十分注意しておりますが、資料内の価格と AWS 公式ウェブサイト記載の価格に相違があった場合、AWS 公式ウェブサイトの価格を優先とさせていただきます
- ・ 価格は税抜表記となっています。日本居住者のお客様には別途消費税をご請求させていただきます

自己紹介

名前：高野 翔史

所属：エンタープライズ技術本部
ソリューションアーキテクト

経歴：国内SIerにてシステム開発に従事



好きなAWSサービス：Amazon CloudWatch、Amazon Timestream

本セミナーの対象者

- これからAWSを利用した運用設計を担当される方
- AWSでの監視設計を担当される方
- AWSを利用した外型監視に興味のある方

アジェンダ

Amazon CloudWatch Synthetics とは

Amazon CloudWatch Synthetics Canaryとは

Amazon CloudWatch Synthetics による Synthetics Monitoring

AWS モニタリングサービスとの連携

Amazon CloudWatch Synthetics 料金

まとめ

アジェンダ

Amazon CloudWatch Synthetics とは

Amazon CloudWatch Synthetics Canaryとは

Amazon CloudWatch Synthetics による Synthetics Monitoring

AWS モニタリングサービスとの連携

Amazon CloudWatch Synthetics 料金

まとめ

モニタリング (Monitoring) とは？

体系的な見直しを続けるために、一定期間にわたって、(何かの) 経過状況や品質を観察し確認する

observe and check the progress or quality of (something) over a period of time; keep under systematic review

引用 : Oxford English Dictionary

ユーザ体験の監視の重要性



サービスやアプリの健全性をチェックする



アプリケーションを元の状態に回復する



トラブルの原因を調査する



ユーザ行動を分析する



キヤパシティを分析する

ユーザ体験の監視の重要性



サービスやアプリの
健全性をチェックする



アプリケーションを
元の性能に回復する



トラブルの原因を
調査する

監視は ユーザ体験を損なわない



ために行う



ユーザ行動を
分析する

キャパシティを
分析する

ユーザ体験の監視手法

Synthetic Monitoring

※ Active monitoring の一種

- 定常に一定間隔で計測プログラムがアクセスしてデータを取得することを通じてパフォーマンスを計測する手法(主に製造業で利用される統計的品質管理手法と同じ思想に基づく計測手法)
- 短期のパフォーマンス計測用途に好適
- 一般に「外形監視」と訳される

Real User Monitoring

※ Passive monitoring の一種

- 実ユーザーアクセスの情報を取得して計測する手法
(実ユーザーによる実ブラウザでの実ロケーションからのアクセステータを取得・計測サーバーに送信)
- 中長期のユーザー傾向把握用途に好適

ユーザ体験の監視手法

Synthetic Monitoring

※ Active monitoring の一種

- 定常に一定間隔で計測プログラムがアクセスしてデータを取得することを通じてパフォーマンスを計測する手法(主に製造業で利用される統計的品質管理手法と同じ思想に基づく計測手法)
- 短期のパフォーマンス計測用途に好適
- 一般に「外形監視」と訳される



Amazon CloudWatch Synthetics

Real User Monitoring

※ Passive monitoring の一種

- 実ユーザーアクセスの情報を取得して計測する手法
(実ユーザーによる実ブラウザでの実ロケーションからのアクセステータを取得・計測サーバーに送信)
- 中長期のユーザー傾向把握用途に好適



Amazon CloudWatch RUM

ユーザ体験の監視手法

Synthetic Monitoring

※ Active monitoring の一種

- 定常的に一定間隔で計測プログラムがアクセスしてデータを取得することを通じてパフォーマンスを計測する手法(主に製造業で利用される統計的品質管理手法と同じ思想に基づく計測手法)
- 短期のパフォーマンス計測用途に好適
- 一般に「外形監視」と訳される



Amazon CloudWatch Synthetics

Real User Monitoring

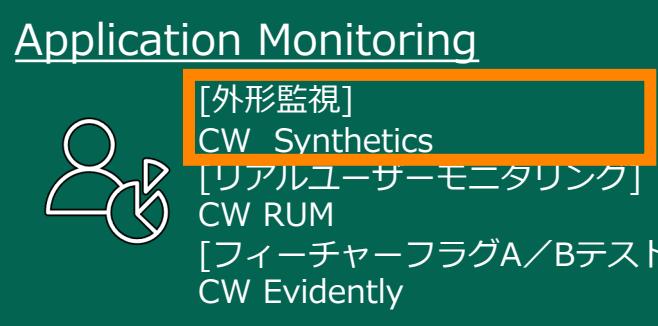
※ Passive monitoring の一種

- 実ユーザーアクセスの情報を取得して計測する手法
(実ユーザーによる実ブラウザでの実ロケーションからのアクセステータを取得・計測サーバーに送信)
- 中長期のユーザー傾向把握用途に好適



Amazon CloudWatch RUM

Amazon CloudWatchの全体像

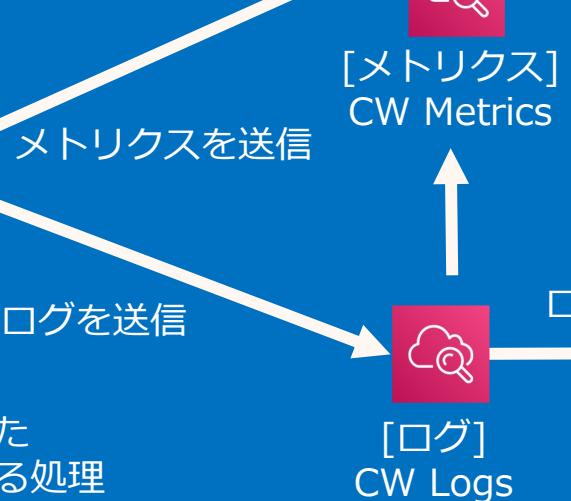


[インターネット監視] CW Internet Monitor
[トレース] CW ServiceLens

[ダッシュボードに統合] CW Dashboard



Infrastructure



CW Metrics Stream



CW Logs Insights



[イベント]
Amazon EventBridge/
Amazon EventBridge Scheduler

Insights

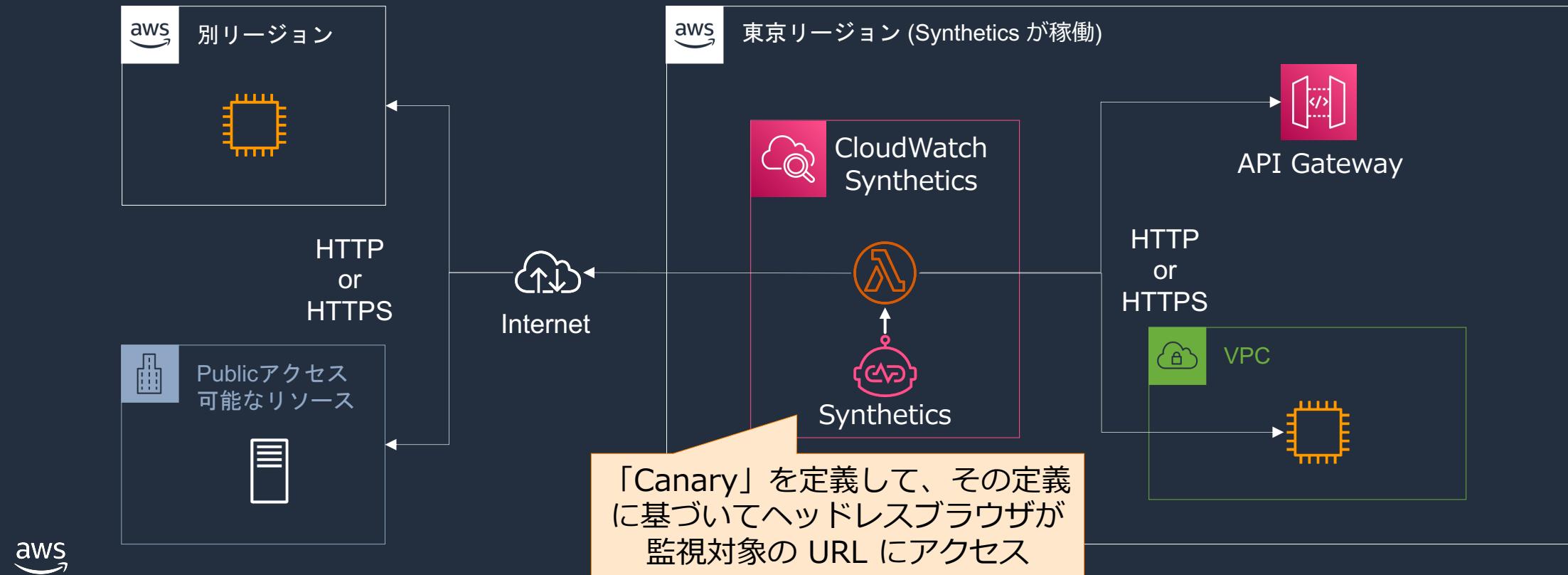


[構造化 ログによる メトリクス]
CW Container Insights / Contributor Insights
[Lambda 拡張機能による メトリクス]
Lambda Insights
[アプリケーション コンポーネント の メトリクス]
Application Insights

※CW = CloudWatch

Amazon CloudWatch Synthetics による Monitoring

- CloudWatch Synthetics は、Web アプリケーションと API を簡単に監視できるようにするサービスです。
- 実際のユーザー トラフィックがなくてもエンドポイントを継続的にテストすることができます。
- 設定したしきい値に基づいてエラーがあればアラートを上げることができます。



Amazon CloudWatch Synthetics Canary 作成画面

設計図

- ハートピートのモニタリング
1つの URL で基本的なページのロードを実行します。
- API Canary
API を HTTP ステップとしてモニタリングします。
- リンク切れチェッカー
指定された URL で基本的なウェブクローラーを実行します。

Canary ビルダー

名前

synthetics-demo

名前は、21 文字までの小文字、数字、ハイフン、またはアンダースコアで構成され、スペースを含めることはできません。

アプリケーションまたはエンドポイント URL 情報

http://[REDACTED] us-east-1.elb.amazonaws.com

削除

エンドポイントを追加

最大でさらに 4 個のエンドポイントを追加できます。スクリプトを変更することで、さらにエンドポイントを追加できます。

スクリーンショット

スクリーンショットを撮る
スクリーンショットは、Canary 実行ごとに Canary の詳細画面に表示されます

aws

Amazon Web Services, Inc. or its affiliates.

Blueprint が用意されており、
基本的なユースケースにおいては、
ノーコードで Canary を
作成できます (詳細は後程詳しく
紹介)。

Amazon CloudWatch Synthetics Canary 実行結果概要

Canary Canary のステータス一覧

3 時間 ▾

ステータス
現在実行中の Canary のステータス配分



● 成功 (5) ● 失敗 (3)

グループ
最終実行による

合計	失敗	アラーム
1	1 ⚠️	1 ⚠️

最も遅いパフォーマンス
最後の 3 時間

最も遅いグループ	最も遅いリージョン
18.2s	1. アジアパシフィック (東京) 2. 米国東部 (バージニア北部)

Canary (8)

リソースを検索 グループを表示 アクション リストを作成 Canary を作成

名前	前回の実行ステータス	成功 %	アラーム	平均時間	状態	ランタイムバージョン	リージョン
DemoGroup	④ 1/2 失敗	74%	⚠️ 1	18.2s	-	-	-
petsite	⑤ 成功	77%	⚠️ 1	17.9s	平均実行時間が表示される	米国東部 (バージニア北部)	
petsite-tokyo	⑥ 失敗	71%	-	18.6s	実行中	syn-nodejs-puppeteer-3.9	アジアパシフィック (東京)
petsite-heartbeat	⑤ 成功	74%	-	18.1s	実行中	syn-nodejs-puppeteer-3.9	米国東部 (バージニア北部)
petsite-python	⑤ 成功	-	-	-	実行中	syn-python-selenium-1.3	米国東部 (バージニア北部)
petsite-visu	⑤ 成功	63%	-	6.7s	実行中	syn-nodejs-puppeteer-3.9	米国東部 (バージニア北部)
petsite-visual	⑥ 失敗	0%	⚠️ 1	0ms	停止	syn-nodejs-puppeteer-3.9	米国東部 (バージニア北部)
test	⑤ 成功	0%	-	-	CloudWatch アラームを設定しており、条件にマッチしている場合は表示される	peteer-3.9	米国東部 (バージニア北部)
test-2-screenshotoff	⑥ 失敗	0%	-	-	-	peteer-3.9	米国東部 (バージニア北部)

aws

Amazon CloudWatch Synthetics のメリット

- 必要な時に素早く Synthetics Monitoring を実現
- モニタリング動作の振る舞いをプログラムで制御可能
 - シンプルな用途であれば、組み込み提供のテンプレート (Blueprint) を用いるだけでプログラミングは不要
 - カスタマイズにより複雑な操作も表現可能
- AWS サービスとの連携による高い柔軟性
 - S3, EventBridge, X-Ray などの AWS サービスと連携

アジェンダ

Amazon CloudWatch Synthetics とは

Amazon CloudWatch Synthetics Canaryとは

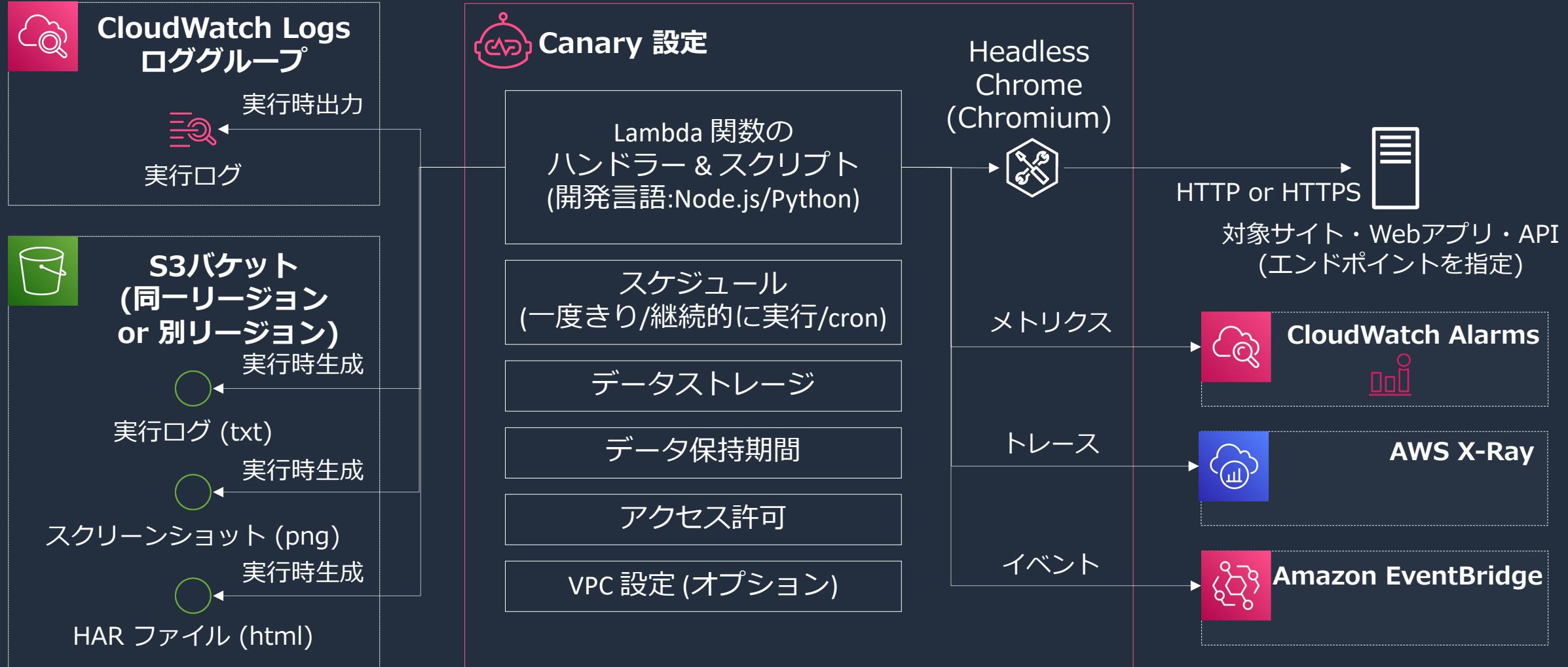
Amazon CloudWatch Synthetics による Synthetics Monitoring

AWS モニタリングサービスとの連携

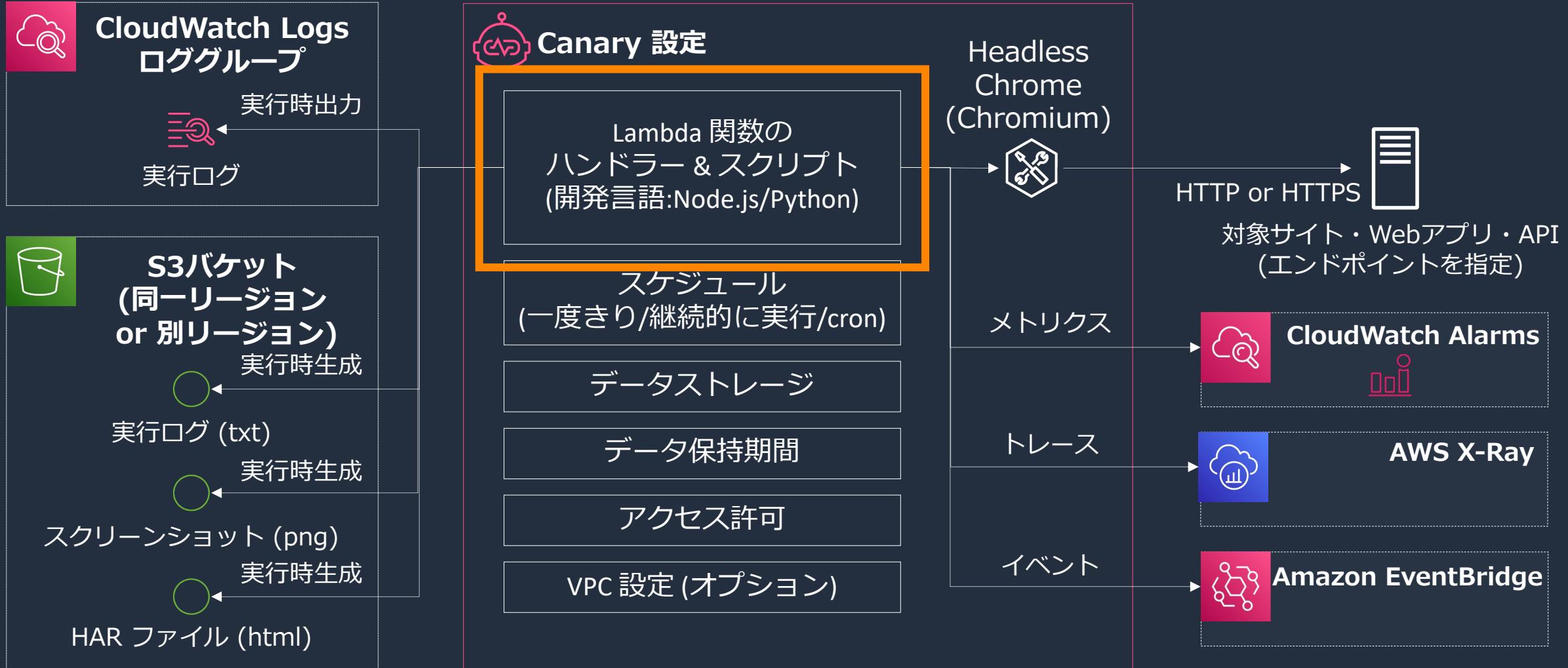
Amazon CloudWatch Synthetics 料金

まとめ

Amazon CloudWatch Synthetics Canary 全体構成イメージ



Amazon CloudWatch Synthetics Canary 全体構成イメージ



Canary 設定（スクリプト）

Lambda Layer として提供される Synthetics ランタイムを利用して、Node.js 又は Python コードでモニタリングしたい操作を記述します。



```
const { URL } = require('url');
const synthetics = require('Synthetics');
const log = require('SyntheticsLogger');
const syntheticsConfiguration = synthetics.getConfiguration();
const syntheticsLogHelper = require('SyntheticsLogHelper');

const loadBlueprint = async function () {
    const urls = [""];

    // Set screenshot option
    const takeScreenshot = true;

    :
}
```

Lambda Layer として AWS が提供するライブラリ

- ・ 設計図 (Blueprint) を利用するか、インラインエディタを使ってコーディングするか、S3 にスクリプトを配置してインポートするか 3 つのやり方を選択可能です。
- ・ Blueprint はユースケースに応じて用意されており、基本的な用途はコーディング不要で利用可能です（後程詳しく紹介）。
- ・ 環境変数を設定可能のため、コードを更新しなくても動作を調整可能です。
※環境変数は暗号化されないため、機密情報を利用する場合は AWS Secrets Manager と連携して下さい

※利用可能なAPIや詳細なコーディングについては

https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Function_Library.html



CloudWatch Synthetics ランタイムについて

- 利用するプログラミング言語ごとに、ブラウザ操作ライブラリが異なります。

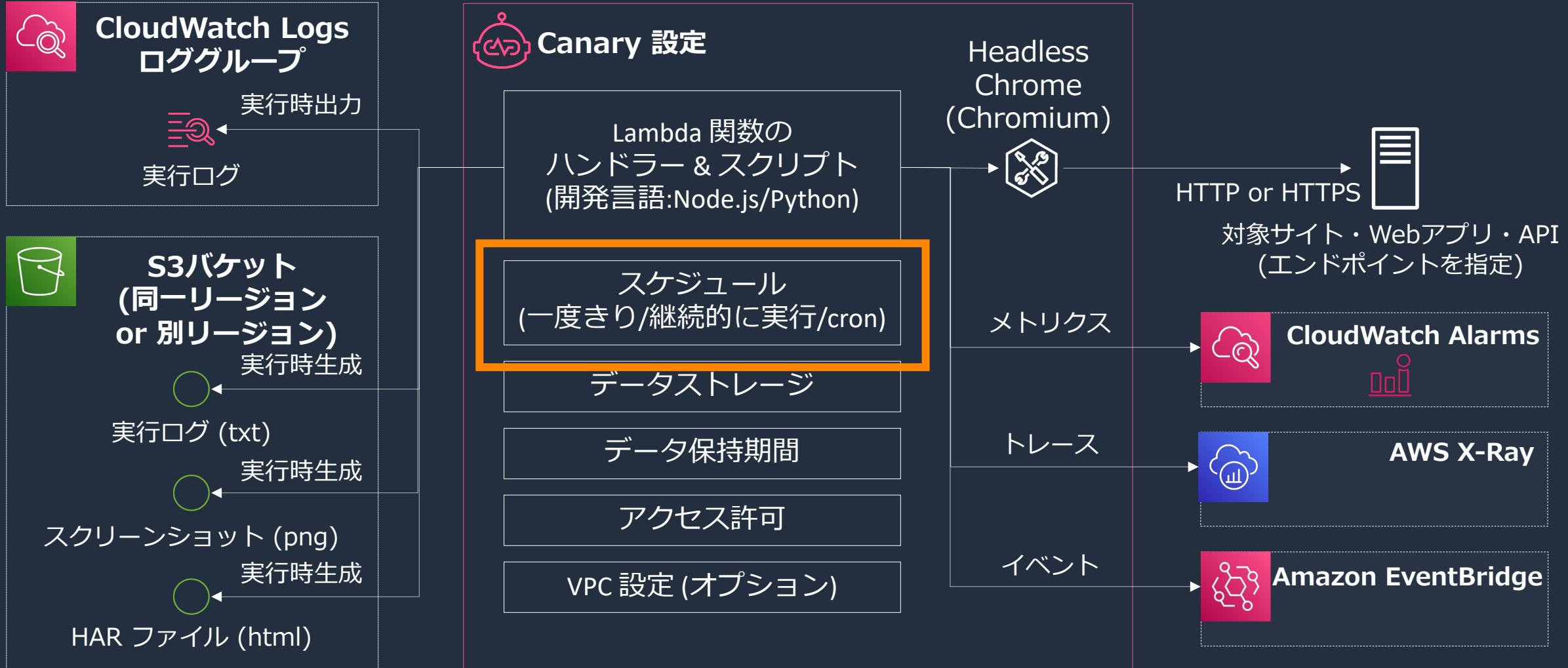
Lambda ランタイム	ブラウザ操作ライブラリ	ヘッドレスブラウザ
Node.js	Puppeteer	Chromium
Python	Selenium	Chromium

- Synthetics ランタイムのバージョンによって、Lambda ランタイム (Node.js/Python)、ブラウザ操作ライブラリ (Puppeteer/Selenium)、ヘッドレスブラウザ (Chromium) のバージョンが異なります。ランタイムのバージョンごとの各コンポーネントバージョンは下記 URL をご確認下さい。
 - Node.js と Puppeteer を利用するランタイムのバージョン
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Library_nodejs_puppeteer.html
 - Python と Selenium Webdriver を使用するランタイムバージョン
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Library_python.selenium.html

CloudWatch Synthetics ランタイム サポートポリシー

- Synthetics ランタイムに含まれているコンポーネントがサポートされなくなると、その Synthetics ランタイムのバージョンは**非推奨**になります。非推奨のランタイムバージョンを使用して新規に Canary を作成することはできません。既存で非推奨のランタイムバージョンを利用している Canary は利用可能ですが、**サポートされているランタイムバージョンへの更新を推奨**します。**60日以内**に非推奨となる予定のランタイムを使用している Canary がある場合に CloudWatch Synthetics は**通知メールを利用者に送信**します。サポートポリシー、ランタイムのバージョンアップ方法については、下記 URL をご確認ください。
- CloudWatch Synthetics ランタイムサポートポリシー
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Library.html#CloudWatch_Synthetics_Canaries_runtime_support

Amazon CloudWatch Synthetics Canary 全体構成イメージ



Canary 設定（スケジュール）

スケジュール 情報
この Canary を編集し、いつでも実行スケジュールを変更できます

「継続的に実行」、「CRON 式」、「1 回実行」の 3 つの中から 1 つを選択

継続的に実行
Canary 実行の頻度をスケジュールする

CRON 式
CRON 式で Canary をスケジュールする

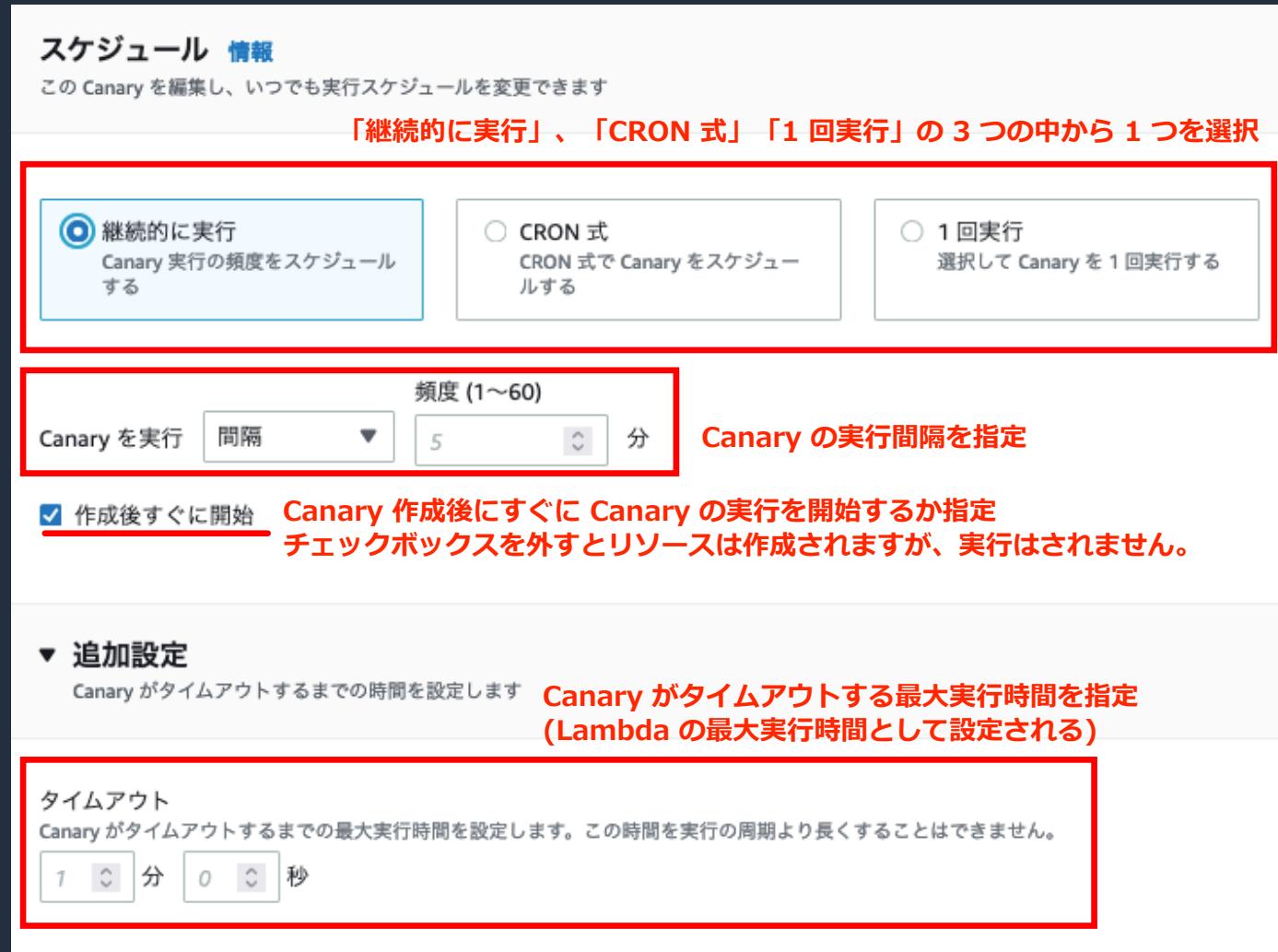
1 回実行
選択して Canary を 1 回実行する

Canary を実行 間隔 ▾ 5 分 Canary の実行間隔を指定

作成後すぐに開始 Canary 作成後にすぐに Canary の実行を開始するか指定
チェックボックスを外すとリソースは作成されますが、実行はされません。

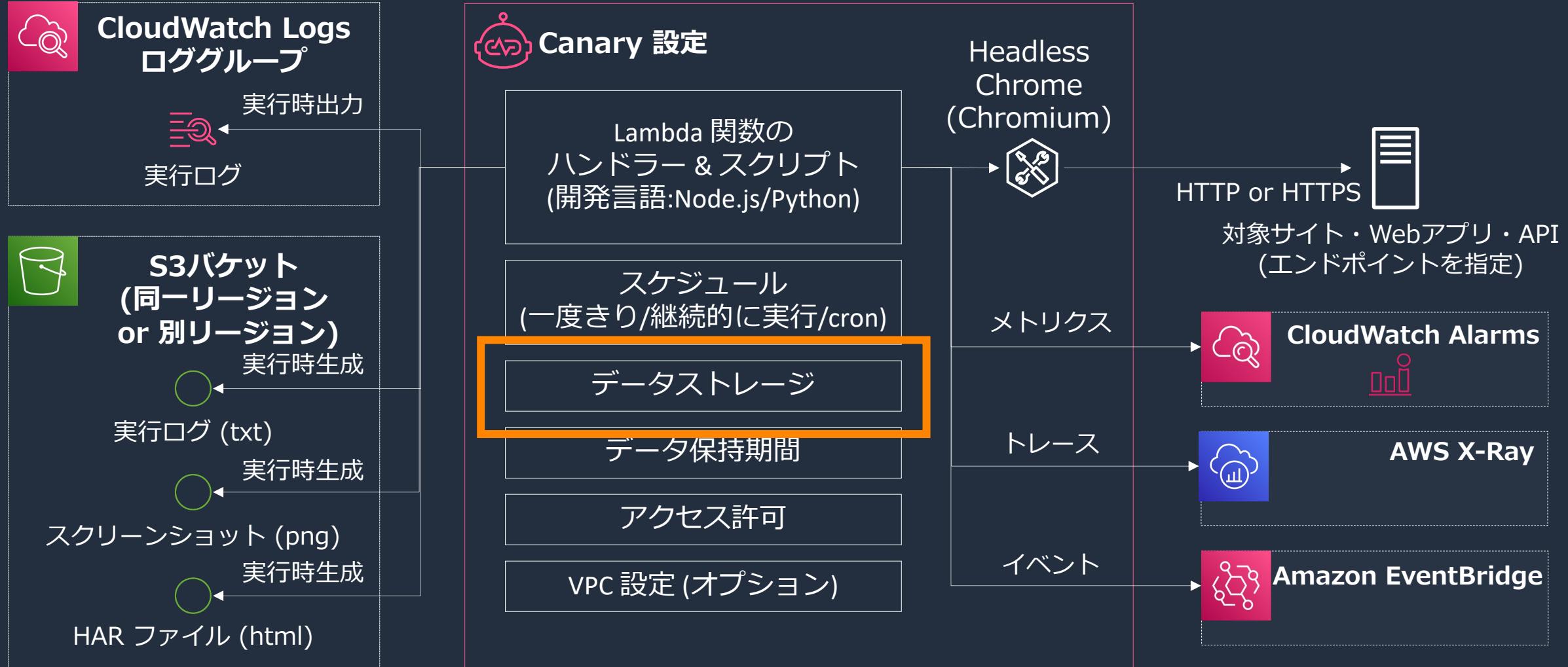
▼ 追加設定
Canary がタイムアウトするまでの時間を設定します Canary がタイムアウトする最大実行時間を指定 (Lambda の最大実行時間として設定される)

タイムアウト
Canary がタイムアウトするまでの最大実行時間を設定します。この時間を実行の周期より長くすることはできません。
1 分 0 秒



- Canary の実行スケジュールを指定します。以下の設定が可能です。
 - 継続的に実行
 - 間隔：1~60 分の間で指定
 - 毎日：タイムゾーン UTC で時刻を指定
 - CRON 式
 - CRON 式を使って柔軟に実行するタイミングを指定して継続的に実行する
 - 1 回実行
 - 1 回のみ実行する
- Canary がタイムアウトするまでの最大実行時間を設定可能です。
実行スケジュール間隔より長くは設定できません。
Lambda コールドスタート等の時間を許容するために 15 秒以上は長く設定して下さい。

Amazon CloudWatch Synthetics Canary 全体構成イメージ



Canary 設定（データストレージ）

データストレージ
各 Canary 実行によって作成されたアーティファクトを保存する S3 フォルダを選択します。

Canary 実行データは、Amazon S3 ストレージリソースバケットに保存されます。
Amazon S3 の詳細は、こちらをご参照ください。[リンク]

S3 の場所
デフォルトの S3 バケットが使用または作成されます。または AWS アカウントから既存の S3 バケットを選択してください。

[表示] [S3 を参照]

▼ 追加設定 情報
SSE-S3 または AWS KMS を使用して Canary アーティファクトを暗号化します。

暗号化
アーティファクトの暗号化方法を選択

暗号化方法を選択
デフォルトでは、Canary アーティファクトは AWS で管理されたキーを使用して保管時に暗号化されます。別の暗号化方法を選択できます。

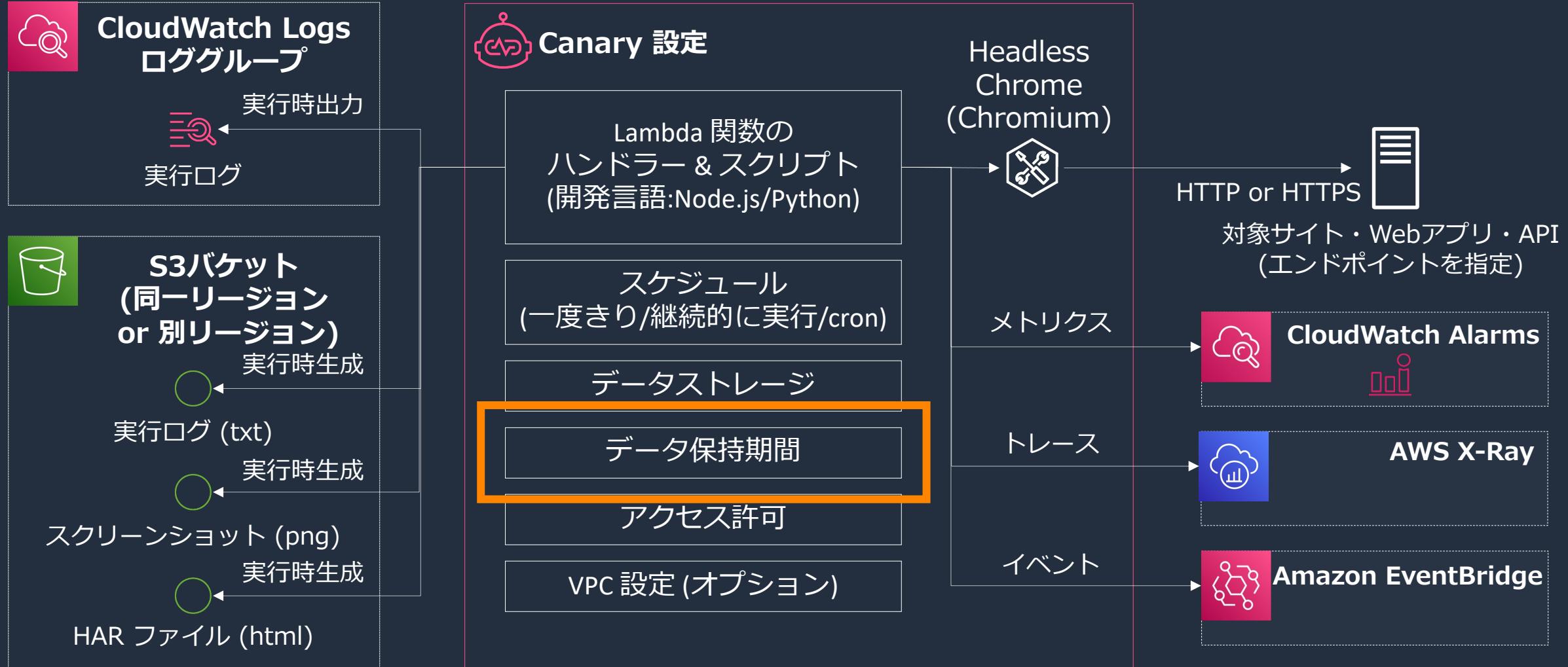
SSE-S3 暗号化を使用
SSE-S3 暗号化を使用

AWS Key Management Service (AWS KMS) を使用
AWS KMS でカスタマーマネージドキーを使用

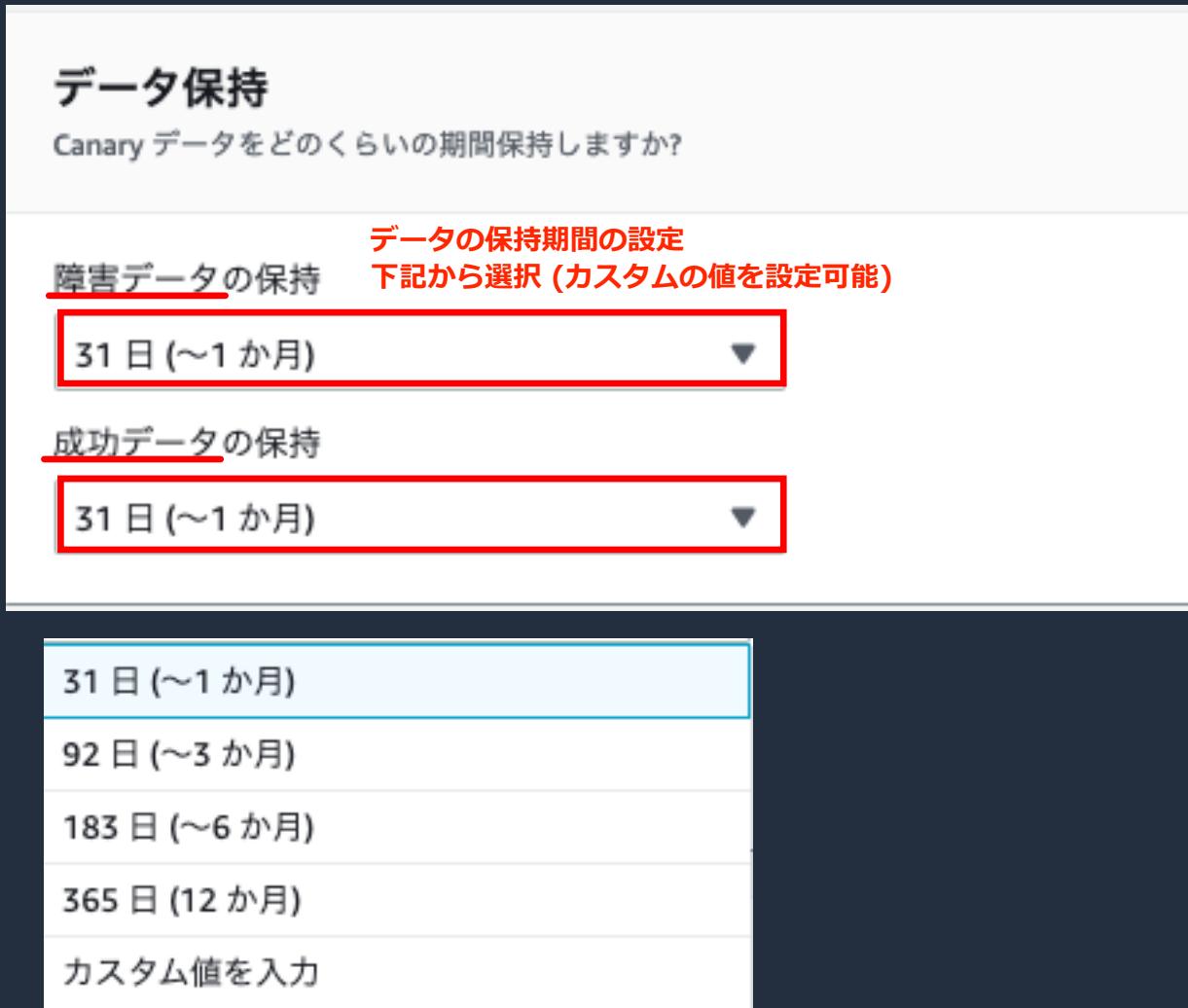
[AWS KMS キーを作成]

- Canary によって作成されるアーティファクト（ログ/スクリーンショット/HARファイル）の出力場所を指定します。
- デフォルトでは一番最初に Canary を作成する際は、同一リージョンに CloudWatch Synthetics 用に S3 バケットを作成して保存し、それ以降は作成された S3 バケットに保存するように設定されます。既存 S3 バケット（別リージョンでも可能）を指定することも可能です。
- 「syn-nodejs-puppeteer-3.3」以降のランタイムバージョンを使用している場合、アーティファクトの暗号化方法を選択でき、SSE-S3 暗号化又は SSE-KMS（カスタマーマネージドキー）を選択可能です。
※ 「syn-python-selenium」を利用する場合は本機能を利用することはできないことにご注意下さい。

Amazon CloudWatch Synthetics Canary 全体構成イメージ



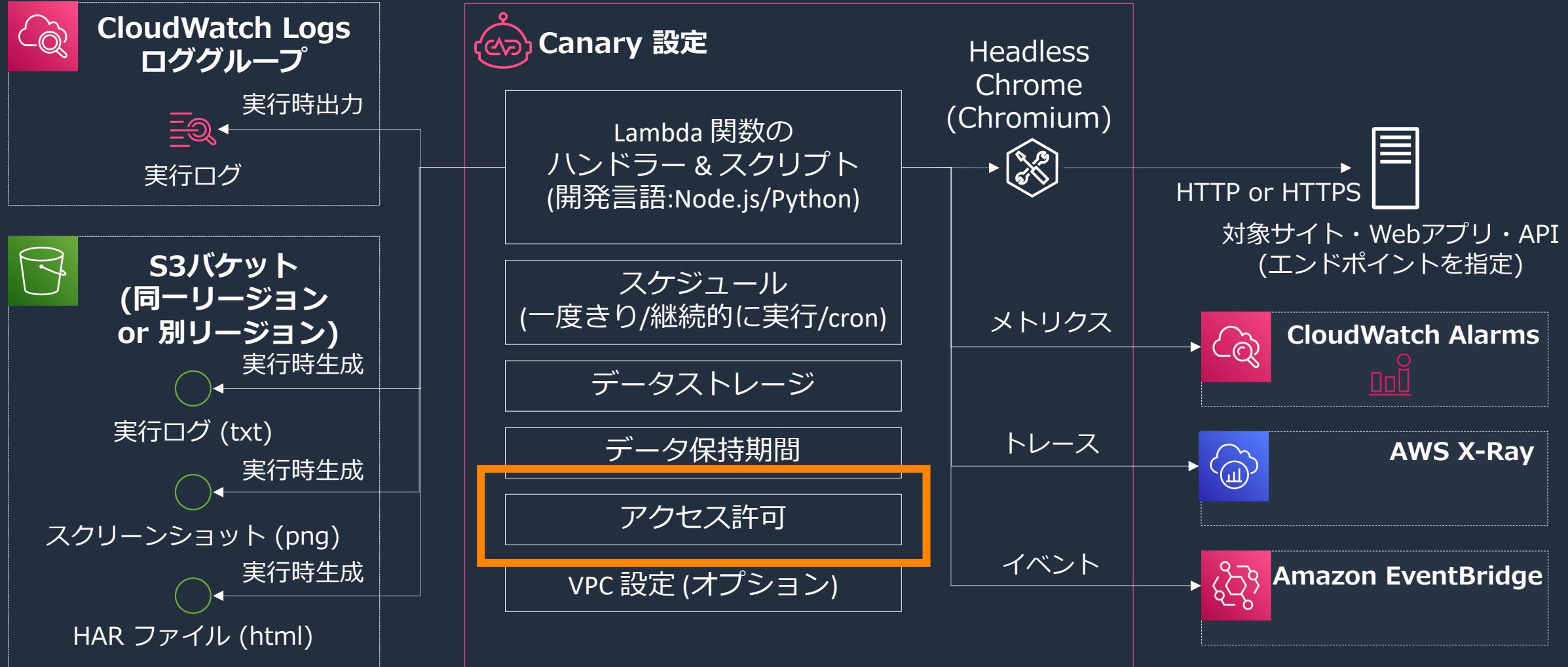
Canary 設定（データ保持期間）



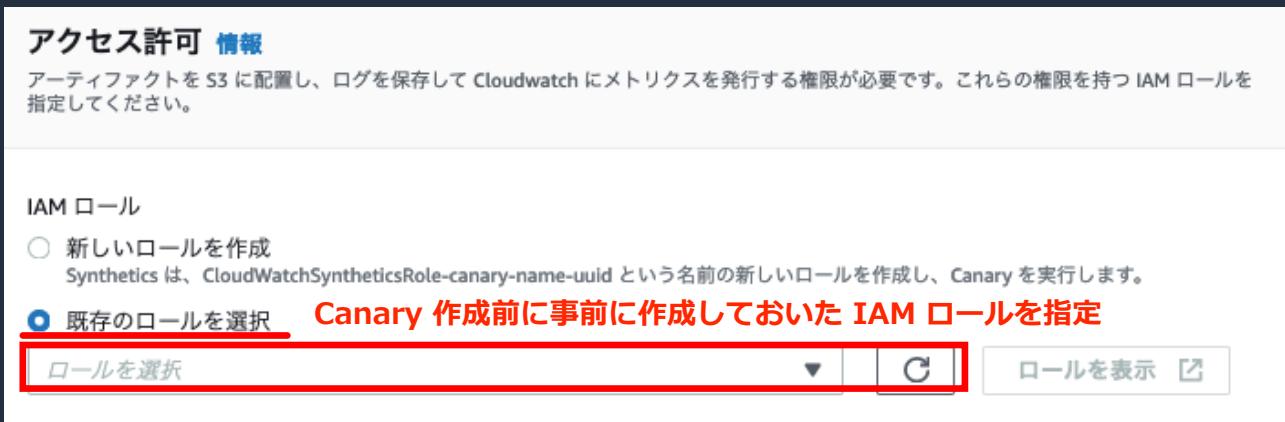
- 成功した Canary 実行結果と失敗した Canary 実行結果のそれぞれに対して、どれくらいの期間情報を保持するのか指定します。指定できる範囲は、1 - 455 日までです。

※あくまで AWS コンソールに保存および表示するデータに関する設定になります。S3 バケットに保存されるアーティファクト等には影響しない点はご注意下さい。

Amazon CloudWatch Synthetics Canary 全体構成イメージ

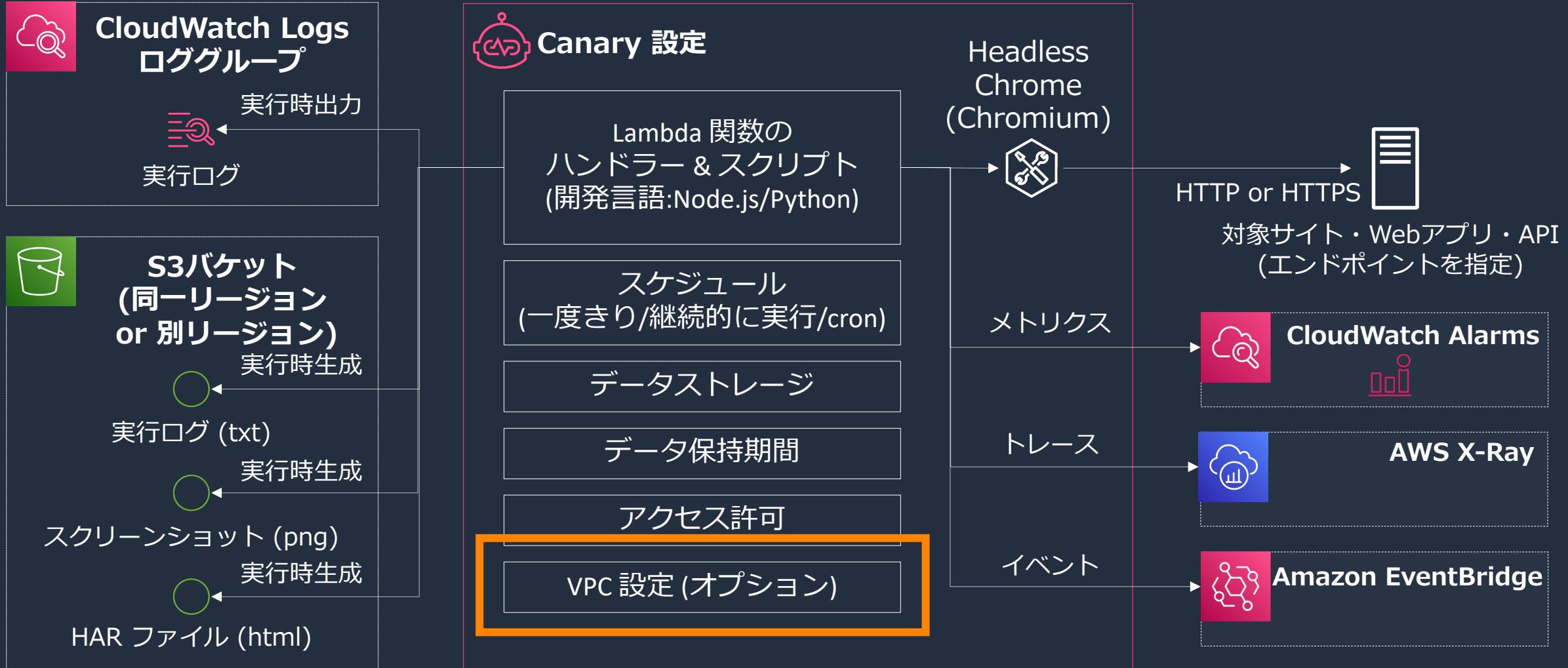


Canary 設定（アクセス許可）



- Canary によって作成されるアーティファクトを S3 バケットに保存したり、CloudWatch にメトリクスを発行するための IAM ロール (Lambda に設定される) を設定します。
- Canary 新規作成時に新しく作成するか、既存の IAM ロールを選択して設定することが可能です。
- 事前に IAM ロールを作成する場合は、構成によって必要な権限が異なります。下記を参照しながら IAM ロールを作成して下さい。
- Canaryに必要なロールとアクセス許可
https://docs.aws.amazon.com/ja_jp/AzonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_CanaryPermissions.html

Amazon CloudWatch Synthetics Canary 全体構成イメージ



Canary 設定 (VPC 設定 - オプション)

▼ VPC 設定 - オプション 情報
エンドポイントがネットワーク下にある場合に使用します

Virtual Private Cloud (VPC) Canary を起動する VPC を指定
vpc- [REDACTED] (10.0.0.0/16) ▾

① Synthetics は CloudWatch でメトリクスを作成し、スクリーンショット、HAR ファイル、ログなどの Canary 実行データを S3 に保存します。パブリックインターネットにアクセスせずに VPC を使用している場合は、特定のリージョンの CloudWatch メトリクスと S3 に VPC エンドポイントを追加してください。詳細はこちら ▾

サブネット Canary を起動するサブネットを指定
VPC 設定のセットアップに使用する Lambda の VPC サブネットを選択します。形式:「subnet-id (cidr-block) | az-name-tag」
オプションの選択 ▾

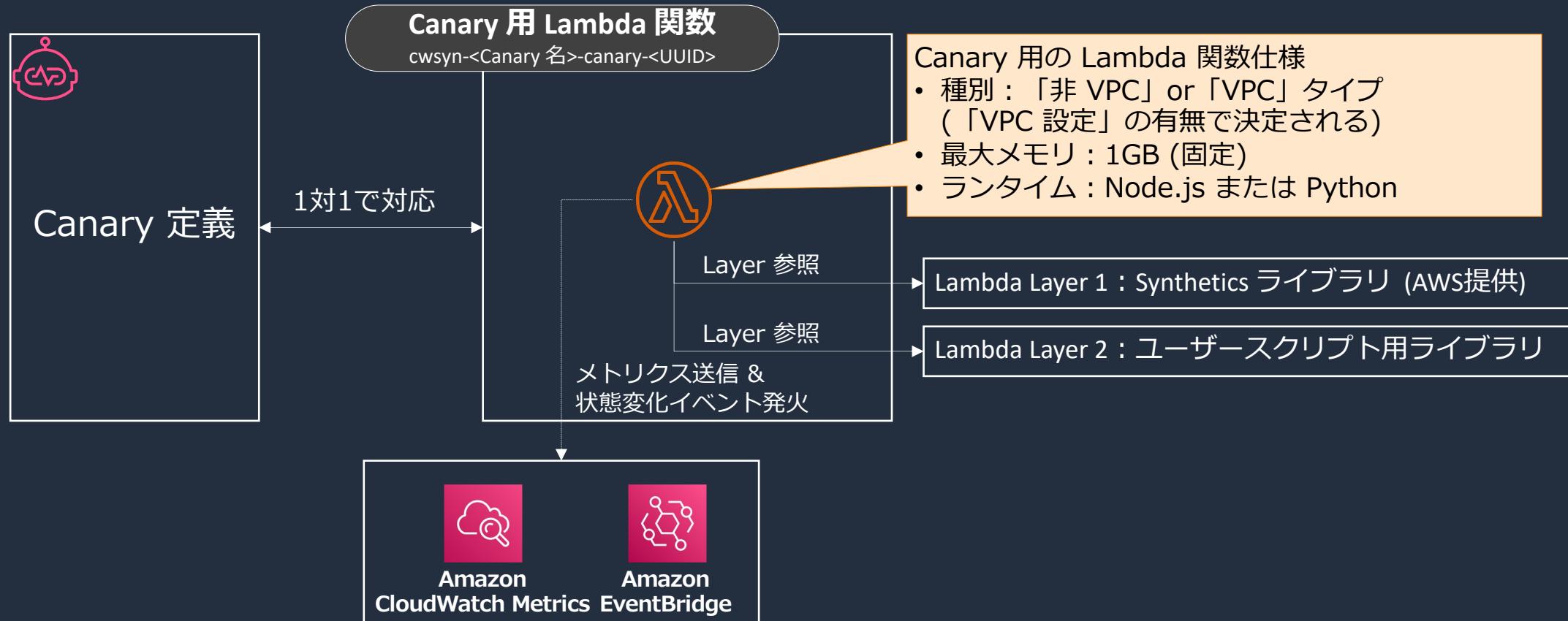
セキュリティグループ Canary に設定するセキュリティグループを指定
VPC 設定のセットアップに使用する Lambda の VPC セキュリティグループを選択します。形式:「sg-id (sg-name) | name-tag」。次の表に、選択したセキュリティグループのインバウンドルールとアウトバウンドルールを示します。
オプションの選択 ▾



- Canary は特定の VPC 内のリソースに対してもモニタリングすることが可能です。
- VPC 内の特定のサブネットから Canary (実体はLambda) を起動し、アーティファクトを S3 に保存したり、CloudWatch にメトリクスを送信できます。通信要件に応じてセキュリティグループを事前に用意して設定して下さい。
- VPC 内からインターネットにアクセスできず S3 等のパブリックエンドポイントにアクセスできない場合は、別途 VPC 内に VPC エンドポイントを作成して下さい。

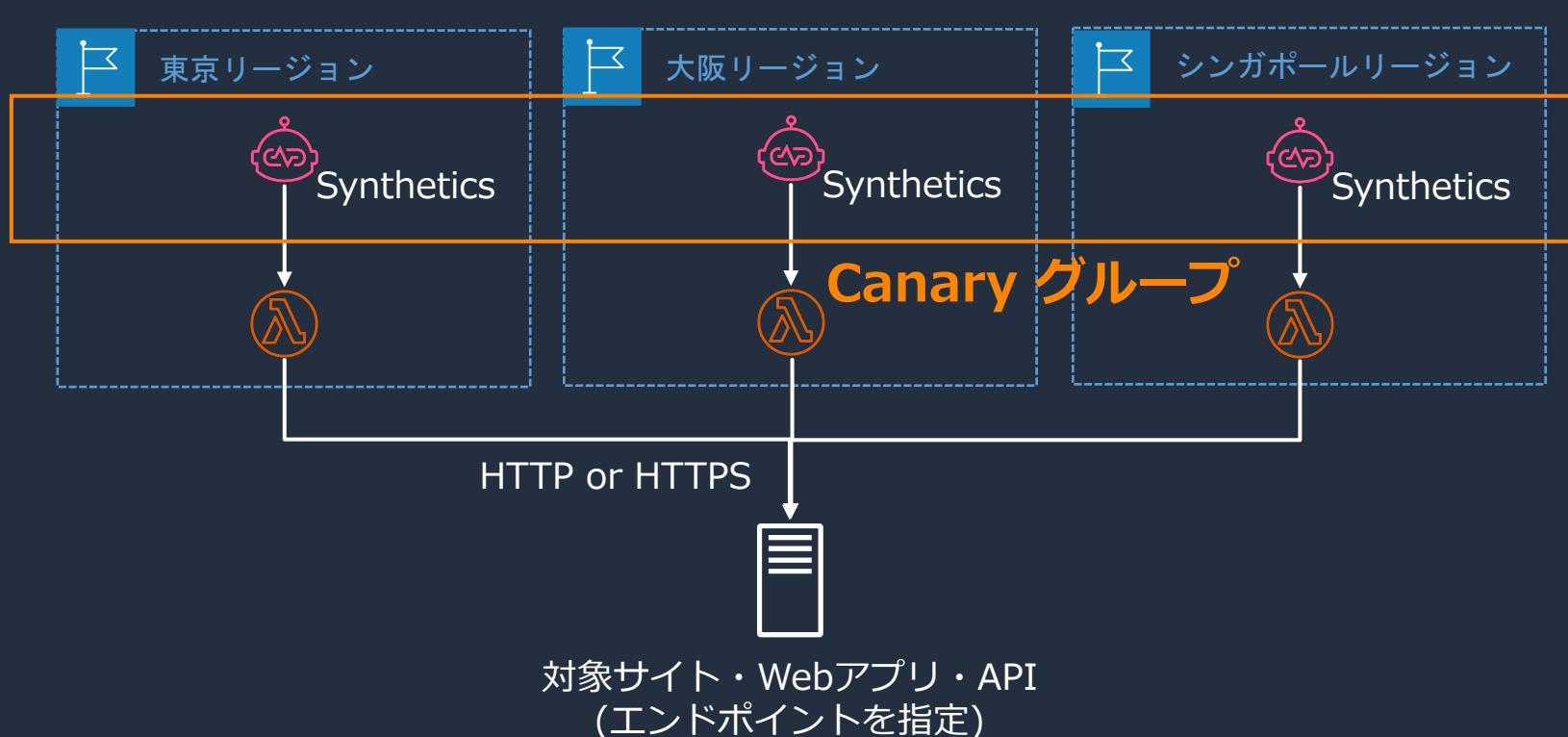
CloudWatch Synthetics Lambda 関数

Canary で定義したコードは、Lambda Layer として登録されて、CloudWatch Synthetics が作成する Lambda 関数コード内で読み込まれて実行されます。



Amazon CloudWatch Synthetics Canary グループ

リージョンを跨って、複数の Canary をグルーピングして
(1 グループ当たり最大 10) 、 Canary の結果をグループとして表示することが可能



- Canary グループ はグローバルリソースとして登録されるため、複数のリージョンから表示されます。
- 1 つの AWS アカウント当たり最大 20 個のグループを作成できます。
- 複数リージョンに跨ってグルーピングすることが可能なため、各リージョン固有の問題かどうかを簡単に切り分けることができるようになります。

Amazon CloudWatch Synthetics Canary グループ 作成

グループ名
グループの名前を入力します。

名前
DemoGroup
名前には最大 64 文字を含めることができます。任意の Unicode 文字を使用できます。

選択
グループ化する最大 10 個の Canary を選択します。

正確な Canary 名 グルーピングしたい Canary を完全一致で入力して検索する
 検索

petsite-tokyo アジアパシフィック (東京) petsite 米国東部 (バージニア北部)

名前 リージョン 検索の結果出力された Canary にチェックを入れると登録される

petsite-tokyo アジアパシフィック (東京)



- ・ グループ名を入力し、作成済みの Canary を検索して、選択することで、グループ内に登録することが可能です。
- ・ グループの検索は、Canary 名を完全一致で入力する必要があることに注意して下さい。
※途中まで入力して、検索してもヒットしません。

アジェンダ

Amazon CloudWatch Synthetics とは

Amazon CloudWatch Synthetics Canaryとは

Amazon CloudWatch Synthetics による Synthetics Monitoring

AWS モニタリングサービスとの連携

Amazon CloudWatch Synthetics 料金

まとめ

Canary Blueprint

Canary の実体は Node.js/Python で実装される Lambda 関数で、6 種類の Blueprint が用意されており、基本的な用途であればコーディングが不要です。

No.	Blueprint	特徴	Node.js	Python
1	ハートビートのモニタリング	指定した URL にアクセスして、ページのスクリーンショットと HTTP アーカイブファイル (HAR ファイル) を保存	○	○
2	API Canary	REST API に対してリクエストを送信して、応答をテスト	○	○
3	リンク切れチェッカー	テスト対象のURL内のすべてのリンクを収集し、リンク切れがないかテスト	○	-
4	Canary レコーダー	Google Chrome の拡張機能である CloudWatch Synthetics Recorder を利用して、ユーザ操作を記録し、テスト	○	-
5	GUI ワークフロービルダー	Web サイト上のユーザ操作ができるかを GUI ベースで作成してテスト	○	○
6	ビジュアルモニタリング	Web サイトの表示が変化していないかをベースラインと比較し、テスト	○	-

Canary 設計図の使用

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html



Canary Blueprint

Canary の実体は Node.js/Python で実装される Lambda 関数で、6 種類の Blueprint が用意されており、基本的な用途であればコーディングが不要です。

No.	Blueprint	特徴	Node.js	Python
1	ハートビートのモニタリング	指定した URL にアクセスして、ページのスクリーンショットと HTTP アーカイブファイル (HAR ファイル) を保存	○	○
2	API Canary	REST API に対してリクエストを送信して、応答をテスト	○	○
3	リンク切れチェッカー	テスト対象のURL内のすべてのリンクを収集し、リンク切れがないかテスト	○	-
4	Canary レコーダー	Google Chrome の拡張機能である CloudWatch Synthetics Recorder を利用して、ユーザ操作を記録し、テスト	○	-
5	GUI ワークフロービルダー	Web サイト上のユーザ操作ができるかを GUI ベースで作成してテスト	○	○
6	ビジュアルモニタリング	Web サイトの表示が変化していないかをベースラインと比較し、テスト	○	-

Canary 設計図の使用

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html



Canary Blueprint - ハートビートのモニタリング

設計図

- ハートビートのモニタリング
- API Canary
- リンク切れチェッカー
- Canary レコーダー
- GUI ワークフロービルダー
- ビジュアルモニタリング

Canary ビルダー

監視対象のURLを入力

名前
synthetics-demo

名前は、21 文字までの小文字、数字、ハイフン、またはアンダースコアで構成され、スペースを含めることはできません。

アプリケーションまたはエンドポイント URL 情報

http:// st-1.elb.a [削除]

エンドポイントを追加

syn-nodejs-puppeteer-3.1以上の場合、エンドポイントを複数追加可能

最大でさらに 4 個のエンドポイントを追加できます。スクリプトを変更することで、さらにエンドポイントを追加できます。

スクリーンショット

スクリーンショットを撮る **スクリーンショットの保存はチェックボックスで指定**

スクリーンショットは、Canary 実行ごとに Canary の詳細画面に表示されます

- 指定した URL にアクセスして、Web ページのスクリーンショット、HTTP アーカイブファイル (HAR ファイル)、実行ログを保存します。スクリーンショットについては保存するかチェックボックスで指定可能です。
- レスポンスのステータスコードをもとに正常/異常を判定します (デフォルトでは、ステータスコード 200 番台なら正常と判定するようにコード内で設定されています)。
- 「syn-nodejs-puppeteer-3.1」以降のランタイムバージョンを使用している場合、複数の URL をモニタリングすることが可能になります。
※ 「syn-python-selenium」を利用する場合は、複数の URL をモニタリングすることはできないため、ご注意下さい。

Canary 実行結果概要

Canary Canary のステータス一覧

ステータス
現在実行中の Canary のステータス配分



● 成功 (5) ● 失敗 (3)

Canary グループのステータス一覧

3 時間 ▾

グループ
最終実行による

合計	失敗	アラーム
1	1 ⚠️	1 ⚠️

最も遅いパフォーマンス
最後の 3 時間

最も遅いグループ
18.2s

最も遅いリージョン
1. アジアパシフィック (東京)
2. 米国東部 (バージニア北部)

Canary (8)

リソースを検索

すべて表示

すべてのリージョン

グループを表示

アクション ▾

グループを作成

Canary を作成

Canary グループはまとめて表示される

名前	前回の実行ステータス	成功 %	アラーム	平均時間	状態	ランタイムバージョン	リージョン
DemoGroup	② 1/2 失敗	74%	⚠️ 1	18.2s	-	-	-
petsite	③ 成功	77%	⚠️ 1	17.9s	実行中	syn-nodejs-puppeteer-3.9	米国東部 (バージニア北部)
petsite-tokyo	④ 失敗	71%	-	18.6s	実行中	syn-nodejs-puppeteer-3.9	アジアパシフィック (東京)
petsite-heartbeat	③ 成功	74%	-	18.1s	実行中	syn-nodejs-puppeteer-3.9	米国東部 (バージニア北部)
petsite-python	③ 成功	-	-	-	実行中	syn-python-selenium-1.3	米国東部 (バージニア北部)
petsite-visu	③ 成功	63%	-	6.7s	実行中	syn-nodejs-puppeteer-3.9	米国東部 (バージニア北部)
petsite-visual	④ 失敗	0%	⚠️ 1	0ms	停止	syn-nodejs-puppeteer-3.9	米国東部 (バージニア北部)
test	③ 成功	0%	-	-	-	peteer-3.9	米国東部 (バージニア北部)
test-2-screenshotoff	④ 失敗	0%	-	-	-	peteer-3.9	米国東部 (バージニア北部)

前回のステータスが表示される

CloudWatch アラームを設定しており、条件にマッチしている場合は表示される

Canary 実行結果詳細 (可用性 - ステップ)

失敗している場合は、選択することで失敗にフォーカスしたログを閲覧可能

問題 (334)
過去 24 時間

Error: Failed to load url: http://servi-petsi-
March 27, 2023 15:41

Error: Failed to load url: http://servi-petsi-
March 27, 2023 15:39

Error: Failed to load url: http://servi-petsi-
March 27, 2023 15:26

Error: Failed to load url: http://servi-petsi-
March 27, 2023 15:20

Error: Failed to load url: http://servi-petsi-
March 27, 2023 15:17

Error: Failed to load url: http://servi-petsi-
March 27, 2023 15:13

Canary 実行

詳細については、Canary のトラブルシューティングドキュメントを参照してください。 詳細は[こちら](#)

Canary の実行履歴が時系列で表示される

各ポイントは Canary 実行を表します。詳細については、各データポイントをクリックしてください。

3 時間 ▾

● 成功 ● 失敗

ステップ | スクリーンショット | ログ | HAR ファイル | トレース

実行されたステップ (1)

失敗したステップのみ < 1 > ⏪

ステップ	ステップ名	ステータス	説明	送信先 URL	期間	スクリーンショット
1	east-1.elb.amazonaws.com	成功	OK	http://[REDACTED]	798 ミリ秒	1

各ステップごとに実行結果が表示される

▶ Canary アーティファクトと S3 の場所

Canary 実行結果詳細 (可用性 - スクリーンショット)

ステップ | **スクリーンショット** | ログ | HAR ファイル | トレス

スクリーンショット (1) ヘッダレスブラウザで取得した Web ページ のスクリーンショットを表示

失敗したステップのみ X

The screenshot shows a web page titled "Observability PET ADOPTIONS". At the top, there are images of various pets: two rabbits, a grey rabbit, a yellow puppy, a black puppy, a calico kitten, and an orange kitten. Below this is a banner with the text "CUTE LITTLE PETS SEEKING A HOME". The main content area displays three dogs for adoption:

- A black and tan dog (Dachshund) with a "Take me home" button below it.
- A black and white puppy (Siberian Husky) with a "Take me home" button below it.
- A brown dog (Cocker Spaniel) with a "Take me home" button below it.

Each dog listing includes the name, price (\$ 89 or \$ 99), and a small image of the dog's face below the main image.

At the bottom of the screenshot, there are three smaller images of dogs: a black and white puppy, a brown dog, and another brown dog.

On the right side of the screenshot, there is a status message: "☑ ステップに合格しました" followed by a link "http://[REDACTED].elb.am...".

Canary 実行結果詳細 (可用性 - ログ)

ログ

Canary の実行結果が出力される
CloudWatch Logs や S3に同じものが表示される

2023-03-14T05-21-34-476Z-log.txt

検索ログ

0/0

1 Start Canary

2 INFO: Event: {"canaryName": "petsite-heartbeat", "s3BaseFilePath": "cw-syn-results-[REDACTED]/canary/us-east-1/petsite-heartbe[REDACTED]"}
3 INFO: Context: {"callbackWaitsForEmptyEventLoop": true, "functionVersion": "1", "functionName": "cwsyn-petsite-heartbeat-[REDACTED]"}
4 INFO: Recording configuration:
5 INFO: Canary Name: petsite-heartbeat
6 INFO: Canary Arn: arn:aws:synthetics:[REDACTED]:canary:petsite-heartbeat
7 INFO: Canary lambda invoked at: Tue Mar 14 2023 05:21:34 GMT+0000 (Coordinated Universal Time)
8 INFO: AWS account Id: [REDACTED] and region us-east-1
9 INFO: S3 Artifact base location: cw-syn-results-[REDACTED]/canary/us-east-1/petsite-heartbeat-[REDACTED]
10 INFO: Artifacts will be encrypted using default KMS key for s3
11 INFO: Configuring tracing: canaryName: petsite-heartbeat canaryArn: arn:aws:synthetics:[REDACTED]:canary:petsite-heartbeat-[REDACTED]
12 INFO: Setting ActiveTracing to: true
13 INFO: memoryLimitInMB: 1000
14 INFO: awsRequestId: [REDACTED]
15 INFO: timeRemainingInMillis: 59998

Canary 実行結果詳細 (可用性 - HAR ファイル)

ステップ | スクリーンショット | ログ | **HAR ファイル** | トレス

Canary の実行した際に取得される HAR ファイルの内容が表示される
S3に同じものが表示される

Requests	Status code	Response size	Duration
http://servi-petsi-[REDACTED]-east-1.elb.amazonaws.com/	200 OK	118.9 KB	125.3ms
GET servi-petsi-[REDACTED]-east-1.elb.amazonaws.com/bootstrap.min.css	200 OK	152.1 KB	19.4ms
GET site.css	200 OK	1.3 KB	3.4ms
GET petstyles.css	200 OK	2.9 KB	24.1ms
GET brand.png	200 OK	3.9 KB	40.7ms
GET main_banner_text.png	200 OK	7.9 KB	78.7ms
GET p9.jpg?X-Amz-Security-Token=[REDACTED]	200 OK	88.2 KB	499.1ms
GET p10.jpg?X-Amz-Security-Token=[REDACTED]	200 OK	99 KB	537.3ms
GET p1.jpg?X-Amz-Security-Token=[REDACTED]	200 OK	78.3 KB	537.9ms
GET p11.jpg?X-Amz-Security-Token=[REDACTED]	200 OK	80 KB	537.5ms
GET p3.jpg?X-Amz-Security-Token=[REDACTED]	200 OK	78.3 KB	538.1ms
GET p6.jpg?X-Amz-Security-Token=[REDACTED]	200 OK	64.5 KB	538.3ms
GET jquery.min.js	200 OK	87.4 KB	62.4ms

Canary Blueprint

Canary の実体は Node.js/Python で実装される Lambda 関数で、6 種類の Blueprint が用意されており、基本的な用途であればコーディングが不要です。

No.	Blueprint	特徴	Node.js	Python
1	ハートビートのモニタリング	指定した URL にアクセスして、ページのスクリーンショットと HTTP アーカイブファイル (HAR ファイル) を保存	○	○
2	API Canary	REST API に対してリクエストを送信して、応答をテスト	○	○
3	リンク切れチェッカー	テスト対象のURL内のすべてのリンクを収集し、リンク切れがないかテスト	○	-
4	Canary レコーダー	Google Chrome の拡張機能である CloudWatch Synthetics Recorder を利用して、ユーザ操作を記録し、テスト	○	-
5	GUI ワークフロービルダー	Web サイト上のユーザ操作ができるかを GUI ベースで作成してテスト	○	○
6	ビジュアルモニタリング	Web サイトの表示が変化していないかをベースラインと比較し、テスト	○	-

Canary 設計図の使用

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html



Canary Blueprint - API Canary

The screenshot shows the configuration interface for a CloudFormation Canary Blueprint. It includes sections for API selection and an HTTP request step.

API 選択 情報
Amazon API Gateway API を使用中 [情報](#)
» Amazon API Gateway API を使用している場合は、このチェックボックスをオンにします。Canary には、Swagger テンプレートのアップロードなど、API Gateway に固有の追加オプションがあります。

Amazon API Gateway API を使用しています **チェックを入れることで、API Gateway の API を選択可能**

API を選択 情報
API とステージを選択するか、Swagger からテンプレートをアップロードできます。

API Gateway から API とステージを選択
 API Gateway Swagger テンプレートを使用

API
テストする API Gateway API
translate-api

同一 AWS アカウント、同一リージョンの API Gateway の API を選択可能

ステージ
API がデプロイされるステージ
dev

API Gateway で設定しているステージを選択する

HTTP リクエスト (1)
HTTP リクエストをステップとして追加します。順序を変更するには、ドラッグアンドドロップします。

HTTP リクエストを追加

呼び出しの順序	ステップ名	リソース	方法
1	検証 https://[REDACTED].execute-api.[REDACTED].amazonaws.com/translate	/translate	GET

- REST API に対して、リクエストを送信して、レスポンスが意図通りかどうかをテストできます。
- Amazon API Gateway と統合されており、API Gateway の API や ステージを Canary と同じ AWS アカウント、同じリージョンから選択したり、API Gateway から Swagger テンプレートをアップロードすることで、クロスアカウントやクロスリージョン API のモニタリングの設定が可能です。
- REST API の URL を直接指定することも可能です。

Canary Blueprint - API Canary HTTP リクエスト設定 1

HTTP リクエストの詳細
HTTP リクエストの詳細を入力します。この Canary に複数のリクエストを追加できます。

リソース
API Gateway 内に設定しているリソースパスを選択する
/translate

方法
API Gateway の対象リソースパスに設定されている HTTP リクエスト
メソッドを選択する
GET

URL クエリ文字列
クエリパラメータを入力
名前 値
ヘッダー 値
新しい文字列を追加 オプションのクエリ文字列を表示

ヘッダー
API リクエストとレスポンスに関連付けられたメタデータ
名前 値
ヘッダー 値
ヘッダーを追加 オプションのヘッダーを表示

データをリクエスト - オプション
必要に応じて、リクエストボディを指定する
1

HTTP リクエストとして下記を設定してテストが可能です。

- メソッド (GET/POST のみサポート)
- リソース
- URL クエリ文字列
- HTTP ヘッダー
- リクエストボディ

Canary Blueprint - API Canary HTTP リクエスト設定 2

レポート設定

ヘッダーとレスポンス本文をキャプチャ。 **Canary のヘッダーとレスポンスをキャプチャするか選択する**
ヘッダーとリクエスト/レスポンス本文に機密データが含まれている場合があります。これらの詳細は取得または保存したり、Canary 実行レポートに表示されたりしません。このデータをキャプチャすることを選択した場合、サービスはこの情報をログに記録または保存しないことに注意してください。データは S3 バケットにのみ保存されます。

ステップ名
後で Canary ステップを追跡できるように、Canary ステップに名前を付けます。

**複数ステップを定義する場合に、追跡しやすいように
Canary ステップに名前をつける**

ステップ失敗時に Canary 実行を続行する
このステップが失敗した場合は、他の Canary スクリプトの実行を続行します。

**複数ステップを定義する場合に、本ステップの後に設定されている
ステップを実行するか選択する**

別の呼び出しを保存して追加
追加の呼び出しにより、Canary の実行時間が長くなる可能性があります。

注意: レポート設定とステップは、ランタイムバージョン syn-nodejs-2.2 以降でのみサポートされます。

「syn-nodejs-2.2」以降のランタイムバージョンを利用してい る場合は、オプションとして、以下の機能が利用可能です。

- ヘッダーとレスポンス本文をキャプチャする
※機密情報が記録される可能性がある場合は取り扱いに注意して下さい。
- API を複数モニタリングする際に、複数ステップに分けて実行結果を表示する Canary の作成

※ 「syn-python-selenium」を利用することはできないため、ご注意下さい。

Canary 実行結果詳細 (可用性 - HTTP リクエスト)

HTTP ステップ | **HTTP リクエスト** | ログ | トレース

HTTP リクエスト (2) 情報
リクエスト/レスポンスのヘッダーと本文はスクリプト設定に基づいて利用できます

ステータスコードやリクエストにかかる時間が表示される

失敗したリクエストのみを表示 < 1 > ○

リクエスト	ステータスコード	説明	期間
https://east-1.amazonaws.com...	200	OK	1863 ミリ秒

GET https://[REDACTED]

リクエストの詳細

リクエストサイズ - 98 Bytes

リクエストヘッダー 98 Bytes

リクエスト本文 0 Bytes

レスポンスサイズ - 489 Bytes

レスポンスヘッダー 466 Bytes

レスポンス本文 23 Bytes

リクエスト詳細 (展開)

リクエストヘッダー

User-Agent : [REDACTED]
X-Amzn-Trace-Id : [REDACTED]

リクエスト本文

レスポンス詳細 (展開)

レスポンスヘッダー

content-type : application/json
content-length : 23
connection : close
date : Tue, 28 Mar 2023 14:10:46 GMT
x-amzn-requestid : [REDACTED]
x-amz-apigw-id : [REDACTED]
x-amzn-trace-id : [REDACTED] 25
x-cache : Miss from cloudfront
via : 1.1 [REDACTED] Front.net (CloudFront)
x-amz-cf-pop : IAD55-P4
x-amz-cf-id : [REDACTED] fMIIaWAWbn6_bQ==

レスポンス本文

ヘッダーとレスポンス本文をキャプチャすると
どのようなリクエストをした結果
どのようなレスポンスが返ってきたか
確認できる

aws

Canary Blueprint

Canary の実体は Node.js/Python で実装される Lambda 関数で、6 種類の Blueprint が用意されており、基本的な用途であればコーディングが不要です。

No.	Blueprint	特徴	Node.js	Python
1	ハートビートのモニタリング	指定した URL にアクセスして、ページのスクリーンショットと HTTP アーカイブファイル (HAR ファイル) を保存	○	○
2	API Canary	REST API に対してリクエストを送信して、応答をテスト	○	○
3	リンク切れチェッカー	テスト対象のURL内のすべてのリンクを収集し、リンク切れがないかテスト	○	-
4	Canary レコーダー	Google Chrome の拡張機能である CloudWatch Synthetics Recorder を利用して、ユーザ操作を記録し、テスト	○	-
5	GUI ワークフロービルダー	Web サイト上のユーザ操作ができるかを GUI ベースで作成してテスト	○	○
6	ビジュアルモニタリング	Web サイトの表示が変化していないかをベースラインと比較し、テスト	○	-

Canary 設計図の使用

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html



Canary Blueprint - リンク切れチェッカー

設計図

- hardt-beat のモニタリング
- API Canary API を HTTP ステップとしてモニタリングします。
- リンク切れチェッカー 指定された URL で基本的なウェブクローラーを実行します。 (選択)
- Canary レコーダー AWS Canary Recorder ブラグインを使用します。
- GUI ワークフロービルダー 実行するアクションを含む GUI ワークフローを作成します。
- ビジュアルモニタリング 実行ごとに視覚的な変更をモニタリングする

Canary ビルダー

名前: synthetics-link-demo

名前は、21 文字までの小文字、数字、ハイフン、またはアンダースコアで構成され、スペースを含めることはできません。

アプリケーションまたはエンドポイント URL 情報
http://[REDACTED].st-1.elb.amazonaws.com

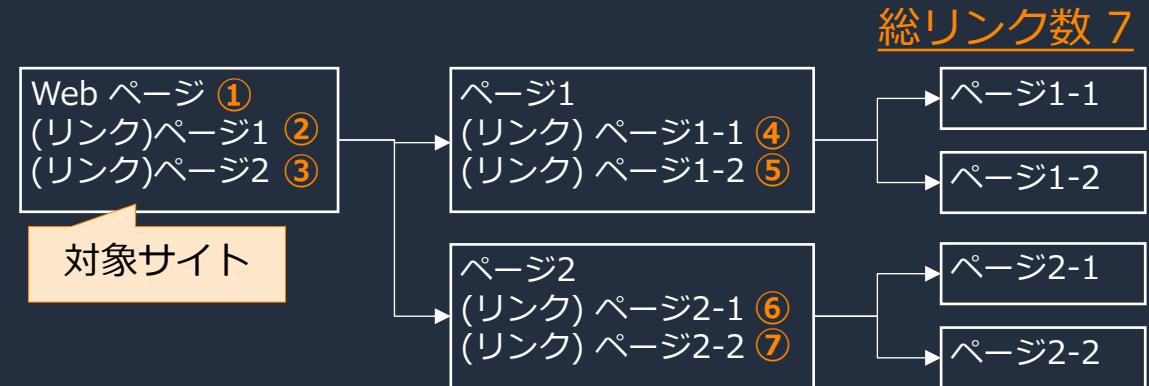
テストするエンドポイント、API、または URL を入力します。

フォローされるリンクの最大数: 5 (選択)

モニタリングするリンクの数を指定する

スクリーンショット 情報
 スクリーンショットを撮る
スクリーンショットは、Canary 実行ごとに Canary の詳細画面に表示されます

- 指定した URL にアクセスして、Web ページに設定されているすべてのリンクを収集し、指定したリンク数までテストを実行します。リンク数のカウントは、指定した URL 自体が最初のリンクとしてカウントされることに注意してください。



- リンクに異常があれば、エラーと判定されます。リンク切れチェッカーで検出できるエラーの種類は下記を参照して下さい。
 - リンク切れチェッカー
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html#CloudWatch_Synthetics_Canaries_Blueprints_Broken_Links

Canary 実行結果詳細 (可用性 - チェックしたリンク)

チェックしたリンク | スクリーンショット | ログ | HAR ファイル

リンク (4) 指定した URL 内にあるリンクをチェックした結果がリンク毎に表示される

壊れたリンクのみ < 1 > ⏪

送信先 URL	ステータスコード	説明	アンカーテキスト	スクリーンショット
http://[REDACTED] [REDACTED]	200	OK	-	2
http://[REDACTED] [REDACTED]	200	OK	See Adoption List	2
http://[REDACTED] [REDACTED]	200	OK	Perform Housekeeping	2

Canary Blueprint

Canary の実体は Node.js/Python で実装される Lambda 関数で、6 種類の Blueprint が用意されており、基本的な用途であればコーディングが不要です。

No.	Blueprint	特徴	Node.js	Python
1	ハートビートのモニタリング	指定した URL にアクセスして、ページのスクリーンショットと HTTP アーカイブファイル (HAR ファイル) を保存	○	○
2	API Canary	REST API に対してリクエストを送信して、応答をテスト	○	○
3	リンク切れチェッカー	テスト対象のURL内のすべてのリンクを収集し、リンク切れがないかテスト	○	-
4	Canary レコーダー	Google Chrome の拡張機能である CloudWatch Synthetics Recorder を利用して、ユーザ操作を記録し、テスト	○	-
5	GUI ワークフロービルダー	Web サイト上のユーザ操作ができるかを GUI ベースで作成してテスト	○	○
6	ビジュアルモニタリング	Web サイトの表示が変化していないかをベースラインと比較し、テスト	○	-

Canary 設計図の使用

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html



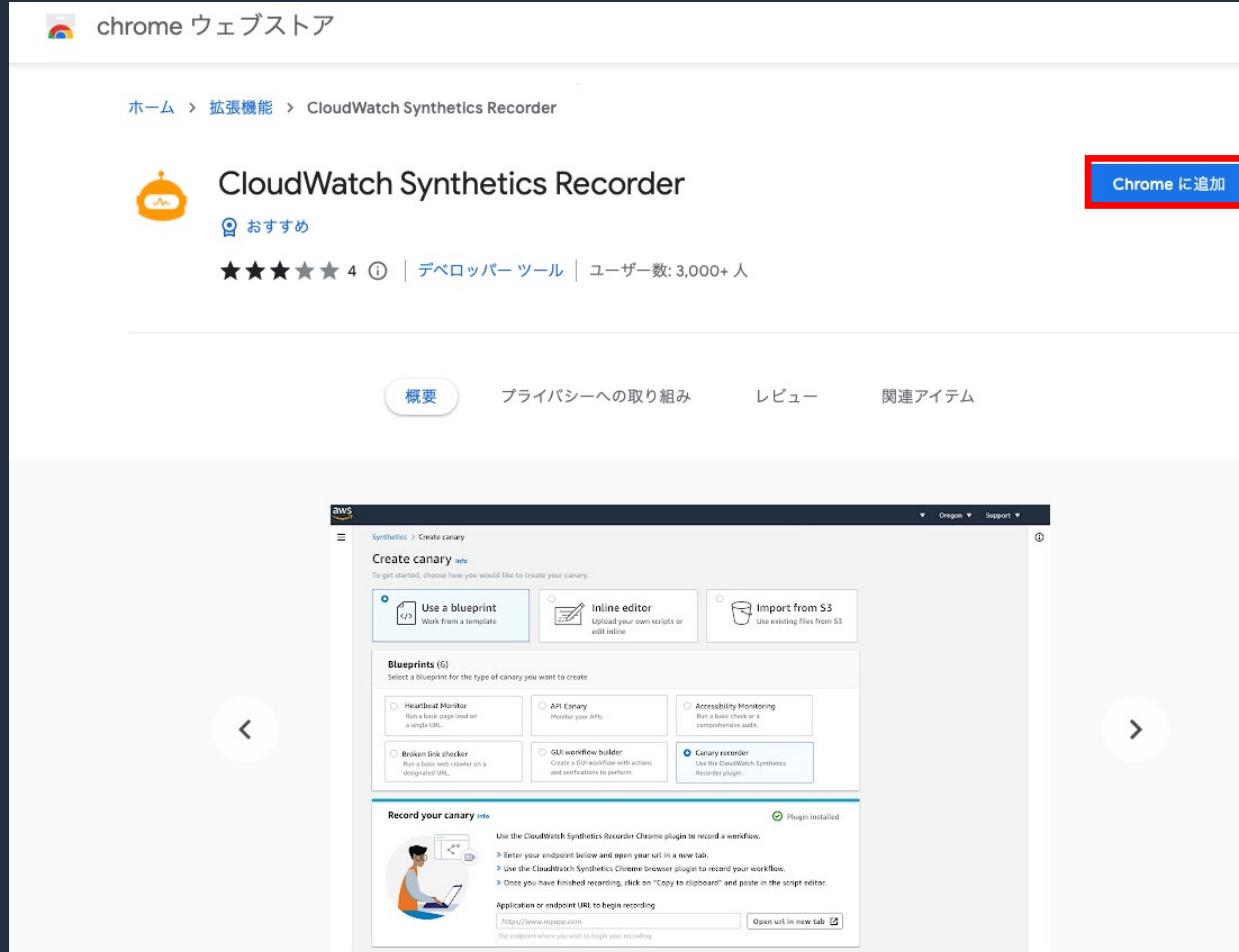
Canary Blueprint - Canary レコーダー

以下流れで記録したユーザ操作を Canary で再現し、モニタリングすることが可能。

- ① Google Chrome の拡張機能である CloudWatch Synthetics Recorder をChrome に追加
- ② CloudWatch Synthetics Recorder を利用して、Web サイトでの操作を記録して、Node.js のコードとして出力
- ③ 出力したコードをコピーし Canary 設定画面にペーストして、登録

Canary Blueprint - Canary レコーダー

① CloudWatch Synthetics Recorder を Chromeに追加

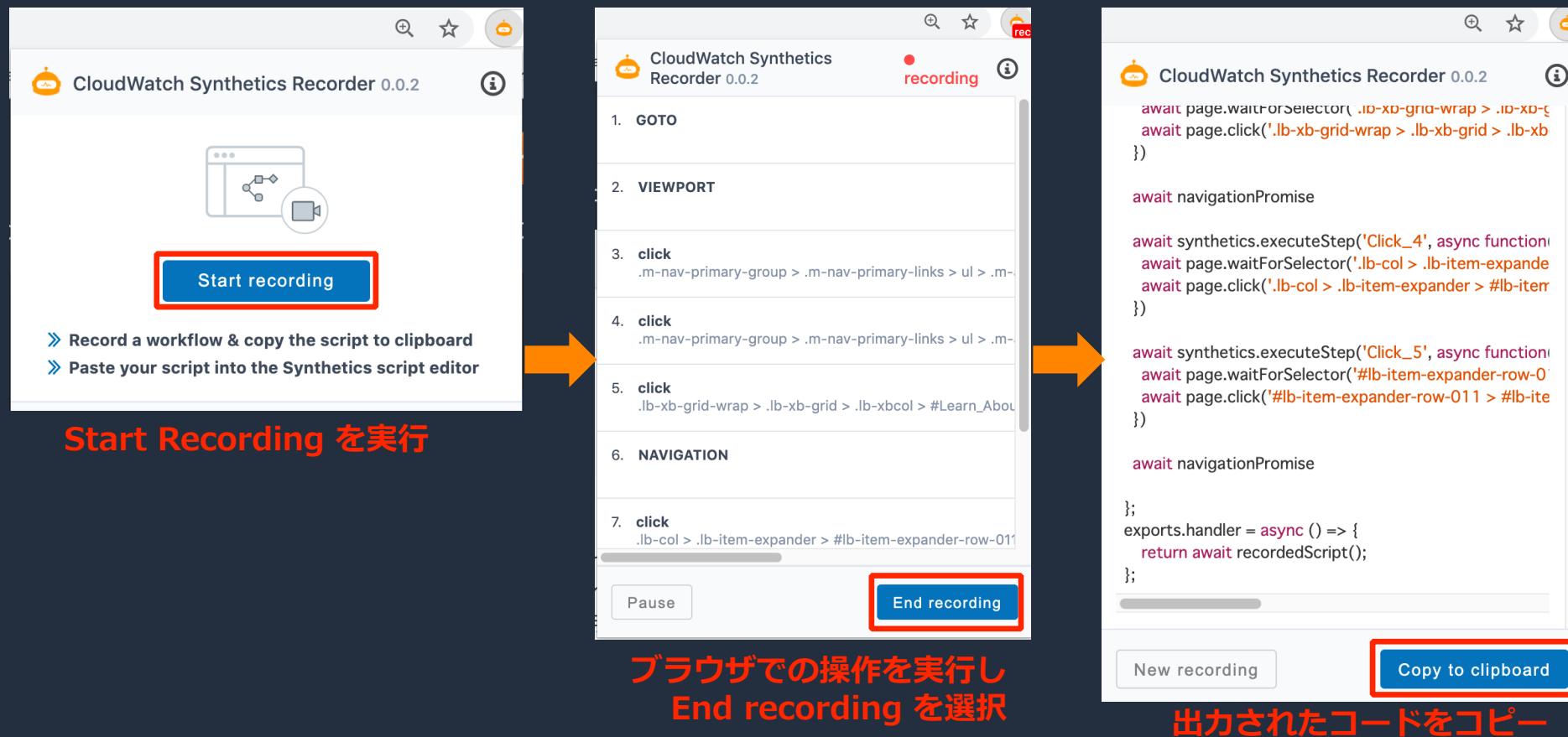


Google Chrome の拡張機能である
CloudWatch Synthetics Recorder を以下の
リンクより Google Chrome に追加する。

<https://chrome.google.com/webstore/detail/cloudwatch-synthetics-rec/bhdnlmmgiplmbcdmkkdfplenecepfno>

Canary Blueprint - Canary レコーダー

② CloudWatch Synthetics Recorder でユーザ操作を記録



Canary Blueprint - Canary レコーダー

③ 出力したコードをコピーし Canary 作成

設計図

Canary レコーダー AWS Canary Recorder プラグインを使用します。

Canary を記録 情報

AWS Synthetics Canary Recorder Chrome プラグインを使用してワークフローを記録します。

以下にエンドポイントを入力し、新しいタブで URL を開きます

CloudWatch Synthetics Chrome ブラウザプラグインを使用してワークフローを記録します

記録が完了したら、[クリップボードにコピー] をクリックし、以下のスクリプトエディタにスクリプトを貼り付けます。

レコーダーをスタートした URL を入力

http://

URL を新しいタブで開く

記録を開始するエンドポイント。

Canary ビルダー

名前

synthetics-recorddemo

名前は、21 文字までの小文字、数字、ハイphen、またはアンダースコアで構成され、スペースを含めることはできません。

スクリプトエディタ

スクリプトをアップロード 元に戻す エディタをクリア

ランタイムバージョン 情報

syn-nodejs-puppeteer-3.9

この Canary を実行する Synthetics ランタイムバージョンを選択します。

先ほどコピーしたコードを貼り付けて Canary 作成

```
1 var synthetics = require('Synthetics');
2 const log = require('SyntheticsLogger');
3
4 const recordedScript = async function () {
5   let page = await synthetics.getPage();
6
7   const navigationPromise = page.waitForNavigation();
8
9   await synthetics.executeStep('Goto_0', async function() {
10     await page.goto("http://servi-petsi-t462mqtv152v-607366110.us-east-1.elb.amazonaws.com/?select");
11   });
12
13   await page.setViewport({ width: 1920, height: 944 });
14 }
```

Canary 実行結果詳細 (可用性 - ステップ) Canary レコーダー

ステップ | スクリーンショット | ログ | HAR ファイル | トレス

実行されたステップ (5) Canary レコーダーで記録した操作のステップ毎に結果が表示される

失敗したステップのみ < 1 > ⚙️

ステップ	ステップ名	ステータス	説明	送信先 URL	期間	スクリーンショット
1	Goto_0	✓ 成功	OK	http://[REDACTED]	792 ミリ秒	2
2	Click_1	✓ 成功	OK	http://[REDACTED]	61 ミリ秒	2
3	Select_2	✓ 成功	OK	http://[REDACTED]	12 ミリ秒	2
4	Click_3	✓ 成功	OK	http://[REDACTED]	38 ミリ秒	2
5	Click_4	✗ 失敗	TimeoutError: waiting for...	http://[REDACTED]	30001 ミリ秒	2

Canary Blueprint

Canary の実体は Node.js/Python で実装される Lambda 関数で、6 種類の Blueprint が用意されており、基本的な用途であればコーディングが不要です。

No.	Blueprint	特徴	Node.js	Python
1	ハートビートのモニタリング	指定した URL にアクセスして、ページのスクリーンショットと HTTP アーカイブファイル (HAR ファイル) を保存	○	○
2	API Canary	REST API に対してリクエストを送信して、応答をテスト	○	○
3	リンク切れチェッカー	テスト対象のURL内のすべてのリンクを収集し、リンク切れがないかテスト	○	-
4	Canary レコーダー	Google Chrome の拡張機能である CloudWatch Synthetics Recorder を利用して、ユーザ操作を記録し、テスト	○	-
5	GUI ワークフロービルダー	Web サイト上のユーザ操作ができるかを GUI ベースで作成してテスト	○	○
6	ビジュアルモニタリング	Web サイトの表示が変化していないかをベースラインと比較し、テスト	○	-

Canary 設計図の使用

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html

Canary Blueprint - GUI ワークフロービルダー

The screenshot shows the 'Design' tab of the 'Canary Blueprint' interface. It lists several actions:

- ハートビートのモニタリング
- API Canary
- リンク切れチェッカー
- Canary レコーダー
- GUI ワークフロービルダー (highlighted with a red box)
- ビジュアルモニタリング

The 'GUI ワークフロービルダー' action is selected, indicated by a blue dot next to its name.

The 'Canary Builder' page displays the configuration for the 'synthetics-gui-demo' blueprint. It includes fields for:

- 名前: synthetics-gui-demo
- アプリケーションまたはエンドポイント URL 情報: http://servi-... .elb.amazonaws.com/

Below these, the 'ワークフロービルダー' section is shown, with the heading 'ユーザ操作を GUI で定義する' (Define user operations via GUI). A red box highlights the 'アクション' dropdown set to 'クリック' (Click), the 'セレクター' input field containing '[id='searchpets']', and the 'テキスト' input field.

- Web ページに対して、ユーザの一連の操作を定義して、意図通りに操作できるかをテストすることができます。
- Web ページ上のアクションを GUI で定義していくことで、ユーザ操作がコードに反映されます。

Canary Blueprint - GUI ワークフロービルダー

定義できるアクション



- 定義できるアクションは、以下5つになります。

No.	アクション	アクション説明
1	クリック	ページ内の要素を指定して、ユーザによる要素のクリックをシミュレーション
2	セレクターを確認	指定した要素が Web ページ上に存在するか検証
3	テキストを確認	指定した文字列が指定した要素に含まれているかを検証
4	テキストを入力	指定したテキストを指定した要素に書き込む
5	ナビゲーションでクリック	指定した要素を選択した後で、ページ全体が読み込まれるまで待つ

- セレクターの定義は選択する言語 (Node.js/Python) で異なります。詳細は下記をご確認ください。
 - GUIワークフロービルダー
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html#CloudWatch_Synthetics_Canaries_Blueprint_GUI_Workflow

Canary Blueprint

Canary の実体は Node.js/Python で実装される Lambda 関数で、6 種類の Blueprint が用意されており、基本的な用途であればコーディングが不要です。

No.	Blueprint	特徴	Node.js	Python
1	ハートビートのモニタリング	指定した URL にアクセスして、ページのスクリーンショットと HTTP アーカイブファイル (HAR ファイル) を保存	○	○
2	API Canary	REST API に対してリクエストを送信して、応答をテスト	○	○
3	リンク切れチェッカー	テスト対象のURL内のすべてのリンクを収集し、リンク切れがないかテスト	○	-
4	Canary レコーダー	Google Chrome の拡張機能である CloudWatch Synthetics Recorder を利用して、ユーザ操作を記録し、テスト	○	-
5	GUI ワークフロービルダー	Web サイト上のユーザ操作ができるかを GUI ベースで作成してテスト	○	○
6	ビジュアルモニタリング	Web サイトの表示が変化していないかをベースラインと比較し、テスト	○	-

Canary 設計図の使用

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_Blueprints.html

Canary Blueprint - ビジュアルモニタリング

設計図

- ハートビートのモニタリング
- API Canary
- リンク切れチェッカー
- Canary レコーダー
- GUI ワークフロービルダー
- ビジュアルモニタリング

① ベースラインのスクリーンショットからエリアを除外したり、視覚的な差異のしきい値を設定したりするなど、
視覚的な Canary を設定する方法についての詳細をご覧ください。 詳細は[こちら](#)

Canary ビルダー

名前
synthetics-visu-demo

名前は、21 文字までの小文字、数字、ハイフン、またはアンダースコアで構成され、スペースを含めることはできません。

アプリケーションまたはエンドポイント URL 情報
.amazonaws.com 削除

エンドポイントを追加

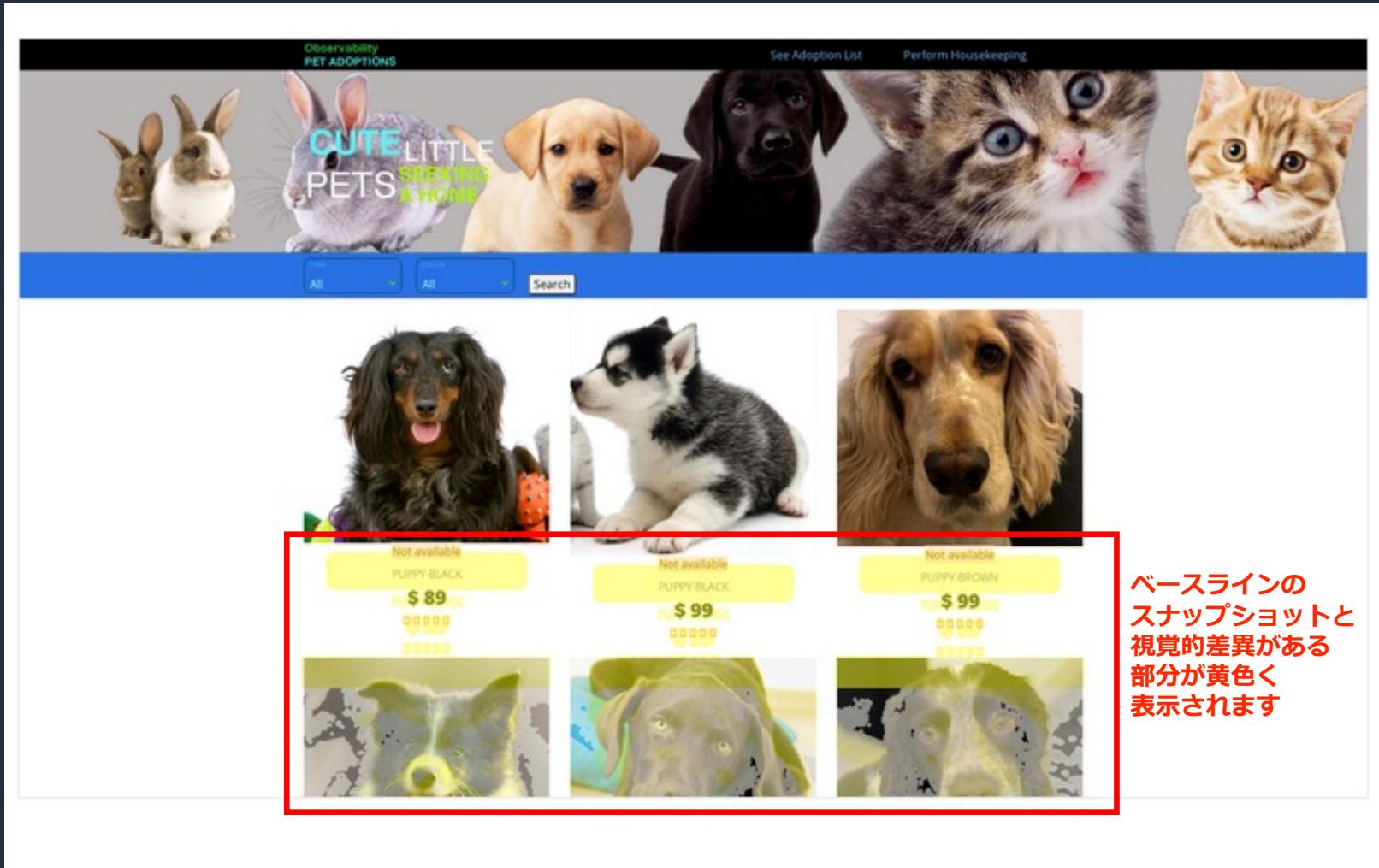
最大でさらに 4 個のエンドポイントを追加できます。スクリプトを変更することで、さらにエンドポイントを追加できます。

しきい値 情報
視覚的な差異が以下より大きい場合 0 % が Canary で失敗

aws Web Services, Inc. or its affiliates.

- 以下流れで、**指定した URL の Web ページの表示に異常な変化がないか**をチェックします。
 - 最初に正常実行した Canary のスクリーンショットをベースラインとして使用
 - その後の Canary 実行で取得したスクリーンショットとベースラインのスクリーンショットを比較して、**定めたしきい値を超えて**いる場合は、**エラー**として検知
- 「syn-nodejs-puppeteer-3.2」以降のランタイムバージョンを使用している場合に利用が可能な機能になります。
※ 「syn-python-selenium」では利用することはできないため、ご注意下さい。

Canary Blueprint - ビジュアルモニタリング 結果の確認



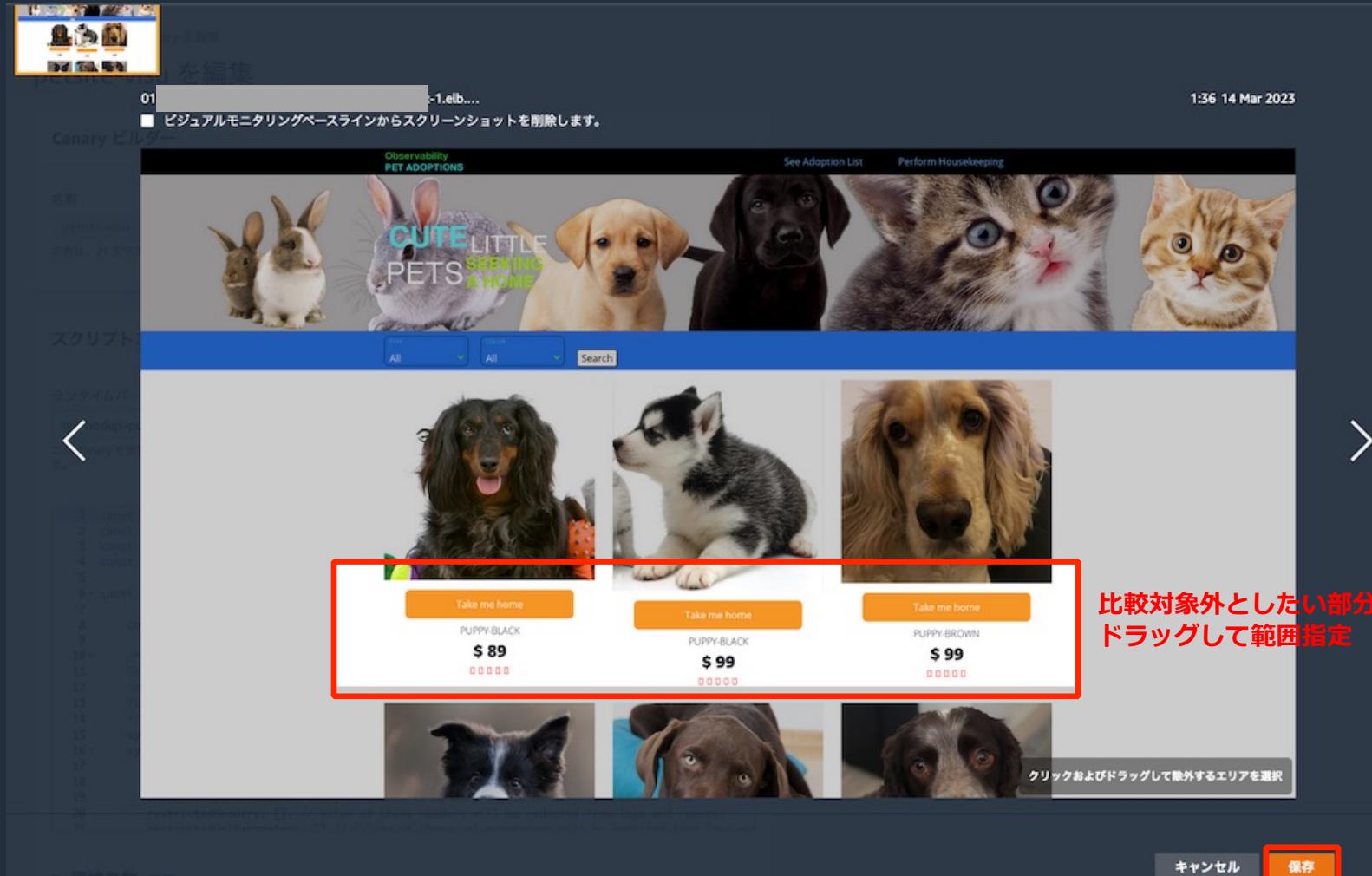
Canary の [可用性] タブの [スクリーンショット] タブを選択することで、視覚的差異が Web ページのどの部分で発生したかが、黄色く表示され、確認できます。

Canary Blueprint - ビジュアルモニタリング ベースライン再設定



- Web ページの変更等でベースラインを再設定したい場合、変更したい Canary の編集画面のビジュアルモニタリングの設定項目から新しくベースラインの設定をすることができます。
- ベースラインの比較対象から、一部を比較対象外としたい場合は、「ベースラインを編集」をクリックして除外する部分を設定することができます。

Canary Blueprint - ビジュアルモニタリング スクリーンショット比較対象領域除外設定



ベースラインのスクリーンショットから、比較対象外としたい箇所をドラッグして、範囲指定することで、指定した領域は比較対象外とすることが可能です。

アジェンダ

Amazon CloudWatch Synthetics とは

Amazon CloudWatch Synthetics Canaryとは

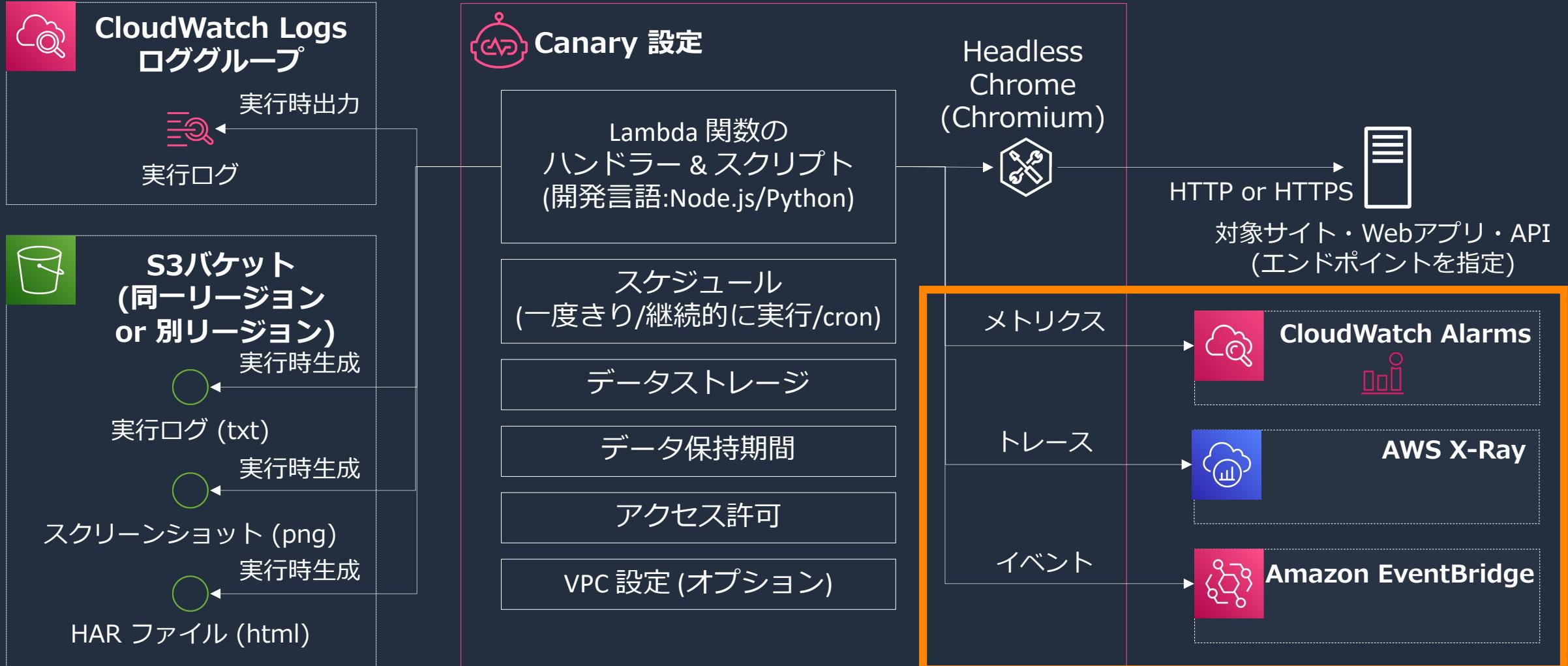
Amazon CloudWatch Synthetics による Synthetics Monitoring

AWS モニタリングサービスとの連携

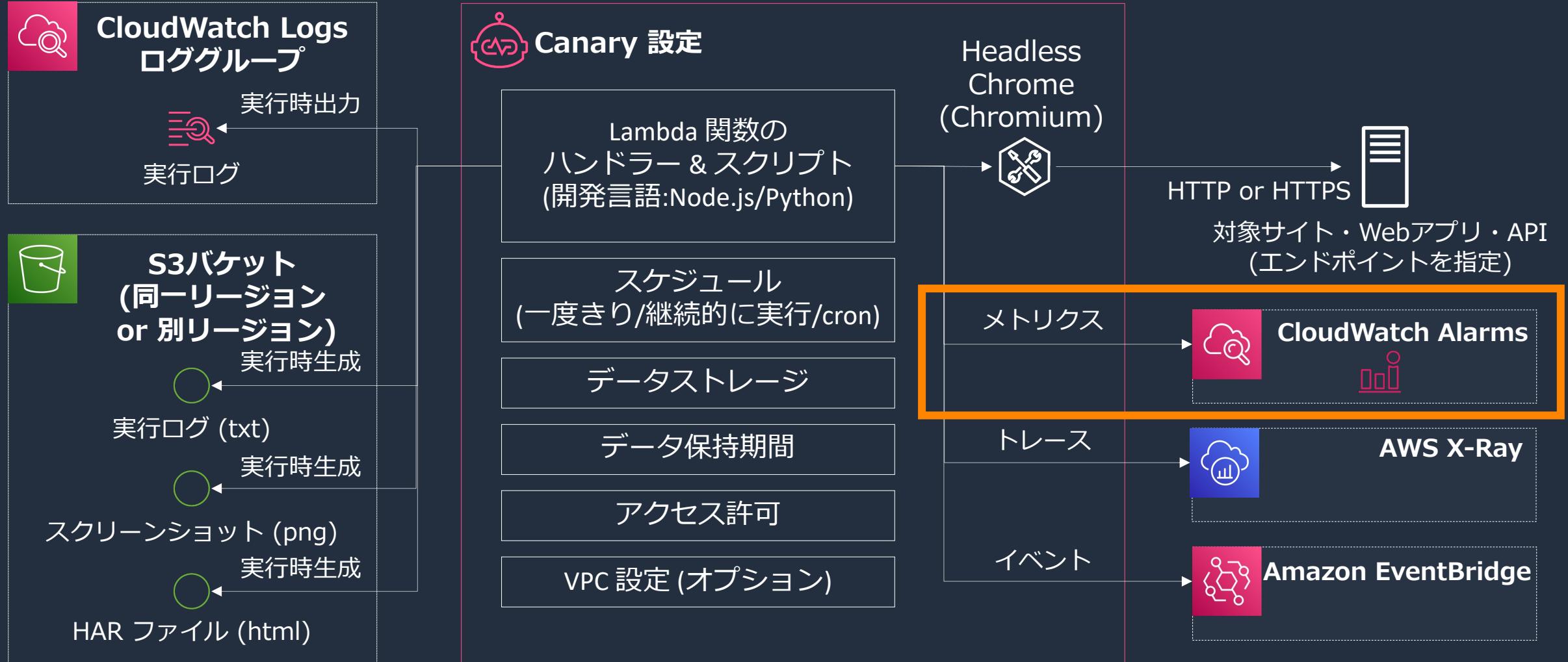
Amazon CloudWatch Synthetics 料金

まとめ

Amazon CloudWatch Synthetics Canary 全体構成イメージ



Amazon CloudWatch Synthetics Canary 全体構成イメージ



Canary モニタリングサービスの連携 – CloudWatch Alarms

The screenshot shows the 'CloudWatch アラーム - オプション' (CloudWatch Alarms - Options) section of the AWS CloudWatch Metrics console. It displays a table of alarms and a configuration panel for setting up notifications.

CloudWatch アラーム - オプション 情報
Synthetics で Canary のアラームを自動的に作成し、後でカスタマイズできます。

メトリクス名	アラームの状態	しきい値	期間	操作
SuccessPer...	より低い	90 %	5 分	削除

新しいアラームを追加

この Canary の通知を設定する
この Canary が定義したレベルに達したときに通知を受け取る場所を選択する

次の SNS トピックに通知を送信します。 [情報](#)
通知を受け取る SNS (Simple Notification Service) トピックを定義する

既存の SNS トピックの選択
 新しいトピックの作成

新しいトピックを作成...
トピック名は一意である必要があります。新しいトピックには Synthetics がプレフィックスとして付けられます。

Synthetics-petsit

SNS トピック名に使用できるのは、英数字、ハイフン (-)、アンダースコアのみです。

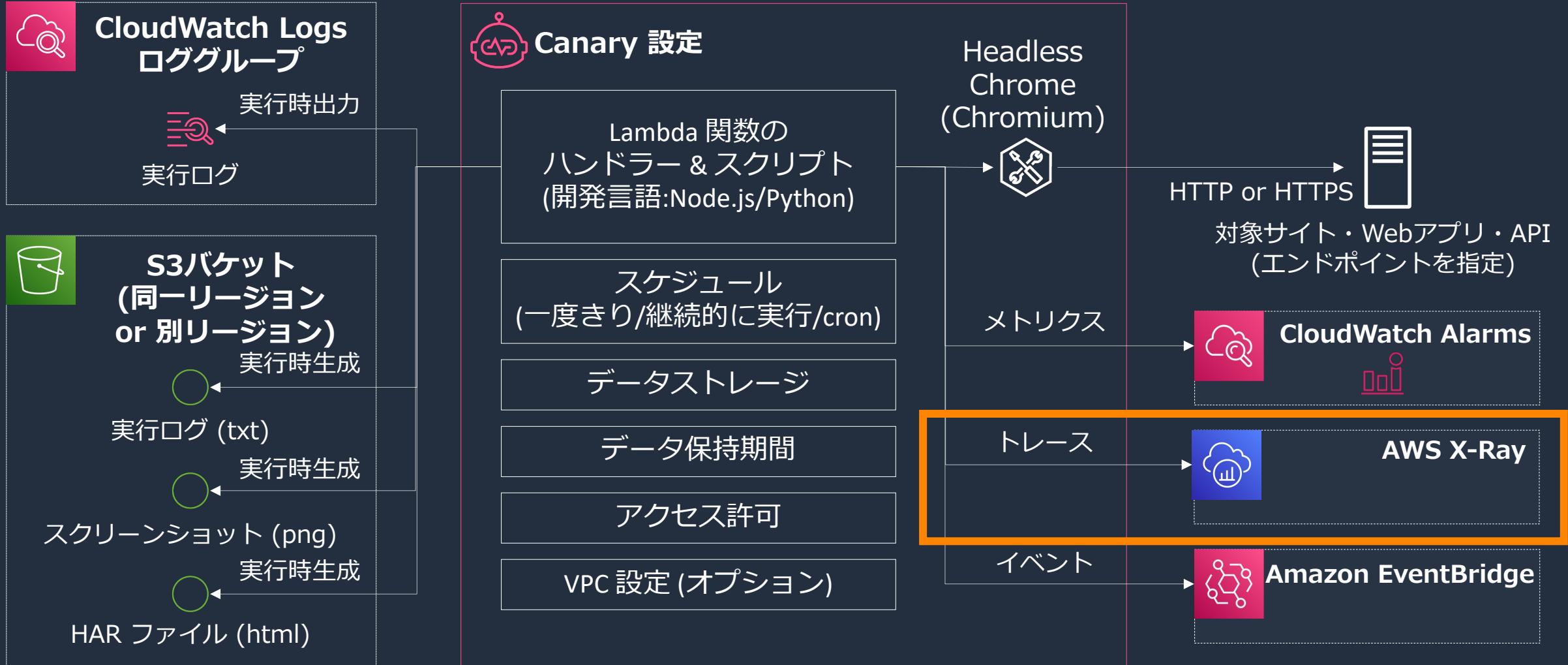
通知を受け取る E メールエンドポイント...
E メールアドレスのカンマ区切りのリストを追加します。各アドレスは、上記のトピックへのサブスクリプションとして追加されます。

xxxxxx@amazon.co.jp
user1@example.com, user2@example.com

トピックの作成

- Canary 作成時、Canary 設定変更時、CloudWatch アラームの設定画面から Synthetics で取得しているメトリクスに対して、アラームを作成することができます。
- CloudWatch Synthetics 設定画面からは通知先としてSNS トピックを設定することも可能です。通知以外の処理を行いたい場合は、CloudWatch アラームの設定画面から設定をして下さい。
- CloudWatch Synthetics で取得している各種メトリクスについては、下記を参照して下さい。
 - Canaryによって発行されるCloudWatchメトリクス https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_Synthetics_Canaries_metrics.html

Amazon CloudWatch Synthetics Canary 全体構成イメージ



Canary モニタリングサービスの連携 – AWS X-Ray

▼ アクティブトレース - オプション 情報

AWS X-Ray でアクティブなトレースを有効にして、トラブルシューティングを行い、解決までの平均時間を短縮します。

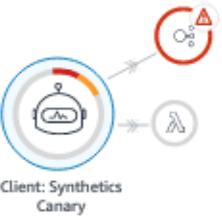
AWS X-Ray でサービスを追跡します。

AWS X-Ray と Synthetics は、分析とアバッテグを行って、進行中の障害の根本原因を見つけ、パフォーマンスのボトルネックや傾向を特定し、レイテンシー率を比較して、API と URL に十分な Canary カバレッジがあるかどうかを確認するのに役立ちます。

[詳細ははこちら](#)

その他の利点

- ✓ AWS X-Ray および CloudWatch ServiceLens サービスマップで Canary を表示します。
- ✓ 各 Canary 実行のトレースとセグメントを表示します。
- ✓ AWS X-Ray 分析を使用して傾向を表示します。


Client: Synthetics Canary

- 「syn-nodejs-2.0」以降のランタイムを使用するCanaryでは、Canary 新規作成時、または、Canary 設定変更時に、アクティブトレース設定画面で、「AWS X-Ray でサービスを追跡します」のチェックボックスにチェックを入れることで、Canary 実行におけるトレース情報を取得することができます。

※ 「syn-python-selenium」を利用する場合は本機能を利用することができないため、ご注意下さい。

Canary 実行結果詳細 (トレース)

ステップ | スクリーンショット | ログ | HAR ファイル | トレース

調査したいトレース ID を選択して、クリックすると該当のトレースマップが表示される

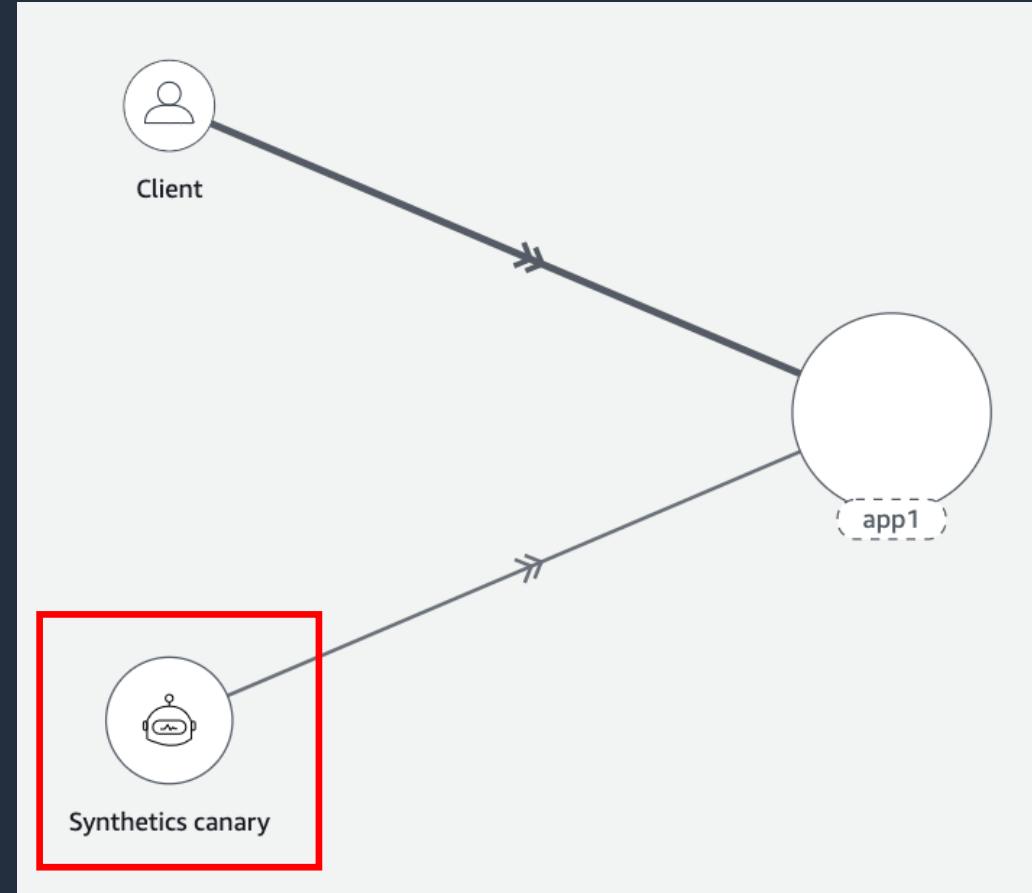
トレース (45) エラーのみを表示 トレースマップに移動

1 2 3 4 5 6 7 8

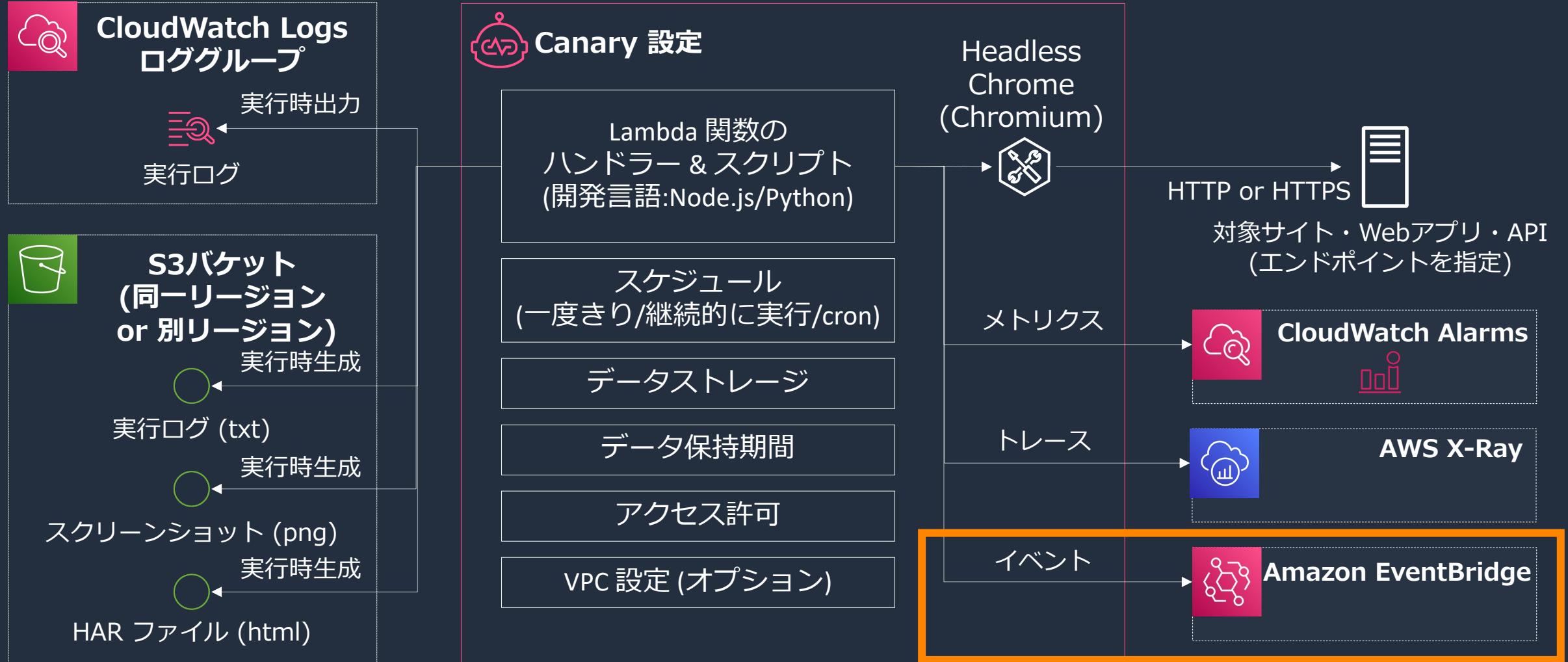
トレース ID	トレースステータス	レスポンスコード	応答時間	URL アドレス	HTTP メソッド
1-6410085b-375a02b4a69f36ef6d72c054	OK	-	0.96秒	-	-
1-6410085b-39287ee394447d9c10f65f2a	OK	200	0.13秒	http://servi-petsi-east-1.elb.amazonaws.com/	GET
1-6410085b-45fd38ae0c23e1c31a17a58	OK	200	0.12秒	http://servi-petsi-east-1.elb.amazonaws.com/css/site.css	GET
1-6410085b-4e113255d272fc138c451625	OK	-	0.90秒	-	-
1-6410085b-596def403747c177ce9cf64c	OK	-	1.12秒	-	-
1-6410085b-5cf9df818aa2210a9093457f	OK	-	0.06秒	-	-

Canary モニタリングサービスの連携 – X-Ray トレースマップ

Canary のトレースは通常のクライアントとは別のアイコンでトレースマップに表示されます。



Amazon CloudWatch Synthetics Canary 全体構成イメージ



Canary モニタリングサービスの連携 – Amazon EventBridge



Event Bridge

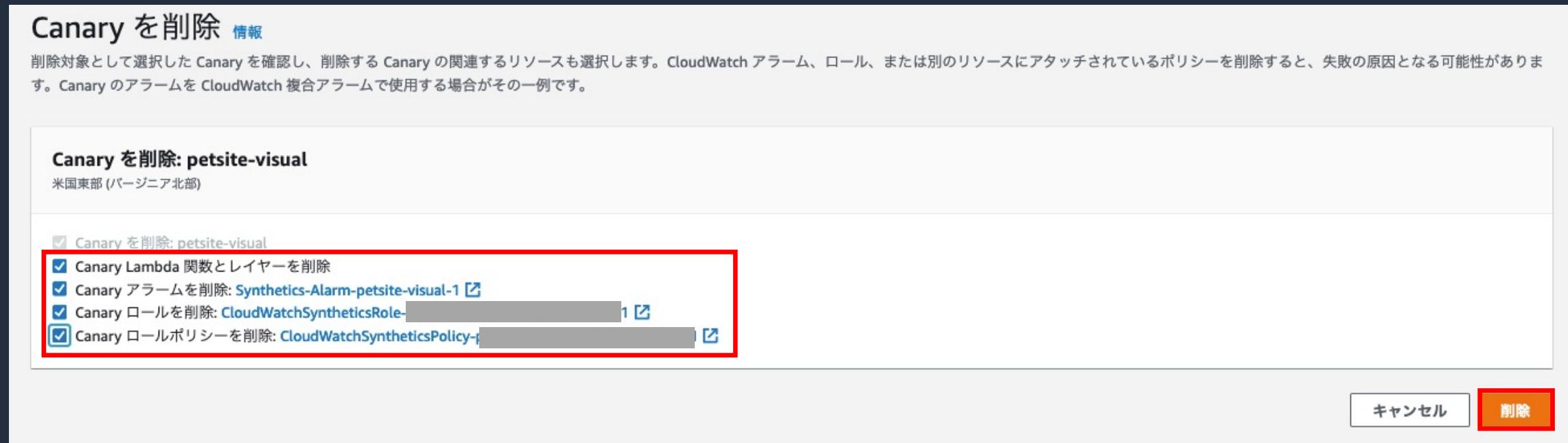
- Canary State Changes
- Canary Testrun Success
- Canary Testrun Failure



CloudWatch Synthetics は Canary の設定を変更したり、Canary の実行が完了した場合にイベントを出力します。Amazon EventBridge のイベントルールを設定することで、特定のイベントパターンをトリガーに通知を行ったり、是正処置を行うことができます。例えば以下のようなことに利用可能です。

- Canary のライフサイクルの追跡 (Canary の設定を変更した際に通知を上げたい)
→ Canary State Changes のイベント発行を機に SNS で通知する
- Canary 実行が失敗した場合の調査 (ワークフローの一部として利用)
→ Canary Testrun Failure のイベント発行を機に ワークフローを動かす

Canary の削除について



- Canary を削除する場合は、Canary を一度停止してから行います。
- Canary を作成時に作成されたリソース (Lambda 関数/IAM ロール・ポリシー /CloudWatch アラーム) も併せて削除することができます。
削除したいリソースを選択の上、削除して下さい。

アジェンダ

Amazon CloudWatch Synthetics とは

Amazon CloudWatch Synthetics Canaryとは

Amazon CloudWatch Synthetics による Synthetics Monitoring

AWS モニタリングサービスとの連携

Amazon CloudWatch Synthetics 料金

まとめ

Amazon CloudWatch Synthetics 料金 (2023/3現在)

① CloudWatch Synthetics 利用料

1回の Canary 実行あたり 0.0019 USD (※東京リージョン)

毎月 100 回 までの無料利用枠あり

月あたりに実行される Canary 実行回数に対しての利用料

<https://aws.amazon.com/jp/cloudwatch/pricing/>

② 関連サービス利用料

1. Canary としての Lambda 関数の実行にかかる利用料
2. S3 ストレージ利用料 (結果データやスクリプトの保存に伴う利用料)
3. CloudWatch Alarms, CloudWatch Logs, X-Ray 等の利用料

アジェンダ

Amazon CloudWatch Synthetics とは

Amazon CloudWatch Synthetics Canaryとは

Amazon CloudWatch Synthetics で実現できる外形監視

AWS モニタリングサービスとの連携

料金・注意点

まとめ

まとめ

Amazon CloudWatch Synthetics を利用すれば

- 必要な時に素早く Synthetics Monitoring を実現
- モニタリング動作の振る舞いをプログラムで制御可能
 - シンプルな用途であれば、組み込み提供のテンプレート (Blueprint) を用いるだけでプログラミングは不要
 - カスタマイズにより複雑な操作も表現可能
- AWS サービスとの連携による高い柔軟性
 - S3, EventBridge, X-Ray などの AWS サービスと連携

本資料に関するお問い合わせ・ご感想

技術的な内容に関しましては、有料のAWSサポート窓口へ
お問い合わせください

<https://aws.amazon.com/jp/premiumsupport/>

料金面でのお問い合わせに関しましては、カスタマーサポート窓口へ
お問い合わせください（マネジメントコンソールへのログインが必要です）

<https://console.aws.amazon.com/support/home#/case/create?issueType=customer-service>

具体的な案件に対する構成相談は、後述する個別相談会をご活用ください



ご感想はTwitterへ！ハッシュタグは以下をご利用ください
#awsblackbelt



その他コンテンツのご紹介

ウェビナーなど、AWSのイベントスケジュールをご参照いただけます

<https://aws.amazon.com/jp/events/>

ハンズオンコンテンツ

<https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-hands-on/>

AWS 個別相談会

AWSのソリューションアーキテクトと直接会話いただけます

<https://pages.awscloud.com/JAPAN-event-SP-Weekly-Sales-Consulting-Seminar-2021-reg-event.html>



Thank you!



Amazon CloudWatchの概要と基本

AWS Black Belt Online Seminar

津和崎 美希

Solutions Architect
2023/03

AWS Black Belt Online Seminarとは

- ・ 「サービス別」「ソリューション別」「業種別」などのテーマに分け、
アマゾン ウェブ サービス ジャパン合同会社が提供するオンラインセミナー
シリーズです
- ・ AWSの技術担当者が、AWSの各サービスやソリューションについてテーマご
とに動画を公開します
- ・ 動画を一時停止・スキップすることで、興味がある分野・項目だけの聴講も
可能、スキマ時間の学習にもお役立ていただけます
- ・ 以下のURLより、過去のセミナー含めた資料などをダウンロードするこ
とができます
 - ・ <https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-service-cut/>
 - ・ <https://www.youtube.com/playlist?list=PLzWGOASvSx6FIwIC2X1nObr1KcMCBBlqY>

内容についての注意点

- ・ 本資料では2023年03月時点のサービス内容および価格についてご説明しています。最新の情報はAWS公式ウェブサイト(<https://aws.amazon.com/>)にてご確認ください
- ・ 資料作成には十分注意しておりますが、資料内の価格とAWS公式ウェブサイト記載の価格に相違があった場合、AWS公式ウェブサイトの価格を優先とさせていただきます
- ・ 価格は税抜表記となっています。日本居住者のお客様には別途消費税をご請求させていただきます

自己紹介

名前 :

津和崎 美希 (つわざき みき)

所属 :

エンタープライズ技術統括本部

経歴 :

前職ではSIerにて、オンプレミス・クラウドでの構築/運用を担当。

主な業務内容はインフラ設計/構築/試験/運用/移行/追加構築計画・実作業と幅広く担当し、
オンコール対応や復旧のための調査・データセンターでのオペレーションも経験。

好きなAWSサービス :

Amazon CloudWatchをはじめとしたObservability関連サービス



Amazon
CloudWatch

本セミナーの対象者

これからAWSで監視やObservability（可観測性）を始めたい方

Amazon CloudWatchの概要を把握したい方

※ Amazon CloudWatchの個別の機能については別途Blackbeltでご紹介予定です。

今後のBlackbeltリリースをご確認下さい。

アジェンダ

1. モニタリング・オブザーバビリティとは？
2. AWSにおけるObservability全体像
3. Amazon CloudWatch全体像
4. 次に何をしたらよいの？

1. モニタリング・オブザーバビリティ とは？

オブザーバビリティ（可観測性）とは



- システムで何がなぜ起こっているかといった動作状況を 把握できている状態のこと
 - 多くの場合、システムを計測（インストルメント）してメトリクス、ログ、またはトレースを収集することによって行われる
- システムがどのように動作しているかを理解することで、運用上の優秀性・ビジネス目標を達成することにも役立つ
 - このサービスでは、メソッドの 90% が 200 ミリ秒以下で完了している
 - この API は 1 秒間に 203 の HTTP リクエストを処理している
 - このサービスの CPU 使用率は 85% に達している

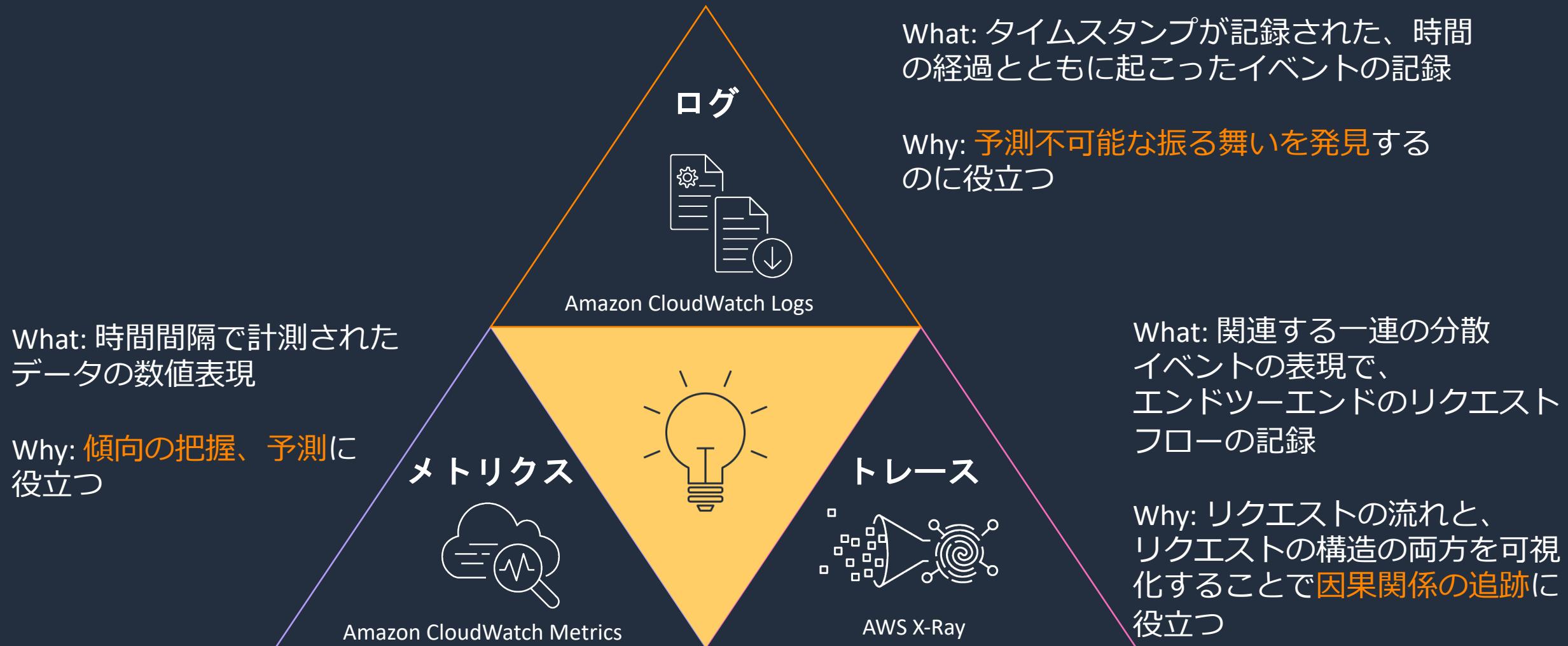
※モニタリング（監視）との違い

明確な定義があるわけではなく、様々な解釈がある。

監視はエラーに対応するためにシステムが機能しているかを検知・把握することを主目的としている場合が多い。

オブザーバビリティはシステムでの計測項目を継続的に追加し、新たなインサイトを得ることで、未知の障害の対処、ダウンタイムの短縮に活かし、システム障害以外のビジネス目標達成の用途にも役立てる。

システムの状態把握に必要な3本柱



ログ・メトリクス・トレース+“イベント”?



イベント=状態の変化。
モニタリングと業務機能双方で活用される

モニタリングでは障害となるイベントが起きたら、リトライ・自動復旧・通知が適切に行えることが重要



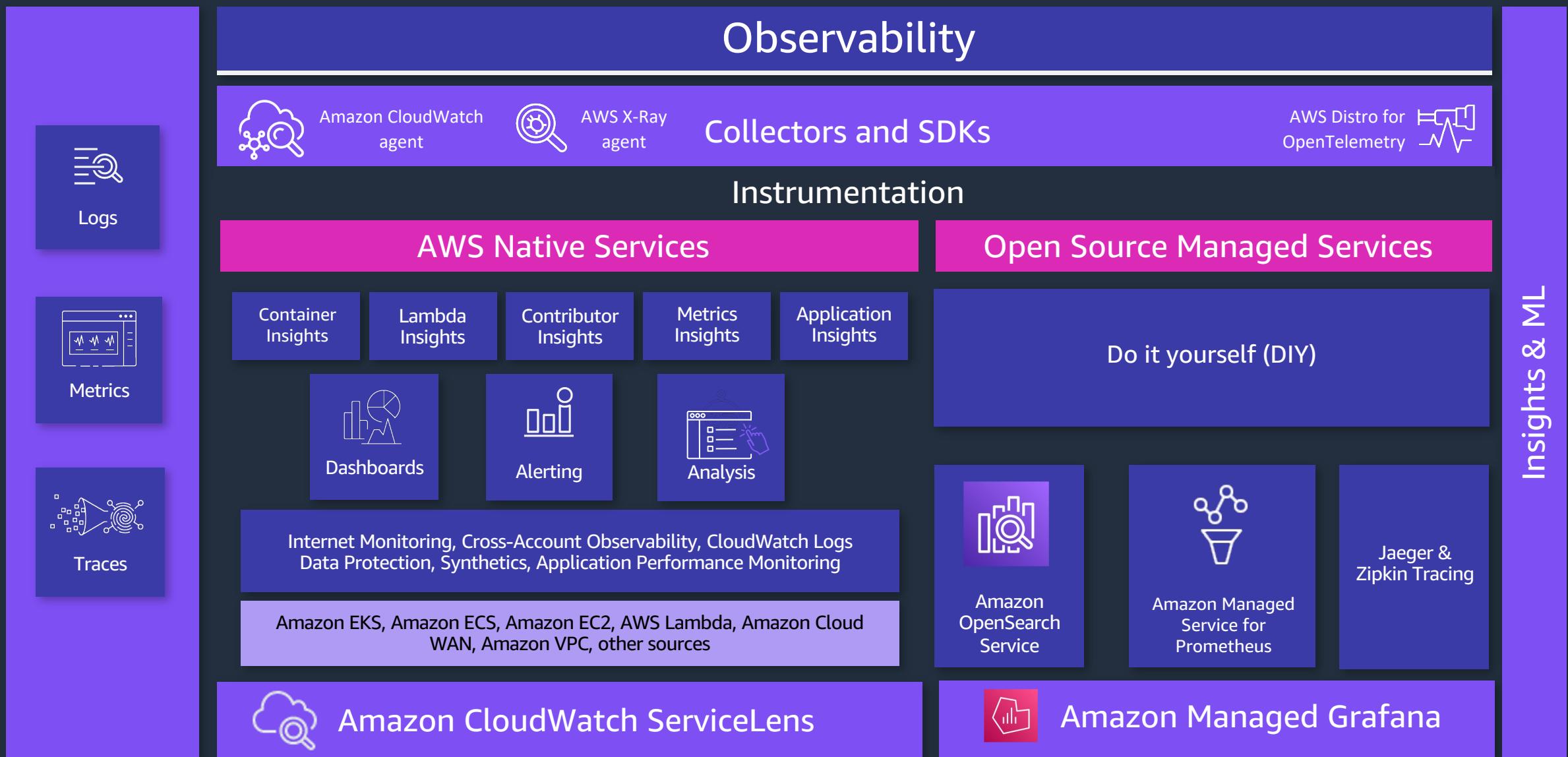
※Amazon EventBridgeはCloudWatch Eventsから派生・独立。
イベントが発生するとターゲットにイベントを送信するが、AWSサービスだけでなく、カスタマイズができるようになっており、パートナーサービスとも連携できる。



2. AWSにおけるObservability全体像

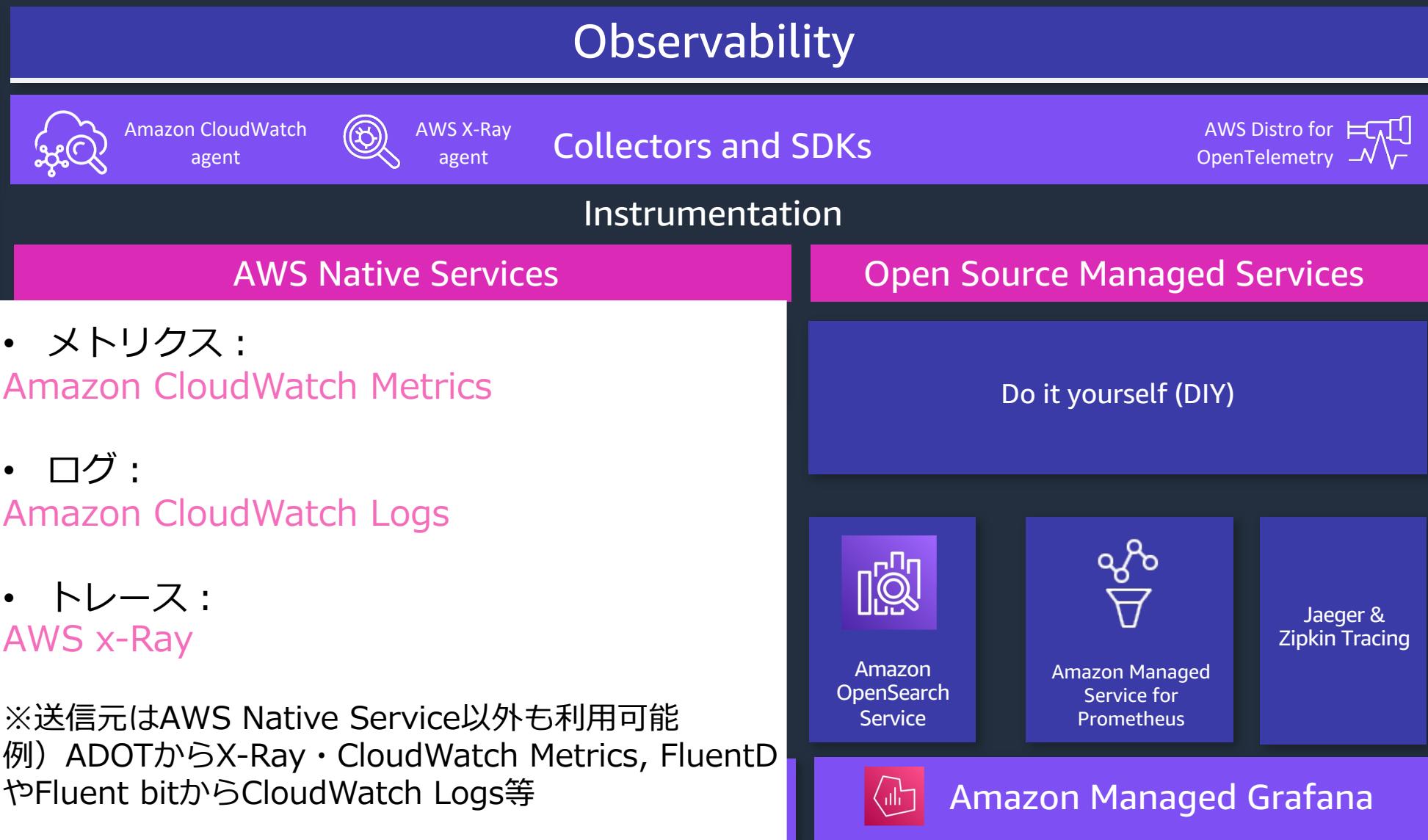
AWS Observability

Observability



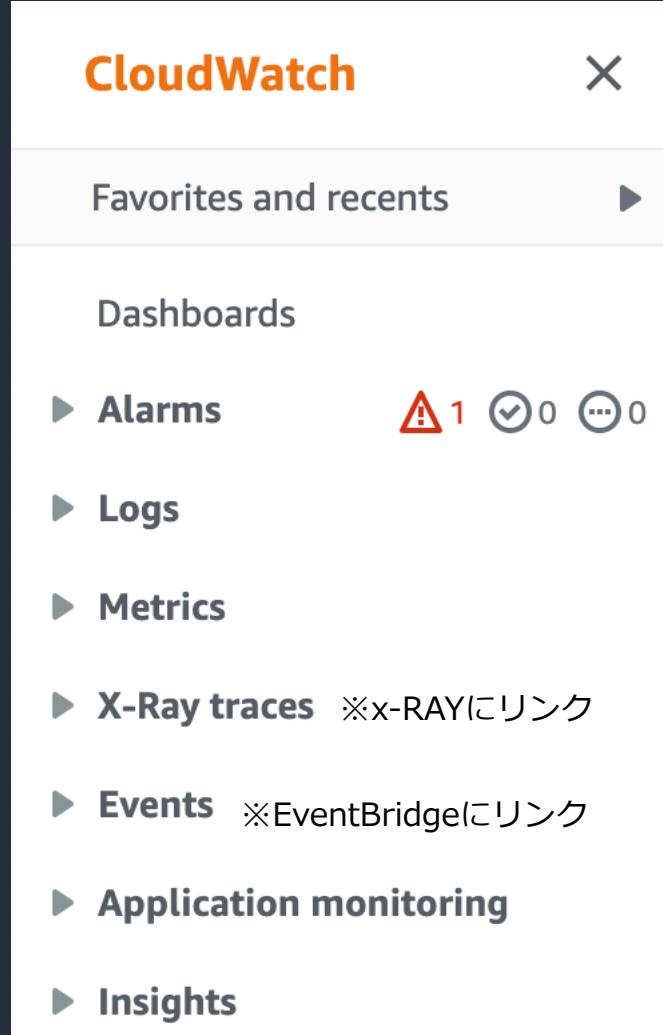
AWS Observability

Observability



3. Amazon CloudWatch全体像

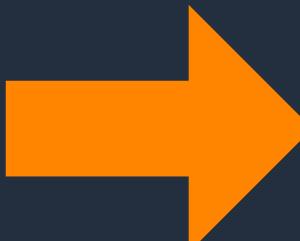
マネジメントコンソールでCloudWatchを検索



様々な機能が
用意されている



This screenshot shows the 'Logs' section of the CloudWatch service menu. It is a dropdown menu with the title 'Logs'. It contains two main categories: 'Log groups' and 'Logs Insights'. Below these are two additional sections: 'Metrics' (which contains 'All metrics', 'Explorer', and 'Streams') and 'Application monitoring' (which contains 'ServiceLens Map', 'Resource Health', 'Internet Monitor', 'Synthetics Canaries', 'Evidently', and 'RUM').



This screenshot shows the 'Insights' section of the CloudWatch service menu. It is a dropdown menu with the title 'Insights'. It contains four main categories: 'Container Insights', 'Lambda Insights', 'Contributor Insights', and 'Application Insights'.

CloudWatchの機能一覧

Infrastructure

AWSサービス名	概要
CloudWatch Metrics	メトリクス
CloudWatch Log	ログ
CloudWatch Alarm	アラーム
CloudWatch Dashboard	ダッシュボード
CloudWatch Metrics Explorer	メトリックス検索
CloudWatch Metrics Stream	メトリックスのリアルタイム連携
CloudWatch Events ※これはEvent Bridgeに統合されています	イベント
CloudWatch Resource Health	EC2の健全性・パフォーマンス可視化

Application Monitoring

AWSサービス名	概要
CloudWatch Synthetics	外形監視
CloudWatch RUM	リアルユーザー モニタリング
CloudWatch Evidently	フィーチャーフラグA/Bテスト
CloudWatch Internet Monitor	インターネット監視
CloudWatch ServiceLens	トレース

コンソールからのメニューで
19機能！

Insights

AWSサービス名	概要
CloudWatch Contributor Insights	ログの時系列分析
CloudWatch Container Insights	コンテナ分析
CloudWatch Lambda Insights	Lambda分析
CloudWatch Application Insights	アプリケーション分析
CloudWatch Logs Insights	LOG分析
CloudWatch Metrics Insights	メトリクス分析

Amazon CloudWatchの全体像

Application Monitoring

- [外形監視]
CW Synthetics
- [リアルユーザー モニタリング]
CW RUM
- [フィーチャーフラグA/Bテスト]
CW Evidently

- [インターネット監視]
CW Internet Monitor
- [トレース]
CW ServiceLens

[ダッシュボードに統合]
CW Dashboard



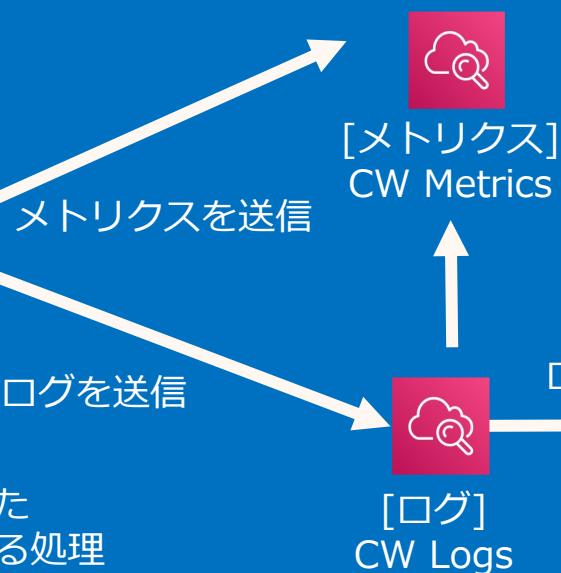
Infrastructure



イベントに応じた
ターゲットによる処理



[イベント]
Amazon EventBridge/
Amazon EventBridge Scheduler



[メトリクス]
CW Metrics

[ログ]
CW Logs

ログの可視化

[メトリクスストリーム]
CW Metrics Stream

[ログ分析]
CW Logs Insights

Amazon Kinesis
Data Firehose

パートナー
サービス、
S3/Redshift



[アラーム]
CW Alarms

メトリクスに応じた
アクション



[リアルタイムメトリクス分析]
CW Metrics Insight
[タグベースの視覚化]
CW Metrics Explorer

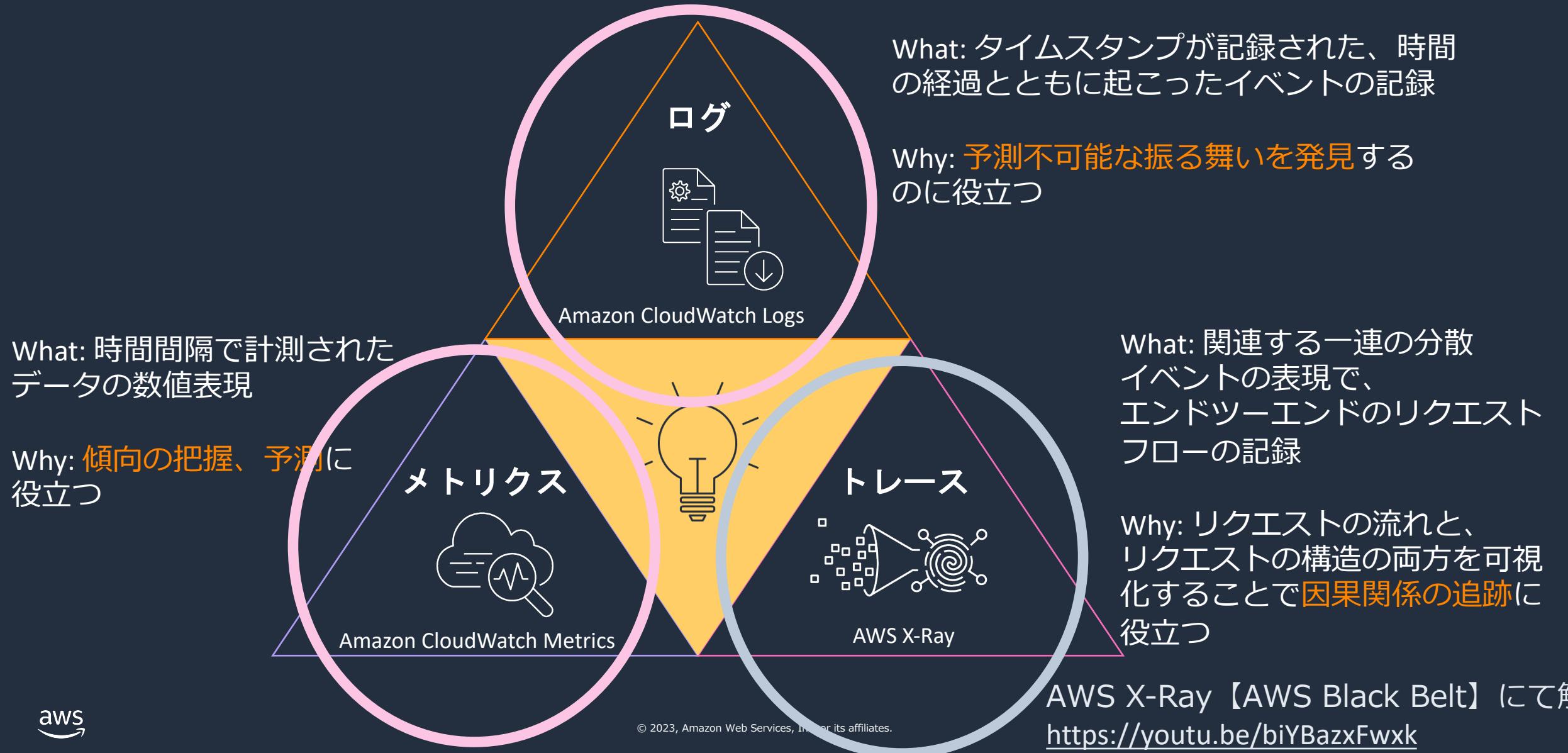
Insights



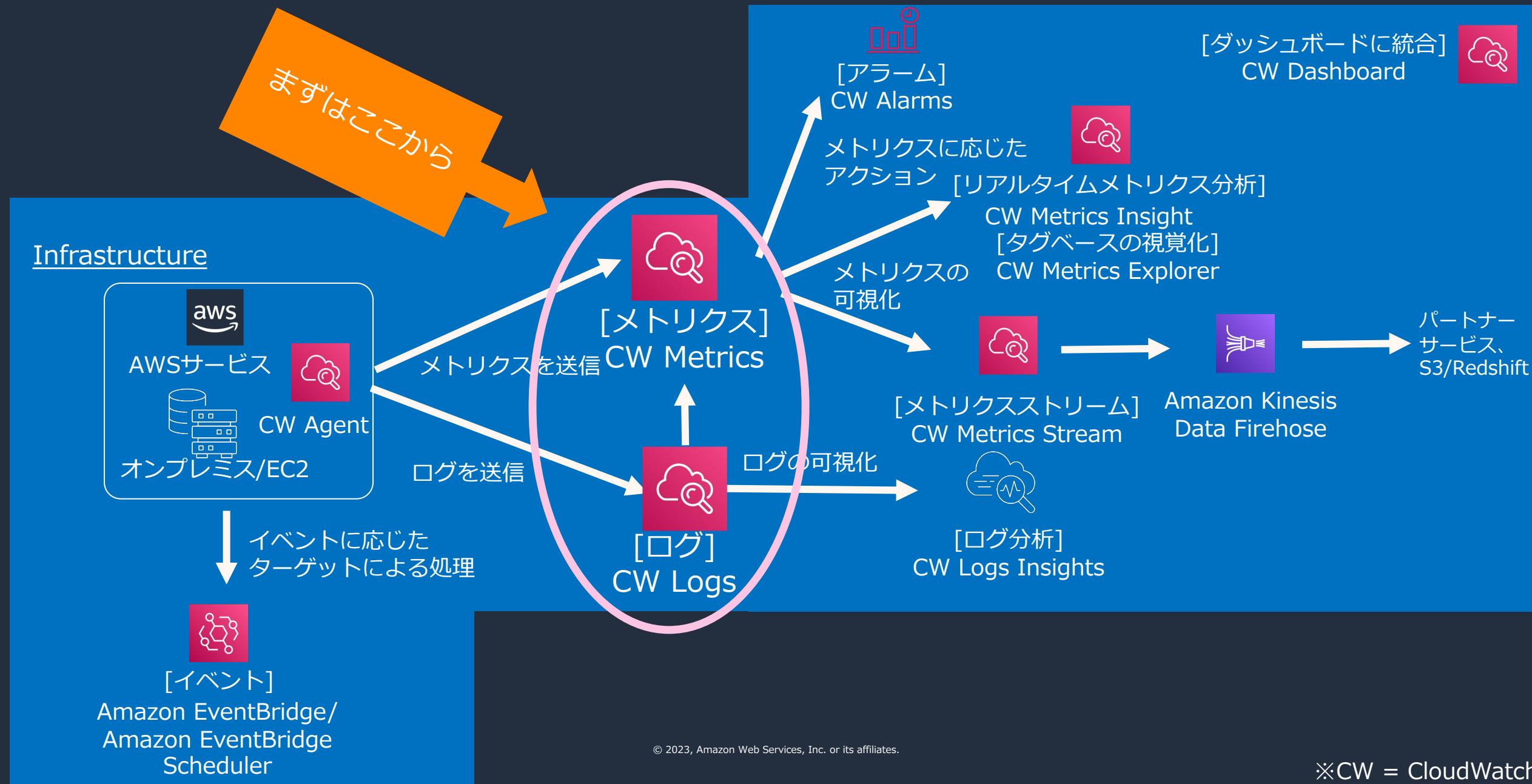
- [構造化ログによるメトリクス]
CW Container Insights / Contributor Insights
- [Lambda拡張機能によるメトリクス]
Lambda Insights
- [アプリケーションコンポーネントのメトリクス]
Application Insights

※CW = CloudWatch

再掲：システムの状態把握に必要な3本柱



Amazon CloudWatchの全体像

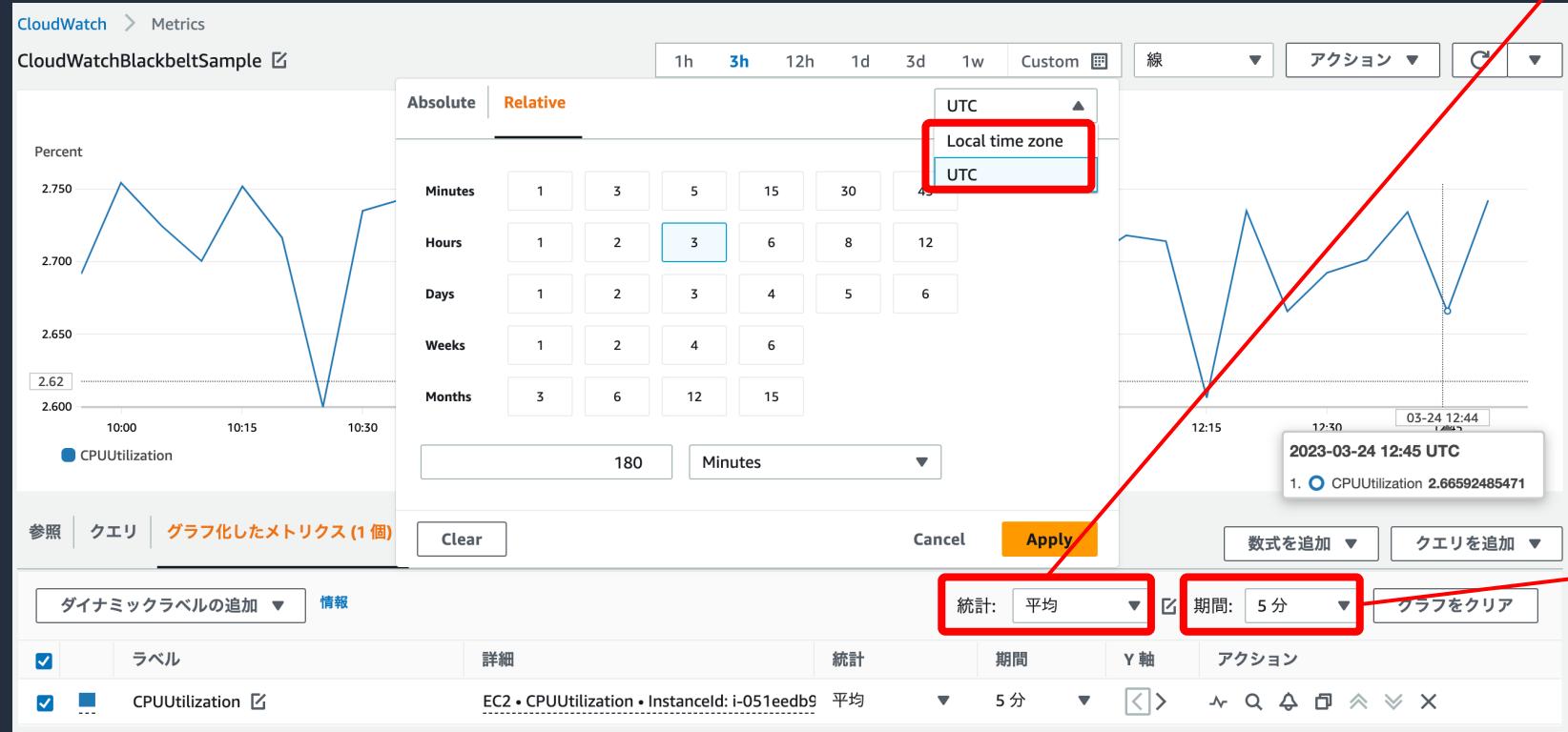


メトリクスとは

時間間隔で計測されたデータの数値表現

期間や統計でサマリーされた情報から、傾向の把握に役立てる

例) EC2の標準メトリクス CPUUtilization



タイムゾーン、統計情報、統計の粒度を設定し表示



Standard
平均
最小
最大
合計
サンプル数

Expandable
PR(n:m)
TC(10%:90%)
TM(10%:90%)
TS(10%:90%)
WM(10%:90%)

※Trim値やPercentileはウェブページのレイテンシ計測など大きなサンプルサイズでネットワークの問題等で起こる外れ値を無視したい場合に活用

詳細はCloudWatch統計定義を参照

期間:
5 分
1 秒
5 秒
10 秒
30 秒
1 分
5 分
15 分
1 時間
6 時間
1 日
7 日
30 日

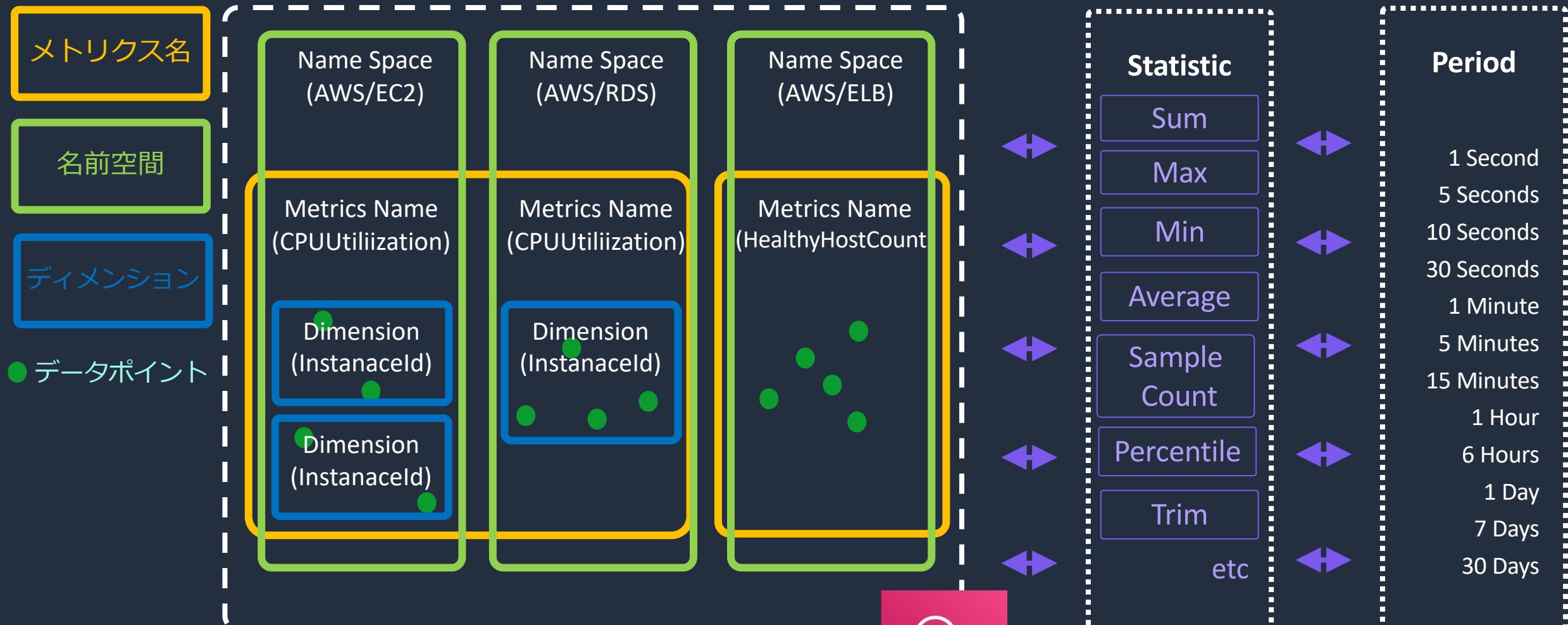
例えは・・・
EC2のインスタンスマトリクスは標準のデフォルト5分間。(無料)
詳細モニタリングの有効化で1分間の取得が可能。

高解像度のカスタムメトリクスを活用すれば1秒の解像度で期間を指定也可能。

※EC2でもステータスチェックメトリクスはデフォルト1分。(無料)

※高解像度のカスタムメトリクスはAPIやプラグインの設定が必要 (参考Blog)

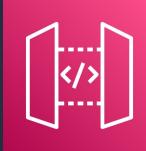
CloudWatch Metricsの考え方



多くのAWSサービスでメトリクスやログを標準で発行

CloudWatch との統合により自動取得可能

メトリクスの例



142サービスでメトリクス
が自動取得可能
※ 2023/03末時点

Amazon API Gateway

- REST API: API request count, Latency, 4XXs, 5XXs, IntegrationLatency, CacheHitCount, CacheMissCount



AWS Lambda

- Invocation count, Invocation errors, DeadLetterErrors, DestinationDeliveryFailures, Throttles など

ログの例



30サービスでログが
自動取得可能
※ 2023/03末時点

Amazon API Gateway

- REST API: **ERROR** と **INFO** の 2 レベルのログ
- HTTP API /WebSocket API: ログ変数でカスタマイズ可能



AWS Lambda

- Lambda 関数の呼び出しの詳細、ログ
- コードから直接ログを取得: e.g. `console.log()`

※1 Amazon CloudWatch Metricsを発行するサービス
<https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/aws-services-cloudwatch-metrics.html>

※2 Lambda Insights/Container Insights/DynamoDB Contributor Insights/Amazon RDS Performance Insights等、有効化することで追加のMetricsが取得できるサービス多数

※3 Amazon CloudWatch Logs にログを発行する サービス
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/aws-services-sending-logs.html

※4 AWS のサービスからのログの記録を有効にできるサービス
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/AWS-logs-and-resource-policy.html

CloudWatch Metric Insights

メトリクスのリアルタイム集計・可視化



メトリクスを効率よく分析

- 柔軟なクエリ機能により、リアルタイムでメトリクスの集計やグルーピング
- SQL ベース のクエリや選択式のビルダーで実行

例) ProductInfoService の名前空間に対して Operation ごとのエラー合計で Top5 を表示

Metrics Insights - Query Editor

```

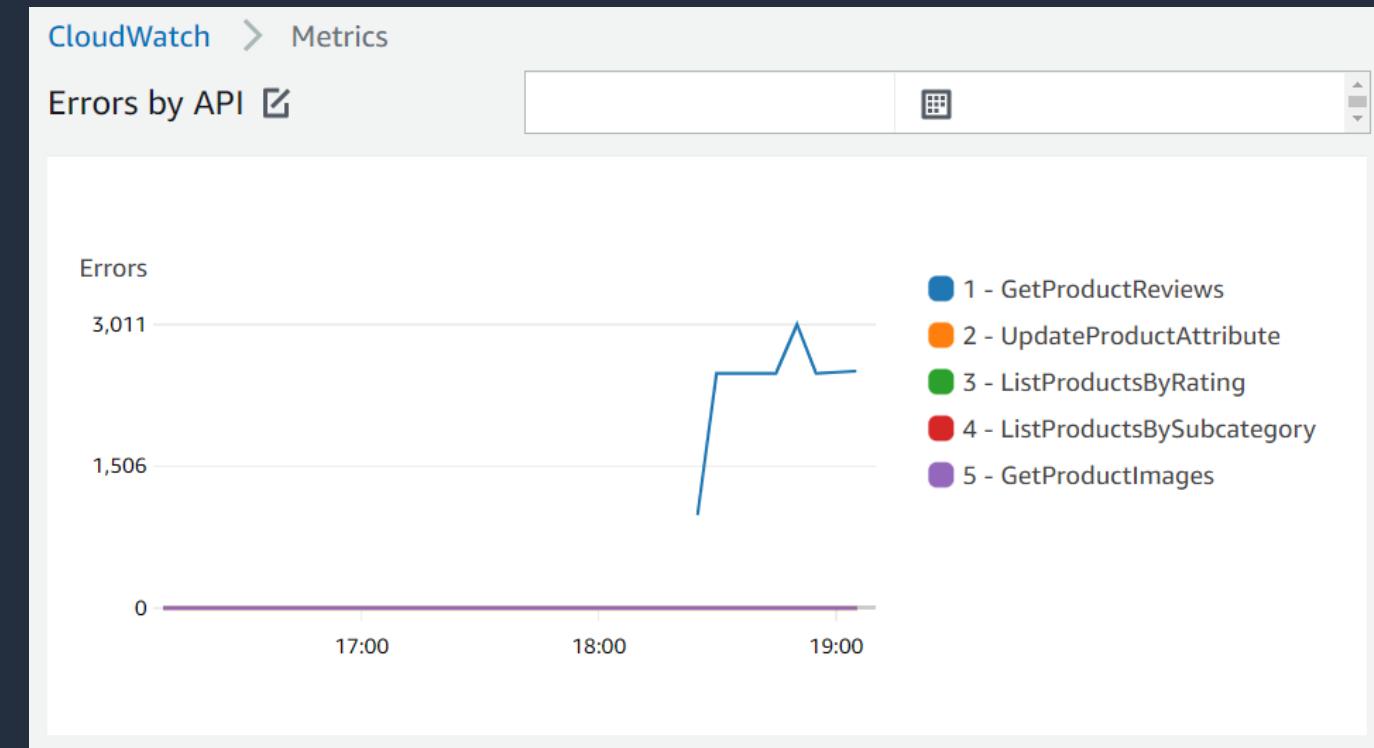
1 SELECT SUM(Errors)
2 FROM ProductInfoService
3 GROUP BY Operation
4 ORDER BY SUM() DESC
5 LIMIT 5

```

OR

Metrics Insights - Query Builder

名前空間	ProductInfoService	X
メトリクス名	SUM(Errors)	X
フィルター条件	Q ディメンションを選択 件 デフォルトでは、すべてのディメンションが選択されています	
グループ化条件	Operation	X
並べ替え条件	Q グループ化するディメンションを選択 件 メトリクスは常に 1 つの時系列に事前に集計されます。	
制限		5 X



CloudWatch メトリクスに数式を使用して、新しいメトリクスを作成

- METRICS関数、基本的な算術関数をはじめとした関数をサポート
- メトリクスに[Id]フィールドを設定し、関数で利用

コマンド	説明	例
AVG	データポイントの平均を表すスカラー	メトリクスの平均値 : AVG(METRICS())
SUM	データポイントの合計値を表すスカラー	メトリクスm1,m2の合計値 : SUM([m1,m2])
METRICS	CloudWatch メトリクスを表す	メトリクスreqall : METRICS("reqall")

etc

ユースケース
全リクエストのうち4xx,5xx
レスポンスの割合を表示

$\text{SUM}([\text{METRICS}("res4xx"), \text{METRICS}("res5xx")]) / \text{SUM}(\text{METRICS}("reqall"))$



CloudWatch Alarmsでメトリクスをアクションにつなげる

アクション

メトリクス



Amazon
CloudWatch

メトリクスアラームの状態

状態が変わったときに実行するアクションが指定できる

OK	ALARM	INSUFFICIENT_DATA
正常値 ✓ 定義された閾値を下回っている	異常値 ✓ 定義された閾値を上回っている	判定不能 ✓ データ不足のため状態を判定できない ✓ CloudWatch特有



Amazon SNS



Amazon EC2



Amazon EC2
Auto Scaling



AWS Systems Manager
OpsCenter

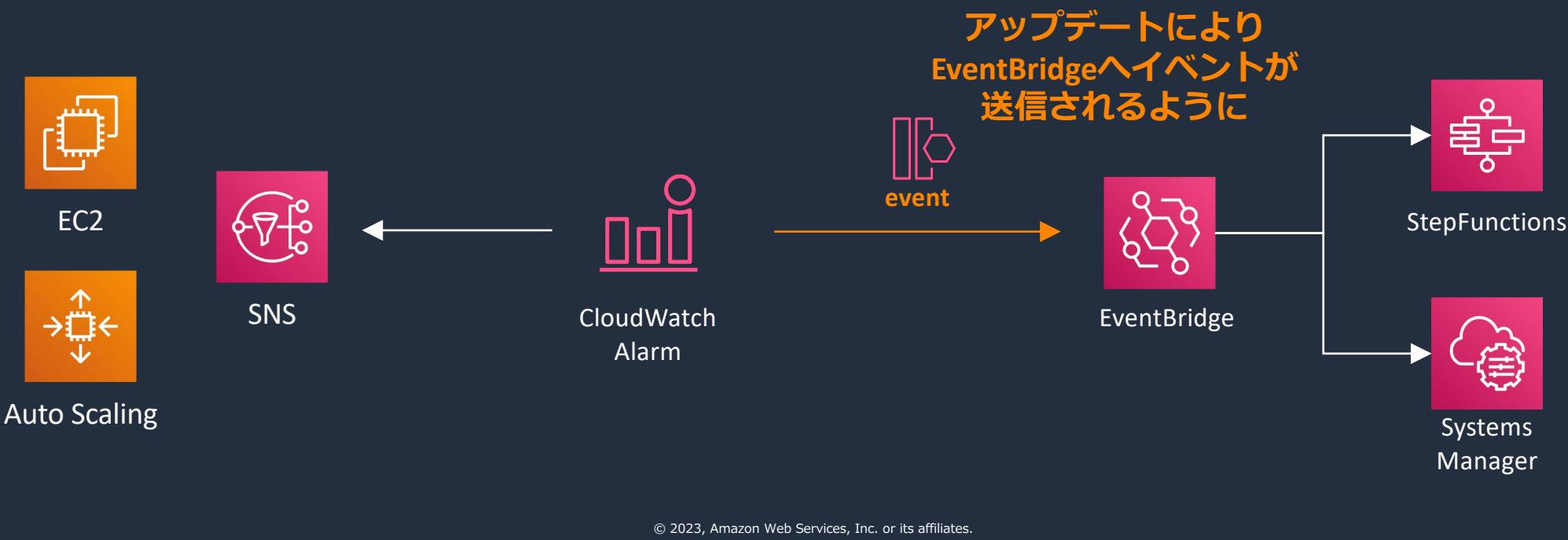


AWS Systems Manager
Incident Manager

※Event Bridgeにも
アラーム状態変化は
配信される

参考：EventBridge でCloudWatch Alarmのイベントを取り扱う例

- CloudWatch Alarmのステータス変化がEventBridgeで取得可能。
- CloudWatch Alarmの機能では送信先としてEC2、SNS、AutoScaling、Systems ManagerでのOpsItem起票/Incident作成の操作のみが選択可能。
- Event Bridgeを介せばAlarmのイベントが**実装なしで様々なサービスに連携可能**



アラームを検知し、メトリクスを確認すると
2時過ぎにインスタンスAでErrorが多かったようだ・・・



原因を特定するには？？

ログを見てみましょう

Everything fails, all the time.備えは忘れずに。基本的には自動復旧が大切！！

- EC2のSystem Status Checkは失敗しうる。原因を特定することは、サービスレベル向上につながらない。
- アラームのアクションやAmazon Event Bridgeの他にも、AWS Step Functions/Amazon SQS/Amazon SNSなどでリトライや障害時のデータの一貫性確保をコントロールしてもよい。

ログとは

タイムスタンプが記録された、時間の経過とともに起こったイベントの記録
システムで何が起こっているか、振る舞いを解析するのに役立てる

CloudWatch > ロググループ > /ecs/PayForAdoption > logs/container/eb7c8084084c4cc19b3571add02f797f

ログイベント

下のフィルターバーを使用して、ログイベント内の用語、語句、値の検索や照合ができます。 [フィルターパターンの詳細](#)

アクション ▾ メトリクスフィルターを作成

検索 イベントをフィルター クリア 1m 30m 1h 12h カスタム 表示 ▾

ログイベント一覧

タイムスタンプ	メッセージ
ロードする古いイベントがあります。 さらにロードします。	
2023-03-28T12:43:06.562+09:00	POST /api/home/cleanupadoptions HTTP/1.1 11.0.12.188:1638 200 POST /api/home/cleanupadoptions HTTP/1.1 11.0.12.188:1638 200
2023-03-28T12:43:06.562+09:00	{"caller": "middlewares.go:84", "err": null, "method": "In CleanupAdoptions", "took": "34.438289ms", "traceId": "..."} {"caller": "repository.go:64", "repo": "sql", "sql": "\n\t\tINSERT INTO transactions (pet_id, transaction_id, ..."} {"caller": "repository.go:124", "method": "UpdateAvailability", "repo": "sql", "success": "(MISSING)", "ts": "..."} {"PetId": "001", "PetType": "puppy", "caller": "middlewares.go:57", "customer": {"ID": 1679974987142511795, "F..."} POST /api/home/completeadoption?petId=001&petType=puppy HTTP/1.1 11.0.105.43:54362 200
2023-03-28T12:43:07.142+09:00	{"caller": "repository.go:64", "repo": "sql", "sql": "\n\t\tINSERT INTO transactions (pet_id, transaction_id, ..."} {"caller": "repository.go:124", "method": "UpdateAvailability", "repo": "sql", "success": "(MISSING)", "ts": "..."} {"PetId": "001", "PetType": "puppy", "caller": "middlewares.go:57", "customer": {"ID": 1679974987142511795, "F..."} POST /api/home/completeadoption?petId=001&petType=puppy HTTP/1.1 11.0.105.43:54362 200
2023-03-28T12:43:07.303+09:00	{"caller": "repository.go:64", "repo": "sql", "sql": "\n\t\tINSERT INTO transactions (pet_id, transaction_id, ..."} {"caller": "repository.go:124", "method": "UpdateAvailability", "repo": "sql", "success": "(MISSING)", "ts": "..."} {"PetId": "020", "PetType": "kitten", "caller": "middlewares.go:57", "customer": {"ID": 1679974987362062218, "F..."} POST /api/home/completeadoption?petId=020&petType=kitten HTTP/1.1 11.0.12.188:13162 200
2023-03-28T12:43:07.362+09:00	{"caller": "repository.go:64", "repo": "sql", "sql": "\n\t\tINSERT INTO transactions (pet_id, transaction_id, ..."} {"caller": "repository.go:124", "method": "UpdateAvailability", "repo": "sql", "success": "(MISSING)", "ts": "..."} {"PetId": "020", "PetType": "kitten", "caller": "middlewares.go:57", "customer": {"ID": 1679974987362062218, "F..."} POST /api/home/completeadoption?petId=020&petType=kitten HTTP/1.1 11.0.12.188:13162 200

ログはタイムスタンプと
メッセージからなる。

使い方 :

トレースやメトリクスを参考に見るべきログを特定し、該当の時間の該当のリソースのログを確認することで何が起こっているか把握する

CloudWatch Logsでログを活用する流れ

各種ログ を収集

- エージェントやログドライバ・サイドカー等でログ収集



EC2 AWS外のServer Fargate

- 統合CloudWatchエージェント
- awslogsログドライバー
- AWS Distro for OpenTelemetry (ADOT)
- Amazon ECS Firelens
(Fluentd や Fluent Bitイメージを活用)
- その他SaaS/OSSのAgentやCollectorなど

- Amazon CloudWatch Logs にログを発行するサービス



CloudTrail Lambda RDS VPC Flow Log など
他多数 ※1

- 設定によりAmazon CloudWatch Logs に追加でログを発行するサービス



API Gatewayアクセスログ Step Functions実行履歴 Route 53クエリログ など
他多数 ※2



CloudWatch Logs



通知:
CloudWatch Alarms



データ連携:
Amazon Kinesis Data
Firehose



可視化:
Amazon OpenSearch Service

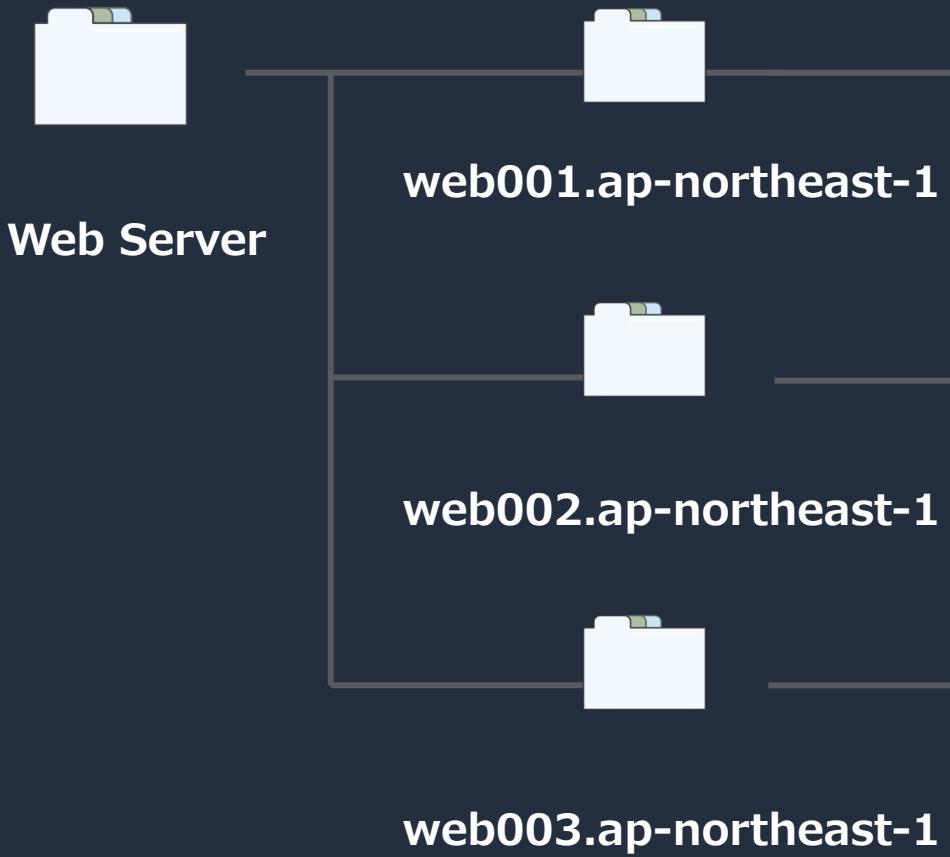
※CloudFront・ALB・NLBのアクセスログなどはS3に直接出力されるなど、
CloudWatch Logs以外に直接ログが出力されるケースもある。

※1 https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/aws-services-sending-logs.html

※2 https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/AWS-logs-and-resource-policy.html

CloudWatch Logsの仕組み

ロググループ ログストリーム ログイベント



ログイベント

```

2014-07-24 17:20:59 UTC+9 10-104 kernel: [0.00000] inlog 5.8.10, log source : /proc/kmsg started.
2014-07-24 17:20:59 UTC+9 10-104 rsyslogd: [origin software="rsyslogd" version="5.8.10" xid="1031"] <http://www.rsyslog.com>
2014-07-24 17:20:59 UTC+9 10-104 kernel: [ 0.00000] Initializing cursor subres cursor
2014-07-24 17:20:59 UTC+9 10-104 kernel: [ 0.00000] Initializing cursor subres cursor
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] Linux version 3.10-42-02-145-vaent-va08_84 (nmbuild@pub/build-E4003) (-)
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] Command line: rootLABEL=/dev/sda1
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] vmlinuz-3.10.0-42-generic physical RAM size: 0x0000000000000000
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] 8196<-0x200 [sw 0x0000000000000000-0x000000000000ffff] usable
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] 8196<-0x200 [sw 0x0000000000000000-0x000000000000ffff] reserved

```



```

2014-07-24 17:20:59 UTC+9 10-104 kernel: [0.00000] inlog 5.8.10, log source : /proc/kmsg started.
2014-07-24 17:20:59 UTC+9 10-104 rsyslogd: [origin software="rsyslogd" version="5.8.10" xid="1031"] <http://www.rsyslog.com>
2014-07-24 17:20:59 UTC+9 10-104 kernel: [ 0.00000] Initializing cursor subres cursor
2014-07-24 17:20:59 UTC+9 10-104 kernel: [ 0.00000] Initializing cursor subres cursor
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] Linux version 3.10-42-02-145-vaent-va08_84 (nmbuild@pub/build-E4003) (-)
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] Command line: rootLABEL=/dev/sda1
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] vmlinuz-3.10.0-42-generic physical RAM size: 0x0000000000000000
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] 8196<-0x200 [sw 0x0000000000000000-0x000000000000ffff] usable
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] 8196<-0x200 [sw 0x0000000000000000-0x000000000000ffff] reserved

```



```

2014-07-24 17:20:59 UTC+9 10-104 kernel: [0.00000] inlog 5.8.10, log source : /proc/kmsg started.
2014-07-24 17:20:59 UTC+9 10-104 rsyslogd: [origin software="rsyslogd" version="5.8.10" xid="1031"] <http://www.rsyslog.com>
2014-07-24 17:20:59 UTC+9 10-104 kernel: [ 0.00000] Initializing cursor subres cursor
2014-07-24 17:20:59 UTC+9 10-104 kernel: [ 0.00000] Initializing cursor subres cursor
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] Linux version 3.10-42-02-145-vaent-va08_84 (nmbuild@pub/build-E4003) (-)
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] Command line: rootLABEL=/dev/sda1
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] vmlinuz-3.10.0-42-generic physical RAM size: 0x0000000000000000
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] 8196<-0x200 [sw 0x0000000000000000-0x000000000000ffff] usable
2014-07-24 17:20:59 UTC+9 Jul 24 08:07:09 i=10-10-104 kernel: [ 0.00000] 8196<-0x200 [sw 0x0000000000000000-0x000000000000ffff] reserved

```

ログイベント

- 1つのログエントリ
- モニタリングしているリソースによって記録されたアクティビティのレコード
- イベント発生時のタイムスタンプおよび生のイベントメッセージで構成

ログストリーム

- 同じソースを共有する一連のログイベント
- モニタリングしているリソースのタイムスタンプ順でイベントを表す

ロググループ

- 複数のログストリームで構成
- 保持、監視、アクセス制御について同じ設定を共有するログストリームのグループを定義

ロググループの設定

ログの保持期間・メトリクスフィルター・データ保護ポリシー・サブスクリプションフィルターはロググループごとに設定する。

The screenshot shows the AWS CloudWatch Log Groups interface. On the left, there's a sidebar with actions like 'Logs Insights で表示' (View in Logs Insights) and 'ロググループの検索' (Search Log Group). A red arrow points from the 'Actions' button to the '保持設定を編集' (Edit Retention Settings) option in the dropdown menu. Another red arrow points from the 'Logs Insights で表示' button to the 'Logs Insights' tab in the main content area.

CloudWatch > ロググループ > /aws/lambda/C9-Observability-Workshop-C9DiskResizeLambda-SGRA3U9M4JRT

/aws/lambda/C9-Observability-Workshop-C9DiskResizeLambda

アクション ▲ Logs Insights で表示 ロググループの検索

ロググループの削除
保持設定を編集
メトリクスフィルターを作成
データ保護ポリシーを作成
サブスクリプションフィルター ▶
寄稿者インサイトルールを作成
データを Amazon S3 にエクスポート
Amazon S3 へのすべてのエクスポートを表示
保持

メトリクスフィルター
Amazon OpenSearch Service サブスクリプションフィルターを作成
Kinesis サブスクリプションフィルターを作成
Kinesis Firehose サブスクリプションフィルターを作成
Lambda サブスクリプションフィルターを作成
すべてのサブスクリプションフィルターを削除

データ
データを非表示

保持期間の設定

次の期間経過後にイベントを失効:
失効しない
失効しない (selected)
1 日
3 日
5 日
1 週間 (7 日)
2 週間 (14 日)
1 か月 (30 日)
2 months (60 日)
3 months (90 日)
4 months (120 日)
5 months (150 日)
6 months (180 日)
12 months (365 日)
13 months (400 日)
18 months (545 日)
2 年 (731 日)
3 年 (1096 日)
5 年 (1827 日)
6 年 (2192 日)
7 年 (2557 日)
8 年 (2922 日)
9 年 (3288 日)
10 年 (3653 日)

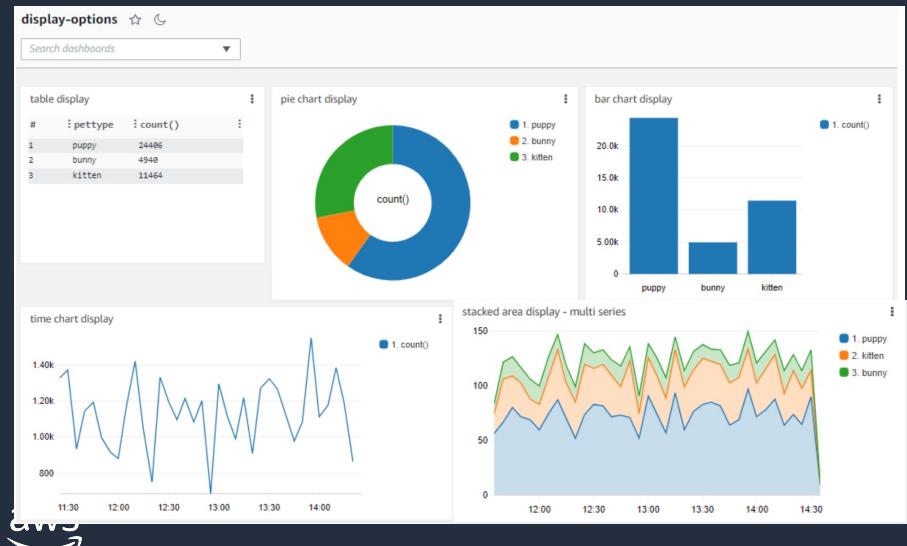
無期限に保存することも可能

© 2023, Amazon Web Services, Inc. or its affiliates.

CloudWatch Logs Insights

ログから実用的な洞察を導き出す

- CloudWatch Logs でログデータをインタラクティブに検索・分析
- 専用のクエリ言語
- 時系列データの可視化、個々のログイベントへのドリルダウン、クエリ結果の CloudWatch ダッシュボードへのエクスポート



ログを効率よく分析

例) 1時間あたりの例外発生件数の一覧を取得する

クエリ

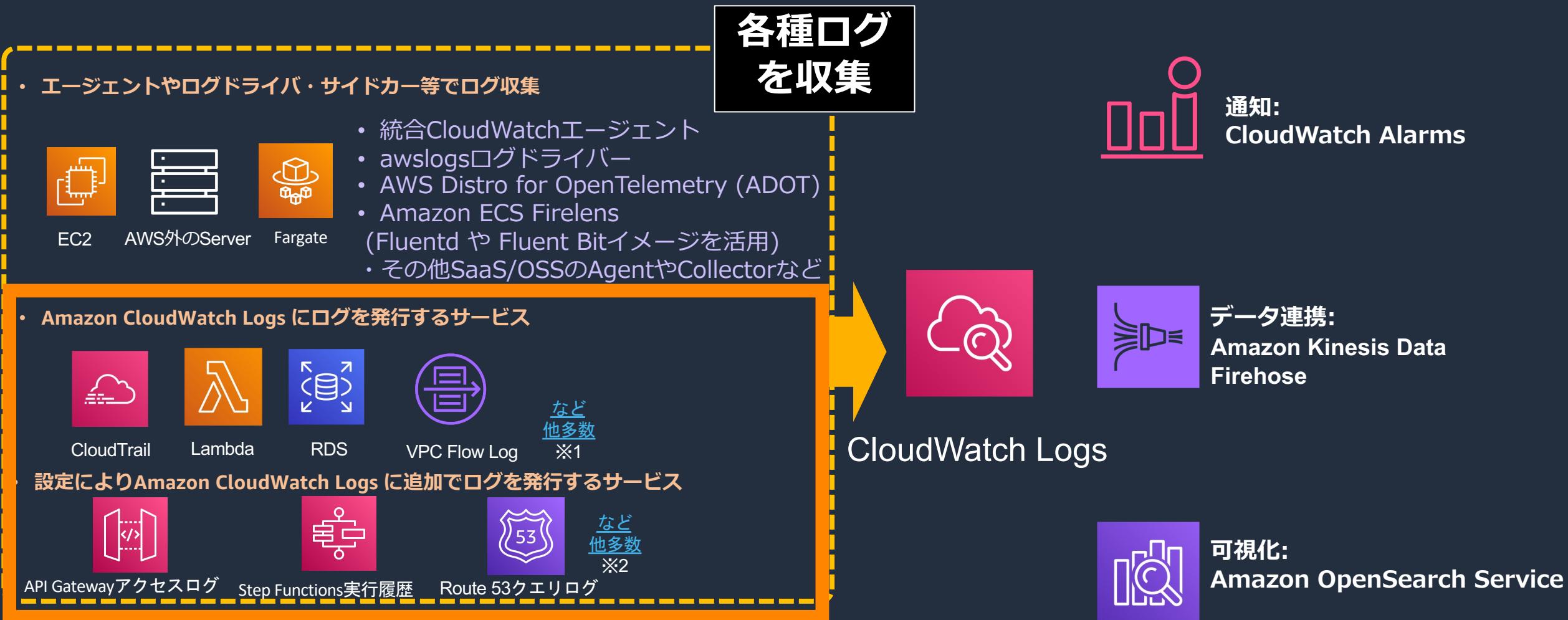
```
1 filter @message like /Exception/
2   | stats count(*) as exceptionCount by bin(1h)
3   | sort exceptionCount desc
```

クエリ結果

#	bin(1h)	exceptionCount
1	2022-07-17T20:00:00.000...	2194
2	2022-07-17T21:00:00.000...	238

CloudWatch Logs Insightsではログデータの検索・分析だけでなく、bin() 関数を使用するクエリを実行して、返された値を期間別にグループ化すると、結果を折れ線グラフ、円グラフ、棒グラフ、積み上げ面グラフとして表示可能。

再掲：CloudWatch Logsでログを活用する流れ



※CloudFront・ALB・NLBのアクセスログなどはS3に直接出力されるなど、CloudWatch Logs以外に直接ログが出力されるケースもある。

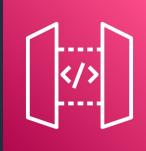
※1 https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/aws-services-sending-logs.html

※2 https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/AWS-logs-and-resource-policy.html

再掲：多くのAWSサービスでメトリクスやログを標準で発行

CloudWatch との統合により自動取得可能

メトリクスの例



142サービスでメトリクス
が自動取得可能
※ 2023/03/30時点

Amazon API Gateway

- REST API: API request count, Latency, 4XXs, 5XXs, IntegrationLatency, CacheHitCount, CacheMissCount



AWS Lambda

- Invocation count, Invocation errors, DeadLetterErrors, DestinationDeliveryFailures, Throttles など

ログの例



30サービスでログが
自動取得可能
※ 2023/03/30時点

Amazon API Gateway

- REST API: ERROR と INFO の 2 レベルのログ
- HTTP API /WebSocket API: ログ変数でカスタマイズ可能



AWS Lambda

- Lambda 関数の呼び出しの詳細、ログ
- コードから直接ログを取得: e.g. console.log()

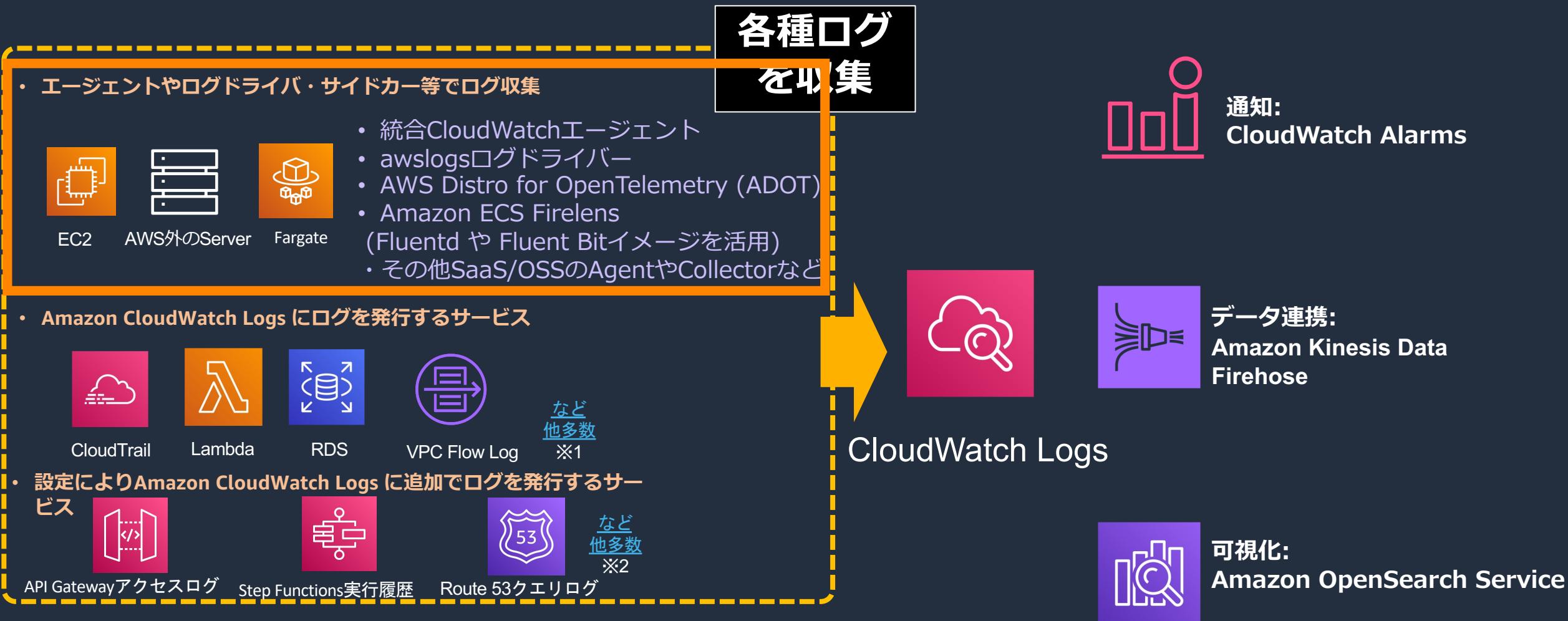
※1 Amazon CloudWatch Metricsを発行するサービス
<https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/aws-services-cloudwatch-metrics.html>

※2 Lambda Insights/Container Insights/DynamoDB Contributor Insights/Amazon RDS Performance Insights等、有効化することで追加のMetricsが取得できるサービス多数

※3 Amazon CloudWatch Logs にログを発行する サービス
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/aws-services-sending-logs.html

※4 AWS のサービスからのログの記録を有効にできるサービス
https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/AWS-logs-and-resource-policy.html

再掲：CloudWatch Logsでログを活用する流れ



※CloudFront・ALB・NLBのアクセスログなどはS3に直接出力されるなど、CloudWatch Logs以外に直接ログが出力されるケースもある。

※1 https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/aws-services-sending-logs.html

※2 https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/AWS-logs-and-resource-policy.html

3本柱のAWSサービスにObservability のデータをカスタムで収集する手段

- ・収集したい情報
- ・収集元の対応
- ・送信先の対応

今必要なもの・将来どの程度
SaaS/OSS含めた送信先の追加が
起こりうるかがTool選びのPoint

統合CloudWatch
エージェント

ログ



Amazon CloudWatch Logs

メトリクス



Amazon CloudWatch Metrics

コンテナの場合のAWSが用意する選択肢

- awslogsログドライバー
- Fluentd や Fluent Bitイメージを活用
 - Amazon ECS Firelens
 - AWS for Fluent Bit image
 - Fluent Bit for Amazon EKS on AWS Fargate

etc

AWS Distro for
OpenTelemetry (ADOT) ※1

X-Ray Agent

トレース

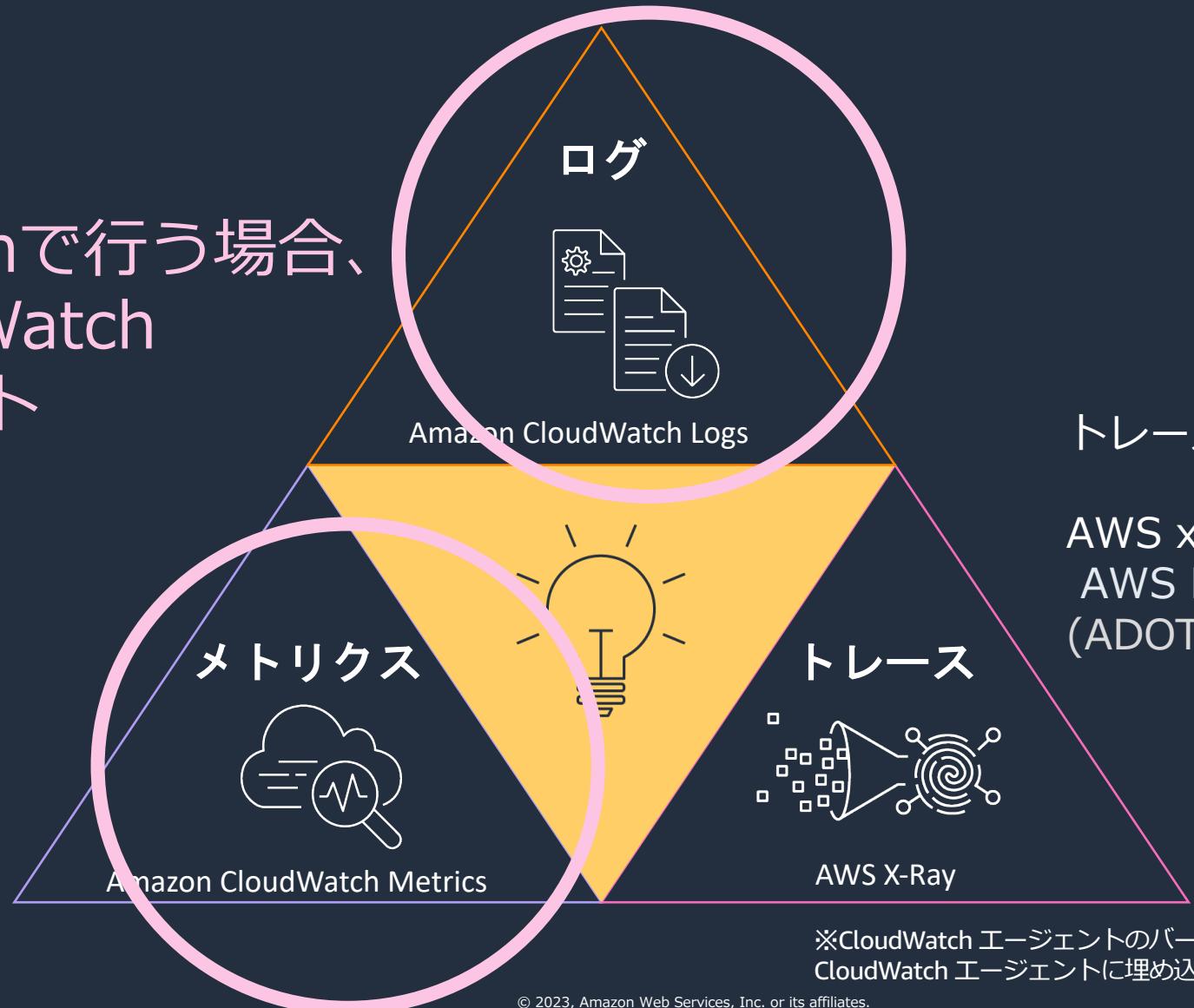


AWS X-Ray



3本柱のAWSサービスにObservability のデータをカスタムで収集する手段

Cloudwatchで行う場合、
統合CloudWatch
エージェント



トレースはどうする？

AWS x-RAYにトレースを送るならば、
AWS Distro for OpenTelemetry
(ADOT)・X-Ray Agentが選択肢



SaaS

その他
OSS

統合CloudWatchエージェント

- CloudWatch Logs・Metrics双方に対応しているエージェント
- クラウドでもオンプレミスでも利用可能。
- Linux, Windowsの両方で稼働。
- 以前のCloudWatch Logs エージェントはサポートされていても非推奨※1

★インストール方法は大きく以下3パターン

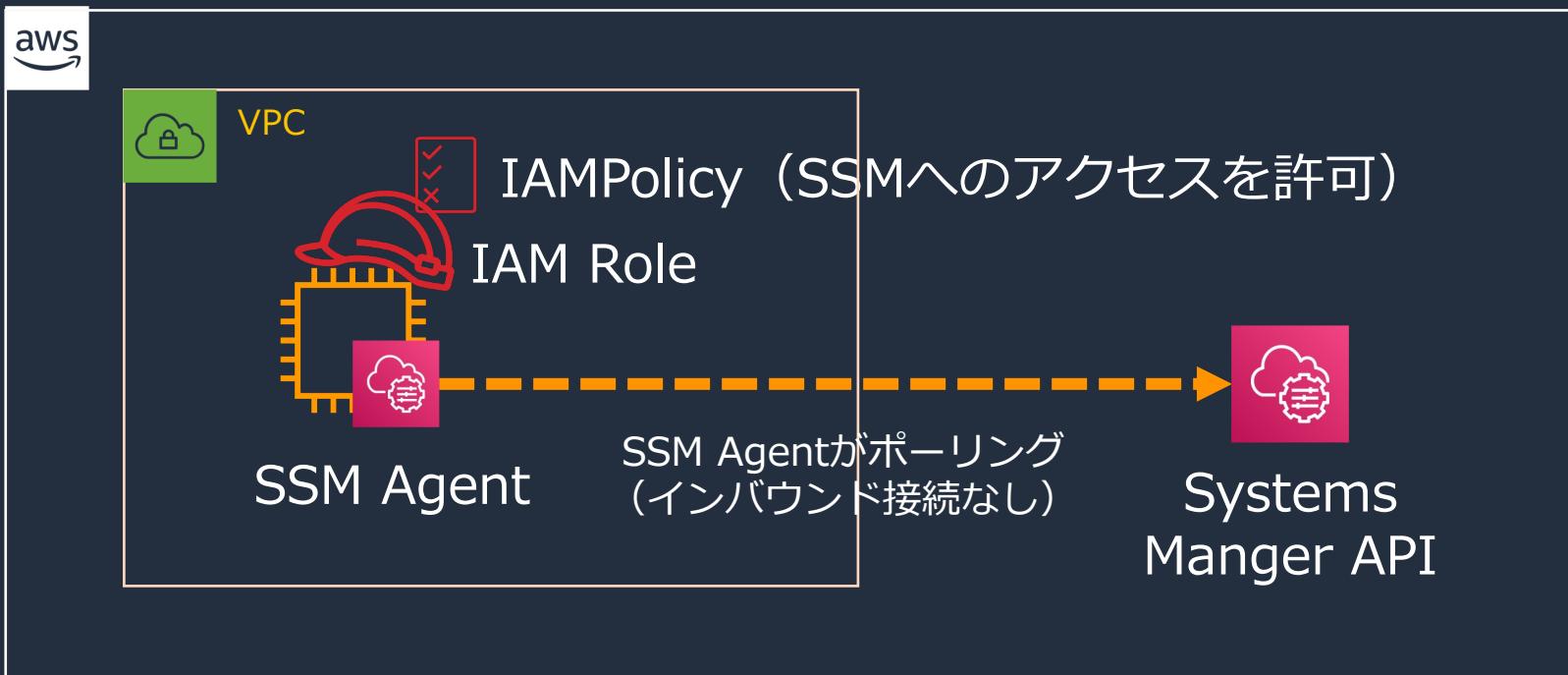
1. コマンドラインで実行（複数サーバーに行うには工夫が必要）→[コマンドラインを使用した CloudWatch エージェントのインストール](#)
2. SSMを活用（SSMなら多数のサーバーにも簡単導入）→[AWS Systems Manager を使用した CloudWatch エージェントのインストール](#)
3. IaCで（CFnで併せてAgentも導入）→[AWS CloudFormationを使用してCloudWatch エージェントのインストール](#)

※1 https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/CWL_GettingStarted.html

Systems Manager を活用した 統合CloudWatchエージェントインストール方法



手順1: Systems Manager Agent の導入



- AWSオフィシャルイメージには導入済みのため、この手順は不要 ※1
- 幅広い対応OS・バージョン一覧は以下URLを参照

<https://docs.aws.amazon.com/systems-manager/latest/userguide/prereqs-operating-systems.html>

–Linux (RHELなど商用含む) /macOS/Windows Serverなど多数

※1 2023年3月29日時点のSSM AgentがプリインストールされたAMI

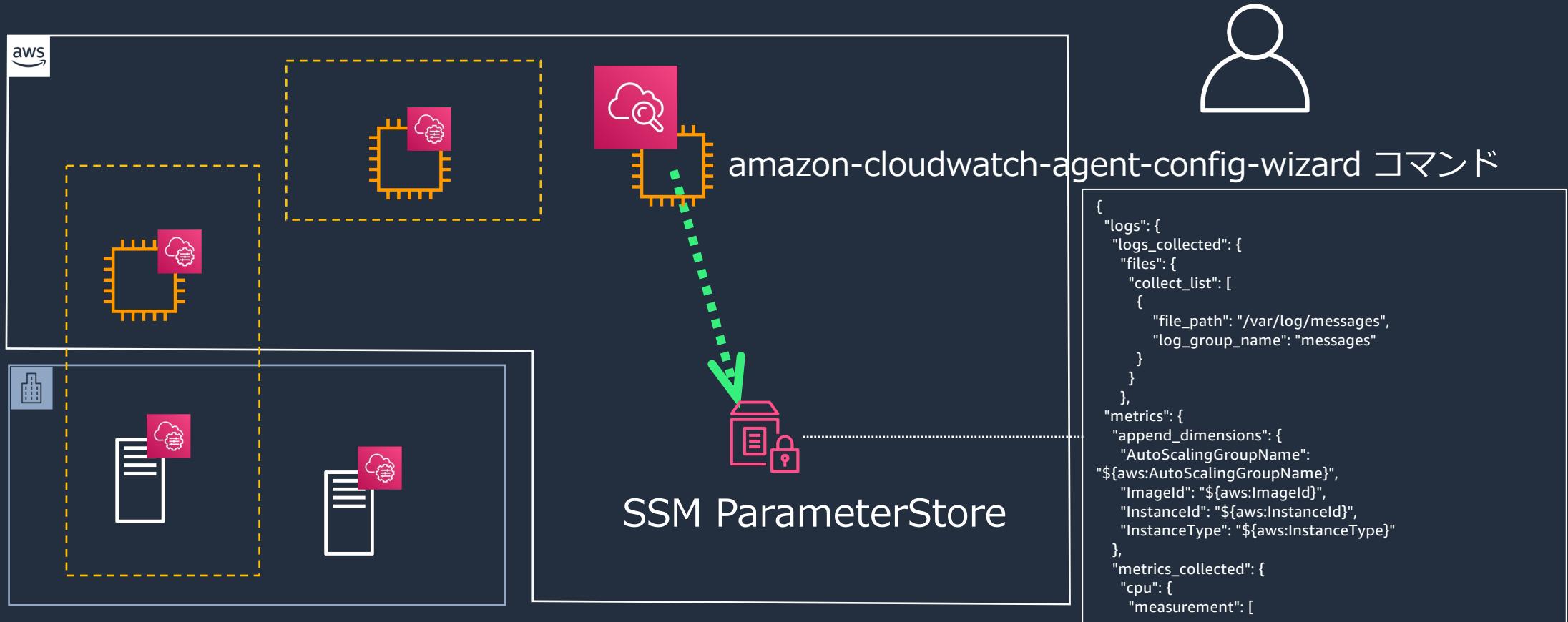
- 2017年9月以降のAmazon Linux Base AMI
- Amazon Linux 2
- Amazon Linux 2 ECSに最適化されたベース AMIs
- Amazon EKS 最適化 Amazon Linux AMIs
- macOS 10.14.x (Mojave)、10.15.x (Catalina)、11.x (Big Sur)
- SUSE Linux Enterprise Server(SLES) 12と15
- Ubuntu Server 16.04、18.04、および 20.04
- 2016年11月以降に公開されたWindows Server 2008-2012 R2 AMIs
- Windows Server 2016、2019、および 2022

https://docs.aws.amazon.com/ja_jp/systems-manager/latest/userguide/ami-preinstalled-agent.html

手順2. 統合CloudWatch Agent の一括セットアップ (1/3)

1. CloudWatch Agent の設定ファイルを生成し SSM ParameterStore へ格納

- amazon-cloudwatch-agent-config-wizard コマンドを実行



参考ドキュメント : CloudWatch エージェント設定ファイルを作成する

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/create-cloudwatch-agent-configuration-file.html



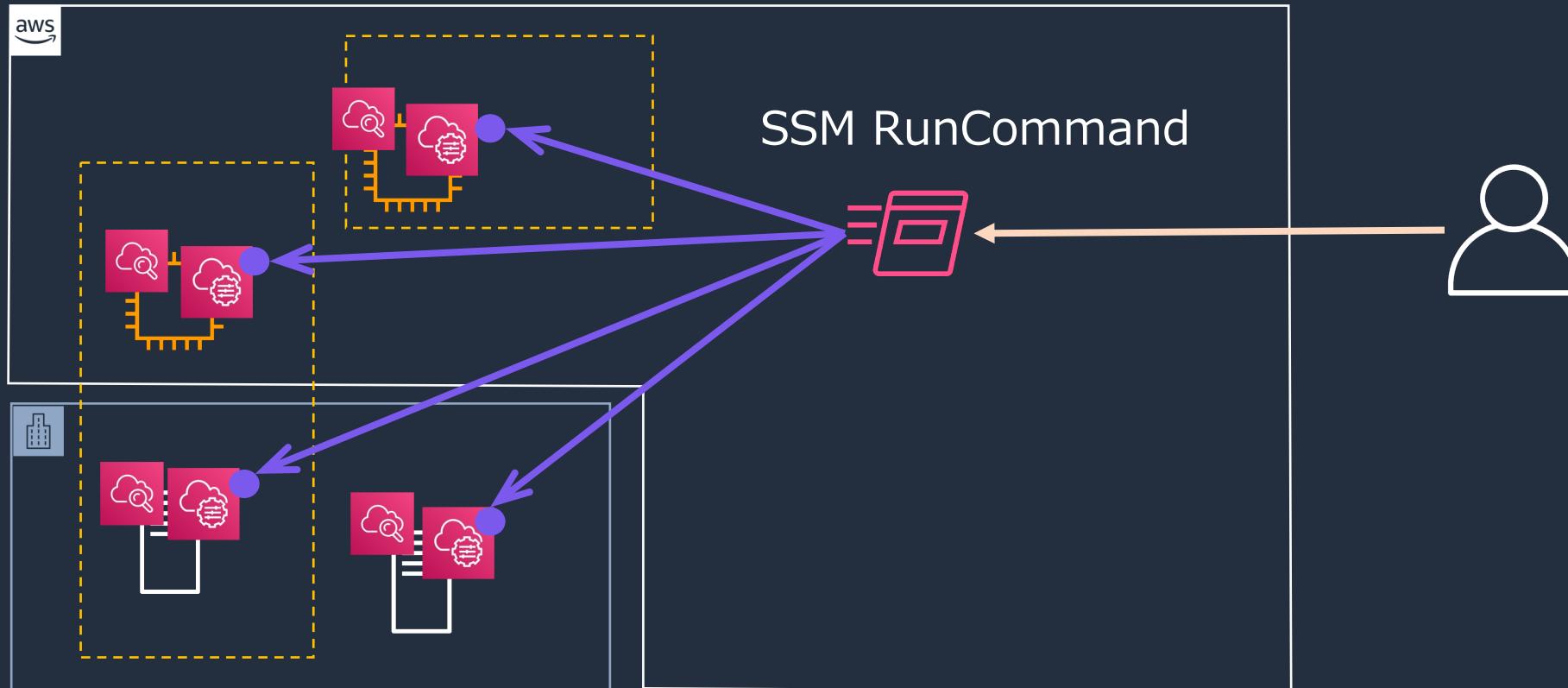
SSM Agent



CW Agent

手順2. 統合CloudWatch Agent の一括セットアップ (2/3)

2. RunCommand で 統合CloudWatch Agentを一括インストール
 - "AWS-ConfigureAWSPackage" ドキュメントを実行



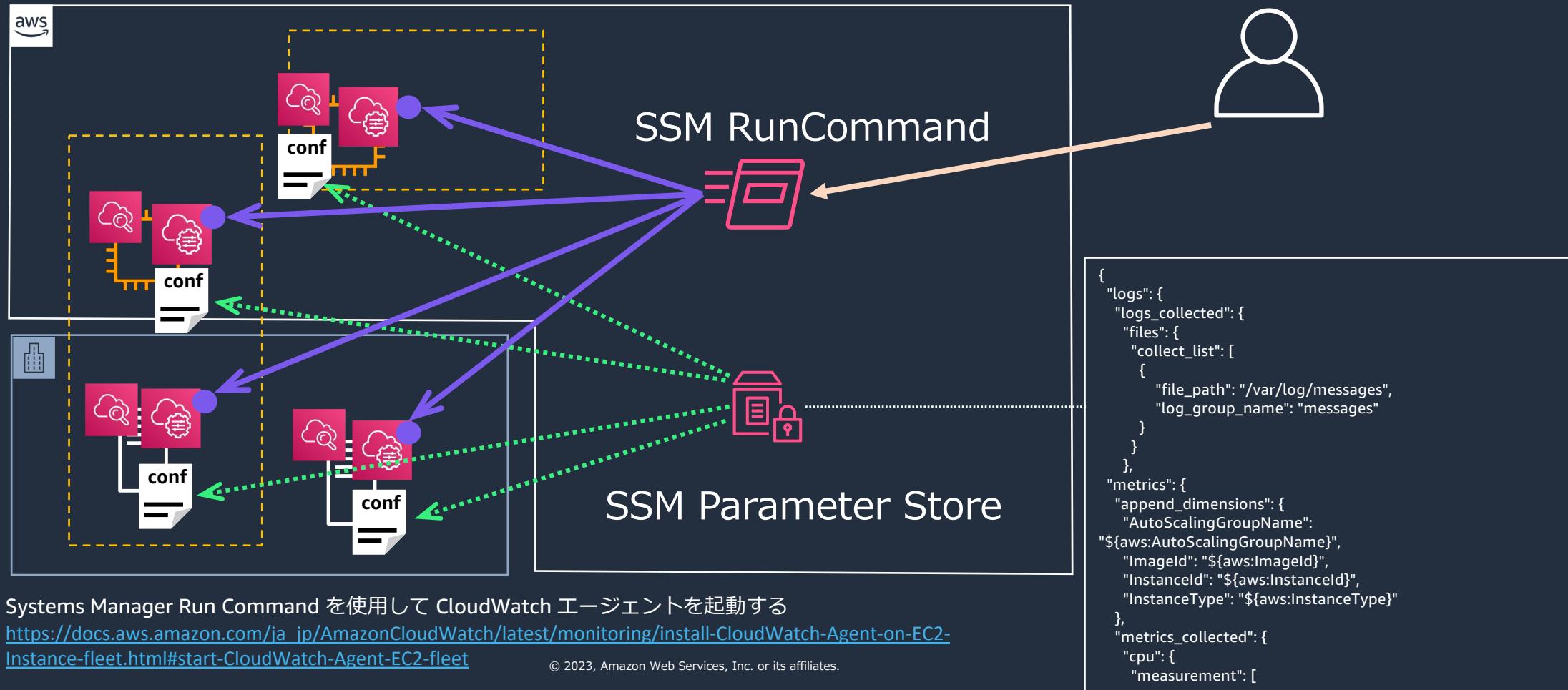
参考ドキュメント : CloudWatch エージェントパッケージをダウンロードする

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/install-CloudWatch-Agent-on-EC2-Instance-fleet.html#download-CloudWatch-Agent-on-EC2-Instance-fleet



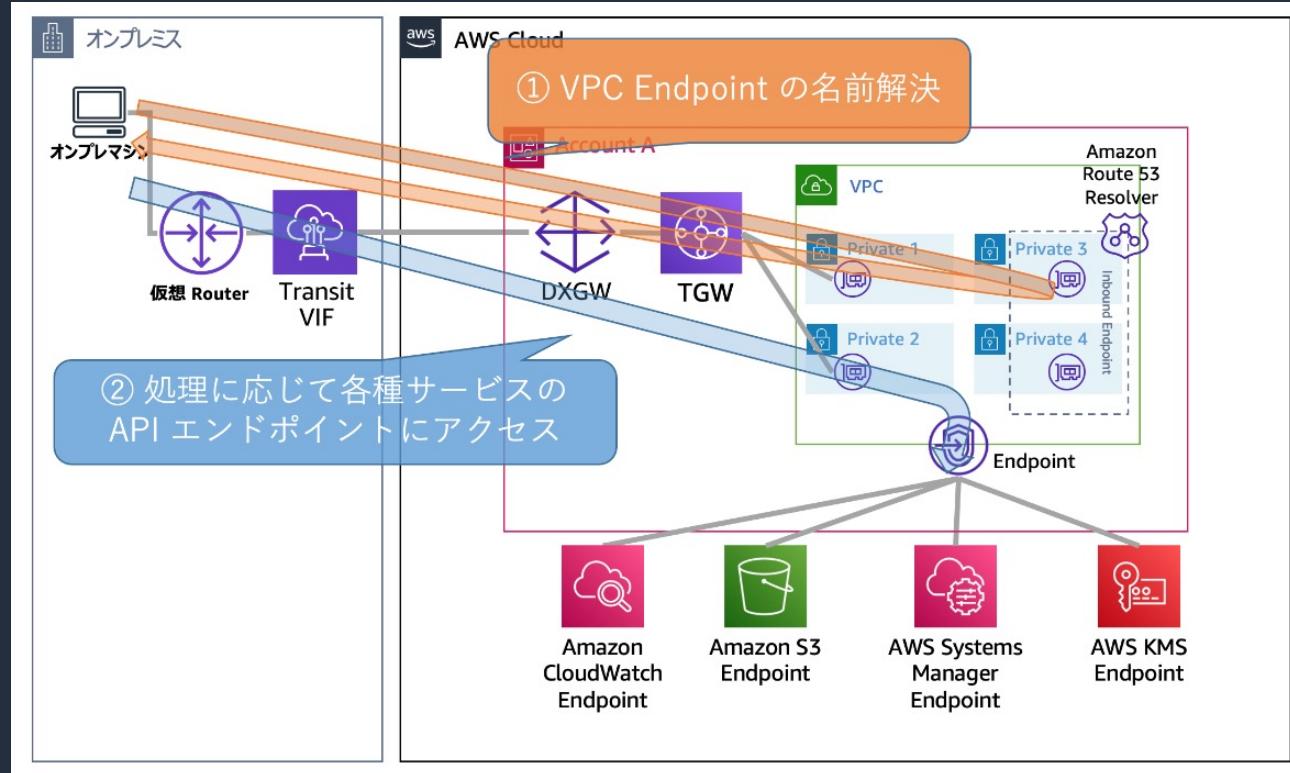
手順3. 統合CloudWatch Agentの一括セットアップ (3/3)

3. RunCommand で Parameter Storeの設定を一括でロード
 - "AmazonCloudWatch-ManageAgent" ドキュメントを実行



ハイブリッド環境の運用・監視

- オンプレミスと閉域網接続の場合にもCloudWatchを活用



- オンプレミスサーバ上の AWS Systems Manager Agent (SSM Agent) と Amazon CloudWatch Agent が通信するサービスエンドポイントへの経路を VPC endpoint で構成

- オンプレミスサーバ上の SSM Agent と CloudWatch Agent が VPC endpoint の DNS 名を解決できるように Amazon Route 53 Resolver を構成

※このブログではオンプレミスと AWS を専用線で接続し、AWS Transit Gateway を経由させた構成ですが、VGW経由など他の構成でも問題ありません。

参考Blog : ハイブリッド環境の運用・監視の実現 – 閉域網で AWS Systems Manager と Amazon CloudWatch を構成する
<https://aws.amazon.com/jp/blogs/news/operation-of-a-hybrid-environment/>

Tips : CloudWatchのVPC Endpoint

利用するCloudWatchのサービス毎に使い分ける

AWS Service	Service name	
Amazon CloudWatch	com.amazonaws.region.monitoring	CloudWatch
	com.amazonaws.region.evidently	CloudWatch Evidentlyのサービス通信
	com.amazonaws.region.evidently-dataplane	CloudWatch Evidentlyの管理通信
	com.amazonaws.region.rum	CloudWatch RUMのサービス通信
	com.amazonaws.region.rum-dataplane	CloudWatch RUMの管理通信
	com.amazonaws.region.synthetics	CloudWatch Synthetics
Amazon EventBridge Amazon CloudWatch Events	com.amazonaws.region.events	AWS EventBridge ※CloudWatch Eventsと共通
Amazon CloudWatch Logs	com.amazonaws.region.logs	CloudWatch Logs

★SSMで統合CloudWatch Agentを使うモニタリングをVPC Endpoint経由で設定する場合、必要なVPC Endpointの例

- com.amazonaws.**region**.ssm
- com.amazonaws.**region**.ssmmessages
- com.amazonaws.**region**.ec2messages
- com.amazonaws.**region**.kms
- com.amazonaws.**region**.s3
- com.amazonaws.**region**.logs
- com.amazonaws.**region**.monitoring

参考 : AWS services that integrate with AWS PrivateLink

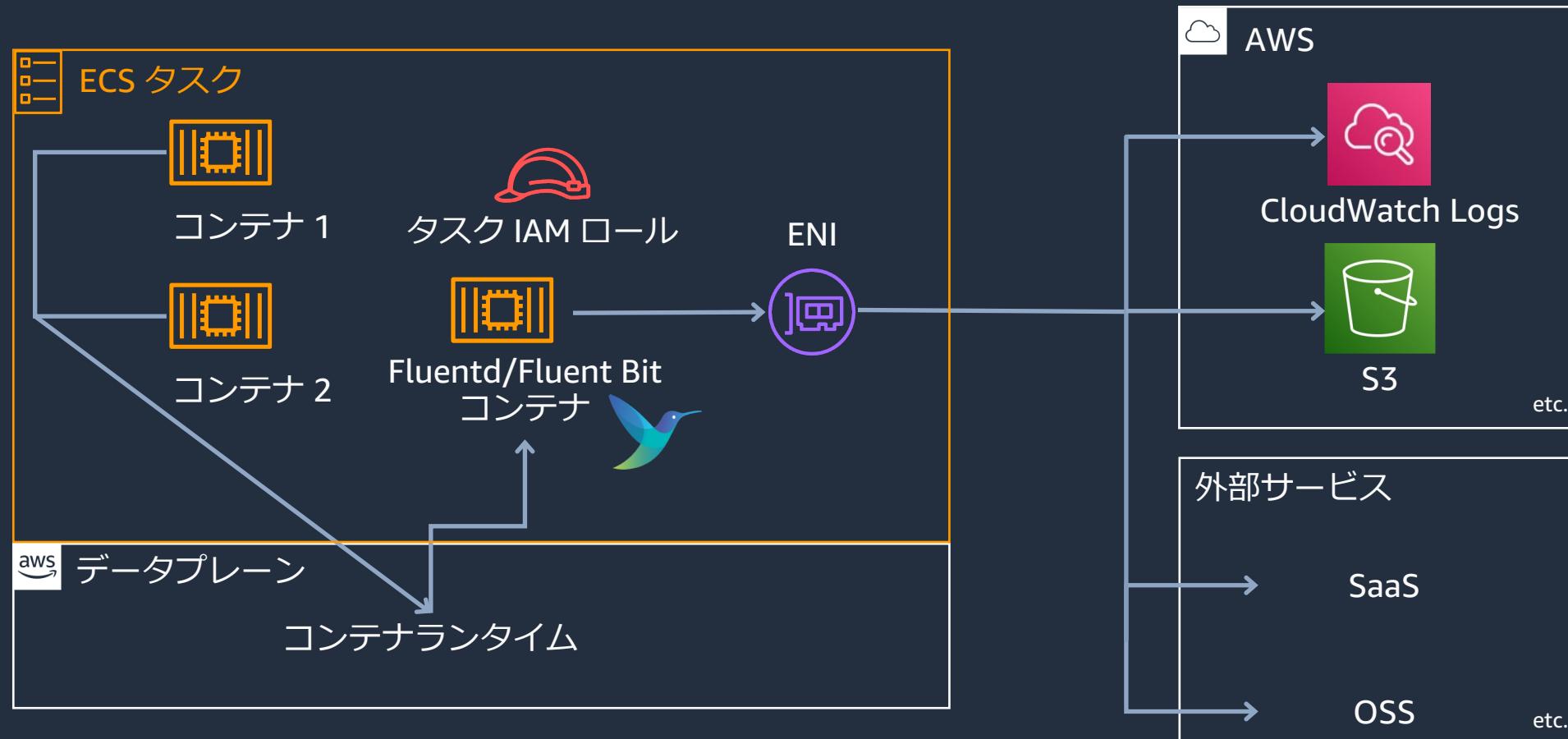
<https://docs.aws.amazon.com/vpc/latest/privatelink/aws-services-privatelink-support.html>



- ・ コンテナ化されたアプリケーションログを **CloudWatch Logs** に送信
 - STDOUT および STDERR をログイベントとして取得
- ・ タスク定義に logConfiguration パラメータを追加するだけで有効化が可能
 - 追加のエージェントなどのインストールは不要
 - **ECS on EC2, ECS on Fargate** のいずれでも利用可能



Fluentd / Fluent Bit を利用した柔軟なログルーティング



https://docs.aws.amazon.com/AmazonECS/latest/developerguide/using_firelens.html

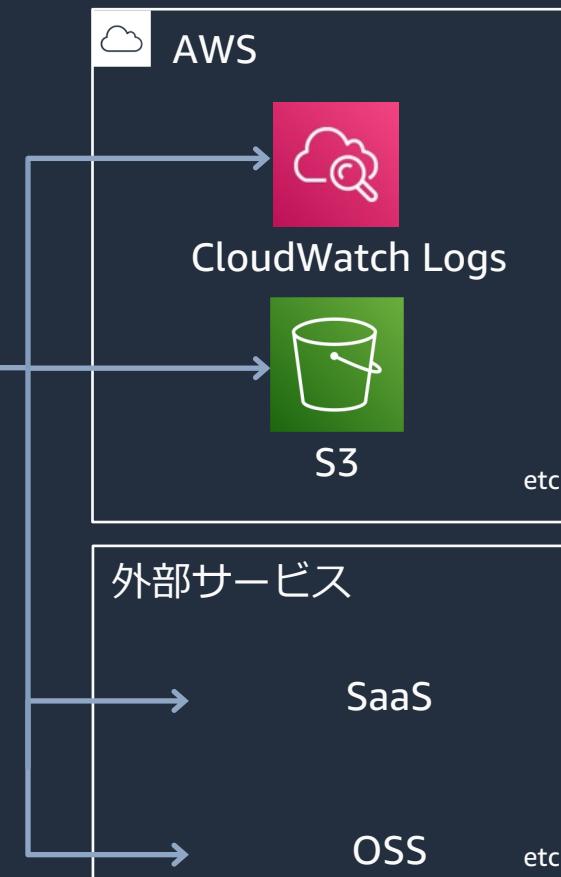
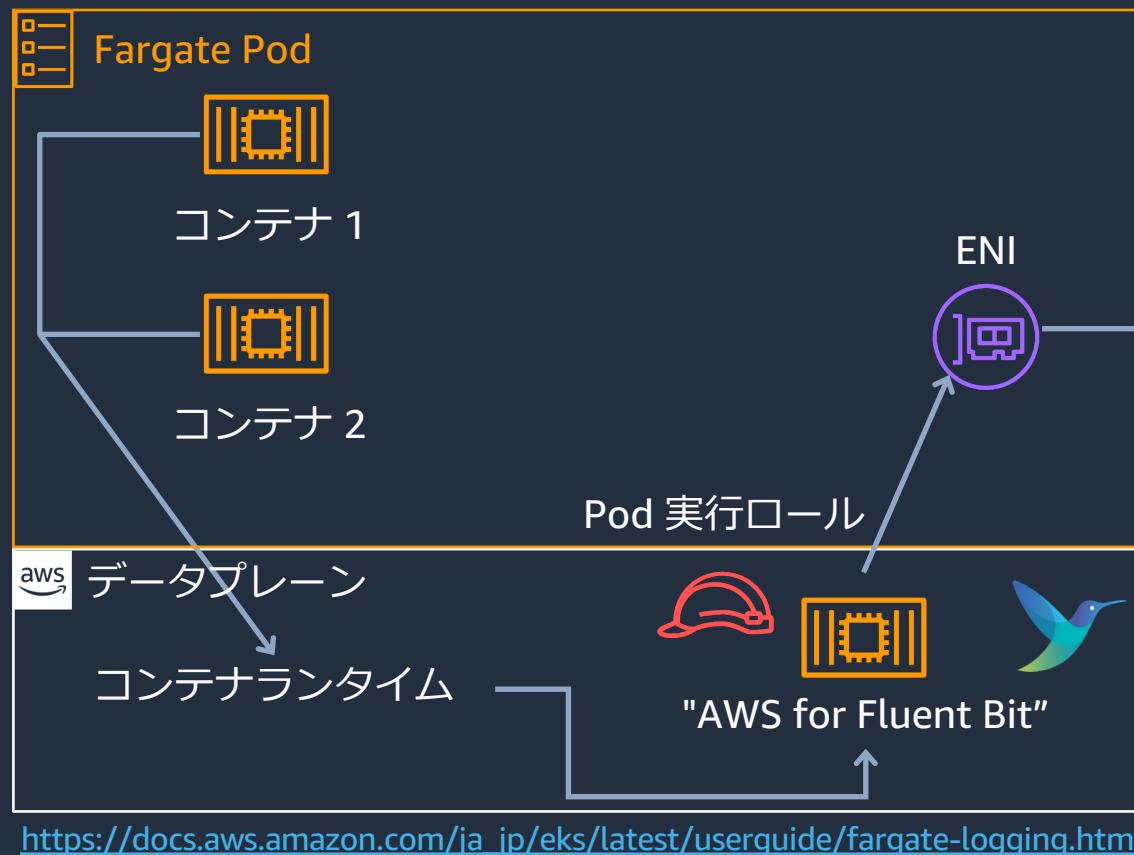
© 2021, Amazon Web Services, Inc. or its Affiliates.



Amazon EKS on Fargate 対応 FireLens

CloudWatch Logsに
EKS on Fargateからログを送る

Fluent Bit をベースにした組み込みのログルーター



※コントロールプレーンのログはEKSで有効化することで取得できる。

https://docs.aws.amazon.com/ja_jp/prescriptive-guidance/latest/implementing-logging-monitoring-cloudwatch/kubernetes-eks-logging.htm

EKS on EC2の場合はFluentBitないしはFluentdをNodeにDaemonSetとして配置。!

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/Container-Insights-setup-logs-FluentBit.html

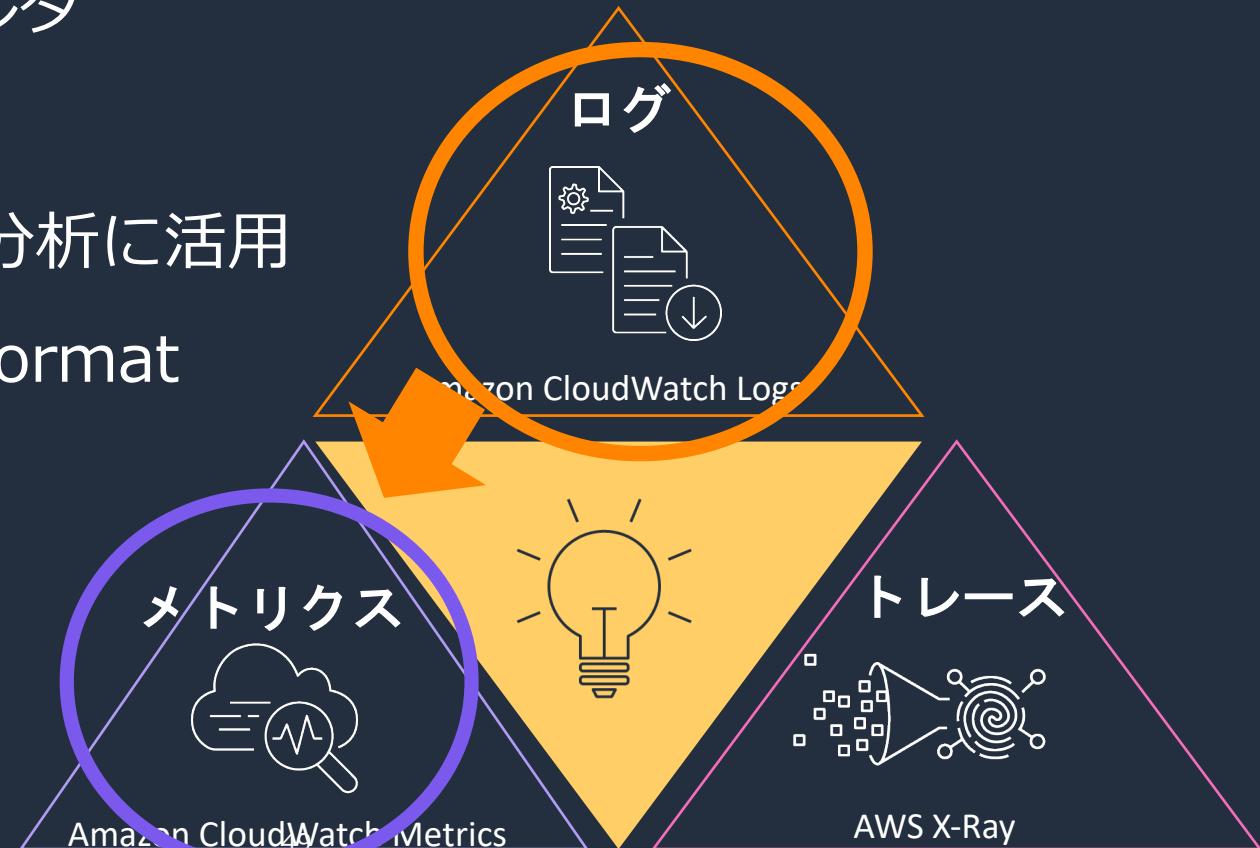
ログからメトリクスを抽出して通知や分析に活用も

① CloudWatch Logsに集めた後、ロググループ単位で分析し、通知に活用

→ CloudWatch Logs メトリクスフィルタ

② ログ出力時にメトリクスを埋め込み、分析に活用

→ CloudWatch Embedded Metrics Format



CloudWatch Logs メトリクスフィルタ

CloudWatch Logsのログをフィルタしてメトリクスにパブリッシュ

- ・ ログデータから特定の文字列のフィルタリングが可能
 - ・ フィルターとパターンの構文で指定できる範囲内で文字列をフィルターが可能
 - ・ 特定文字列のエントリ頻度等によりアラーム作成、SNS連携が可能

CloudWatch > ロググループ > /aws/lambda/connect-mtsuwaza20210331-1-initContactDetails-5DJGASSIKAGC > メトリクスフィルターを作成

Step 1
パターンを定義

Step 2
メトリクスの割り当て

Step 3
Review and create

パターンを定義

フィルターパターンを作成

メトリクスフィルターを使用し、ロググループ内のイベントが CloudWatch Logs に送信されるときに、セーリングできます。特定の用語のモニタリングやカウントを行ったり、ログイベントから値を抽出したりするに関連付けることができます。 [パターン構文の詳細については、こちらをご参照ください。](#)

フィルターパターン

メトリクスを作成するためのログイベントに対して一致する用語やパターンを指定します。

フィルターパターンを入力

用語を含むログイベントの照合:

ERROR

[INFO]

"Warning:"

ログイベントと 400 レベルの HTTP 応答との照合:

[host, logName, user, timestamp, request, statusCode=4*, size]

JSON ログイベントとの照合:

{ \$.errorCode = "AccessDenied" }

メトリクスの詳細

メトリクス名前空間

Namespace lets you group similar metrics. [詳細はこちら](#)

メトリクス名前空間を入力

新規作成

名前空間の長さは最大 255 文字です。コロン (:)、アスタリスク (*)、ドル (\$)、スペース () を除くすべての文字が使用できます。

メトリクス名

メトリクス名はこのメトリクスを識別し、名前空間内で一意である必要があります。 [詳細はこちら](#)

メトリクス名を入力

メトリクス名の長さは最大 255 文字です。コロン (:)、アスタリスク (*)、ドル (\$)、スペース () を除くすべての文字が使用できます。

メトリクス値

メトリクス値は、フィルターパターンの一数が発生したときにメトリクスの名前にパブリッシュされる値です。

メトリクス値を入力

有効なメトリクス値は、浮動小数点数 (1、99.9 など)、数値フィールド識別子 (\$1、\$2 など)、または名前付きフィールド識別子です (区切りフィルターパターンの場合は \$requestSize、JSON ベースのフィルターパターンの場合は \$.status(\$)- ドル (\$) またはドルドット (\$) に、英数字やアンダースコア (_) が続きます)。

デフォルト値 - オプション

デフォルト値は、パターンが一致しないときにメトリクスに発行されます。この値を空白のままにすると、一致がないときの値は発行されません。 [詳細はこちら](#)

デフォルト値を入力

Unit - オプション

Select a unit

- 定義したパターンに一致した時に
- メトリクスにパブリッシュされる値を設定 (ERRORが含まれた単純なパターン数を求める時はメトリクス値 1 を指定)
- 一致したログをカウントした値をメトリクスにできる

CloudWatch Embedded Metrics Format (EMF)

詳細なログイベントと一緒にカスタムメトリクスを埋め込む

EMF の例

- 複雑なアプリケーションデータをログの形式で取り込み、実用的なメトリクスを生成
- CloudWatch はカスタムメトリクスをログから自動的に抽出
- クライアントライブラリを使用してEMF形式のログを生成、もしくは手動で生成した EMF 形式のログを PutLogEvents API や CloudWatch エージェントで送信
- ADOTにもCloudWatch Embedded Metric Format(EMF) Exporterがあり、対応している。

```
{  
  "_aws": {  
    "Timestamp": 1574109732004,  
    "CloudWatchMetrics": [  
      {  
        "Namespace": "lambda-function-metrics",  
        "Dimensions": [{"functionVersion"}],  
        "Metrics": [  
          {  
            "Name": "time",  
            "Unit": "Milliseconds",  
            "StorageResolution": 60  
          }  
        ]  
      },  
      "functionVersion": "$LATEST",  
      "time": 100,  
      "requestId": "989ffbf8-9ace-4817-a57c-e4dd734019ee"  
    ]  
  }  
}
```

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/monitoring/CloudWatch_EMBEDDED_Metric_Format.html
<https://aws-otel.github.io/docs/getting-started/cloudwatch-metrics#cloudwatch-emf-exporter-awsemf>



Amazon S3 へのログデータのエクスポート

コンソールかAWS CLIにより利用

コンソールの使用

- エクスポートするデータの期間
- S3バケットの指定
- バケットプレフィックスの指定

AWS CLIを使用したエクスポート

エクスポートタスクの作成

```
aws logs create-export-task --task-name "my-Log-group-09-10-2015" --log-group-name "my-Log-group" --from 1441490400000 --to 1441494000000 --destination "my-exported-Logs" --destination-prefix "export-task-output"
```

タスクの作成ステータス確認

```
aws logs describe-export-tasks --task-id "cda45419-90ea-4db5-9833-aade86253e66"
```

※オプションでストリームプレフィックスを指定すると、指定されたプレフィックスに一致するログストリームのみをエクスポートします。

ログを長期保管したい！
他サービスで活用したい！



※ログデータは、エクスポートできるようになるまで最大 12 時間かかる場合があります

CloudWatch Logs サブスクリプションフィルタ

CloudWatch Logsに集めたログをフィルタパターンに応じて外部にリアルタイムに転送

CloudWatch Logsに集めたログをフィルタパターンに応じてリアルタイムに
Kinesis Data Streams/Kinesis Data Firehose/Lambdaへ転送

- 1つのロググループにつき、2つのサブスクリプションフィルタを設定可能



CloudWatch Logs



サブスクリプション
フィルタ



AWS Lambda



Amazon Kinesis Data
Streams



Amazon Kinesis Data
Firehose

AWS CLIからのみ設定可能

```
aws logs put-subscription-filter ¥
  --log-group-name "xxxxxxxx" ¥
  --filter-name "xxxxxxxx" ¥
  --filter-pattern "{xxxxxxxx = xxxxxxxx}" ¥
  --destination-arn "arn:aws:kinesis:ap-northeast-1:123456789012:stream/xxxxxxxx" ¥
  --role-arn "arn:aws:iam::123456789012:role/xxxxxxxx"
```



ユースケース：サブスクリプションフィルタの利用例

Lambdaを活用したOpenSearch Serviceへのストリーミング



Kinesis Data Firehoseを活用したS3等へのデータ転送



Kinesis Data Firehoseのカスタムプレフィックスによりタイムスタンプ情報をApache Hiveフォーマットに設定可能

myPrefix/year={!timestamp:yyyy}/month={!timestamp:MM}/day={!timestamp:dd}/hour={!timestamp:HH}/

パスの例
myPrefix/year=2018/month=07/day=06/hour=23/



https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/CWL_OpenSearch_Stream.html

https://docs.aws.amazon.com/ja_jp/AmazonCloudWatch/latest/logs/Subscriptions.html

https://docs.aws.amazon.com/ja_jp/firehose/latest/dev/s3-prefixes.html

© 2023, Amazon Web Services, Inc. or its affiliates.

CloudWatch Metric streams

継続的なメトリクスのニアリアルタイムストリーム

- CloudWatch メトリクスを選択した宛先に継続的にストリーミングし、ニアリアルタイムで配信
 - 送信先: AWS のサービス (Amazon Kinesis Data Firehose など) またはサードパーティのサービス プロバイダー
 - ユースケース: Amazon S3等 でデータレイクを構築するか、サードパーティのツールを使用してメトリクスを監視する



例: metrics を Amazon S3 bucketへ

Your metric stream **MyMetrics-US-East-1A** was successfully created.

The following resources have been created: [S3 bucket](#), [S3 write role](#), [Firehose](#), [Firehose write role](#)

[View details](#) [X](#)

CloudWatch > Metric Streams

Metric streams [Monitoring](#)

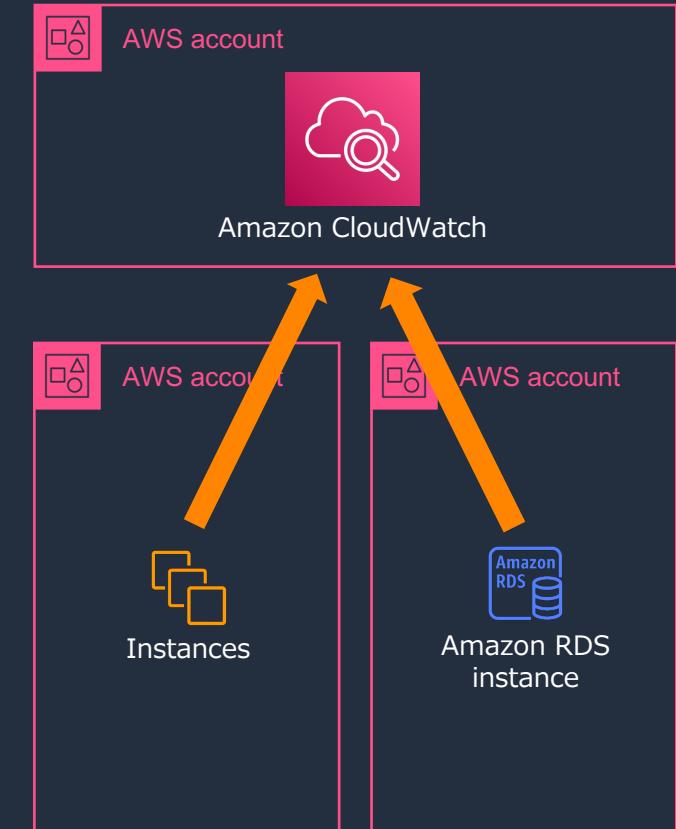
Metric streams (1) Info		Create metric stream	
<input type="text" value="Find by name"/>		C Stop View details Edit Delete	
Name	Status	Last Updated Time	Destination
MyMetrics-US-East-1A	Running	03/23/2021 5:48 PM	S3

S3バケットのファイル例

```
new 26 ms.yaml MetricStreams-MyMetrics-US-East-1A:1YvyRge-1-2021-03-24-00-51-08-2e9e2a5f3b8a-4fb-bbd4-2200f744925
1 {"metric_stream_name": "MyMetrics-US-East-1A", "account_id": "██████████", "region": "us"
ame": "WLMQueueLength", "dimensions": {"ClusterIdentifier": "test-cluster", "service
class": "100"}, "timestamp": 1616546940000, "value": {"count": 1.0, "sum": 0.0, "max": 0.0, "min
2 {"metric_stream_name": "MyMetrics-US-East-1A", "account_id": "██████████", "region": "us"
ame": "WriteThroughput", "dimensions": {"ClusterIdentifier": "test-cluster", "NodeID": "Com
count": 1.0, "sum": 0.0, "max": 0.0, "min": 0.0}, "unit": "Bytes/Second"}
3 {"metric_stream_name": "MyMetrics-US-East-1A", "account_id": "██████████", "region": "us"
ame": "NetworkReceiveThroughput", "dimensions": {"ClusterIdentifier": "test-cluster", "Node
value": {"count": 1.0, "sum": 7062.616666666667, "max": 7062.616666666667, "min": 7062.616666
4 {"metric_stream_name": "MyMetrics-US-East-1A", "account_id": "██████████", "region": "us"
ame": "NetworkTransmitThroughput", "dimensions": {"EngineName": "mariadb"}, "timestamp": 1616546
662039, "max": 2583.340277662039, "min": 2583.340277662039}, "unit": "Bytes/Second"}
5 {"metric_stream_name": "MyMetrics-US-East-1A", "account_id": "██████████", "region": "us"
ame": "ReadLatency", "dimensions": {"ClusterIdentifier": "prod-cluster", "NodeID": "Compute
t": 1.0, "sum": 0.0, "max": 0.0, "min": 0.0}, "unit": "Seconds"}
6 {"metric_stream_name": "MyMetrics-US-East-1A", "account_id": "██████████", "region": "us"
ame": "WriteLatency", "dimensions": {"ClusterIdentifier": "prod-cluster", "NodeID": "Compute
t": 1.0, "sum": 0.0, "max": 0.0, "min": 0.0}, "unit": "Seconds"}
```

Amazon CloudWatchでアカウント横断の監視が可能に

- Amazon CloudWatchで複数のアカウントを横断的にモニタリングできるようになった
 - メトリクス、ログ、AWS X-Rayによるトレースなどの可視化や分析をアカウント境界を気にせずに一元的に実現
 - 例えばアカウントを跨ってAWS Lambdaを介して連携するアプリケーションのエンドツーエンドでのトレースが容易に
 - 単一リージョンで展開される複数アカウント環境の監視と障害対応を支援
- メトリクスとログは追加料金なし、トレースは1つの監視アカウントでの利用は追加料金なしで提供
- 全商用リージョンで利用可能



新着 – Amazon CloudWatch のクロスアカウントオブザーバビリティ

<https://aws.amazon.com/jp/blogs/aws/new-amazon-cloudwatch-cross-account-observability/>

<https://aws.amazon.com/jp/blogs/news/new-amazon-cloudwatch-cross-account-observability/>

再掲：CloudWatchの機能一覧

Infrastructure

AWSサービス名	概要
CloudWatch Metrics	メトリクス
CloudWatch Log	ログ
CloudWatch Alarm	アラーム
CloudWatch Dashboard	ダッシュボード
CloudWatch Metrics Explorer	メトリックス検索
CloudWatch Metrics Stream	メトリックスのリアルタイム連携
CloudWatch Events ※これはEvent Bridgeに統合されています	イベント
CloudWatch Resource Health	EC2の健全性・パフォーマンス可視化

Application Monitoring

AWSサービス名	概要
CloudWatch Synthetics	外形監視
CloudWatch RUM	リアルユーザー モニタリング
CloudWatch Evidently	フィーチャーフラグA/Bテスト
CloudWatch Internet Monitor	インターネット監視
CloudWatch ServiceLens	トレース

Insights

AWSサービス名	概要
CloudWatch Contributor Insights	ログの時系列分析
CloudWatch Container Insights	コンテナ分析
CloudWatch Lambda Insights	Lambda分析
CloudWatch Application Insights	アプリケーション分析
CloudWatch Logs Insights	LOG分析
CloudWatch Metrics Insights	メトリクス分析

グレーアウトはご紹介済み

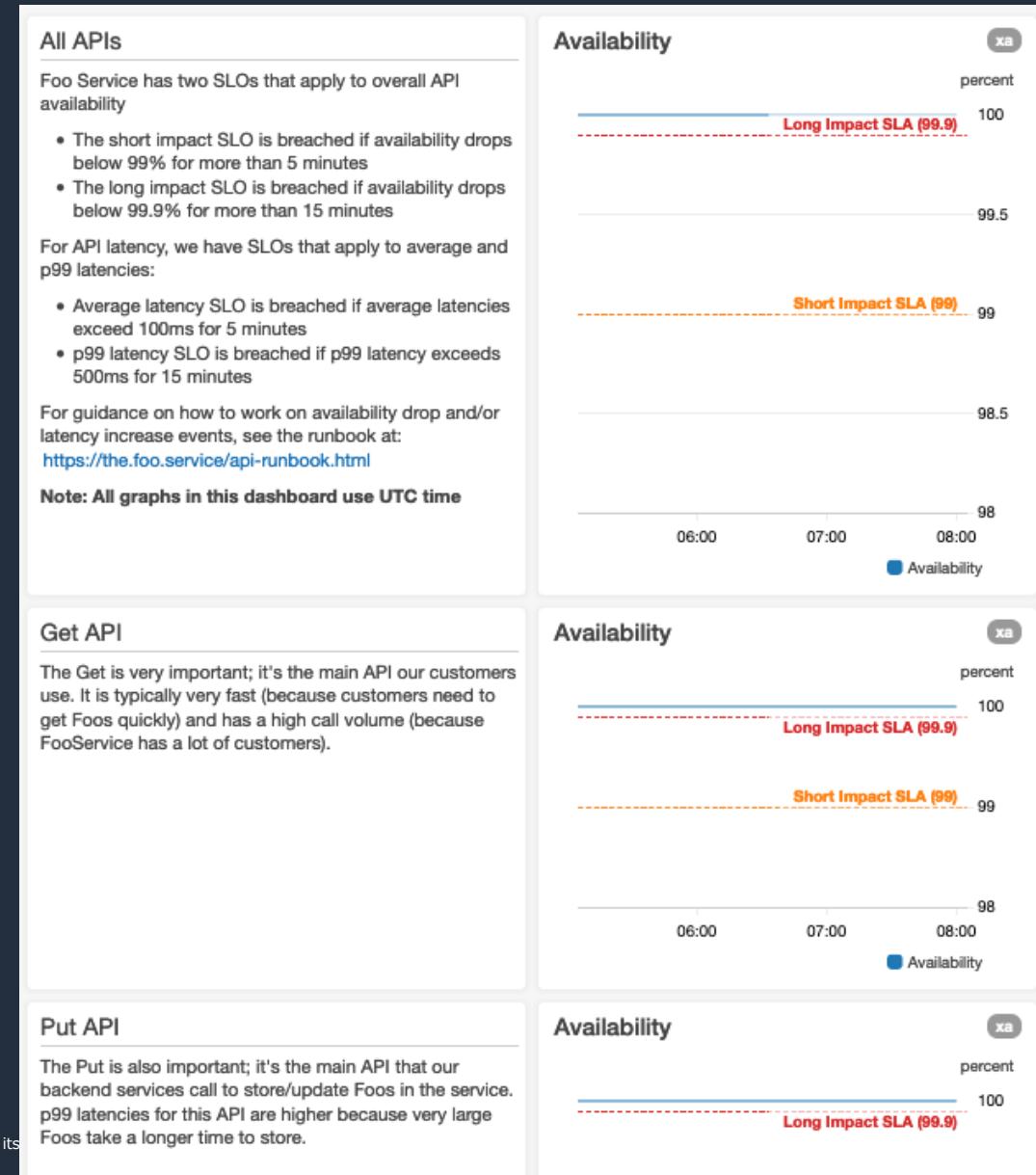
★ここからはその他のサービスを簡単にご紹介
※個別の詳細は別途Blackbeltセッションで紹介予定

CloudWatch Dashboard

カスタマイズ可能なダッシュボードを作成

- 折れ線グラフ、数値、ゲージ、テキスト、アラーム、Lambdaによるカスタムウィジェットでお好みのダッシュボードを作成
 - 異なるアカウント、異なるリージョンのリソースでも、ダッシュボード化が可能
 - 自動更新間隔(10s, 1m, 2m, 5m, 15m)、時間範囲、タイムゾーンの調整が可能

目的別のダッシュボードを構築する



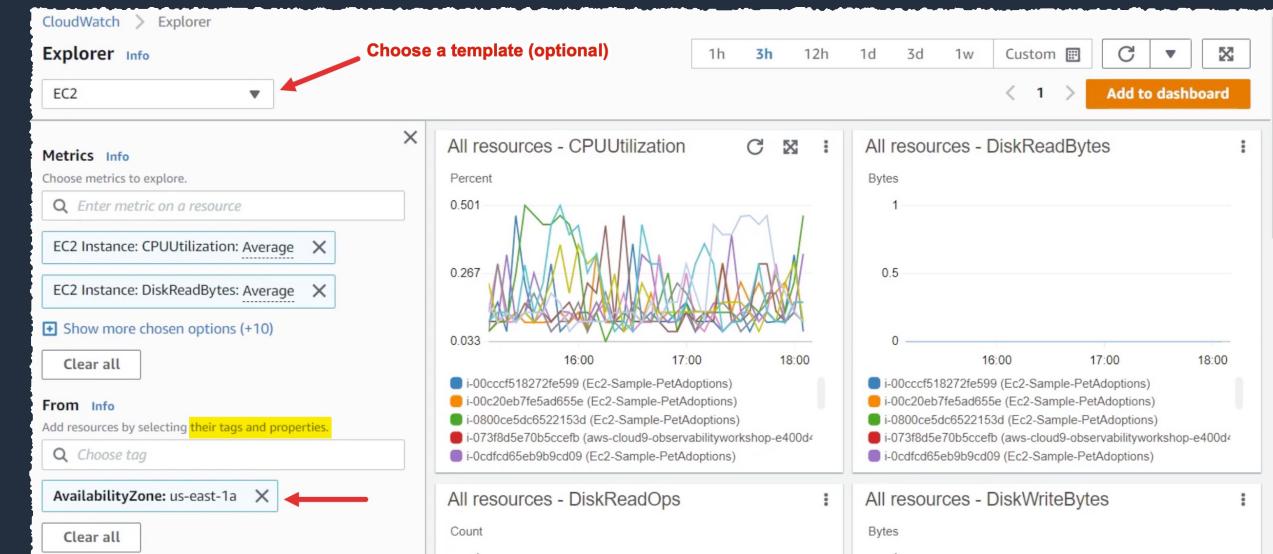
CloudWatch Metrics explorer

タグとプロパティでメトリクスを監視する

- タグとリソース、プロパティによるメトリクスのフィルタリング、集計、および視覚化のためのツール
- 検索条件に当てはまる場合、動的に新しいリソースのメトリクスが追加される
- 付属のテンプレートを使用すると、ワンクリックで便利にグラフを作成可能

メトリクス検索をタグで柔軟に

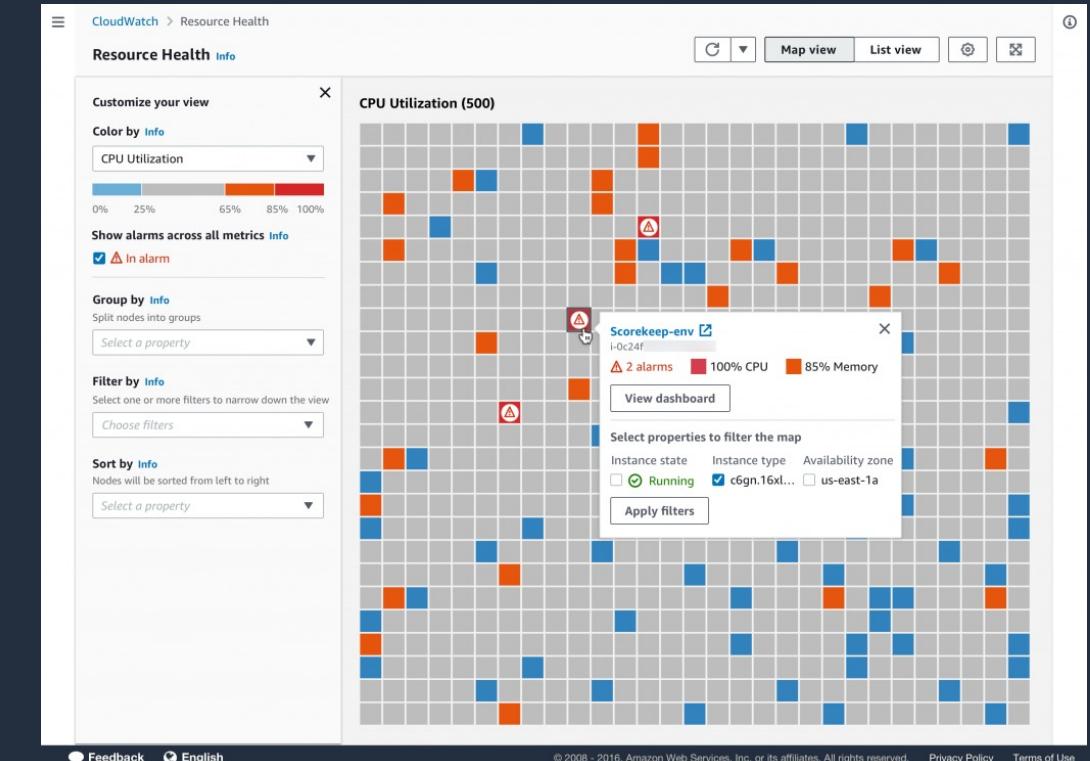
例: us-east-1a アベイラビリティーゾーンのすべてのAmazon EC2 インスタンスのメトリクスを表示します



CloudWatch Resource Health (Amazon EC2)

Amazon EC2 ホストの正常性を自動的に検出して視覚化する

- Amazon EC2 インスタンスの正常性とパフォーマンスを自動的に検出して視覚化するフルマネージドソリューション
- AZ・VPC・AutoScalingGroup・CPUなどキャパシティの状況やタグなどの情報ごとにグループ化/フィルタリング/ソート可能



再掲：CloudWatchの機能一覧

Infrastructure

AWSサービス名	概要
CloudWatch Metrics	メトリクス
CloudWatch Log	ログ
CloudWatch Alarm	アラーム
CloudWatch Dashboard	ダッシュボード
CloudWatch Metrics Explorer	メトリックス検索
CloudWatch Metrics Stream	メトリックスのリアルタイム連携
CloudWatch Events ※これはEvent Bridgeに統合されています	イベント
CloudWatch Resource Health	EC2の健全性・パフォーマンス可視化

Application Monitoring

AWSサービス名	概要
CloudWatch Synthetics	外形監視
CloudWatch RUM	リアルユーザー モニタリング
CloudWatch Evidently	フィーチャーフラグA/Bテスト
CloudWatch Internet Monitor	インターネット監視
CloudWatch ServiceLens	トレース

Insights

AWSサービス名	概要
CloudWatch Contributor Insights	ログの時系列分析
CloudWatch Container Insights	コンテナ分析
CloudWatch Lambda Insights	Lambda分析
CloudWatch Application Insights	アプリケーション分析
CloudWatch Logs Insights	LOG分析
CloudWatch Metrics Insights	メトリクス分析

グレーアウトはご紹介済み

★ここからはその他のサービスを簡単にご紹介
※個別の詳細は別途Blackbeltセッションで紹介予定

Application monitoring

サーバーとクライアントサイドのパフォーマンスマニタリング

Amazon CloudWatch Synthetics (canaries)



- ・アプリケーションのエンドポイントを監視
- ・Blueprintで以下のようなモニタリングが可能
 - ハートビートモニター
 - API Canary
 - リンク切れチェッカー
 - ビジュアルモニタリング
 - Canary Recorder
 - GUI ワークフロー

Amazon CloudWatch Real-user monitoring

- ・Web アプリケーションのパフォーマンスに関するクライアント側のデータをリアルタイムで収集
- ・ユーザー数、位置情報、ブラウザなど、エンドユーザーへのさまざまな影響をモニター



Amazon CloudWatch Evidently

- ・一般的なユーザーに向けてロールアウトする前に、新機能の実験を行う
- ・A/B テストを実施してEvidenceとデータに基づいて機能の設計を決定可能に



CloudWatch Synthetics

Canary を使用し、ユーザートラフィックがない場合でも
24時間365日、ユーザー体験を継続的に監視



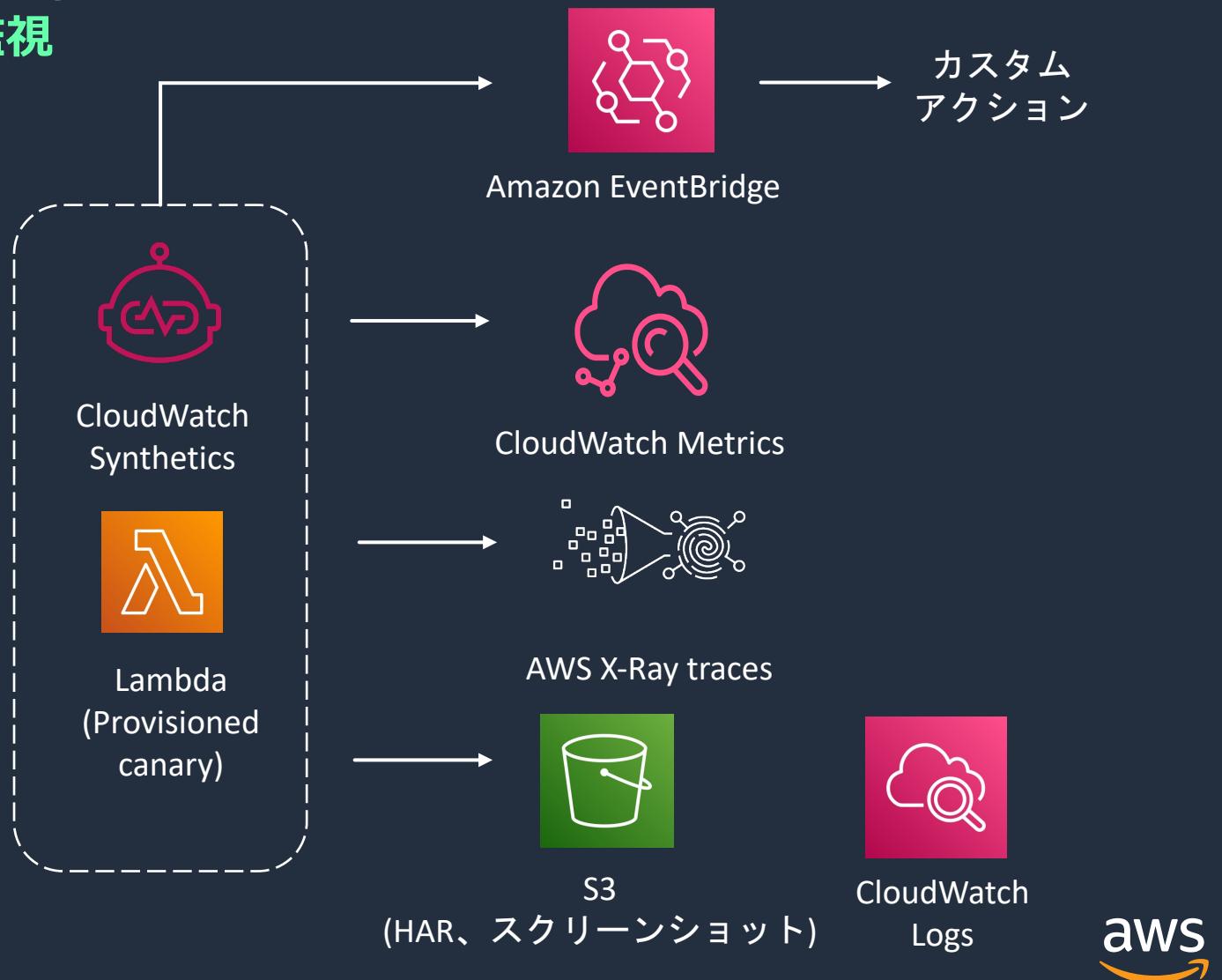
ウェブサイトおよび API
のエンドポイント監視



クライアントサイドか
らサーバーサイドへ



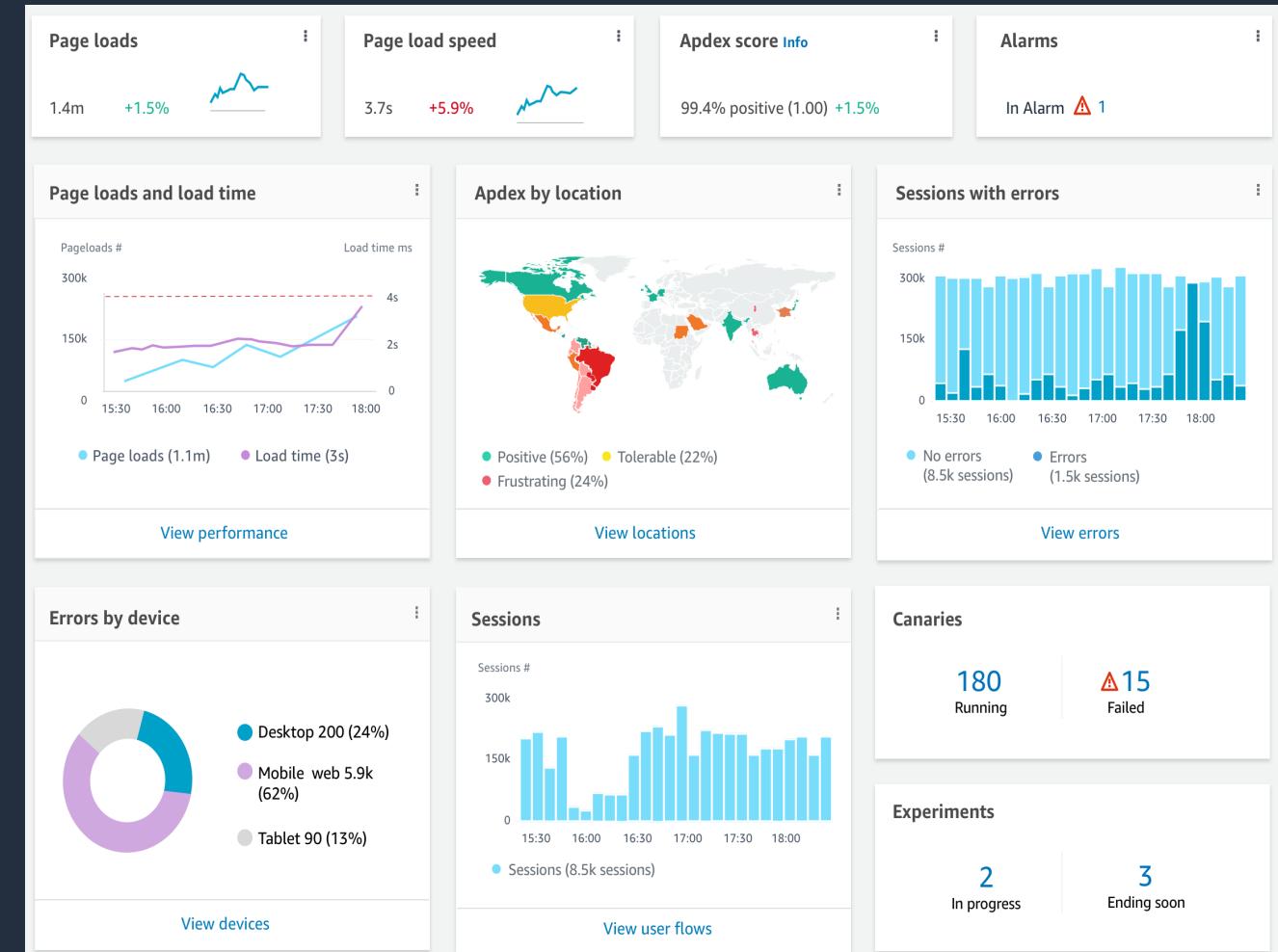
継続的なモニタリング



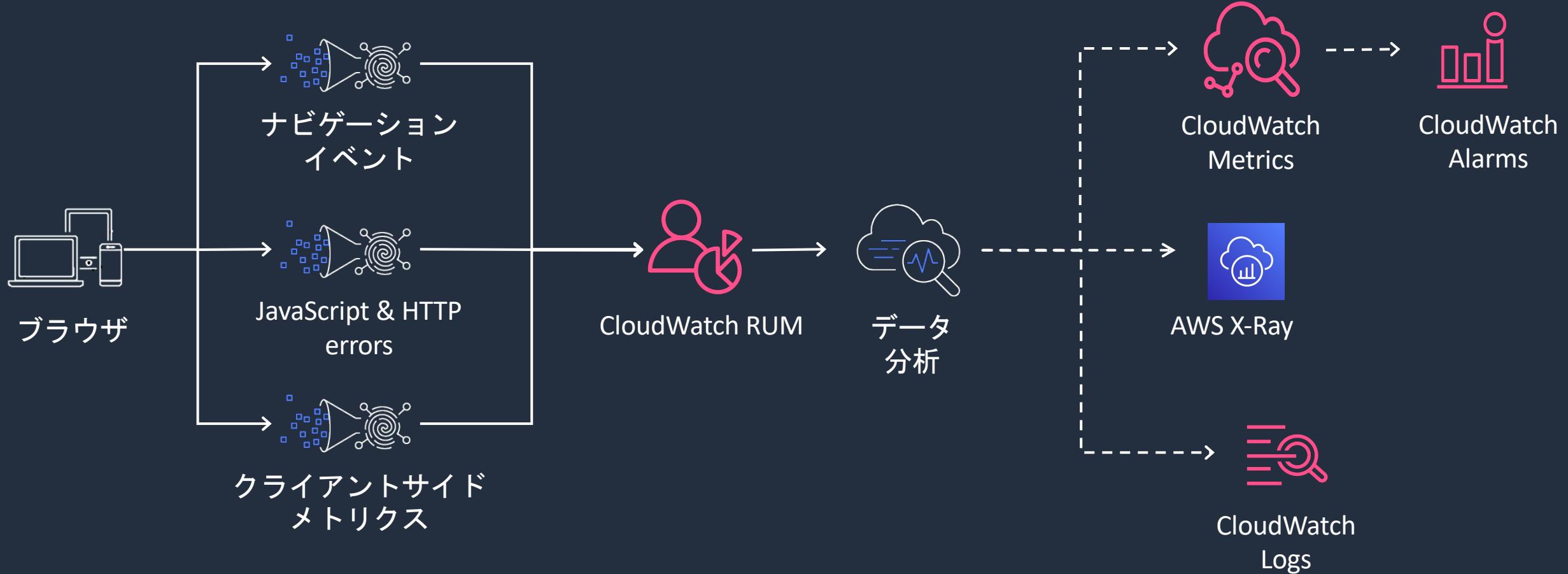
CloudWatch RUM

アプリのパフォーマンスに関するクライアントサイドのデータをリアルタイムで取得し、顧客体験を最適化

- リアルユーザーのパフォーマンスをモニタし、ブラウザやデバイスの種類、位置、ネットワークの接続性の問題などを把握できる
 - ダッシュボードでページの読み込み順序や JavaScript / HTTP レスポンスのエラーなど、パフォーマンス問題に関する情報を可視化
 - 同じ問題の影響下にあるユーザセッション数を提示するため、改修の優先順位を付けることが容易
 - Amazon CloudWatch ServiceLens、AWS X-Ray と組み合わせることも可能



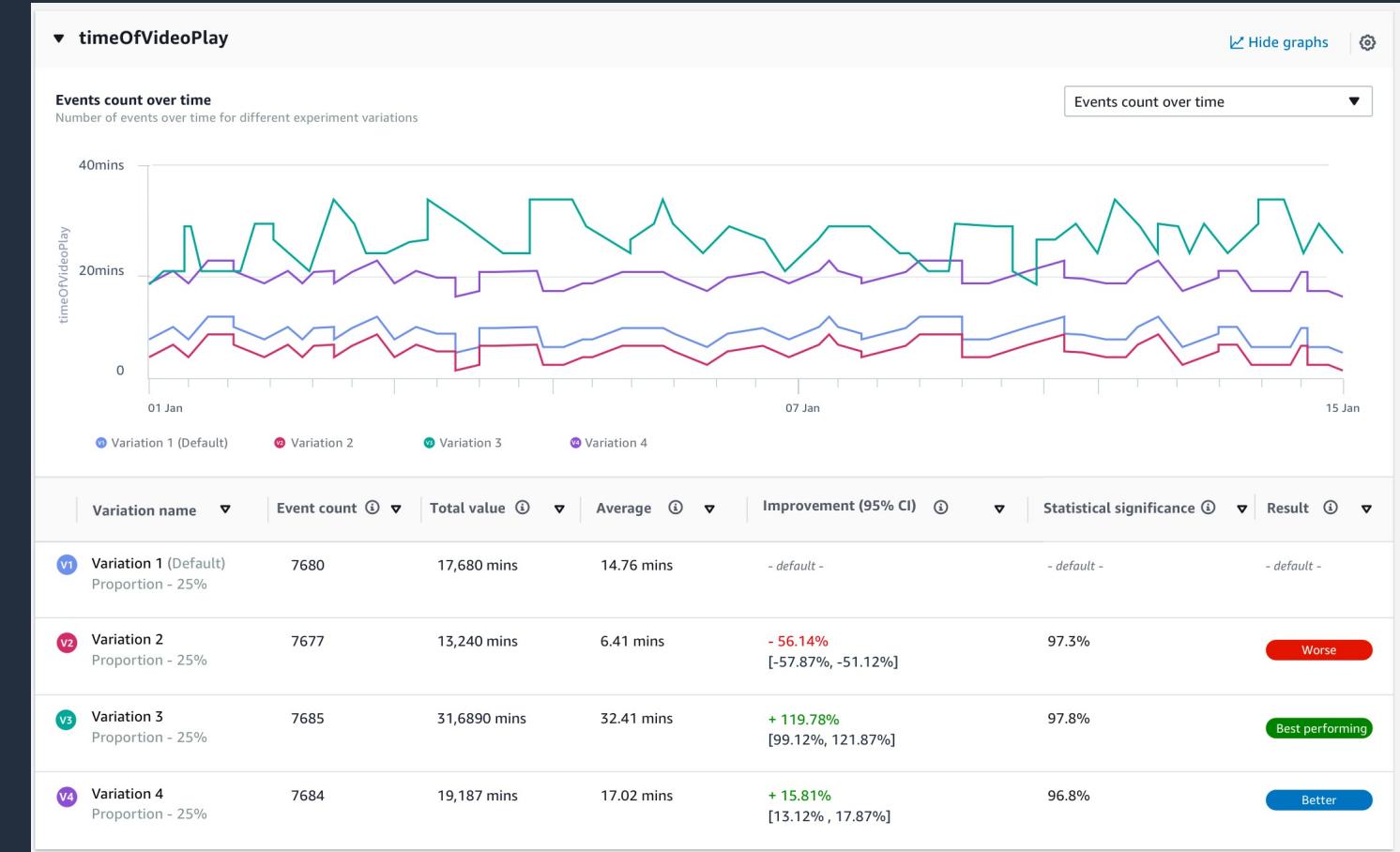
CloudWatch RUM の動作イメージ



CloudWatch Evidently

機能変更・リリース時に、データに基づいた良否の判断を可能に

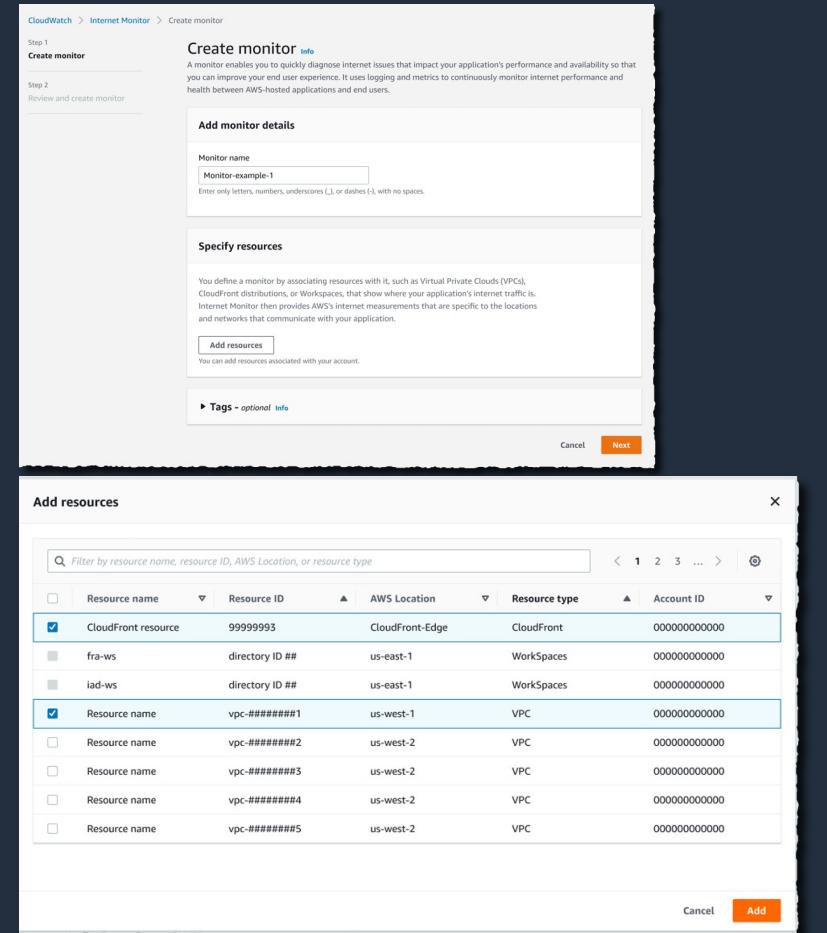
- A/Bテストやフィーチャーフラグといった手法でどちらの選択肢が望ましいか検証する際に、ユーザの挙動をモニタリングするサービス
- A/Bテスト、フィーチャーフラグで検証を行う際に、データに基づいた判断を支援。開発者が新機能を安全に評価できる
- ラフィックをコントロールするスケジュール機能や、問題発生時にロールバックするアラーム機能を備える
- Amazon CloudWatch RUMと統合されており、RUMのメトリクスを利用可能



CloudWatch Internet Monitor

インターネットアクセスの状況評価

- AWS 上のアプリケーションに対してインターネットからアクセスした際の可用性とパフォーマンスマトリクスを CloudWatch で可視化可能に
 - ユーザから見た場合の可用性と性能をチェックできる
 - AWS のグローバルネットワークから取得した接続データに基づきモニタリング。問題がある場合はその影響や場所、プロバイダーなどを可視化し、改善アクションを起こしやすくする
 - 例えば「概ね正常だがラスベガスからアクセスしているユーザはパフォーマンスが落ちている」といった状況が検出できる
 - VPC フローログや CloudFront ログの有効化は不要
- 2023/02/28に一般提供開始**



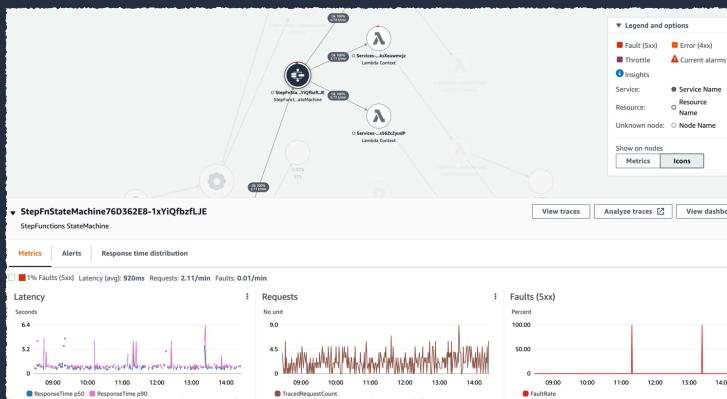
顧客が利用するアプリケーションをホストする、VPC /CloudFront ディストリビューション/WorkSpacesディレクトリを選択

CloudWatch ServiceLens

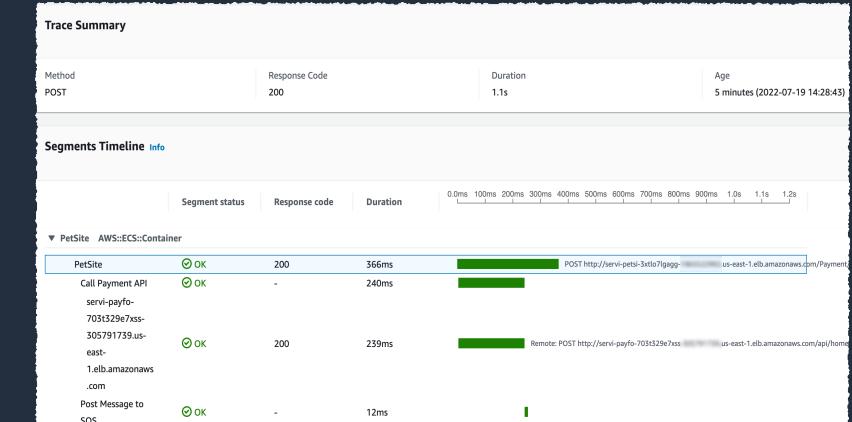
1箇所からサービスの健全性・可用性・パフォーマンスを確認

- Service map: サービスのコンテキストリンクを視覚化
- CloudWatch メトリクスとログ、および AWS X-Ray からのトレースを 1 カ所に結び付ける

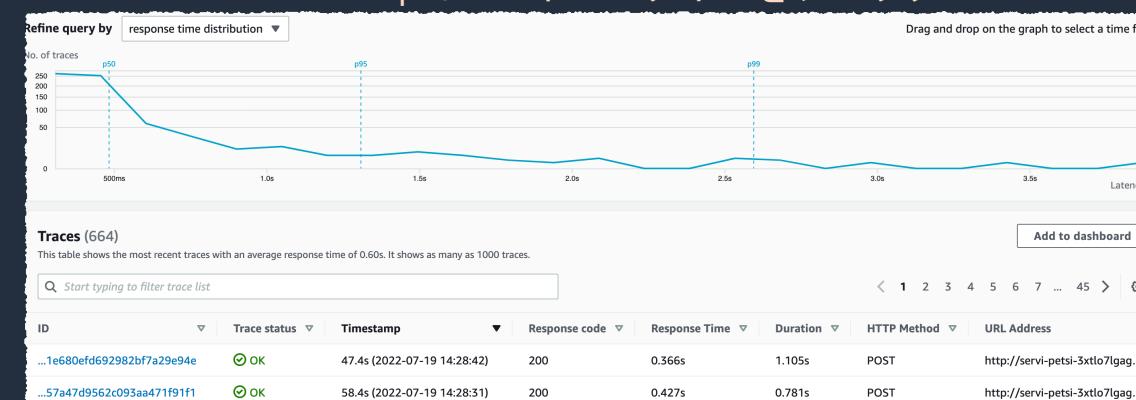
Service mapでノードとメトリクスを関連付けて確認



Service mapでトレースの詳細を見る



Service mapでレスポンスタイムをチェック



再掲：CloudWatchの機能一覧

Infrastructure

AWSサービス名	概要
CloudWatch Metrics	メトリクス
CloudWatch Log	ログ
CloudWatch Alarm	アラーム
CloudWatch Dashboard	ダッシュボード
CloudWatch Metrics Explorer	メトリックス検索
CloudWatch Metrics Stream	メトリックスのリアルタイム連携
CloudWatch Events ※これはEvent Bridgeに統合されています	イベント
CloudWatch Resource Health	EC2の健全性・パフォーマンス可視化

Application Monitoring

AWSサービス名	概要
CloudWatch Synthetics	外形監視
CloudWatch RUM	リアルユーザー モニタリング
CloudWatch Evidently	フィーチャーフラグA/Bテスト
CloudWatch Internet Monitor	インターネット監視
CloudWatch ServiceLens	トレース

Insights

AWSサービス名	概要
CloudWatch Contributor Insights	ログの時系列分析
CloudWatch Container Insights	コンテナ分析
CloudWatch Lambda Insights	Lambda分析
CloudWatch Application Insights	アプリケーション分析
CloudWatch Logs Insights	LOG分析
CloudWatch Metrics Insights	メトリクス分析

グレーアウトはご紹介済み

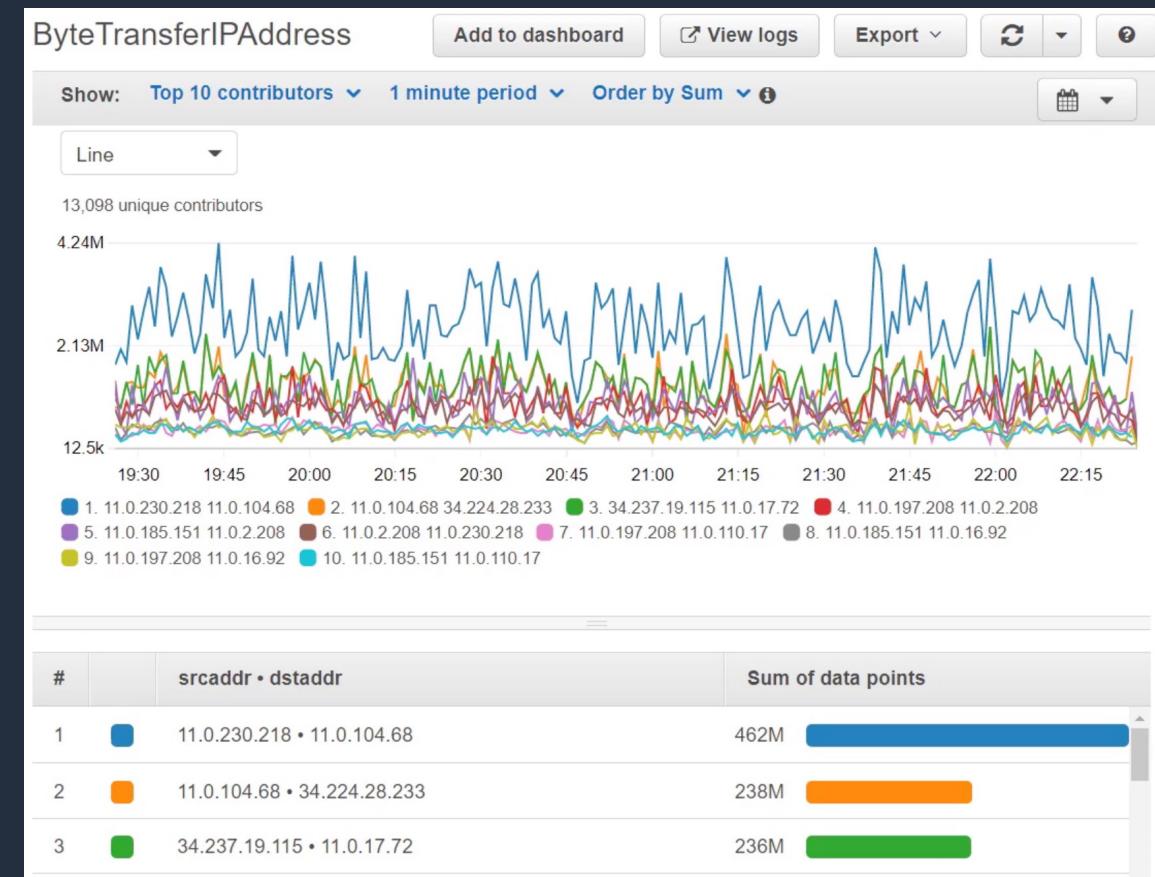
★ここからはその他のサービスを簡単にご紹介
※個別の詳細は別途Blackbeltセッションで紹介予定

CloudWatch Contributor Insights

ログを解析して「上位N」のコントリビューターを見つける

- CloudWatch Logs の構造化されたログデータを解析
- コントリビューターデータを表示する時系列グラフを作成
- システムのパフォーマンスに影響を与えていている人や物を判断するのに有効
 - 例1) エラーを最も多く生成する Product Id
 - 例2) 最もネットワークが重いユーザー

例: VPCフローログで送信元および宛先 IP アドレスごとのデータ数上位3位を見つける

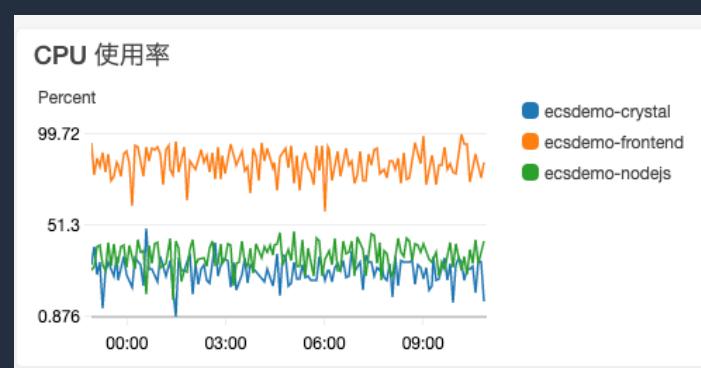


CloudWatch Container Insights

コンテナのメトリクス

コンテナ単位でのメトリクスを取得

- コンテナ化されたアプリケーションの メトリクスと ログ を収集、集計、要約できる CloudWatch の機能の一つ
- タスクや Pod、コンテナレベルでのメトリクスの収集が可能
- CPU やメモリ、ディスク、ネットワークなど、多数のリソースのメトリクスを自動的に収集
- 収集されたメトリクスは自動的に作成されるダッシュボードに集約
- Amazon ECS、Amazon EKS (on EC2)、および Amazon EC2 の Kubernetes プラットフォームで利用可能
ECS サービス



A screenshot of the "Task performance" dashboard. It lists nine tasks, filtered by "ecsdemo-frontend". The table includes columns for "Task definition", "サービス", "平均 CPU (%)", and "平均メモリ (%)". The second task in the list has a red box around it, highlighting its high CPU usage.

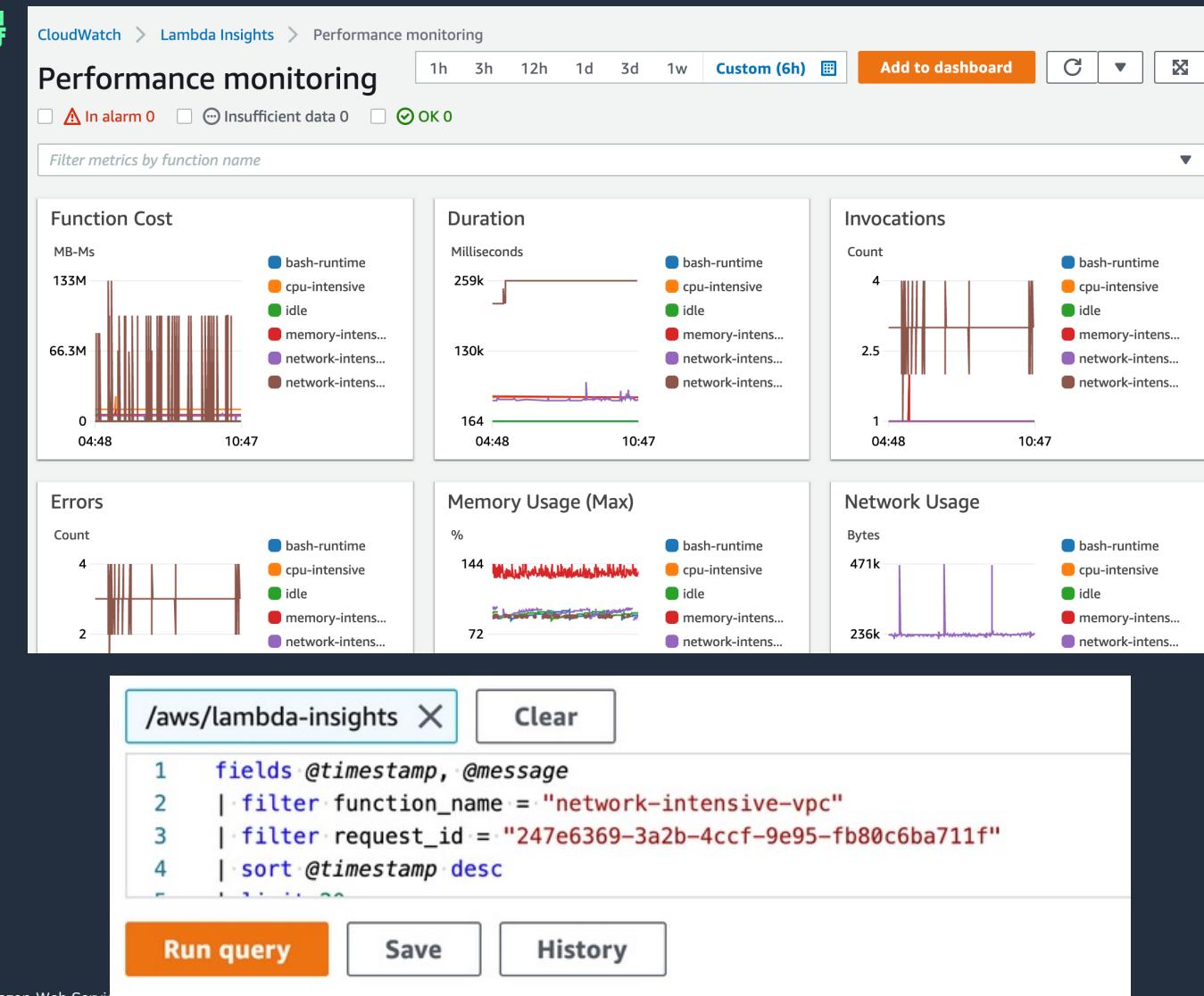
Task definition	サービス	平均 CPU (%)	平均メモリ (%)
ecsdemo-frontend	ecsdemo-frontend	1.6258	16.0156
ecsdemo-frontend	ecsdemo-frontend	99.574	96.0156
ecsdemo-frontend	ecsdemo-frontend	1.5725	16.0156

CloudWatch Lambda Insights

Lambda関数のデフォルトに追加でメトリクスを取得

- システムレベルのメトリクスを使用して、Lambda 関数単位でより深い洞察を得ることができる
 - CPU 時間、メモリ、ディスク、ネットワーク 使用率、コールドスタートなど
- 自動で構築されたダッシュボードを使用可能
- Lambda Insights でクエリを実行し、パフォーマンスの問題をトラブルシューティング可能
- Lambda Layer として提供される拡張機能を有効にするだけで使用可能に

Lambdaのメトリクス



CloudWatch Application Insights

最小限の労力で深いエンタープライズアプリケーションインサイトを

- エンタープライズ アプリケーションのモニタリング
- アプリケーションのリソース全体およびテクノロジースタック (Microsoft SQL Server データベース、ウェブ (IIS) サーバー、アプリケーションサーバー、OS、ロードバランサー、キューなど) で主要なメトリクス、ログ、およびアラームを簡単に設定が可能
- 例えば、CloudWatch Application Insights for .NET and SQL Server は、アプリケーションリソースをスキャンして推奨されるメトリクスとログのリスト (カスタマイズ可能) を提供するため、CloudWatch でこのメトリクスやログを簡単に監視設定できる

Application Insights for .NET and SQL Server [Edit configuration](#)

Problems detected

Severity	Problem summary	Source	Start-time	Status
Medium	EC2: High CPU A problem is going on with the resource awseb-e-jcmmrgsmpx-stack-AWSEBAutoScalingGroup-1VNFYAH14I5RS in DOT_NET_WEB_TIER tier.	CloudWatch: Application Insights Problem Id: p-2fc4d610-2ed3-4da6-8927-b07c30d08c0b Edit configuration	Time range 1h 3h 12h 1d 3d 1w custom (20h)	Actions
Low				

Problem summary

Severity	Problem summary	Source	Start-time	Status	Resource group
Medium	EC2: High CPU	i-0bf88dbe687f32678	2019-07-24T22:29:06Z	In progress	SampleAppGroup

Insight

High CPU on the .NET application layer. This may result in application performance degradation due to high load on web servers or application errors. If you experience high load conditions for long periods of time, use Auto Scaling to add additional resources to process the load. To troubleshoot further, collect a full user dump with task manager on the high CPU process and collect PerfMon logs with the thread counter to identify the thread ID causing high CPU.

Help us improve our models: This insight is useful This insight is not useful [Submit feedback](#)

© 2023, Amazon Web Services, Inc. or its affiliates.



4. 次に何をしたらよいの？



そもそも、サービスが使っているかを監視したい？

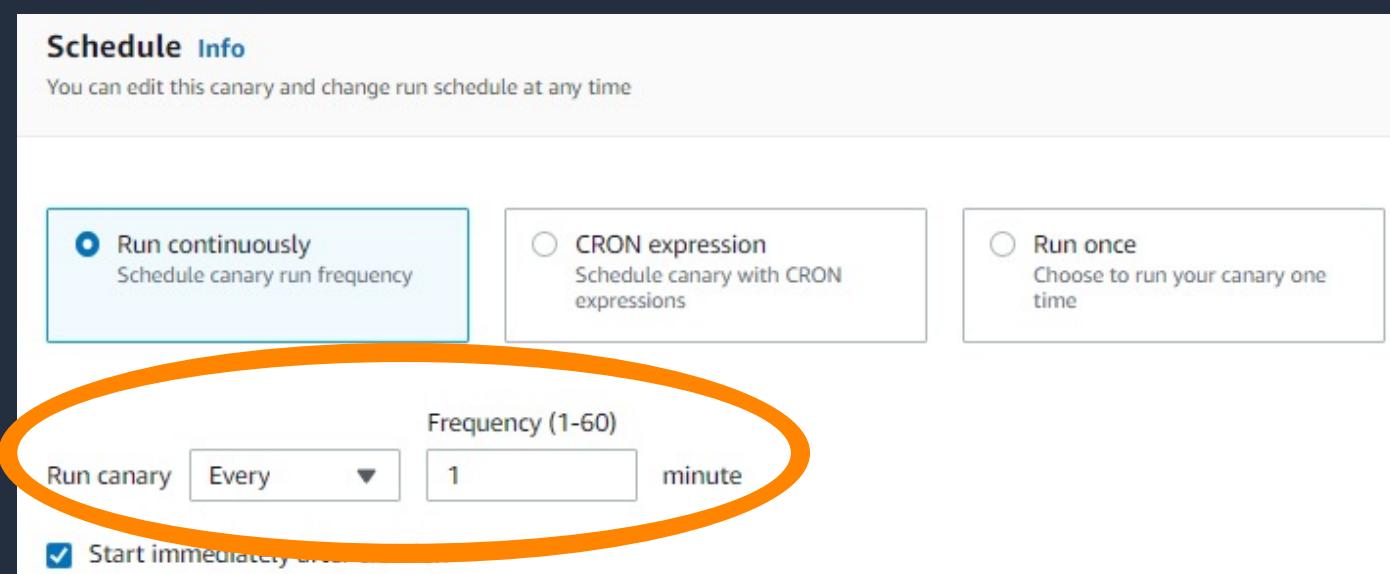
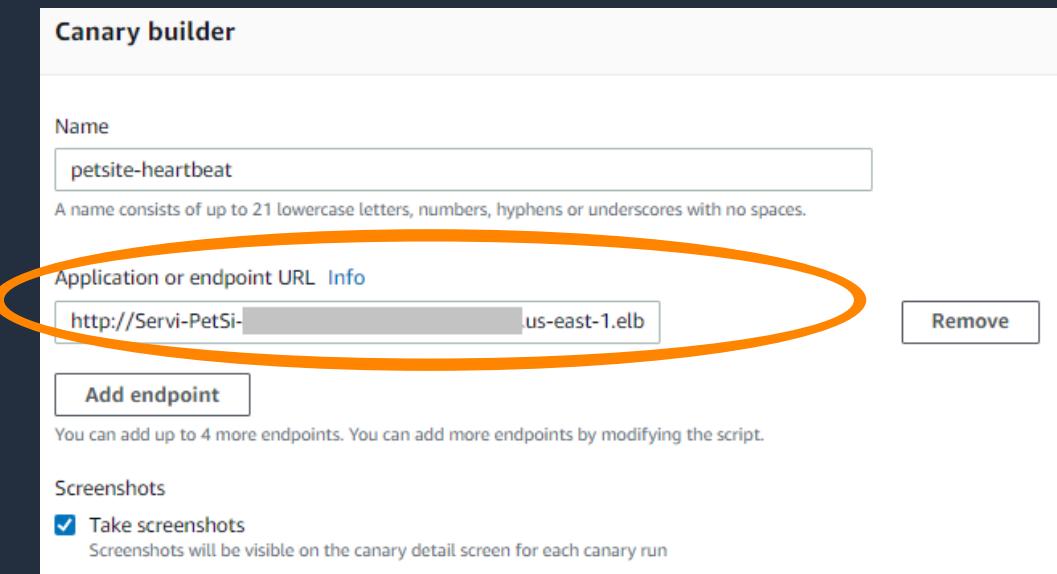
ビジネスに直結する
メトリクスの取得と監視

→CloudWatch Syntheticsもご活用ください！

- **Canary（カナリア）を作成してサービスのエンドポイントとAPIを監視する**

Heartbeat Canaryのblueprint例

※blueprintは、Heartbeatのような特定のユースケースのためにカスタマイズ可能なコードを提供

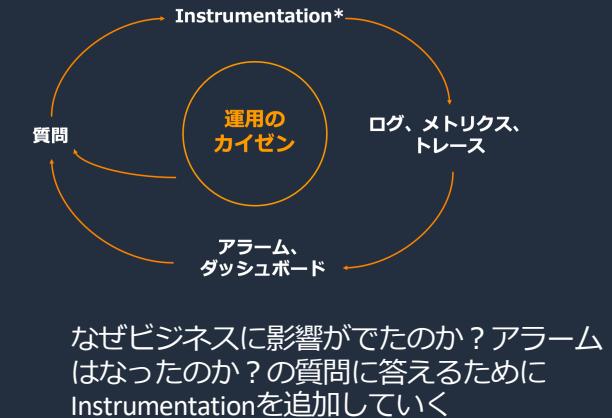
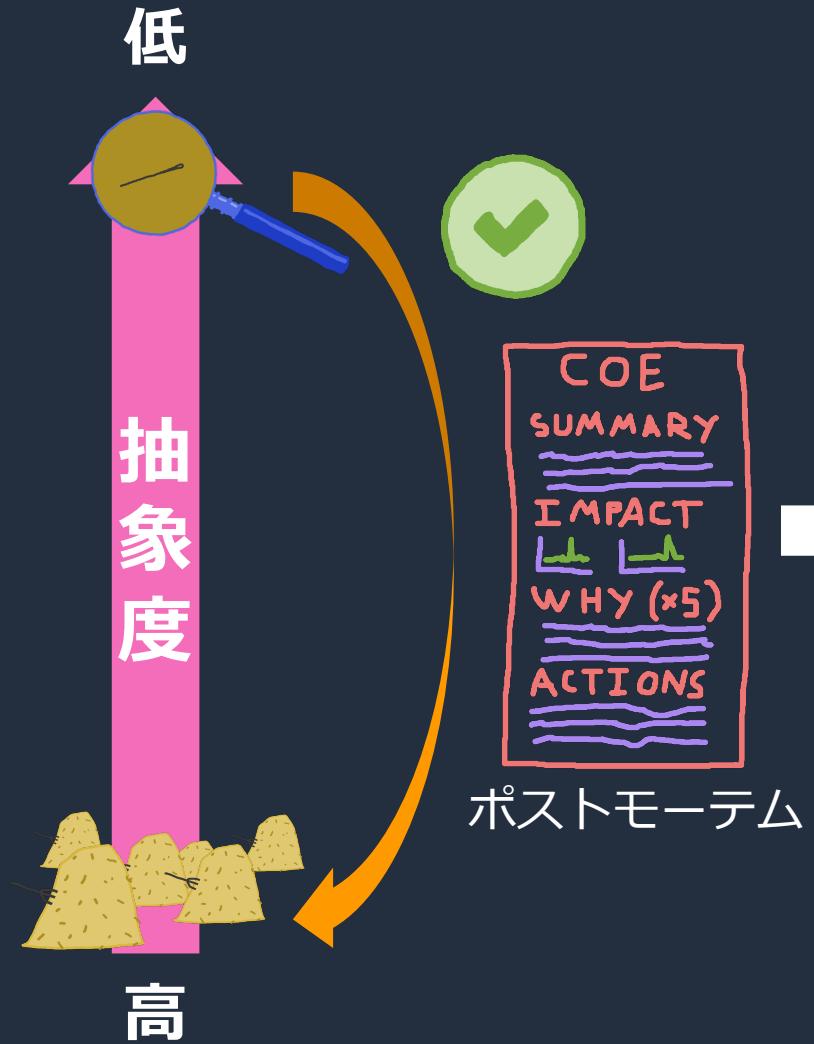
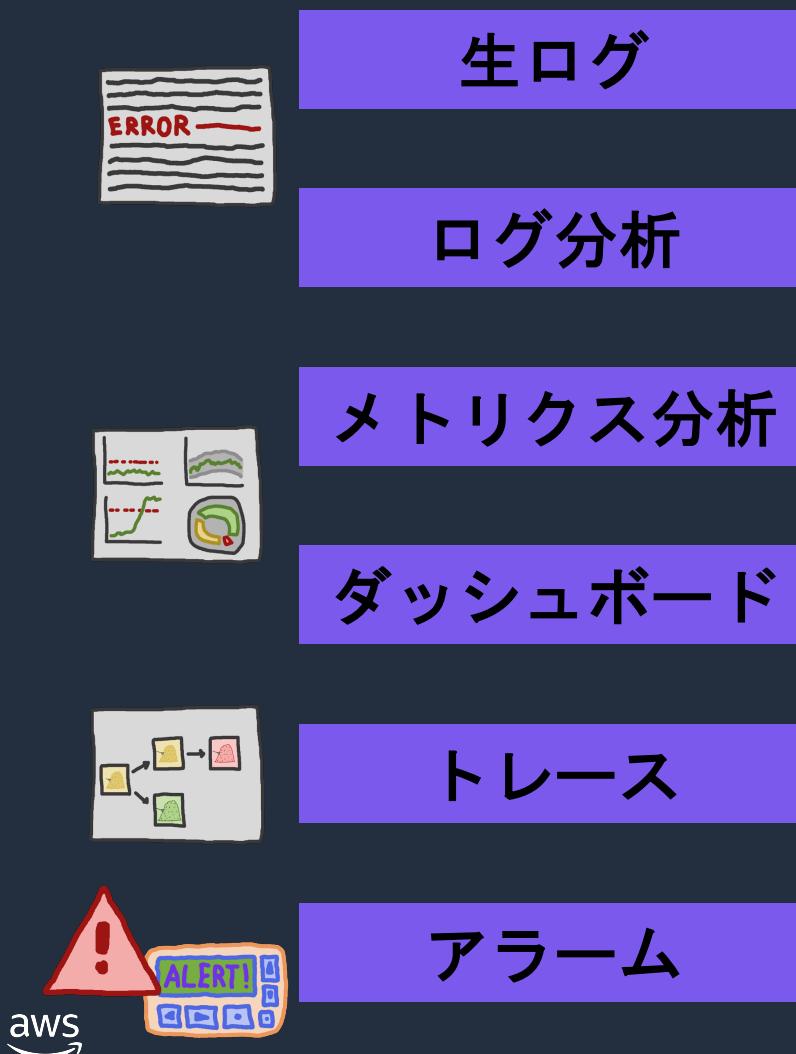


Canaryが1分毎にendpoint URLを
Heartbeatする

個々の機能やカテゴリについては、別途Blackbeltを公開予定です。
CloudWatch Syntheticsについても近日公開します。

で、実際どうやって運用しているの？

Amazon での対応例



*Instrumentation
ログ、メトリクス、トレースなどのデータを取得し、外部に送信できるようシステムに組み込むこと



運用を可視化するための
ダッシュボードの構築

<https://aws.amazon.com/jp/builders-library/building-dashboards-for-operational-visibility/>

最も重要なテレメトリデータはモニタリングできていますか？

アラームとダッシュボードのチェックで発生する質問に答えられる準備をしておきましょう。

Amazon CloudWatchをはじめとした
Observability Serviceが力になります！

本資料に関するお問い合わせ・ご感想

技術的な内容に関しましては、有料のAWSサポート窓口へ
お問い合わせください

<https://aws.amazon.com/jp/premiumsupport/>

料金面でのお問い合わせに関しましては、カスタマーサポート窓口へ
お問い合わせください（マネジメントコンソールへのログインが必要です）

<https://console.aws.amazon.com/support/home#/case/create?issueType=customer-service>

具体的な案件に対する構成相談は、後述する個別相談会をご活用ください



ご感想はTwitterへ！ハッシュタグは以下をご利用ください
#awsblackbelt

その他コンテンツのご紹介

ウェビナーなど、AWSのイベントスケジュールをご参照いただけます

<https://aws.amazon.com/jp/events/>

ハンズオンコンテンツ

<https://aws.amazon.com/jp/aws-jp-introduction/aws-jp-webinar-hands-on/>

AWS 個別相談会

AWSのソリューションアーキテクトと直接会話いただけます

<https://pages.awscloud.com/JAPAN-event-SP-Weekly-Sales-Consulting-Seminar-2021-reg-event.html>



Thank you!