ユーザーガイド

AWS IoT SiteWise

AWS IoT SiteWise: ユーザーガイド

Table of Contents

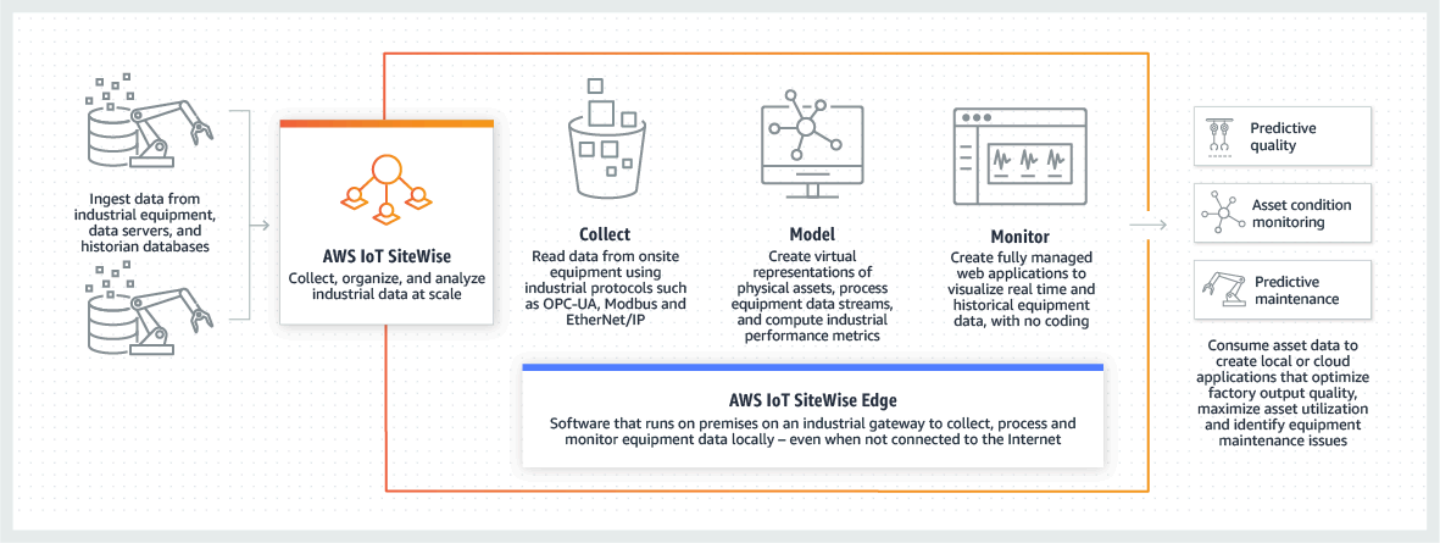
とは何ですか AWS IoT SiteWise?

AWS IoT SiteWise は、産業機器からのデータを大規模に簡単に収集、保存、整理、監視できるようにするマネージドサービスです。これにより、データ主導型のより良い意思決定が可能になります。施設全体の運用を監視したり、 AWS IoT SiteWise 一般的な産業パフォーマンス指標を迅速に計算したり、産業機器データを分析するアプリケーションを作成したりすることで、コストのかかる機器の問題を防ぎ、生産におけるギャップを減らすことができます。

AWS IoT SiteWise Monitor 運用担当者は、産業データをリアルタイムで表示、分析するための Web アプリケーションをすばやく作成できます。平均故障間隔や総合設備効率 (OEE) などのメトリクスを設定およびモニタリングすることで、産業オペレーションに関するインサイトを得ることができます。

AWS IoT SiteWise エッジは、ローカルデバイスでのデータの収集、保存、 AWS IoT SiteWise 処理を可能にするコンポーネントです。これは、インターネットへのアクセスが制限されている場合や、データを非公開にしておく必要がある場合に便利です。

以下の図は、以下の基本アーキテクチャを示しています AWS IoT SiteWise。



トピック

- AWS IoT SiteWise の仕組み
- AWS IoT SiteWise コンセプト
- のユースケース AWS IoT SiteWise

AWS IoT SiteWise の仕組み

AWS IoT SiteWise は、産業デバイス、プロセス、および施設の表現を作成するために使用できるリソースモデリングフレームワークを提供します。機器とプロセスの表現は、 ではアセットモデルと呼ばれます AWS IoT SiteWise。アセットモデルでは、消費する raw データとその処理方法を有用なメトリクスに定義します。AWS IoT SiteWise コンソール で、産業オペレーション用のアセットとモデルを構築および視覚化します。エッジまたは AWS クラウドでデータを収集して処理するようにアセットモデルを設定することもできます。

トピック

- 産業データの取り込み
- 収集したデータをコンテキスト化するためのモデルアセット
- クエリ、アラーム、予測を使用した分析
- オペレーションを視覚化する
- データストア
- 他の サービスとの統合

産業データの取り込み

産業データの取り込み AWS IoT SiteWise から の使用を開始します。データの取り込みは、次のいずれかの方法で行われます。

- オンサイトサーバーからの直接取り込み： OPC-UA などのプロトコルを使用して、オンサイトデバイスからデータを直接読み取ります。と互換性のある SiteWise Edge ゲートウェイソフトウェアを AWS IoT Greengrass V2、一般的な産業用ゲートウェイや仮想サーバーなどの幅広いプラットフォームにデプロイします。最大 100 台の OPC-UA サーバーを 1 つの AWS IoT SiteWise ゲートウェイに接続できます。詳細については、「SiteWise エッジゲートウェイの要件」を参照してください。

  Modbus TCP やイーサネット/IP (EIP) などのプロトコルは、 のコンテキストDomaticaで とのパートナーシップを通じてサポートされることに注意してください AWS IoT Greengrass V2。
- パックによるエッジデータ処理： パックを追加して SiteWise エッジゲートウェイを強化し、包括的なエッジ機能を有効にします。Edge を で SiteWise 利用できる場合 AWS IoT Greengrass V2、データ処理は AWS IoT Greengrass ストリームを使用して AWS クラウドに安全に転送される前に、オンサイトで直接実行されます。詳細については、「パックを使用する。」を参照してください。

- 一括操作による Amazon S3 経由のアダプティブ取り込み： 多数のアセットまたはアセットモデルを使用する場合は、一括操作を使用して Amazon S3 バケットからリソースを一括インポートおよびエクスポートします。詳細については、「アセットとモデルによる一括オペレーション」を参照してください。
- AWS IoT コアルールを使用した MQTT メッセージ： AWS IoT Core に接続されたデバイスで MQTT メッセージを送信する場合は、 AWS IoT Core ルールエンジンを使用してそれらのメッセージを に転送します AWS IoT SiteWise。 AWS IoT Core に接続されたデバイスが MQTT メッセージを送信する場合は、 AWS IoT Core ルールエンジンを使用してそれらのメッセージを にルーティングします AWS IoT SiteWise。詳細については、「ルールを使ったデータの取り込み AWS IoT Core」を参照してください。
- イベントによってトリガーされるデータインジェスト： AWS IoT Events アクションを使用して、イベントが発生した AWS IoT SiteWise ときに にデータを送信する AWS IoT Events ように の IoT SiteWise アクションを設定します。詳細については、「からのデータの取り込み AWS IoT Events」を参照してください。
- AWS IoT SiteWise API: Edge またはクラウドのアプリケーションは、データを に直接送信できます AWS IoT SiteWise。詳細については、「API を使用してデータを取り込む AWS IoT SiteWise」を参照してください。

収集したデータをコンテキスト化するためのモデルアセット

データを取り込んだ後、そのデータを使用して、物理オペレーションのモデルを構築することで、アセット、プロセス、および施設の仮想表現を作成できます。デバイスまたはプロセスを表すアセットは、データストリームを AWS クラウドに送信します。アセットは、論理デバイスグループ化も意味します。階層は、複雑なオペレーションをミラーリングするためにアセットを関連付けることによって形成されます。これらの階層により、アセットは関連する子アセットのデータにアクセスできます。アセットはアセットモデルから作成されます。アセットモデルは、アセット形式を標準化する宣言構造です。アセットのコンポーネントを再利用して、モデルの整理と保守を行います。詳細については、「産業用アセットのモデリング」を参照してください。

を使用すると AWS IoT SiteWise、受信データをコンテキストメトリクスと変換に変換するようにアセットを設定できます。

- 機器データを受信するときの作業を変換します。
- メトリクスは、定義した間隔で計算されます。

メトリクスと変換は、個々のアセットまたは複数のアセットの両方に適用されます。AWS IoT SiteWise は、機器データ、メトリクス、変換に関連するさまざまな時間枠にわたって、平均、合計、カウントなどの一般的に使用される統計集計を自動的に計算します。

アセットは を使用して同期できます AWS IoT TwinMaker。詳細については、「AWS IoT SiteWise と AWS IoT TwinMaker の統合」を参照してください。

クエリ、アラーム、予測を使用した分析

クエリを実行し、アラームを設定 AWS IoT SiteWise して、 で収集された日付を分析します。Amazon Lookout を使用して、メトリクス内の異常を自動的に検出し、その根本原因を特定することもできます。

- 特定のアラームを設定して、機器やプロセスが最適なパフォーマンスから逸脱したときにチームに警告し、問題を迅速に特定して解決できるようにします。詳細については、「アラームによるデータのモニタリング。」を参照してください。
- AWS IoT SiteWise API オペレーションを使用して、特定の時間間隔におけるアセットプロパティの現在の値、履歴値、集計をクエリします。詳細については、「からのデータのクエリ AWS IoT SiteWise」を参照してください。
- Amazon Lookout for Equipment で異常検出を使用して、機器や動作状態の変化を特定して視覚化します。異常検出を使用すると、オペレーションの予防的メンテナンス対策を決定できます。この統合により、お客様は AWS IoT SiteWise と Amazon Lookout for Equipment の間でデータを同期できます。詳細については、「Amazon Lookout for Equipment による機器の異常の検出」を参照してください。

オペレーションを視覚化する

SiteWise Monitor を設定して、運用担当者用のウェブアプリケーションを作成します。ウェブアプリケーションは、従業員がオペレーションを視覚化するのに役立ちます。IAM Identity Center または IAM を使用して、従業員のさまざまなレベルのアクセスを処理します。産業オペレーション全体の特定のサブセットを表示するように、従業員ごとに一意のログインとアクセス許可を設定します。SiteWise は、これらの従業員向けに Monitor の使用方法を学習するためのアプリケーションガイド AWS IoT SiteWise を提供します。

オペレーションの視覚化の詳細については、「」を参照してくださいによるデータの監視 AWS IoT SiteWise Monitor。

データストア

時系列ストレージを産業データレイクと統合できます。 AWS IoT SiteWise には産業データ用の 3 つのストレージ階層があります。

- リアルタイムアプリケーション用に最適化されたホットストレージ層。
- 分析ワークロード用に最適化されたウォームストレージ階層。
- 高いレイテンシー耐性を持つ運用データアプリケーションに Amazon S3 を使用する、カスタマー管理のコールドストレージ層。

AWS IoT SiteWise は、最新のデータをホットストレージ階層に保持することで、ストレージコストを管理するのに役立ちます。次に、履歴データをウォーム階層またはコールド階層のストレージに移動するデータ保持ポリシーを定義します。詳細については、「ストレージの管理」を参照してください。

アセットメタデータをインポートおよびエクスポートすることもできます。詳細については、アセットメタデータを参照してください。

他の サービスとの統合

AWS IoT SiteWise は、 AWS クラウドで完全な AWS IoT ソリューションを開発するために、複数の AWS サービスと統合されています。詳細については、「AWS 他のサービスとのやり取り」を参照してください。

AWS IoT SiteWise コンセプト

AWS IoT SiteWiseのコアコンセプトは以下のとおりです。

集計

集計は、 AWS IoT SiteWise すべての時系列データを自動的に計算する基本的な指標、つまり測定値です。詳細については、「アセットプロパティ集計のクエリ」を参照してください。

アセット

AWS IoT SiteWise 産業機器からデータを入力または取り込むと、デバイス、機器、プロセスがそれぞれ資産として表示されます。各資産には関連するデータがあります。たとえば、機器にはシリアル番号、場所、製造元とモデル、設置日などが記載されている場合があります。また、可

用性、性能、品質、温度、圧力などの時系列値が含まれている場合もあります。資産を階層にグループ化し、資産が子資産に保存されているデータにアクセスできるようにします。詳細については、「産業用アセットのモデリング」を参照してください。

アセット階層

資産階層を設定して、産業活動を論理的に表現できます。そのためには、アセットモデルに階層を定義し、そのモデルから作成されたアセットを指定した階層に関連付けます。親アセットのメトリックは、子アセットのプロパティのデータを組み合わせることができるため、運用全体またはその特定の部分に関するインサイトを提供するメトリックを計算できます。詳細については、「アセットモデル階層の定義」を参照してください。

アセットモデル

すべての資産は資産モデルを使用して作成されます。アセットモデルは、アセットのフォーマットを定義し標準化する構造です。これにより、同じ種類の複数のアセットにわたって情報の整合性が保たれるため、デバイスのグループを表すアセット内のデータを処理できます。各アセットモデルでは、属性、時系列入力 (測定) 、時系列変換 (変換) 、時系列集約 (メトリクス) 、アセット階層を定義できます。詳細については、「産業用アセットのモデリング」を参照してください。

アセットモデルをエッジ用に設定して、アセットモデルのプロパティをどこで処理するかを決めます。この機能を利用して、ローカルデバイス上のアセットデータを処理および監視してください。

アセットプロパティ

資産プロパティは、産業データを保持する各資産内の構造です。各プロパティにはデータタイプがあり、単位を設定することもできます。プロパティには、属性、測定、変換、または メトリクスを指定できます。詳細については、「データのプロパティを定義する。」を参照してください。

エッジで計算するようにアセットプロパティを設定します。エッジでのデータ処理については、the section called “エッジデータ処理を有効にする。” を参照してください。

属性

属性は、デバイスメーカーやデバイスの場所など、通常は変わらないアセットのプロパティです。属性にはプリセット値を設定できます。アセットモデルから作成されたすべてのアセットには、そのモデルで定義されているアトリビュートのデフォルト値が含まれます。詳細については、「静的データ (属性) の定義」を参照してください。

ダッシュボード

各プロジェクトには、ダッシュボードのセットが含まれています。ダッシュボードは、一連のアセットの値に対する一連の可視化を提供します。プロジェクトの所有者は、ダッシュボードとそれに含まれる可視化を作成します。プロジェクト所有者がダッシュボードのセットを共有する準備ができたら、所有者はプロジェクトに閲覧者を招待し、プロジェクト内のすべてのダッシュボードにアクセスできるようにします。ダッシュボードごとに異なる閲覧者のセットが必要な場合は、プロジェクト間でダッシュボードを分割する必要があります。閲覧者がダッシュボードを見るとき、特定のデータを見るように時間範囲をカスタマイズできます。

データストリーム

AWS IoT SiteWise 資産モデルや資産を作成する前でも、産業データを入力、または取り込むことができます。 AWS IoT SiteWise データストリームを自動的に生成して、機器から未加工のデータストリームを収集します。

データストリームのエイリアス

データストリームのエイリアスは、データストリームを簡単に識別するのに役立ちます。たとえば、エイリアスは風力発電所 #3 のタービン #7 `server1-windfarm/3/turbine/7/temperature` からの温度値を示します。`server1`この用語は OPC-UA サーバーの識別に役立つデータソース名で、この OPC-UA `server1-` サーバーから報告されるすべてのデータストリームに付けられるプレフィックスです。

データストリームの関連付け

アセットモデルとアセットを作成したら、データストリームをアセットに定義されているアセットプロパティに関連付けてデータを構造化します。 AWS IoT SiteWise その後、アセットモデルとアセットを使用して、データストリームからの受信データを処理できます。また、アセットプロパティからのデータストリームの関連付けを解除することもできます。詳細については、「データストリームの管理」を参照してください。

式

トランスフォームプロパティとメトリックプロパティにはそれぞれ、そのプロパティがどのようにデータを変換または集計するかを説明する数式が付属しています。これらの式には、プロパティ入力、演算子、およびが提供する関数が含まれます。 AWS IoT SiteWise詳細については、「数式の使用」を参照してください。

測定

測定値は、デバイスまたは機器からの未加工のセンサー時系列データストリームを表すアセットのプロパティです。詳細については、「機器からのデータストリームの定義 (測定値) 」を参照してください。

メトリクス

メトリックは、集計された時系列データを表すアセットのプロパティです。各指標には、データポイントの集計方法を説明する数式 (式) と、その集計を計算するための時間間隔が付いています。メトリクスは、指定した時間間隔ごとに 1 つのデータポイントを生成します。詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください。

パック

SiteWise Edge ゲートウェイはパックを使用してデータの収集、処理、ルーティングの方法を決定します。現在、 AWS IoT SiteWise はデータ収集パックとデータ処理パックをサポートしています。 SiteWise Edge ゲートウェイで使用できるパックの詳細については、を参照してくださいthe section called “パックを使用する。”。

データ収集パック

データ収集パックを使用すると、 SiteWise Edge ゲートウェイが産業データを収集し、 AWS 選択した宛先にルーティングできるようになります。このパックは SiteWise Edge ゲートウェイに自動的に追加され、削除することはできません。

データ処理パック

データ処理パックを使用してデータをエッジで処理し、ローカルアプリケーションで使用できるように 30 日間保持します。

Portal

AWS IoT SiteWise Monitor ポータルは、 AWS IoT SiteWise データを視覚化して共有できる Web アプリケーションです。ポータルには 1 人以上の管理者があり、0 個以上のプロジェクトが含まれています。

ポータル管理者

SiteWise 各モニターポータルには 1 人以上のポータル管理者がいます。ポータル管理者は、ポータルを使用して、アセットとダッシュボードのコレクションを含むプロジェクトを作成します。その後で、ポータル管理者はアセットと所有者を各プロジェクトに割り当てます。プロジェクトへのアクセスを制御することにより、ポータル管理者は、プロジェクトの所有者とビューワーが参照できるアセットを指定します。

プロジェクト

各 SiteWise Monitor ポータルには一連のプロジェクトが含まれています。各プロジェクトには、AWS IoT SiteWise アセットと関連付けられたのサブセットがあります。プロジェクト所有者は、1 つ以上のダッシュボードを作成して、それらのアセットに関連付けられたデータの整合性ある表示方法を提供します。プロジェクトの所有者は、プロジェクトに閲覧者を招待して、プロジェクト内のアセットとダッシュボードを表示できるようにすることができます。 SiteWise プロジェクトはモニター内での共有の基本単位です。プロジェクトオーナーは、 AWS 管理者からポータルへのアクセス権を与えられたユーザーを招待できます。ユーザーは、そのポータル内のプロジェクトをそのユーザーと共有する前に、ポータルへのアクセス権を持っている必要があります。

プロジェクトの所有者

各 SiteWise Monitor プロジェクトにはオーナーがいます。プロジェクトの所有者は、一貫した方法で運用データを表すために、ダッシュボードの形式で可視化を作成します。ダッシュボードを共有する準備ができたら、プロジェクト所有者はビューワーをプロジェクトに招待できます。プロジェクトの所有者は、他の所有者もプロジェクトに割り当てることができます。プロジェクト所有者は、アラームのしきい値や通知設定を設定することができます。

プロジェクトビューワー

各 SiteWise Monitor プロジェクトには視聴者がいます。プロジェクト閲覧者は、ポータルに接続して、プロジェクト所有者が作成したダッシュボードを表示できます。各ダッシュボードでは、プロジェクトビューワーが時間範囲を調整することで、運用データをより深く理解することができます。プロジェクト閲覧者は、アクセス権のあるプロジェクトのダッシュボードのみを表示できます。プロジェクトビューワーは、アラームを確認したり、スヌーズさせたりすることができます。

プロパティエイリアス

OPC-UA サーバーのデータストリームパス (/company/windfarm/3/turbine/7/temperature など) などのアセットプロパティにエイリアスを作成できるため、アセットデータの取り込みまたは取得時にアセットプロパティを簡単に識別できます。SiteWise Edge ゲートウェイを使用してサーバーからデータを取り込む場合、プロパティエイリアスは未加工データストリームのパスと一致する必要があります。詳細については、「アセットプロパティへの産業データストリームのマッピング」を参照してください。

プロパティ通知

アセットプロパティのプロパティ通知を有効にすると、 AWS IoT Core そのプロパティが新しい値を受け取るたびに MQTT AWS IoT SiteWise メッセージを公開します。メッセージペイロード

には、そのプロパティ値の更新に関する詳細が含まれます。資産価値通知を使用して、 AWS 産業データを他のサービスと結び付けるソリューションを作成できます。 AWS IoT SiteWise 詳細については、「AWS 他のサービスとのやり取り」を参照してください。

SiteWise エッジゲートウェイ

SiteWise エッジゲートウェイはお客様の敷地内に設置され、データを収集、処理、転送します。 SiteWise エッジゲートウェイは OPC-UA プロトコルを介して産業用データソースに接続し、データを収集して処理し、クラウドに送信します AWS 。 SiteWise Edge ゲートウェイはパートナーのデータソースにも接続できます。 SiteWise エッジゲートウェイは、データ収集、エッジ処理などにパックを使用します。使用できるパックの詳細については、the section called “パックを使用する。” を参照してください。

SiteWise Edge ゲートウェイは、 AWS IoT Greengrass実行可能なデバイスやプラットフォームであればどれでも柔軟に作成できます。詳細については、「 SiteWise Edge ゲートウェイを使用する」を参照してください。

変換

トランスフォームは、変換された時系列データを表すアセットのプロパティです。すべての変換には、データポイントをある形式から別の形式に変換する方法を指定する数式 (式) が付いています。 one-to-one 変換されたデータポイントは入力データポイントと関係があります。詳細については、「データの変換 (変換) 」を参照してください。

視覚化

各ダッシュボードでは、プロジェクトに関連するアセットのプロパティとアラームをどのように表示するかをプロジェクト所有者が決定します。可用性は折れ線グラフで、その他の数値は棒グラフや重要業績評価指標 (KPI、 Key Performance Indicator) で表示されるかもしれません。アラームは、状態グリッドと状態タイムラインでの表示が最適です。プロジェクトの所有者は、それぞれの可視化をカスタマイズして、そのアセットのデータを最もよく理解できるようにしています。

のユースケース AWS IoT SiteWise

AWS IoT SiteWise は、さまざまな業界で多くの産業用データ収集および分析アプリケーションに使用されています。

あらゆるソースから一貫してデータを収集することで、問題を迅速に解決できます。 AWS IoT SiteWise データを現場で直接収集したり、多数の施設にまたがる複数のソースから収集したりする

リモート監視機能を提供します。 AWS IoT SiteWise 産業用 IoT データソリューションに必要な柔軟性を提供します。

製造

AWS IoT SiteWise 機器からデータを収集して利用するプロセスを簡素化し、非効率性を特定して最小限に抑え、産業運営を強化できます。 AWS IoT SiteWise 製造ラインや機器からデータを収集するのに役立ちます。を使用すると AWS IoT SiteWise、 AWS データをクラウドに転送して、特定の機器やプロセスのパフォーマンスメトリクスを構築できます。生成された指標を使用して、業務の全体的な有効性を把握し、革新と改善の機会を特定できます。また、製造工程を表示し、設備や工程の欠陥、生産ギャップ、製品不良を特定することができます。

食品飲料

食品飲料業界の施設は、穀物製粉、精肉と加工、電子レンジ調理可能な料理のアセンブリ、調理、冷凍など、多岐にわたる食品加工を手掛けます。食品加工工場は複数の場所にまたがっていることが多く、工場や機器のオペレーターが一元管理してプロセスや設備を監視しています。たとえば、冷凍ユニットは原料の取り扱いと有効期限を評価します。施設全体の廃棄物の発生を監視して、運用効率を確保します。を使用すると AWS IoT SiteWise、複数の場所からのセンサーデータストリームを生産ラインや施設ごとにグループ化できるため、プロセスエンジニアは施設全体をよりよく理解して改善することができます。

エネルギーとユーティリティ

を使用すると AWS IoT SiteWise、機器の問題をより簡単かつ効率的に解決できます。資産のパフォーマンスをリモートでリアルタイムに監視できます。どこからでも機器の履歴データにアクセスして、潜在的な問題を特定し、正確なリソースを派遣し、問題の防止と解決を迅速に行えます。

の開始方法 AWS IoT SiteWise

を使用すると AWS IoT SiteWise、データを収集、整理、分析、視覚化できます。

AWS IoT SiteWise は、実際のデータソースを設定せずにサービスを調べるために使用できるデモを提供します。詳細については、「AWS IoT SiteWise デモを使う」を参照してください。

以下のチュートリアルを完了して、 の特定の機能を調べることができます AWS IoT SiteWise。

- モノからのデータの取り込み AWS IoT
- Monitor SiteWise での風力発電所データの視覚化と共有
- プロパティ値の更新を Amazon DynamoDB に公開する

の詳細については、以下のトピックを参照してください AWS IoT SiteWise。

- へのデータの取り込み AWS IoT SiteWise
- 産業用アセットのモデリング
- エッジデータ処理を有効にする。
- によるデータの監視 AWS IoT SiteWise Monitor
- からのデータのクエリ AWS IoT SiteWise
- AWS 他のサービスとのやり取り

トピック

要件

の使用を開始する AWS アカウント には、 が必要です AWS IoT SiteWise。アカウントをお持ちでない場合は、「のセットアップ AWS アカウント」を参照してください。

が利用可能な リージョン AWS IoT SiteWise を使用します。詳細については、「AWS IoT SiteWise エンドポイントとクォータ」を参照してください。のリージョンセレクタを使用して AWS Management Console 、これらのリージョンのいずれかに切り替えることができます。

のセットアップ AWS アカウント

トピック

にサインアップする AWS アカウント

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、通話呼び出しを受け取り、電話キーパッドで検証コードを入力するように求められます。

   にサインアップすると AWS アカウント、 AWS アカウントのルートユーザーが作成されます。ルートユーザーには、アカウントのすべての AWS のサービス とリソースへのアクセス権があります。セキュリティのベストプラクティスとして、ユーザーに管理アクセスを割り当て、ルートユーザーのみを使用してルートユーザーアクセスが必要なタスクを実行してください。

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ の [マイアカウント] を選んで、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理できます。

管理アクセスを持つユーザーを作成する

にサインアップしたら AWS アカウント、 を保護し AWS アカウントのルートユーザー、 を有効にして AWS IAM Identity Center、日常的なタスクにルートユーザーを使用しないように管理ユーザーを作成します。

のセキュリティ保護 AWS アカウントのルートユーザー

1. ルートユーザーを選択し、 AWS アカウント E メールアドレスを入力して、アカウント所有者AWS Management Consoleとして にサインインします。次のページでパスワードを入力します。

ルートユーザーを使用してサインインする方法については、AWS サインイン ユーザーガイドの「ルートユーザーとしてサインインする」を参照してください。

2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、「IAM ユーザーガイド」の AWS アカウント 「ルートユーザーの仮想 MFA デバイスを有効にする (コンソール)」を参照してください。

管理アクセスを持つユーザーを作成する

1. IAM アイデンティティセンターを有効にします。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「AWS IAM Identity Centerの有効化」を参照してください。

2. IAM アイデンティティセンターで、ユーザーに管理アクセスを付与します。

   を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、「 ユーザーガイド」の「デフォルト でユーザーアクセス IAM アイデンティティセンターディレクトリを設定するAWS IAM Identity Center 」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

  IAM Identity Center ユーザーを使用してサインインする方法については、「 AWS サインイン ユーザーガイド」の AWS 「 アクセスポータルにサインインする」を参照してください。

追加のユーザーにアクセス権を割り当てる

1. IAM アイデンティティセンターで、最小特権のアクセス許可を適用するというベストプラクティスに従ったアクセス許可セットを作成します。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」を参照してください。

2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

手順については、「AWS IAM Identity Center ユーザーガイド」の「グループの参加」を参照してください。

AWS IoT SiteWise デモを使う

AWS IoT SiteWise AWS IoT SiteWise デモを使うと簡単に探索できます。 AWS IoT SiteWise AWS CloudFormation デモをテンプレートとして提供しており、これをデプロイしてアセットモデル、アセット、 SiteWise モニターポータルを作成し、最大 1 週間分のサンプルデータを生成できます。

> **Important**
>
> デモを作成すると、このデモで作成および消費されるリソースに対して課金が開始されます。

トピック

- デモの作成 AWS IoT SiteWise
- AWS IoT SiteWise デモを削除する。

デモの作成 AWS IoT SiteWise

AWS IoT SiteWise AWS IoT SiteWise デモはコンソールから作成できます。

> **Note**
>
> デモでは、Lambda 関数、1 CloudWatch つのイベントルール、およびデモに必要な AWS Identity and Access Management (IAM) ロールを作成します。次のようなリソースがに表示される場合があります。 AWS アカウントデモを終了するまで、これらのリソースを維持することをお勧めします。これらのリソースを削除すると、デモが正常に動作しなくなることがあります。

AWS IoT SiteWise コンソールでデモを作成するには

1. AWS IoT SiteWise コンソールに移動し、SiteWise ページの右上隅でデモを探します。

2. (オプション) SiteWise デモで、「デモアセットを保管する日数」フィールドを変更して、デモを削除する前に保存する日数を指定します。
3. (オプション) SiteWise サンプルデータを監視するモニターポータルを作成するには、次の操作を行います。

> **Note**
>
> SiteWise このデモで作成および消費されるモニターリソースに対して課金されます。詳しくは、AWS IoT SiteWise 価格の「SiteWise モニター」を参照してください。

a. [リソースをモニタリング]を選択します。
b. 「許可」を選択します。
c. フェデレーティッド IAM ユーザーにポータルへのアクセス権を付与する既存の IAM ロールを選択します。

> **Important**
>
> IAM ロールには、以下のアクセス許可が必要です。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:Describe*",
                "iotsitewise:List*",
                "iotsitewise:Get*",
                "cloudformation:DescribeStacks",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedRolePolicies",
                "sso:DescribeRegisteredRegions",
                "organizations:DescribeOrganization"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Monitor の使用方法について詳しくは、「 SiteWise Monitor とは AWS IoT SiteWise Monitor?」を参照してください。 『AWS IoT SiteWise Monitor アプリケーションガイド』の。

4. [デモの作成] を選択します。

   デモの作成には約 3 分かかります。デモの作成に失敗した場合は、アカウントに十分な権限がない可能性があります。管理者権限を持つアカウントに切り替えるか、次の手順に従ってデモを削除し、もう一度試してください。

   a. [デモの削除] を選択します。

      デモの削除には約 15 分かかります。

   b. デモが削除されない場合は、AWS CloudFormation コンソールを開いて IoT という名前のスタックを選択しSiteWiseDemoAssets、右上隅の [削除] を選択します。

   c. デモを再度削除できない場合は、 AWS CloudFormation コンソールの手順に従い、削除に失敗したリソースをスキップして、もう一度試してください。

5. デモが正常に作成されたら、[AWS IoT SiteWise console] (コンソール) でデモアセットとデータを調べることができます。

AWS IoT SiteWise デモを削除する。

AWS IoT SiteWise デモは 1 週間後、 AWS CloudFormation またはコンソールからデモスタックを作成した場合は選択した日数後に自動的に削除されます。デモリソースの使用が完了したら、決められた日数が経過する前にデモを削除することができます。デモの作成に失敗した場合も、デモを削除することができます。デモを手動で削除するには、次の手順に従います。

デモを削除するには AWS IoT SiteWise

1. AWS CloudFormation コンソールに移動します。
2. [スタック] のリストから [IoTSiteWiseDemoAssets] を選択します。
3. [Delete (削除)] を選択します。

   スタックを消去すると、デモ用に作成されたすべてのリソースが消去されます。

4. 確認ダイアログで、[スタックの削除] を選択します。

スタックの消去には約 15 分かかります。デモの消去に失敗した場合は、右上隅の [消去] を再度選択します。デモを再度削除できない場合は、 AWS CloudFormation コンソールの手順に従って削除に失敗したリソースをスキップし、もう一度試してください。

AWS IoT SiteWise チュートリアル

AWS IoT SiteWise チュートリアルページへようこそ。増え続けるこのチュートリアルのコレクションは、の複雑さを乗り越えるために必要な知識とスキルを身に付けるのに役立ちます。 AWS IoT SiteWiseこれらのチュートリアルでは、ユーザーのニーズに応えられるよう、さまざまな基本トピックを用意しています。チュートリアルを掘り下げていくうちに、のさまざまな側面に関する貴重な洞察を発見してください。 AWS IoT SiteWise

各チュートリアルでは、特定の機器の例を使用しています。これらのチュートリアルはテスト環境を対象としており、架空の会社名、モデル、アセット、プロパティなどを使用しています。目的は、一般的なガイダンスを提供することです。チュートリアルは、組織固有のニーズに合わせて入念なレビューと調整を行わずに、実稼働環境で直接使用することを意図したものではありません。

トピック

- でのOEEの計算 AWS IoT SiteWise
- モノからのデータの取り込み AWS IoT
- Monitor SiteWise での風力発電所データの視覚化と共有
- プロパティ値の更新を Amazon DynamoDB に公開する

でのOEEの計算 AWS IoT SiteWise

このチュートリアルでは、製造工程の総合設備効率 (OEE) をコンピューティングする方法の例を示します。その結果、実際の OEE の計算または式は、ここに示すものとは異なる場合があります。一般に、OEE は `Availability * Quality * Performance` として定義されます。OEEの算出について詳しく知りたい方は、[Wikipedia] (ウィキペディア) の[Overall equipment effectiveness] (機器全体の効果) をご覧ください。

前提条件

このチュートリアルを完了するには、次の 3 つのデータストリームを持つデバイスのデータ取り込みを設定する必要があります。

- `Equipment_State` - アイドル、障害、計画された停止、通常の稼働など、マシンの状態を表す数値コード。
- `Good_Count` - 最後のデータポイント以降に成功した操作の数が各データポイントに含まれているデータストリーム。

- Bad_Count - 最後のデータポイント以降に失敗した操作の数が各データポイントに含まれているデータストリーム。

データ取り込みを設定するには、「へのデータの取り込み AWS IoT SiteWise」を参照してください。使用可能な産業オペレーションがない場合は、 AWS IoT SiteWise API を使用してサンプルデータを生成およびアップロードするスクリプトを作成できます。

OEE の計算方法

このチュートリアルでは、Equipment_State、Good_Count、Bad_Count の 3 つのデータ入力ストリームから OEE を計算するアセットモデルを作成します。この例では、砂糖、ポテトチップス、またはペンキなどの包装に使用される汎用包装機器を考えてみます。AWS IoT SiteWise コンソールで、以下の測定値、変換、 AWS IoT SiteWise 指標を含む資産モデルを作成します。次に、パッケージングマシンを表すアセットを作成して、OEE AWS IoT SiteWise がどのように計算されるかを確認できます。

包装機器からの生データストリームを表すために、次の測定を定義します。

測定

- Equipment_State - 数値コードで包装機器の現在の状態を提供するデータストリーム (または測定)。
  - 1024 - マシンはアイドル状態です。
  - 1020 - エラーや遅延などの障害。
  - 1000 - 計画された停止。
  - 1111 - 通常のオペレーション。
- Good_Count - 最後のデータポイント以降に成功した操作の数が各データポイントに含まれているデータストリーム。
- Bad_Count - 最後のデータポイント以降に失敗した操作の数が各データポイントに含まれているデータストリーム。

Equipment_State 測定値データストリームとそれに含まれるコードを使用して、次の変換 (または派生の測定値) を定義できます one-to-one 変換は未加工の測定値と関係があります。

変換

- `Idle = eq(Equipment_State, 1024)` - マシンのアイドル状態を含む変換されたデータストリーム。
- `Fault = eq(Equipment_State, 1020)` - マシンの故障状態を含む変換されたデータストリーム。
- `Stop = eq(Equipment_State, 1000)` - マシンの計画された停止状態を含む変換されたデータストリーム。
- `Running = eq(Equipment_State, 1111)` - マシンの通常の動作状態を含む変換されたデータストリーム。

生の測定値と変換された測定値を使用して、指定した時間間隔でマシンデータを集計する次のメトリクスを定義します。このセクションでメトリクスを定義するときは、各メトリクスに同じ時間間隔を選択します。

メトリクス

- `Successes = sum(Good_Count)` - 指定した時間間隔に正常に包装されたパッケージの数。
- `Failures = sum(Bad_Count)` 指定した時間間隔に正常に包装されなかったパッケージの数。
- `Idle_Time = statetime(Idle)` - 指定した時間間隔あたりのマシンの合計アイドル時間 (秒単位)。
- `Fault_Time = statetime(Fault)` - 指定した時間間隔あたりのマシンの合計故障時間 (秒単位)。
- `Stop_Time = statetime(Stop)` - 指定した時間間隔あたりのマシンの計画された停止時間の合計 (秒単位)。
- `Run_Time = statetime(Running)` - 指定した時間間隔あたりのマシンの問題なく実行されている合計時間 (秒単位)。
- `Down_Time = Idle_Time + Fault_Time + Stop_Time` 指定された時間間隔でのマシンの合計ダウンタイム (秒単位)。 以外のマシン状態の合計としてコンピューティングされます。`Run_Time`
- `Availability = Run_Time / (Run_Time + Down_Time)` - 指定された時間間隔のマシンの稼働時間またはマシンが動作できるようスケジュールされた時間の割合。
- `Quality = Successes / (Successes + Failures)` - 指定した時間間隔に正常に包装されたパッケージのマシンの割合。

- `Performance = ((Successes + Failures) / Run_Time) / Ideal_Run_Rate` - 指定された時間間隔でのマシンのパフォーマンスを、プロセスの理想的な実行レート (秒単位) からのパーセンテージで示します。

  たとえば、`Ideal_Run_Rate` が 1 分あたり 60 個のパッケージ (1 秒あたり 1 個のパッケージ) であるとします。`Ideal_Run_Rate` が 1 分単位または 1 時間単位の場合、 `Run_Time`は秒単位であるため適切な単位変換係数で割る必要があります。

- `OEE = Availability * Quality * Performance` - 指定された時間間隔における機械の全体的な設備効率。この式は、OEE を 1 に対する分数として計算します。

モノからのデータの取り込み AWS IoT

このチュートリアルでは、 AWS IoT SiteWise AWS IoT デバイスシャドウを使用してさまざまなものからデータを取り込む方法を学びます。デバイスシャドウは、デバイスの現在の状態情報を保存する JSON オブジェクトです。 AWS IoT 詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Device shadow service] (デバイスシャドウサービス) を参照してください。

このチュートリアルを完了すると、 AWS IoT SiteWise AWS IoT 事柄に基づいて操作を設定できます。 AWS IoT Thing を使用すると、作成した操作をの他の便利な機能と統合できます AWS IoT。たとえば、 AWS IoT 以下のタスクを実行するように機能を設定できます。

- Amazon DynamoDB AWS IoT Eventsなどにデータをストリーミングするための追加ルールを設定します。 AWS のサービス詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Rules] (ルール) を参照してください。
- AWS IoT フリートインデックスサービスを使用して、デバイスデータのインデックス、検索、集計を行います。詳細については、「AWS IoT デベロッパーガイド」の「フリートインデックス作成サービス」を参照してください。
- AWS IoT Device Defenderを使用してデバイスを監査し、保護します。詳細については、「AWS IoT デベロッパーガイド」の 「AWS IoT Device Defender」を参照してください。

このチュートリアルでは、 AWS IoT モノのデバイスシャドウからアセットにデータを取り込む方法を学びます。 AWS IoT SiteWiseそのためには、1 AWS IoT つまたは複数のモノを作成し、CPU とメモリの使用状況データで各モノのデバイスシャドウを更新するスクリプトを実行します。このチュートリアルの CPU およびメモリ使用率データを使用して、現実的なセンサーデータを模倣します。次に、 AWS IoT SiteWise AWS IoT SiteWise モノのデバイスシャドウが更新されるたびにこのデータ

をアセットに送信するアクションを含むルールを作成します。詳細については、「ルールを使ったデータの取り込み AWS IoT Core」を参照してください。

トピック

前提条件

このチュートリアルを完了するには、以下が必要です。

- と AWS アカウント。アカウントをお持ちでない場合は、「のセットアップ AWS アカウント」を参照してください。
- Windows、macOSLinux、Unixを実行してにアクセスする開発用コンピュータ AWS Management Console。詳細については、「AWS Management Consoleの開始方法」を参照してください。
- 管理者権限を持つ AWS Identity and Access Management (IAM) ユーザー。
- Python3 開発用コンピュータにインストールされている、または AWS IoT THING として登録したいデバイスにインストールされている。

ステップ 1: AWS IoT ポリシーの作成

この手順では、このチュートリアルで使用されているリソースに AWS IoT Things AWS IoT がアクセスできるようにするポリシーを作成します。

AWS IoT ポリシーを作成するには

1. AWS Management Consoleにサインインします。

2. AWSAWS IoT SiteWise サポートされているリージョンを確認してください。必要に応じて、これらのサポートされているいずれかのリージョンに切り替えます。
3. AWS IoT コンソールに移動します。[デバイスのConnect] ボタンが表示されたら、それを選択します。
4. 左のナビゲーションペインで、[安全性] を選択し、[ポリシー] を選択します。
5. [作成] を選択します。
6. AWS IoT ポリシーの名前 (例:**SiteWiseTutorialDevicePolicy**) を入力します。
7. [ポリシードキュメント] で [JSON] を選択して JSON 形式で次のポリシーを入力します。お客様のリージョンとアカウント ID で *region* と *account-id* を置き換えます (**us-east-1**、**123456789012** など)。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:Connect",
      "Resource": "arn:aws:iot:region:account-id:client/SiteWiseTutorialDevice*"
    },
    {
      "Effect": "Allow",
      "Action": "iot:Publish",
      "Resource": [
        "arn:aws:iot:region:account-id:topic/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/update",
        "arn:aws:iot:region:account-id:topic/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/delete",
        "arn:aws:iot:region:account-id:topic/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/get"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "iot:Receive",
      "Resource": [
        "arn:aws:iot:region:account-id:topic/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/update/accepted",
        "arn:aws:iot:region:account-id:topic/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/delete/accepted",
```

```
        "arn:aws:iot:region:account-id:topic/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/get/accepted",
        "arn:aws:iot:region:account-id:topic/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/update/rejected",
        "arn:aws:iot:region:account-id:topic/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/delete/rejected"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "iot:Subscribe",
      "Resource": [
        "arn:aws:iot:region:account-id:topicfilter/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/update/accepted",
        "arn:aws:iot:region:account-id:topicfilter/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/delete/accepted",
        "arn:aws:iot:region:account-id:topicfilter/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/get/accepted",
        "arn:aws:iot:region:account-id:topicfilter/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/update/rejected",
        "arn:aws:iot:region:account-id:topicfilter/$aws/things/
${iot:Connection.Thing.ThingName}/shadow/delete/rejected"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "iot:GetThingShadow",
        "iot:UpdateThingShadow",
        "iot:DeleteThingShadow"
      ],
      "Resource": "arn:aws:iot:region:account-id:thing/SiteWiseTutorialDevice*"
    }
  ]
}
```

このポリシーにより、 AWS IoT デバイスは接続を確立し、MQTT メッセージを使用してデバイスシャドウと通信できるようになります。MQTT メッセージについて詳しくは、「MQTT とは?」を参照してください。 。デバイスシャドウを操作するには、で始まるトピックの AWS IoT MQTT メッセージをパブリッシュおよび受信します。$aws/things/*thing-name*/shadow/このポリシーには、という Thing ポリシー変数が組み込まれています。${iot:Connection.Thing.ThingName}この変数は、各トピック内の接続されて

いる Thing の名前を置き換えます。`iot:Connect`このステートメントは、接続を確立できるデバイスに制限を設け、Thing ポリシー変数がで始まる名前だけに置き換えられるようにしています。`SiteWiseTutorialDevice`

詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Thing policy variables] (シングポリシー変数) を参照してください。

> **Note**
>
> このポリシーは、名前が `SiteWiseTutorialDevice` で始まるモノに適用されます。モノに別の名前を使用するには、それに応じてポリシーを更新する必要があります。

8. [作成] を選択します。

ステップ 2: AWS IoT のモノを作成および設定する

この手順では、 AWS IoT 何でも作成して設定します。開発用コンピューターは任意の AWS IoT Thing として指定できます。進歩するにつれ、ここで学んでいる原則は実際のプロジェクトにも適用できることを覚えておいてください。FreeRTOS を含め、 AWS IoT AWS IoT Greengrass SDKを実行できるあらゆるデバイスで柔軟に作成および設定できます AWS IoT 。詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[AWS IoT SDKs] (SDK) を参照してください。

何でも作成して設定するには AWS IoT

1. コマンドラインを開き、次のコマンドを実行して、このチュートリアル用のディレクトリを作成します。

```
mkdir iot-sitewise-rule-tutorial
cd iot-sitewise-rule-tutorial
```

2. 次のコマンドを実行して、モノの証明書用のディレクトリを作成します。

```
mkdir device1
```

追加のモノを作成する場合は、それに応じてディレクトリ名の番号を増やして、どの証明書がどのモノに属しているかをわかるようにします。

3. AWS IoT コンソールに移動します。

4. 左のナビゲーションペインで、「管理」セクションの「すべてのデバイス」を選択します。[モノ] を選択します。
5. [まだモノがありません] ダイアログボックスが表示された場合は、[モノの作成] を選択します。それ以外の場合は、[モノの作成] を選択します。
6. [モノを作成する] ページで、[単一のモノを作成する] を選択し、[次へ] を選択します。
7. [モノのプロパティを指定する] ページで、 AWS IoT のモノの名前 (例: **`SiteWiseTutorialDevice1`**) を入力し、[次へ] を選択します。追加のモノを作成する場合は、それに応じてモノの名前の番号を増やします。

   > **Important**
   >
   > Thing の名前は、「ステップ 1: ポリシーを作成する」 AWS IoT で作成したポリシーで使用した名前と一致する必要があります。そうしないと、デバイスはに接続できません AWS IoT。

8. [デバイス証明書を構成する - オプション] ページで [新しい証明書を自動生成する (推奨)] を選択し、[次へ] を選択します。証明書を使用すると AWS IoT 、デバイスを安全に識別できます。
9. 「ポリシーを証明書に添付-オプション」ページで、「ステップ 1: ポリシーの作成」 AWS IoT で作成したポリシーを選択し、「Create thing」を選択します。
10. [証明書とキーのダウンロード] ダイアログボックスで、次の操作を行います。

    a. [ダウンロード] リンクを選択して、モノの証明書、パブリックキー、プライベートキーをダウンロードします。モノの証明書用に作成したディレクトリ (例: `iot-sitewise-rule-tutorial/device1`) に 3 つのファイルをすべて保存します。

       > **Important**
       >
       > モノの証明書とキーをダウンロードできるのは今回だけです。この証明書とキーは、デバイスが AWS IoTに正常に接続するために必要です。

    b. [ダウンロード] のリンクを選択し、ルート CA 証明書をダウンロードします。ルート CA 証明書を `iot-sitewise-rule-tutorial` に保存します。Amazon Root CA 1 をダウンロードすることをお勧めします。

11. [完了] をクリックします。

これで、 AWS IoT コンピューターへのモノの登録が完了しました。次のステップのいずれかを実行してください。

- 「ステップ 3: デバイス資産モデルを作成する」に進んでください。 AWS IoT 追加項目は作成しないでください。このチュートリアルは、1 つのモノだけで完了できます。
- 別のコンピュータまたはデバイスでこのセクションの手順を繰り返して、さらに多くの AWS IoT のモノを作成します。このチュートリアルでは、複数のデバイスから一意の CPU およびメモリ使用率データを取り込むことができるよう、このオプションに従うことをお勧めします。
- 同じデバイス (コンピュータ) でこのセクションのステップを繰り返して、さらに多くの AWS IoT のモノを作成します。 AWS IoT それぞれがコンピューターから同等の CPU とメモリ使用量データを受け取るので、この方法を使用して複数のデバイスから一意でないデータを取り込む方法を示します。

ステップ 3: デバイスアセットモデルを作成する

この手順では、CPU とメモリの使用状況データをストリーミングするデバイスを表すアセットモデルを作成します。 AWS IoT SiteWise デバイスのグループを表すアセット内のデータを処理するために、アセットモデルは同じタイプの複数のアセットにわたって一貫した情報を適用します。詳細については、「産業用アセットのモデリング」を参照してください。

デバイスを表すアセットモデルを作成するには

1. AWS IoT SiteWise コンソールに移動します。
2. 左のナビゲーションペインで [モデル] を選択します。
3. [モデルの作成] を選択します。
4. [モデルの詳細] で、モデルの名前を入力します。例えば **`SiteWise Tutorial Device Model`** です。
5. [測定定義] で、次の操作を実行します。

   a. [名前] に「**`CPU Usage`**」と入力します。

   b. [単位] に「`%`」と入力します。

   c. [データ型] は [倍精度浮動小数点型] のままにします。

   測定プロパティは、デバイスの raw データストリームを表します。詳細については、「機器からのデータストリームの定義 (測定値) 」を参照してください。

6. 2 つ目の測定プロパティを追加する場合は、[新しい測定の追加] を選択します。
7. [測定定義] の 2 行目で、次の操作を実行します。
   a. [名前] に「**Memory Usage**」と入力します。
   b. [単位] に「**%**」と入力します。
   c. [データ型] は [倍精度浮動小数点型] のままにします。
8. [メトリクス定義] で、次の操作を実行します。
   a. [名前] に「**Average CPU Usage**」と入力します。
   b. [計算式] に「**avg(CPU Usage)**」と入力します。表示されたら、自動入力リストから CPU Usage を選択します。
   c. [時間間隔] に「**5 minutes**」と入力します。

   メトリクスプロパティは、ある間隔ですべての入力データポイントを処理し、間隔ごとに 1 つのデータポイントを出力する集計計算を定義します。このメトリクスプロパティは、各デバイスの平均 CPU 使用率を 5 分ごとに計算します。詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください。
9. [新しいメトリクスの追加] を選択して、2 つ目のメトリクスプロパティを追加します。
10. [メトリクス定義] の 2 行目で、次の操作を実行します。
    a. [名前] に「**Average Memory Usage**」と入力します。
    b. [計算式] に「**avg(Memory Usage)**」と入力します。表示されたら、自動入力リストから Memory Usage を選択します。
    c. [Time interval (時間間隔)] に「**5 minutes**」と入力します。

    このメトリクスプロパティは、各デバイスの平均メモリ使用率を 5 分ごとに計算します。
11. (オプション) デバイスごとのコンピューティングに関心のあるその他のメトリクスを追加します。興味深い関数には、`min`、`max` などがあります。詳細については、「数式の使用」を参照してください。「ステップ 4: デバイスフリートアセットモデルを作成する」では、デバイスのフリート全体のデータを使用してメトリクスを計算できる親アセットを作成します。
12. [モデルの作成] を選択します。

ステップ 4: デバイスフリートアセットモデルを作成する

この手順では、 AWS IoT SiteWise デバイスコレクションをシンボル化するアセットモデルを作成します。このアセットモデルでは、多数のデバイスアセットを 1 つの包括的なフリートアセットにリンクできる構造を確立します。その後、フリートアセットモデルのメトリクスの概要を説明して、接続されているすべてのデバイスアセットのデータを統合します。このアプローチにより、保有車両全体の総合的なパフォーマンスに関する包括的なインサイトが得られます。

デバイスフリートを表すアセットモデルを作成するには

1. AWS IoT SiteWise コンソールに移動します。
2. 左のナビゲーションペインで [モデル] を選択します。
3. [モデルの作成] を選択します。
4. [モデルの詳細] で、モデルの名前を入力します。例えば **SiteWise Tutorial Device Fleet Model** です。
5. [階層定義] で、次の操作を実行します。

   a. [階層名] に「**Device**」と入力します。

   b. [階層モデル] で、デバイスアセットモデル (**SiteWise Tutorial Device Model**) を選択します。

   階層は、親 (フリート) アセットモデルと子 (デバイス) アセットモデルの間の関係を定義します。親アセットは、子アセットのプロパティデータにアクセスできます。後でアセットを作成する場合は、親アセットモデルの階層定義に従って、子アセットを親アセットに関連付ける必要があります。詳細については、「アセットモデル階層の定義」を参照してください。
6. [Metric definitions (メトリクス定義)] で、次の操作を実行します。

   a. [名前] に「**Average CPU Usage**」と入力します。

   b. [計算式] に「**avg(Device | Average CPU Usage)**」と入力します。自動入力リストが表示されたら、[Device] を選択して階層を選択し、[Average CPU Usage] を選択して前に作成したデバイスアセットからメトリクスを選択します。

   c. [Time interval (時間間隔)] に「**5 minutes**」と入力します。

   このメトリクスプロパティは、**Device** 階層を通じてフリートアセットに関連付けられているすべてのデバイスアセットの平均 CPU 使用率を計算します。

7. [新しいメトリクスの追加] を選択して、2 つ目のメトリクスプロパティを追加します。
8. [メトリクス定義] の 2 行目で、次の操作を実行します。

   a. [名前] に「**Average Memory Usage**」と入力します。

   b. [計算式] に「**avg(Device | Average Memory Usage)**」と入力します。自動入力リストが表示されたら、[Device] を選択して階層を選択し、[Average Memory Usage] を選択して前に作成したデバイスアセットからメトリクスを選択します。

   c. [Time interval (時間間隔)] に「**5 minutes**」と入力します。

   このメトリクスプロパティは、**Device** 階層を通じてフリートアセットに関連付けられているすべてのデバイスアセットの平均メモリ使用率を計算します。
9. (オプション) デバイスのフリート全体のコンピューティングに関心のあるその他のメトリクスを追加します。
10. [モデルの作成] を選択します。

ステップ 5: デバイスアセットを作成および設定する

この手順では、デバイスアセットモデルに基づいてデバイスアセットを生成します。次に、測定プロパティごとにプロパティエイリアスを定義します。プロパティエイリアスは、アセットのプロパティを識別する一意の文字列です。後で、アセット ID とプロパティ ID の代わりにエイリアスを使用して、データをアップロードするプロパティを特定できます。詳細については、「アセットプロパティへの産業データストリームのマッピング」を参照してください。

デバイスアセットを作成し、プロパティエイリアスを定義するには

1. AWS IoT SiteWise コンソールに移動します。
2. 左側のナビゲーションペインで [アセット] を選択します。
3. [アセットの作成] を選択します。
4. [モデル情報] で、デバイスアセットモデル **SiteWise Tutorial Device Model** を選択します。
5. [アセット情報] で、アセットの名前を入力します。例えば **SiteWise Tutorial Device 1** です。
6. [アセットの作成] を選択します。
7. 新しいデバイスアセットの場合は、 [編集] を選択します。

8. CPU Usage で、プロパティエイリアスとして **/tutorial/device/SiteWiseTutorialDevice1/cpu** と入力します。 AWS IoT プロパティエイリアスにはモノの名前を含めると、1 AWS IoT つのルールを使用してすべてのデバイスからデータを取り込むことができます。
9. Memory Usage で、プロパティエイリアスとして **/tutorial/device/SiteWiseTutorialDevice1/memory** と入力します。
10. [保存] を選択します。

AWS IoT 以前に複数のものを作成した場合は、デバイスごとにステップ 3 ~ 10 を繰り返し、それに応じてアセット名とプロパティエイリアスの数を増やします。たとえば、2 つ目のデバイスアセットの名前は **SiteWise Tutorial Device 2** となり、そのプロパティエイリアスは **/tutorial/device/SiteWiseTutorialDevice2/cpu** および **/tutorial/device/SiteWiseTutorialDevice2/memory** となります。

ステップ 6: デバイスフリートアセットを作成および設定する

この手順では、デバイスフリートアセットモデルから派生したデバイスフリートアセットを作成します。次に、個々のデバイスアセットをフリートアセットにリンクします。この関連付けにより、フリートアセットのメトリックプロパティで複数のデバイスからのデータをコンパイルして分析できるようになります。このデータにより、フリート全体のパフォーマンスをまとめて把握できます。

デバイスフリートアセットを作成してデバイスアセットを関連付けるには

1. AWS IoT SiteWise コンソールに移動します。
2. 左側のナビゲーションペインで [アセット] を選択します。
3. [アセットの作成] を選択します。
4. [モデル情報]で、デバイスフリートアセットモデル **SiteWise Tutorial Device Fleet Model** を選択します。
5. [アセット情報] で、アセットの名前を入力します。例えば **SiteWise Tutorial Device Fleet 1** です。
6. [アセットの作成] を選択します。
7. 新しいデバイスフリートアセットの場合は、 [編集] を選択します。
8. 「この資産に関連する資産」で「関連資産を追加」を選択し、次の操作を行います。

a. [階層] で、[Device] を選択します。この階層は、デバイスとフリートアセットの間の階層関係を識別します。この階層は、このチュートリアルで前にデバイスフリートアセットモデルで定義しました。

b. [アセット] で、デバイスアセット (SiteWise Tutorial Device 1) を選択します。

9. (オプション) 以前に複数のデバイスアセットを作成した場合は、作成したデバイスアセットごとにステップ 8～10 を繰り返します。

10. [保存] を選択します。

これで、デバイスアセットが階層として編成されているのがわかります。

ステップ 7: AWS IoT Core でデバイスアセットにデータを送信するルールを作成する

この手順では、でルールを設定します AWS IoT Core。このルールは、デバイスシャドウからの通知メッセージを解釈し、そのデータをデバイスアセットに送信するように設計されています AWS IoT SiteWise。デバイスのシャドウが更新されるたびに、MQTT AWS IoT メッセージが送信されます。MQTT メッセージに基づいてデバイスのシャドウが変更されたときにアクションを実行するルールを作成できます。この場合の目的は、更新メッセージを処理し、プロパティ値を抽出し、その値を内のデバイスアセットに送信することです。 AWS IoT SiteWise

AWS IoT SiteWise アクションを含むルールを作成するには

1. AWS IoT コンソールに移動します。
2. 左側のナビゲーションペインで [メッセージルーティング] を選択し、[ルール] を選択します。
3. [ルールの作成] を選択します。
4. ルールの名前と説明を入力し、[次へ] を選択します。
5. 次の SQL ステートメントを入力し、[次へ] を選択します。

```
SELECT
  *
FROM
  '$aws/things/+/shadow/update/accepted'
WHERE
  startsWith(topic(3), "SiteWiseTutorialDevice")
```

このルールクエリステートメントは、デバイスシャドウサービスがシャドウの更新を `$aws/things/thingName/shadow/update/accepted` にパブリッシュするため、機能します。デバイスシャドウの詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の [Device shadow service] (デバイスシャドウサービス) を参照してください。

`WHERE` 句では、このルールクエリステートメントは `topic(3)` 関数を使用して、トピックの 3 番目のセグメントからモノの名前を取得します。次に、このステートメントは、名前がチュートリアルのデバイス名と一致しないデバイスを除外します。 AWS IoT SQL について詳しくは、『AWS IoT 開発者ガイド』AWS IoT の「SQL リファレンス」を参照してください。

6. [ルールのアクション] で、[AWS IoT SiteWiseのアセットプロパティにメッセージデータを送信する] を選択し、次の操作を行います。

   a. [プロパティエイリアスによる] を選択します。

   b. [Pプロパティエイリアス] に「**`/tutorial/device/${topic(3)}/cpu`**」と入力します。

      `${...}`構文は代替テンプレートです。 AWS IoT 中括弧内の内容を評価します。この置換テンプレートは、トピックからモノの名前を取得し、モノごとに固有のエイリアスを作成します。詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Substitution templates] (置換テンプレート) を参照してください。

      > **Note**
      >
      > 置換テンプレート内の式は `SELECT` ステートメントとは別に評価されるため、`AS` 句を使用して作成されたエイリアスは置換テンプレートを使用して参照することはできません。サポートされている関数と演算子に加えて、元のペイロードに存在する情報のみを参照できます。

   c. [エントリ ID - オプション] に **`${concat(topic(3), "-cpu-", floor(state.reported.timestamp))}`** と入力します。

      エントリ ID は、各値エントリの試行を一意に識別します。エントリがエラーを返す場合は、エラーの出力でエントリ ID を検索して、問題をトラブルシューティングすることができます。このエントリ ID の置換テンプレートは、モノの名前とデバイスの報告されたタイムスタンプを組み合わせたものです。たとえば、結果のエントリ ID は `SiteWiseTutorialDevice1-cpu-1579808494` のようになります。

   d. [時間 (秒)] に「**`${floor(state.reported.timestamp)}`**」と入力します。

この置換テンプレートは、デバイスの報告されたタイムスタンプから秒単位で時間を計算します。このチュートリアルでは、デバイスは Unix エポック時間でタイムスタンプ (秒) を浮動小数点数として報告します。

e. [オフセット (ナノ秒) - オプション] に **`${floor((state.reported.timestamp % 1) * 1E9)}`** と入力します。

この置換テンプレートは、デバイスの報告されたタイムスタンプの小数部分を変換して、秒単位の時間からのナノ秒オフセットを計算します。

> **Note**
>
> AWS IoT SiteWise データには Unix エポックタイムでの現在のタイムスタンプが必要です。デバイスが正確に時間を報告しない場合、[timestamp ()] (タイムスタンプ) を使って AWS IoT ルールエンジンから現在の時間を取得することができます。この関数は、時刻をミリ秒単位で報告するため、ルールアクションの時間パラメータを次の値に更新する必要があります。
>
> - [Time in seconds (時間 (秒))] に「**`${floor(timestamp() / 1E3)}`**」と入力します。
> - [オフセット (ナノ秒)] に「**`${(timestamp() % 1E3) * 1E6}`**と入力します。

f. [データ型] で、[Double (倍精度浮動小数点型)] を選択します。

このデータ型は、アセットモデルで定義したアセットプロパティのデータ型と一致する必要があります。

g. [値] には「**`${state.reported.cpu}`**」と入力します。置換テンプレートでは、. 演算子を使用して JSON 構造内から値を取得します。

h. [エントリの追加] を選択して、メモリ使用率プロパティに新しいエントリを追加し、そのプロパティに対して次の手順を再度実行します。

i. [プロパティエイリアスによる] を選択します。

ii. [Pプロパティエイリアス] に「**`/tutorial/device/${topic(3)}/memory`**」と入力します。

iii. [エントリ ID - オプション] に **`${concat(topic(3), "-memory-", floor(state.reported.timestamp))}`** と入力します。

iv. [時間 (秒)] に「**`${floor(state.reported.timestamp)}`**」と入力します。

v. [オフセット (ナノ秒) - オプション] に **${floor((state.reported.timestamp % 1) * 1E9)}** と入力します。

vi. [データ型] で、[倍精度浮動小数点型] を選択します。

vii. [値] には「**${state.reported.memory}**」と入力します。

i. [ロール] で、[ロールの作成] を選択して、このルールアクションに対する IAM ロールを作成します。 AWS IoT このロールにより、デバイスフリートアセットとそのアセット階層内のプロパティにデータをプッシュできます。

j. ロール名を入力し、[作成] を選択します。

7. (オプション) ルールのトラブルシューティングに使用できるエラーアクションを設定します。詳細については、「ルールのトラブルシューティング」を参照してください。

8. [次へ] を選択します。

9. 設定を確認し、[作成] を選択してルールを作成します。

ステップ 8: デバイスクライアントスクリプトを実行する

このチュートリアルでは、実際のデバイスを使用してデータを報告することはありません。代わりに、スクリプトを実行して AWS IoT Thing のデバイスシャドウを CPU とメモリの使用量で更新し、実際のセンサーデータを模倣します。スクリプトを実行するには、Pythonまず必要なパッケージをインストールする必要があります。この手順では、Python必要なパッケージをインストールし、デバイスクライアントスクリプトを実行します。

デバイスクライアントスクリプトを設定および実行するには

1. AWS IoT コンソールに移動します。

2. 左側のナビゲーションペインの下部にある [設定] を選択します。

3. デバイスクライアントスクリプトで使用するためにカスタムエンドポイントを保存します。このエンドポイントを使用して、モノのシャドウとやり取りします。このエンドポイントは、現在のリージョンのアカウントに固有です。

   カスタムエンドポイントは、次の例のようになります。

   ```
   identifier.iot.region.amazonaws.com
   ```

4. コマンドラインを開き、次のコマンドを実行して、先ほど作成したチュートリアル用のディレクトリに移動します。

```
cd iot-sitewise-rule-tutorial
```

5. 以下のコマンドを実行して、 AWS IoT Device SDK for Pythonをインストールします。

```
pip3 install AWSIoTPythonSDK
```

詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[AWS IoT Device SDK for Python]を参照してください。

6. 次のコマンドを実行して、psutil をインストールします。psutil は、クロスプラットフォームプロセスおよびシステムユーティリティライブラリです。

```
pip3 install psutil
```

詳細については、[Python Package Index] (パッケージインデックス) の[psutil]を参照してください。

7. `iot-sitewise-rule-tutorial` ディレクトリ `thing_performance.py` にというファイルを作成し、次の Python コードをファイルにコピーします。

```
import AWSIoTPythonSDK.MQTTLib as AWSIoTPyMQTT

import json
import psutil
import argparse
import logging
import time


# Configures the argument parser for this program.
def configureParser():
    parser = argparse.ArgumentParser()
    parser.add_argument(
        "-e",
        "--endpoint",
        action="store",
        required=True,
        dest="host",
        help="Your AWS IoT custom endpoint",
    )
    parser.add_argument(
```

```
        "-r",
        "--rootCA",
        action="store",
        required=True,
        dest="rootCAPath",
        help="Root CA file path",
    )
    parser.add_argument(
        "-c",
        "--cert",
        action="store",
        required=True,
        dest="certificatePath",
        help="Certificate file path",
    )
    parser.add_argument(
        "-k",
        "--key",
        action="store",
        required=True,
        dest="privateKeyPath",
        help="Private key file path",
    )
    parser.add_argument(
        "-p",
        "--port",
        action="store",
        dest="port",
        type=int,
        default=8883,
        help="Port number override",
    )
    parser.add_argument(
        "-n",
        "--thingName",
        action="store",
        required=True,
        dest="thingName",
        help="Targeted thing name",
    )
    parser.add_argument(
        "-d",
        "--requestDelay",
        action="store",
```

```
        dest="requestDelay",
        type=float,
        default=1,
        help="Time between requests (in seconds)",
    )
    parser.add_argument(
        "-v",
        "--enableLogging",
        action="store_true",
        dest="enableLogging",
        help="Enable logging for the AWS IoT Device SDK for Python",
    )
    return parser


# An MQTT shadow client that uploads device performance data to AWS IoT at a
 regular interval.
class PerformanceShadowClient:
    def __init__(
        self,
        thingName,
        host,
        port,
        rootCAPath,
        privateKeyPath,
        certificatePath,
        requestDelay,
    ):
        self.thingName = thingName
        self.host = host
        self.port = port
        self.rootCAPath = rootCAPath
        self.privateKeyPath = privateKeyPath
        self.certificatePath = certificatePath
        self.requestDelay = requestDelay

    # Updates this thing's shadow with system performance data at a regular
 interval.
    def run(self):
        print("Connecting MQTT client for {}...".format(self.thingName))
        mqttClient = self.configureMQTTClient()
        mqttClient.connect()
        print("MQTT client for {} connected".format(self.thingName))
        deviceShadowHandler = mqttClient.createShadowHandlerWithName(
```

```
            self.thingName, True
        )

        print("Running performance shadow client for {}...
\n".format(self.thingName))
        while True:
            performance = self.readPerformance()
            print("[{}]".format(self.thingName))
            print("CPU:\t{}%".format(performance["cpu"]))
            print("Memory:\t{}%\n".format(performance["memory"]))
            payload = {"state": {"reported": performance}}
            deviceShadowHandler.shadowUpdate(
                json.dumps(payload), self.shadowUpdateCallback, 5
            )
            time.sleep(args.requestDelay)

    # Configures the MQTT shadow client for this thing.
    def configureMQTTClient(self):
        mqttClient = AWSIoTPyMQTT.AWSIoTMQTTShadowClient(self.thingName)
        mqttClient.configureEndpoint(self.host, self.port)
        mqttClient.configureCredentials(
            self.rootCAPath, self.privateKeyPath, self.certificatePath
        )
        mqttClient.configureAutoReconnectBackoffTime(1, 32, 20)
        mqttClient.configureConnectDisconnectTimeout(10)
        mqttClient.configureMQTTOperationTimeout(5)
        return mqttClient

    # Returns the local device's CPU usage, memory usage, and timestamp.
    def readPerformance(self):
        cpu = psutil.cpu_percent()
        memory = psutil.virtual_memory().percent
        timestamp = time.time()
        return {"cpu": cpu, "memory": memory, "timestamp": timestamp}

    # Prints the result of a shadow update call.
    def shadowUpdateCallback(self, payload, responseStatus, token):
        print("[{}]".format(self.thingName))
        print("Update request {} {}\n".format(token, responseStatus))


# Configures debug logging for the AWS IoT Device SDK for Python.
def configureLogging():
    logger = logging.getLogger("AWSIoTPythonSDK.core")
```

```
    logger.setLevel(logging.DEBUG)
    streamHandler = logging.StreamHandler()
    formatter = logging.Formatter(
        "%(asctime)s - %(name)s - %(levelname)s - %(message)s"
    )
    streamHandler.setFormatter(formatter)
    logger.addHandler(streamHandler)


# Runs the performance shadow client with user arguments.
if __name__ == "__main__":
    parser = configureParser()
    args = parser.parse_args()
    if args.enableLogging:
        configureLogging()
    thingClient = PerformanceShadowClient(
        args.thingName,
        args.host,
        args.port,
        args.rootCAPath,
        args.privateKeyPath,
        args.certificatePath,
        args.requestDelay,
    )
    thingClient.run()
```

8. 以下のパラメータを使用して、コマンドラインから thing_performance.py を実行します。

   - -n、 --thingName - モノの名前 (**SiteWiseTutorialDevice1** など)。
   - -e, --endpoint — AWS IoT この手順の前半で保存したカスタムエンドポイント。
   - -r, --rootCA — AWS IoT ルート CA 証明書へのパス。
   - -c, --cert — AWS IoT Thing 証明書へのパス。
   - -k, --key — AWS IoT Thing 証明書のプライベートキーへのパス。
   - -d、 --requestDelay -(オプション) 各デバイスシャドウ更新間の待機時間 (秒)。デフォルトは 1 秒です。
   - -v、 --enableLogging -(オプション) このパラメータが存在する場合、スクリプトによって AWS IoT Device SDK for Pythonからのデバッグメッセージが出力されます。

   コマンドは、次の例のようになります。

```
python3 thing_performance.py \
  --thingName SiteWiseTutorialDevice1 \
  --endpoint identifier.iot.region.amazonaws.com \
  --rootCA AmazonRootCA1.pem \
  --cert device1/thing-id-certificate.pem.crt \
  --key device1/thing-id-private.pem.key
```

AWS IoT 他の目的でスクリプトを実行する場合は、それに応じて Thing 名と証明書ディレクトリを更新してください。

9. デバイスでプログラムを開いたり閉じたりして、CPU およびメモリ使用率がどのように変化するかを確認します。スクリプトにより、各 CPU およびメモリ使用率の読み取りが出力されます。スクリプトがデータをデバイスシャドウサービスに正常にアップロードする場合、スクリプトの出力は次の例のようになります。

```
[SiteWiseTutorialDevice1]
CPU:    24.6%
Memory: 85.2%

[SiteWiseTutorialDevice1]
Update request e6686e44-fca0-44db-aa48-3ca81726f3e3 accepted
```

10. スクリプトがデバイスシャドウを更新していることを確認するには、次の手順を実行します。

    a. AWS IoT コンソールに移動します。

    b. 左側のナビゲーションペインで、[すべてのデバイス] を選択し、[モノ] を選択します。

    c. 好きなものを選んでくださいSiteWiseTutorialDevice。

    d. [デバイスシャドウ] タブを選択し、[クラシックシャドウ] を選択して、シャドウの状態が次の例のようになっていることを確認します。

```
{
  "reported": {
    "cpu": 24.6,
    "memory": 85.2,
    "timestamp": 1579567542.2835066
  }
}
```

Thing のシャドウステートが空の場合、または前の例と異なる場合は、スクリプトが実行されていて、に正常に接続されていることを確認してください AWS IoT。に接続してもスクリプトがタイムアウトし続ける場合は AWS IoT、Thing ポリシーがこのチュートリアルに従って設定されていることを確認してください。

11. ルールアクションが AWS IoT SiteWiseにデータを送信していることを確認するには、次の手順を実行します。

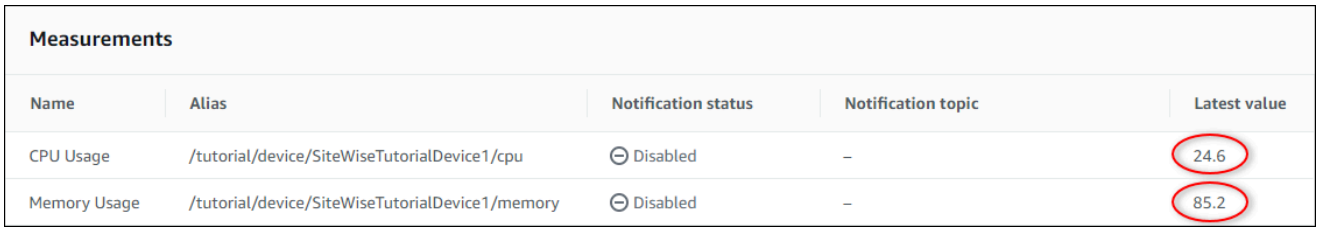
    a. AWS IoT SiteWise コンソールに移動します。

    b. 左側のナビゲーションペインで [Assets (アセット)] を選択します。

    c. デバイスフリートアセット (SiteWise Tutorial Device Fleet 1 1) の横にある矢印を選択してそのアセット階層を展開し、デバイスアセット (SiteWise Tutorial Device 1) を選択します。

    d. [測定] を選択します。

    e. 最新値 セルに、CPU Usage および Memory Usage プロパティの値があることを確認します。



    f. CPU Usage プロパティと Memory Usage プロパティ に最新の値がない場合は、ページを更新します。数分経過しても値が表示されない場合は、「ルールのトラブルシューティング」を参照してください。

12. これで、このチュートリアルは終了です。データのライブ可視化を調べる場合は、 AWS IoT SiteWise Monitorでポータルを設定できます。詳細については、「によるデータの監視 AWS IoT SiteWise Monitor」を参照してください。それ以外の場合は、コマンドプロンプトで Ctrl+C キーを押して、デバイスクライアントスクリプトを停止できます。Python プログラムが、料金がかかる量のメッセージを送信することはほとんどありませんが、終了したらプログラムを停止することをお勧めします。

ステップ 9: チュートリアルの終了後にリソースをクリーンアップする

AWS IoT モノからのデータの取り込みに関するチュートリアルを完了したら、追加料金が発生しないようにリソースをクリーンアップしてください。

で階層型アセットを削除するには AWS IoT SiteWise

1. [AWS IoT SiteWise console] (コンソール) に移動します。
2. 左側のナビゲーションペインで [Assets (アセット)] を選択します。
3. でアセットを削除するときは AWS IoT SiteWise、まずその関連付けを解除する必要があります。

   デバイスフリートアセットからデバイスアセットの関連付けを解除するには、次の手順を実行します。

   a. デバイスフリートアセット (SiteWise Tutorial Device Fleet 1) を選択します。
   b. [編集] を選択します。
   c. [このアセットに関連付けられているアセット] で、このデバイスフリートアセットに関連付けられている各デバイスアセットに対して [関連付け解除] を選択します。
   d. [保存] を選択します。

      これで、デバイスアセットが階層として編成されなくなったことがわかります。
4. デバイスアセット (SiteWise Tutorial Device 1) を選択します。
5. [削除] をクリックします。
6. 確認フィールドに「**Delete**」と入力し、[削除] を選択します。
7. デバイスアセットとデバイスフリートアセット (SiteWise Tutorial Device Fleet 1) ごとに、ステップ 4～6 を繰り返します。

で階層型アセットモデルを削除するには AWS IoT SiteWise

1. AWS IoT SiteWise コンソールに移動します。
2. デバイスおよびデバイスフリートアセットを削除していない場合は、削除します。詳細については、前述の手順を参照してください。モデルから作成されたアセットがある場合は、そのモデルを削除できません。
3. 左のナビゲーションペインで [モデル] を選択します。
4. デバイスフリートアセット (SiteWise Tutorial Device Fleet Model 1) を選択します。

   階層型アセットモデルを削除する場合は、まず親アセットモデルを削除することから始めてください。
5. [削除] をクリックします。

6. 確認フィールドに「**Delete**」と入力し、[削除] を選択します。
7. デバイスのアセットモデル (SiteWise Tutorial Device Model) について、ステップ 4～6 を繰り返します。

のルールを無効化または削除するには AWS IoT Core

1. AWS IoT コンソールに移動します。
2. 左側のナビゲーションペインで [メッセージルーティング] を選択し、[ルール] を選択します。
3. ルールを選択し、[削除] を選択します。
4. 確認ダイアログボックスでルール名を入力し、[削除] を選択します。

Monitor SiteWise での風力発電所データの視覚化と共有

このチュートリアルでは、ポータルと呼ばれるマネージド Web アプリケーションを通じて産業データを視覚化して共有する方法について説明します。 AWS IoT SiteWise Monitor 各ポータルには複数のプロジェクトが用意されているため、各プロジェクト内でどのデータにアクセスできるかを柔軟に選択できます。次に、各ポータルにアクセスできる組織内のユーザーを指定します。 AWS IAM Identity Center ユーザーはアカウントを使用してポータルにサインインするので、 AWS既存のアイデンティティストアまたはが管理するストアを使用できます。

ユーザーおよび十分なアクセス許可を持つユーザーは、各プロジェクトでダッシュボードを作成し、産業データを有意義な方法で視覚化することができます。その後、ユーザーはこれらのダッシュボードを表示して、データに関するインサイトをすばやく取得し、オペレーションを監視できます。各プロジェクトに対する管理権限または読み取り専用のアクセス許可は、社内の各ユーザーに対して設定できます。詳細については、「によるデータの監視 AWS IoT SiteWise Monitor」を参照してください。

チュートリアル全体を通して、 AWS IoT SiteWise 風力発電所のサンプルデータセットを提供してデモを強化します。 SiteWise Monitor でポータルを設定し、プロジェクトを作成し、風力発電所のデータを視覚化するダッシュボードを作成します。このチュートリアルでは、プロジェクトとそれに関連するダッシュボードを所有または表示する権限の割り当てに加えて、追加ユーザーの作成についても説明します。

> **Note**
>
> SiteWise Monitor を使用すると、ポータルにサインインするユーザーごとに (1 か月あたり) 課金されます。このチュートリアルでは、3 人のユーザーを作成しますが、サインインする

必要があるのは 1 人のユーザーだけです。このチュートリアルを完了すると、1 人のユーザーに対して料金が発生します。詳細については、「AWS IoT SiteWise の料金」を参照してください。

トピック

前提条件

このチュートリアルを完了するには、以下が必要です。

- と AWS アカウント. アカウントをお持ちでない場合は、「のセットアップ AWS アカウント」を参照してください。
- Windows、macOSLinux、Unixまたはを実行してにアクセスする開発用コンピュータ AWS Management Console。詳細については、「AWS Management Consoleの開始方法」を参照してください。
- 管理者権限を持つ AWS Identity and Access Management (IAM) ユーザー。
- AWS IoT SiteWise 実行中の風力発電所のデモ。デモをセットアップすると、 AWS IoT SiteWise 風力発電所を表すモデルとアセットを定義し、それらにデータをストリーミングします。詳細については、「AWS IoT SiteWise デモを使う」を参照してください。
- アカウントで IAM Identity Center を有効にしている場合は、 AWS Organizations 管理アカウントにサインインします。詳しくは、[AWS Organizations terminology and concepts] (用語と概念) をご覧ください。IAM Identity Center を有効にしなかった場合は、このチュートリアルで有効にして、自分のアカウントを管理アカウントに設定します。

  AWS Organizations 管理アカウントにサインインできない場合は、組織に IAM Identity Center ユーザーがいる限り、チュートリアルを部分的に完了できます。この場合、ポータルやダッシュ

ボードは作成できますが、プロジェクトに割り当てる IAM Identity Center ユーザーを新規作成することはできません。

ステップ 1: Monitor SiteWise でポータルを作成する

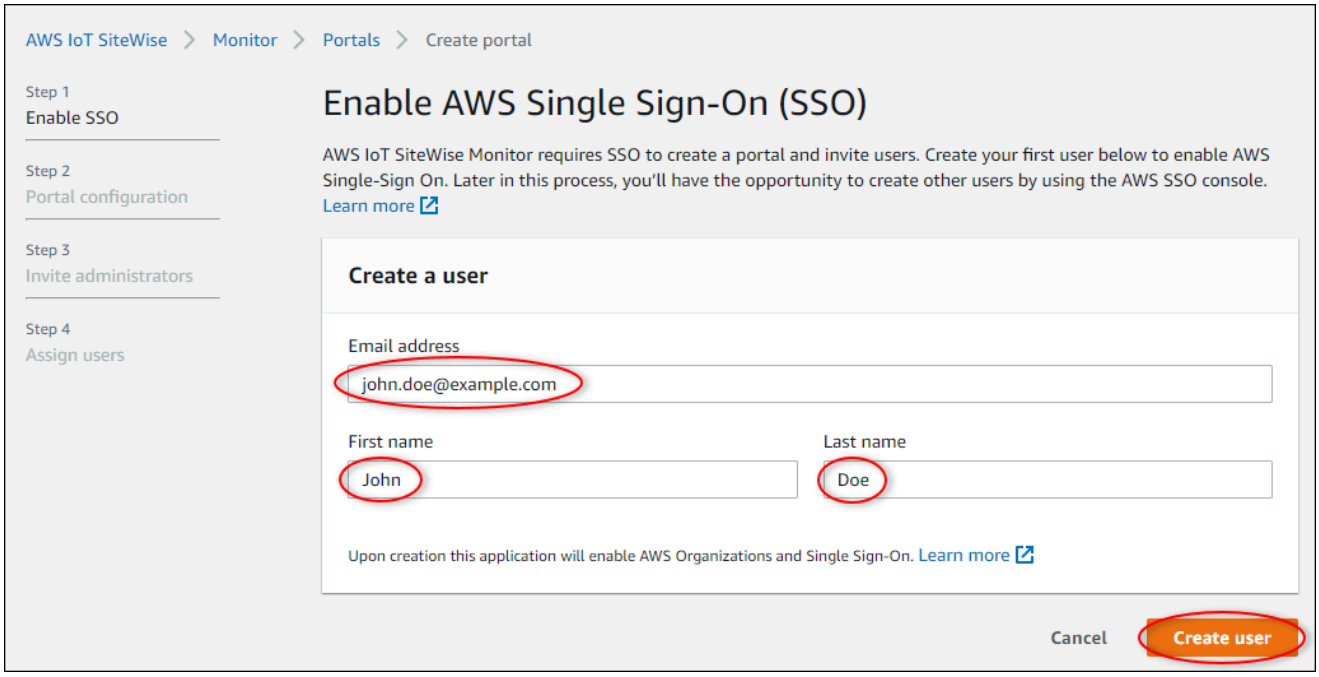
この手順では、 AWS IoT SiteWise Monitorでポータルを作成します。各ポータルは、管理対象の Web アプリケーションであり、 AWS IAM Identity Center 管理者とユーザーはアカウントを使用してサインインできます。IAM Identity Center では、会社の既存のアイデンティティストアを使用することも、 AWS管理するアイデンティティストアを作成することもできます。会社の従業員は、 AWS アカウント別途作成しなくてもサインインできます。

ポータルを作成するには

1. AWS IoT SiteWise コンソールにサインインします。
2. AWS IoT SiteWise サポートされているエンドポイントとクォータを確認し、 AWS IoT SiteWise 必要に応じてリージョンを切り替えてください。 AWS IoT SiteWise デモは同じリージョンで実行する必要があります。
3. 左のナビゲーションペインで、[ポータル] を選択します。
4. [ポータルの作成] を選択します。
5. すでに IAM Identity Center を有効にしている場合は、ステップ 6 に進みます。それ以外の場合は、以下のステップを実行して IAM Identity Center を有効にします。

   a. 「有効化 AWS IAM Identity Center (SSO)」ページで、「メールアドレス」、「名」、「姓」を入力して、自分がポータル管理者になる IAM Identity Center ユーザーを作成します。新しい IAM Identity Center ユーザーのパスワードを設定するための E メールを受信できるように、アクセスできる E メールアドレスを使用します。

      ポータルでは、ポータル管理者がプロジェクトを作成し、ユーザーをプロジェクトに割り当てます。後でさらにユーザーを作成できます。



b. [ユーザーの作成] を選択します。

6. [ポータルの設定] ページで、次のステップを実行します。

   a. **`WindFarmPortal`** などのポータルの名前を入力します。

   b. (オプション) ポータルの説明を入力します。複数のポータルがある場合は、わかりやすい説明を使用すると、各ポータルに含まれる内容を追跡できます。

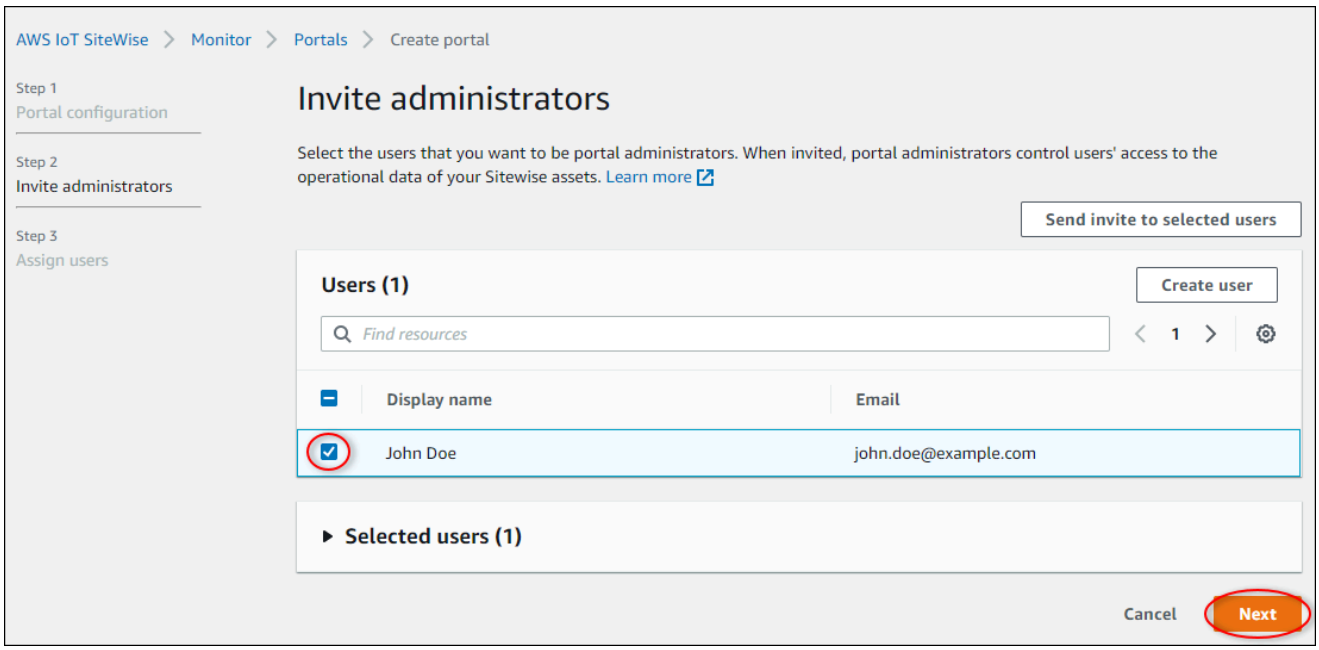
   c. (オプション) ポータルに表示するイメージをアップロードします。

   d. ポータルユーザーがポータルで問題が発生し、 AWS 問題を解決するために会社の管理者の支援が必要な場合に連絡できるメールアドレスを入力します。

   e. [ポータルの作成] を選択します。

7. [管理者を招待する] ページでは、IAM Identity Center ユーザーを管理者としてポータルに割り当てることができます。ポータル管理者は、ポータル内のアクセス許可とプロジェクトを管理します。このページで、次のことを行ってください。

   a. ポータル管理者となるユーザーを選択します。のチュートリアルの前半で IAM Identity Center を有効にした場合、作成したユーザーを選択します。



b. (オプション) [選択したユーザーに招待を送信] を選択します。メールソフトを開くと、メッセージの本文に招待状が表示されます。E メールは、ポータル管理者に送信する前にカスタマイズできます。後でポータル管理者に E メールを送信することもできます。 SiteWise Monitor を初めて利用する場合で、ポータル管理者になる場合は、自分宛にメールを送信する必要はありません。

c. [次へ] をクリックします。

8. [ユーザーの割り当て] ページでは、ポータルに IAM Identity Center ユーザーを割り当てることができます。ポータル管理者は、これらのユーザーを後でプロジェクトの所有者やビューワーとして割り当てることができます。プロジェクト所有者は、プロジェクト内にダッシュボードを作成することができます。プロジェクトのビューワーは、割り当てられたプロジェクトに対して読み取り専用のアクセス権を持ちます。このページでは、ポータルに追加する IAM Identity Center ユーザーを作成することができます。

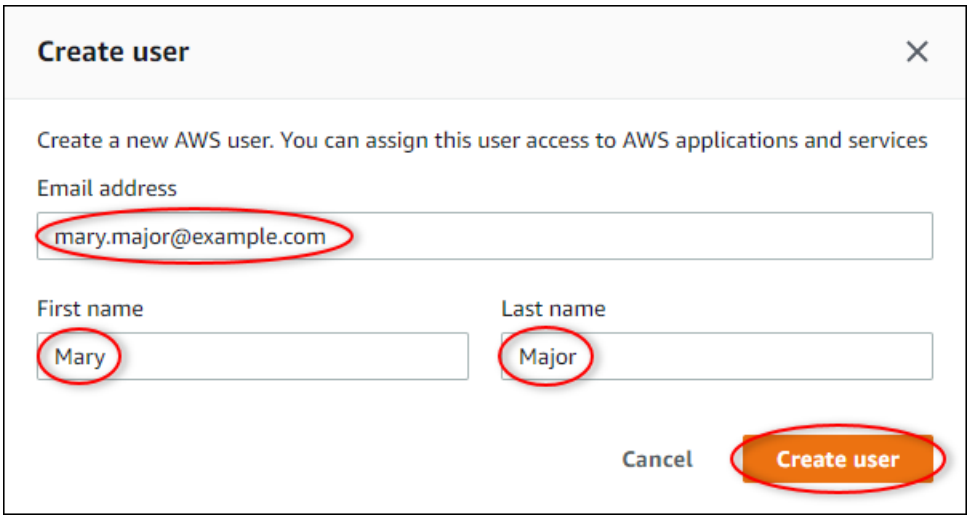
> **Note**
>
> AWS Organizations 管理アカウントにサインインしていない場合、IAM Identity Center ユーザーを作成することはできません。[ユーザーの割り当て] を選択して、ポータルユーザーなしでポータルを作成し、このステップをスキップします。

このページで、次のことを行ってください。

a. 次のステップを 2 回行い、2 人の IAM Identity Center ユーザーを作成します。

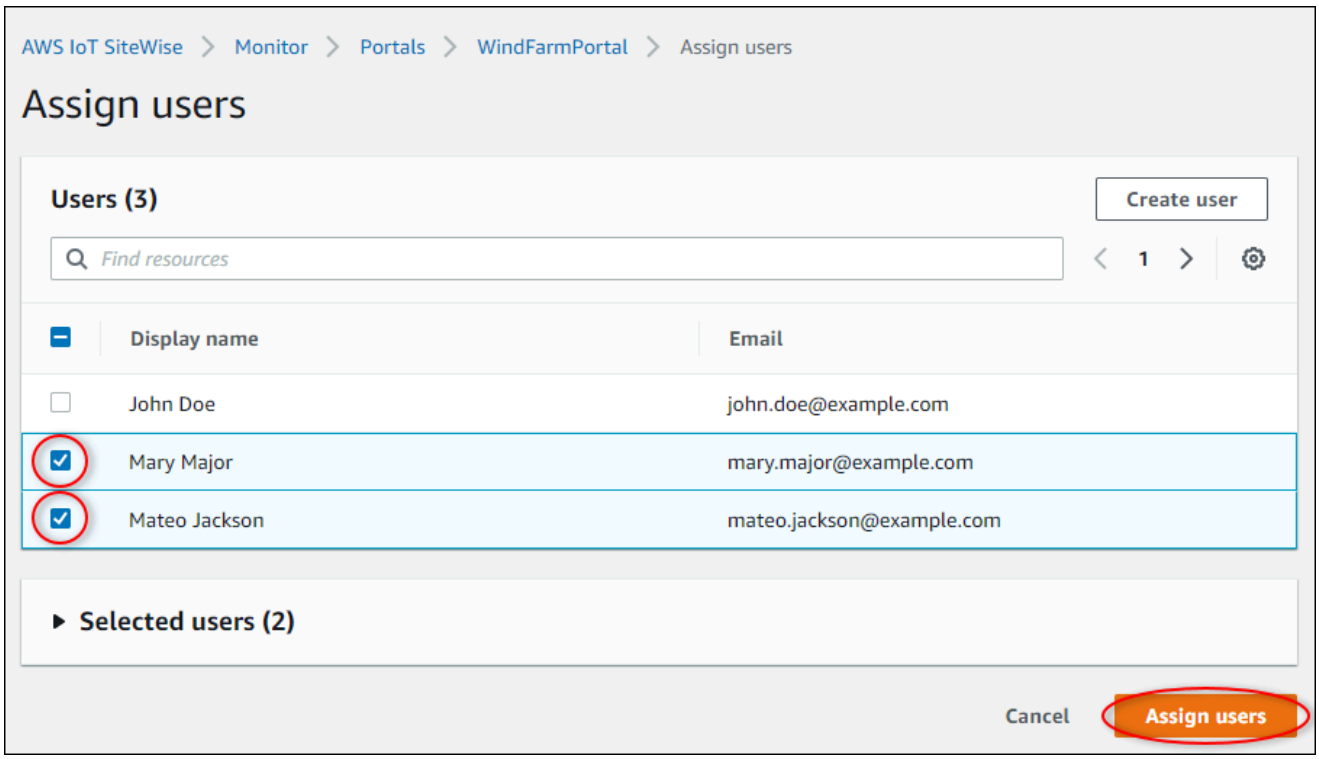
i. [ユーザーの作成] を選択すると、新規ユーザーの詳細を入力するダイアログボックスが表示されます。

ii. 登録するユーザーの E メールアドレスと名、姓を入力します。IAM Identity Center からユーザーに、パスワードを設定するための E メールが送信されます。これらのユーザーとしてポータルにサインインする場合、アクセス可能なメールアドレスを選択します。各メールアドレスは異なるメールアドレスでなければなりません。ユーザーは、メールアドレスをユーザー名としてポータルにサインインします。



iii. [ユーザーの作成] を選択します。

b. 前のステップで作成した 2 人の IAM Identity Center ユーザーを選択します。



c. [ユーザーの割り当て] を選択して、これらのユーザーをポータルに追加します。

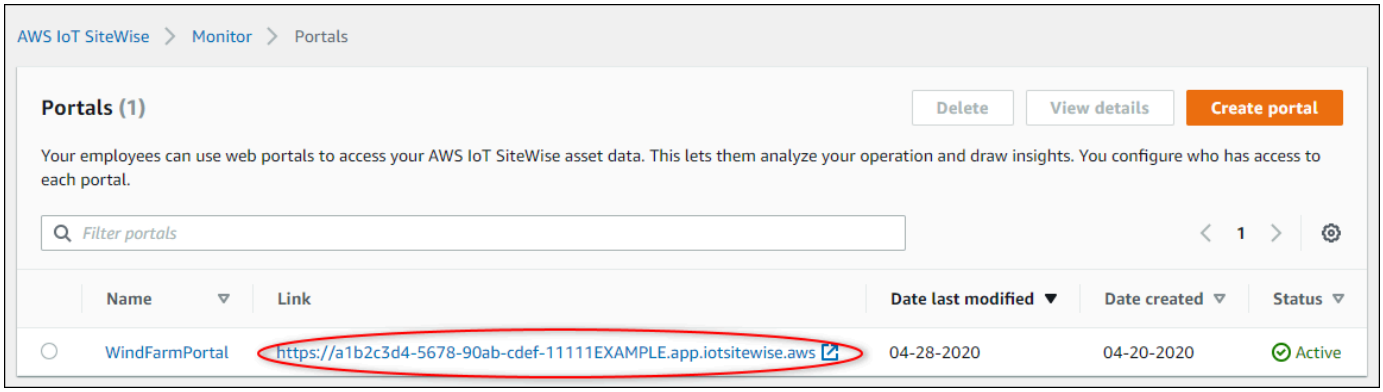
ポータルページが開き、新しいポータルが一覧表示されます。

ステップ 2: ポータルにサインインする

この手順では、 AWS IAM Identity Center ポータルに追加したユーザーを使用して新しいポータルにサインインします。

ポータルにサインインするには

1. [ポータル] ページで、新しいポータルの リンク を選択し、新しいタブでポータルを開きます。



2. チュートリアルの前半で最初の IAM Identity Center ユーザーを作成した場合は、次のステップに従ってユーザーのパスワードを作成します。

   a. メールで件名 (Invitation to join AWS IAM Identity Center) を確認します。

   b. 招待メールを開き、[Accept invitation] を選択します。

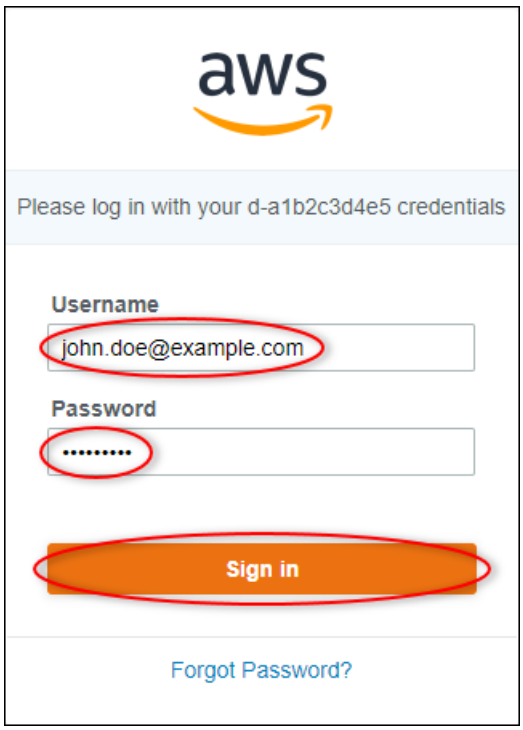
   c. 新しいウィンドウで、IAM Identity Center ユーザーのパスワードを設定します。

   先に作成した 2 人目、3 人目の IAM Identity Center ユーザーとして後でポータルにサインインしたい場合は、次のステップでパスワードを設定することもできます。

   > **Note**
   >
   > メールが届かない場合は、IAM Identity Center コンソールでユーザーのパスワードを生成できます。詳しくは、[AWS IAM Identity Center User Guide] (ユーザーガイド) の[Reset a user password] (ユーザーパスワードをリセットする) をご覧ください。

3. IAM Identity Center の Usernameと Passwordを入力します。このチュートリアルの前半で IAM Identity Center ユーザーを作成した場合、Usernameは作成したポータル管理者ユーザーの E メールアドレスです。

   ポータル管理者を含むすべてのポータルユーザーは、自分の IAM Identity Center ユーザー認証情報を使用してサインインする必要があります。通常、この認証情報は、 AWS Management Consoleへのサインインに使用する認証情報とは異なります。



4. [Sign in] を選択します。

   ポータルが開きます。

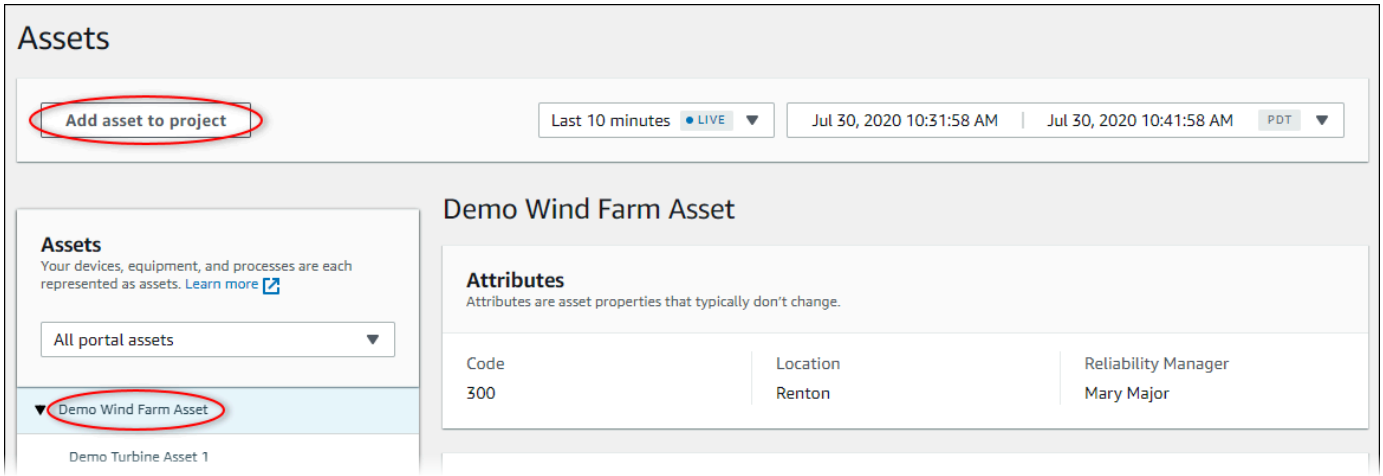
ステップ 3: 風力発電プロジェクトを作成する

この手順では、ポータルでプロジェクトを作成します。プロジェクトは、パーミッション、アセット、ダッシュボードのセットを定義するリソースで、プロジェクト内のアセットデータを視覚化するように設定できます。プロジェクトでは、オペレーションのどのサブセットにアクセスできるか、およびそれらのサブセットのデータを視覚化する方法を定義します。ポータルユーザーは、各プロジェクトの所有者やビューワーとして割り当てることができます。プロジェクト所有者は、データを視覚化するダッシュボードを作成し、他のユーザーとプロジェクトを共有することができます。プロジェクトビューワーはダッシュボードを表示できますが、編集はできません。 SiteWise Monitor のロールについて詳しくは、を参照してください。SiteWise モニターの役割

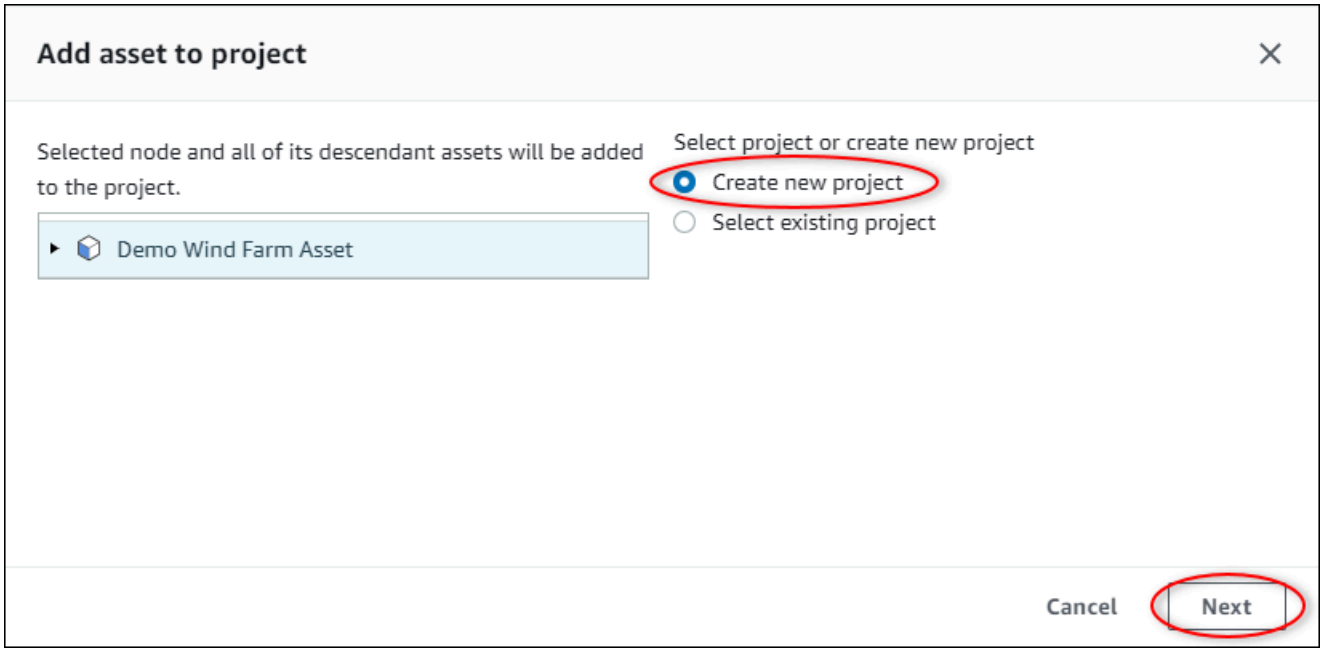
風力発電施設プロジェクトを作成するには

1. ポータルの左側のナビゲーションペインで、[アセット] タブを選択します。[アセット] ページでは、ポータルで利用可能なすべてのアセットを探索し、プロジェクトにアセットを追加することができます。

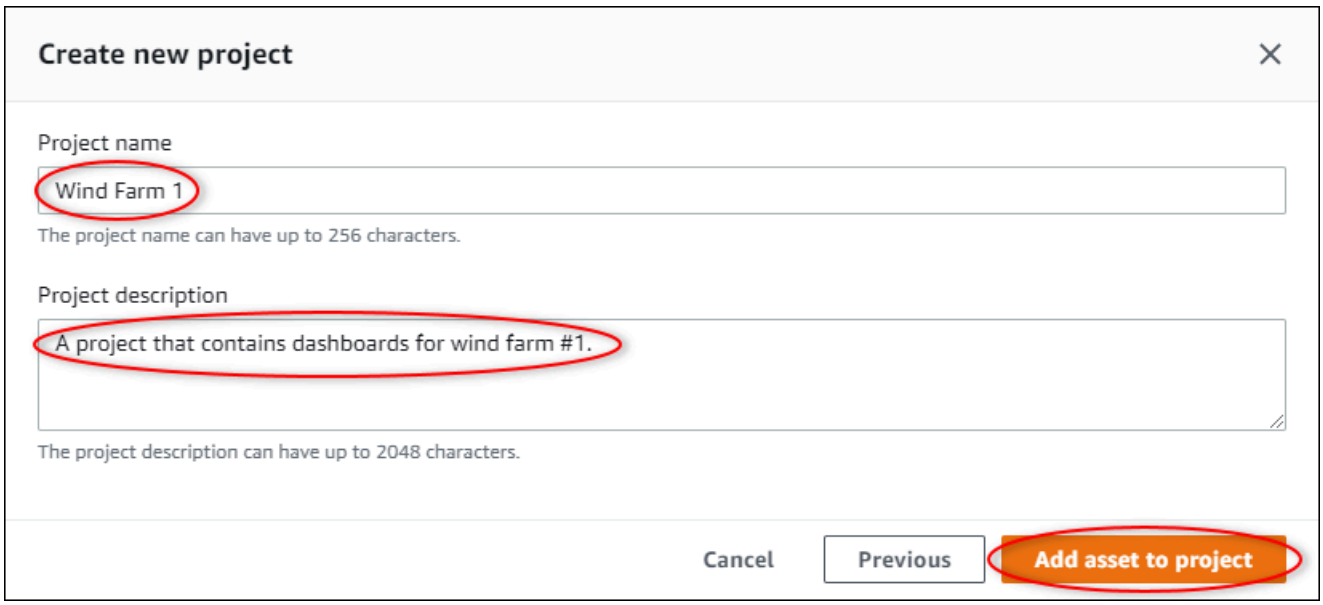
2. アセットブラウザで、[Demo Wind Farm Asset] を選択します。アセットを選択すると、そのアセットのライブデータと履歴データを調べることができます。Shiftを押して複数のアセットを選択し、それらのデータを比較することもできます side-by-side。
3. 左上の [プロジェクトにアセットを追加] を選択します。プロジェクトには、ポータルユーザーがデータを探索するために表示できるダッシュボードが含まれています。各プロジェクトは、AWS IoT SiteWiseにあるアセットのサブセットにアクセスできます。プロジェクトにアセットを追加すると、そのプロジェクトへのアクセス権を持つすべてのユーザーは、そのアセットとその子のデータにもアクセスできます。



4. [プロジェクトにアセットを追加する] ダイアログボックスで、[新しいプロジェクトを作成する] を選択し、[次へ] を選択します。



5. [プロジェクトの新規作成] ダイアログボックスで、[プロジェクト名] と[プロジェクトの説明] を入力し、[アセットをプロジェクトに追加する] を選択します。



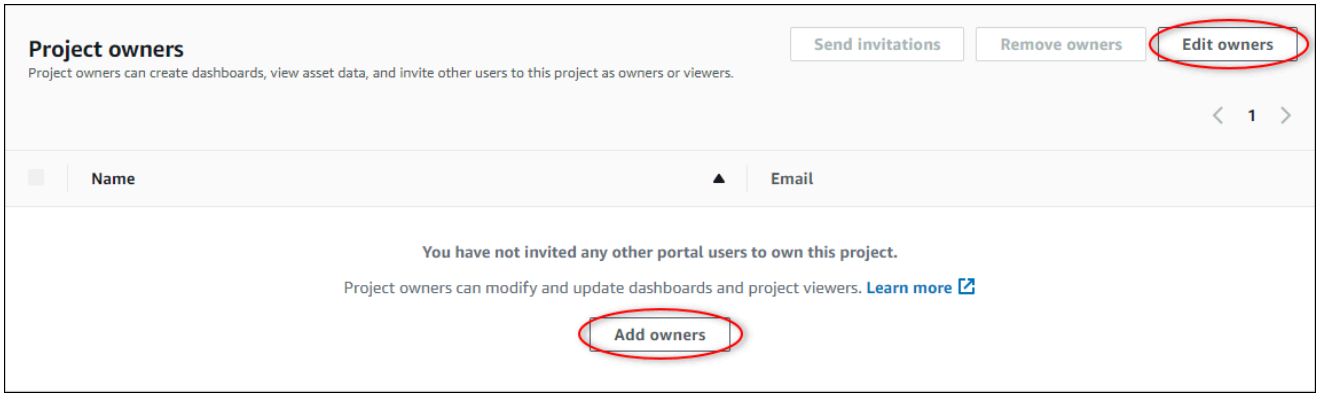
新しいプロジェクトのページが開きます。

6. プロジェクトのページでは、ポータルユーザーをこのプロジェクトの所有者やビューワーとして追加することができます。

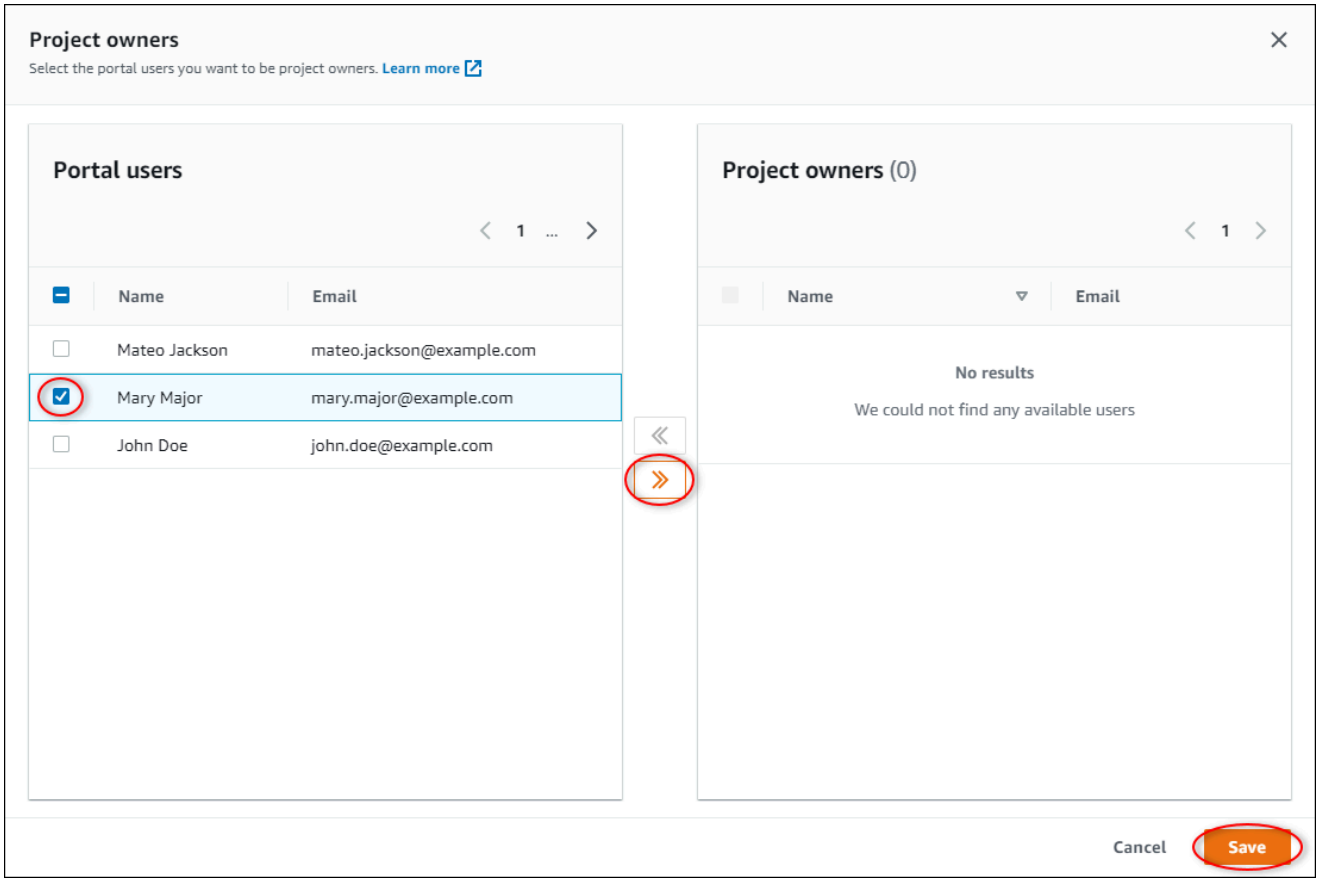
> **Note**
>
> AWS Organizations 管理アカウントにサインインしていない場合は、このプロジェクトに割り当てるポータルユーザーがいない可能性があるため、この手順は省略できます。

このページで、次のことを行ってください。

a. [プロジェクトの所有者] で[所有者の追加] または[ユーザーの編集] を選択します。



b. プロジェクト所有者として追加するユーザー (Mary Major など) を選択し、[>>] アイコンを選択します。



c. [保存] を選択します。

IAM Identity Center ユーザーの Mary Majorは、このポータルにサインインしてこのプロジェクトのダッシュボードを編集したり、このポータルの他のユーザーとこのプロジェクトを共有したりできます。

d. [プロジェクトのビューワー] で[ビューワーの追加] または[ユーザーの編集] を選択します。

e. プロジェクトビューワーとして追加するユーザー (Mateo Jackson など) を選択し、>> アイコンを選択します。

f. [保存] を選択します。

IAM Identity Center ユーザーの Mateo Jacksonは、このポータルにサインインして、風力発電施設のプロジェクトのダッシュボードを表示できますが、編集はできません。

ステップ 4: 風力発電所のデータを視覚化するダッシュボードを作成する

この手順では、ダッシュボードを作成して、デモ風力発電施設のデータを視覚化します。ダッシュボードには、プロジェクトのアセットデータのカスタマイズ可能な視覚化が含まれています。各ビ

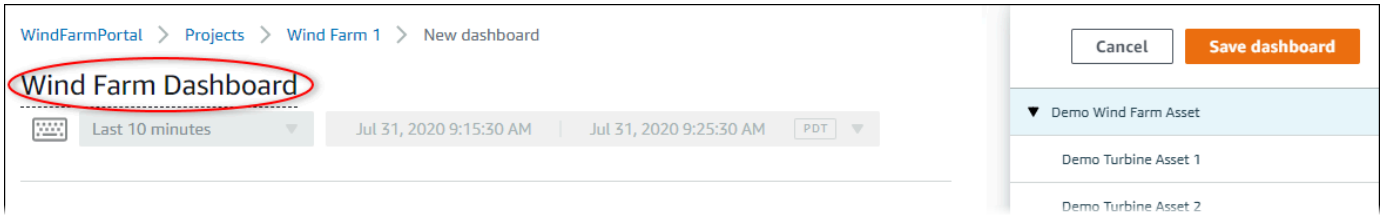
ジュアライゼーションには、折れ線グラフ、棒グラフ、重要業績評価指標 (KPI) 表示など、異なるタイプを使用できます。データに最も適した視覚化のタイプを選択できます。プロジェクトオーナーはダッシュボードを編集できますが、プロジェクト閲覧者はインサイトを得るためにダッシュボードを閲覧することしかできません。

視覚化を含むダッシュボードを作成するには

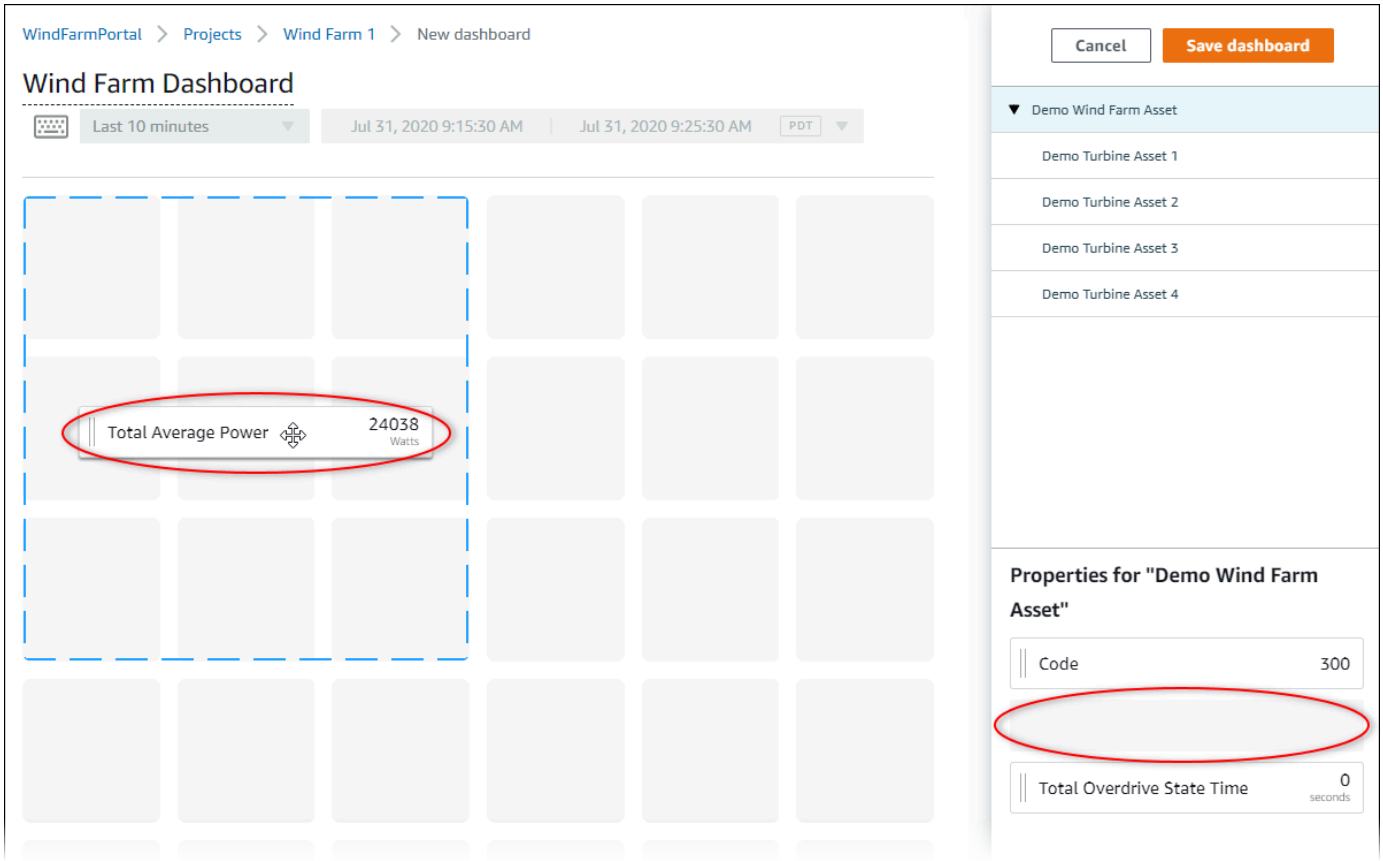
1. 新規プロジェクトのページで、[ダッシュボードの作成] を選択してダッシュボードを作成し、その編集ページを開きます。

   ダッシュボードの編集ページで、アセット階層からダッシュボードにアセットプロパティをドラッグして、視覚化を作成できます。次に、各視覚化のタイトル、凡例タイトル、タイプ、サイズ、ダッシュボード内の場所を編集できます。

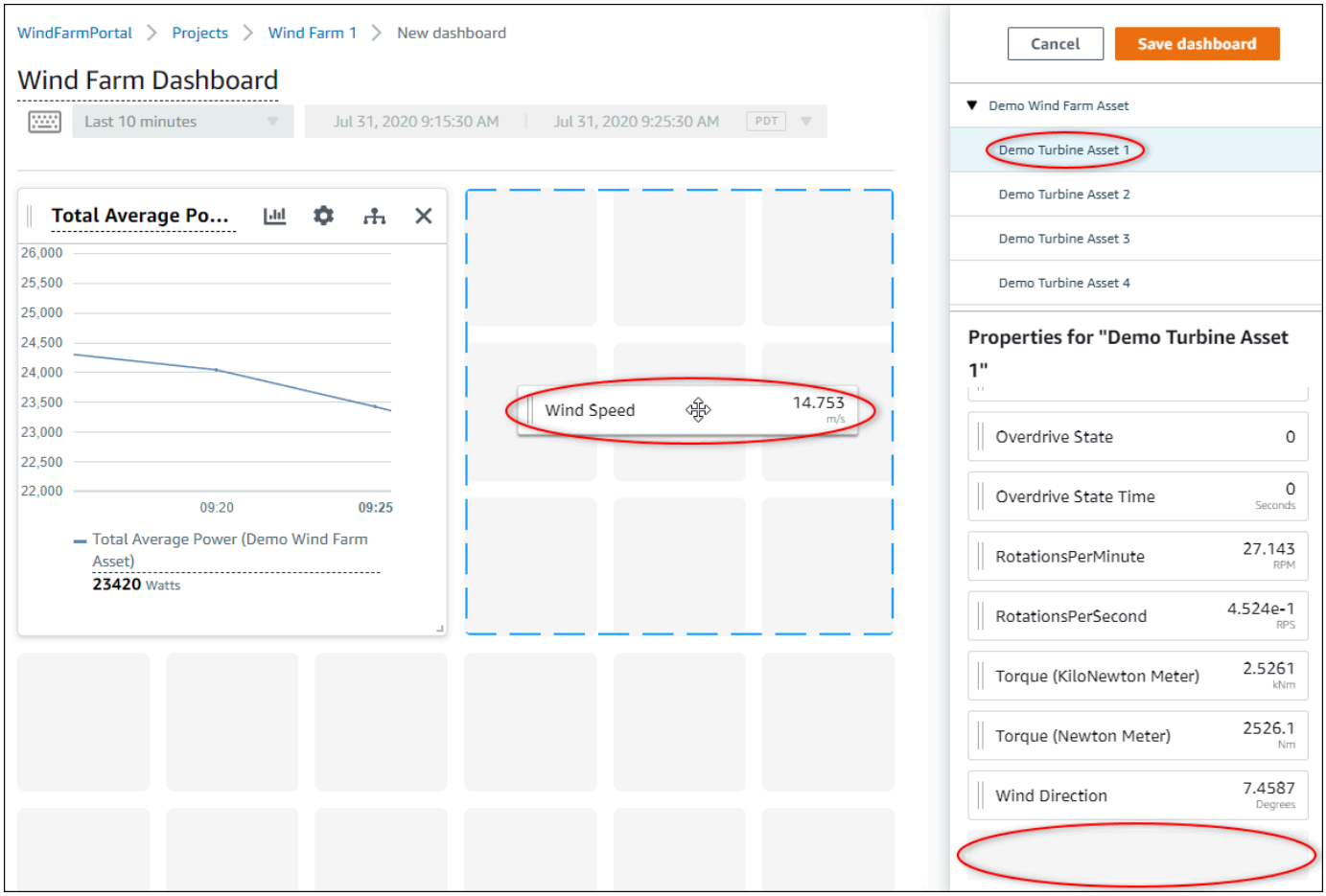
2. ダッシュボードに名前を入力します。



3. Total Average Power から Demo Wind Farm Asset をダッシュボードにドラッグして視覚化する。

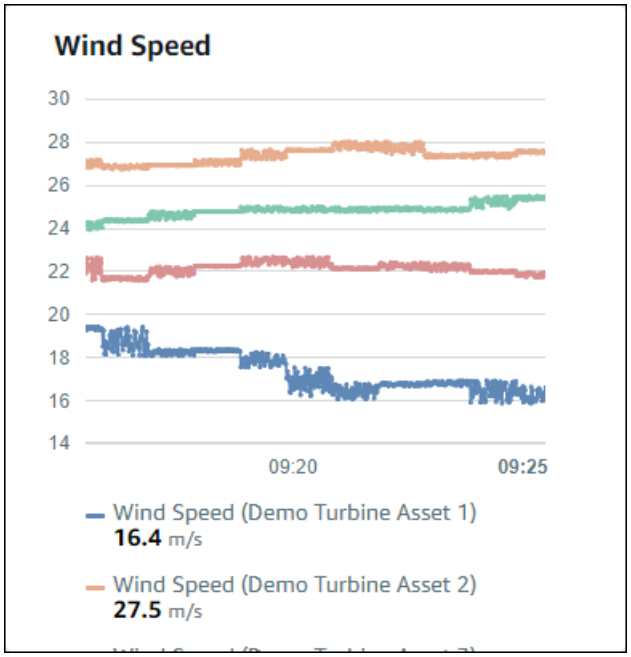


4. Demo Turbine Asset 1を選択してそのアセットのプロパティを表示し、Wind Speedをダッシュボードにドラッグして風速の視覚化を作成します。

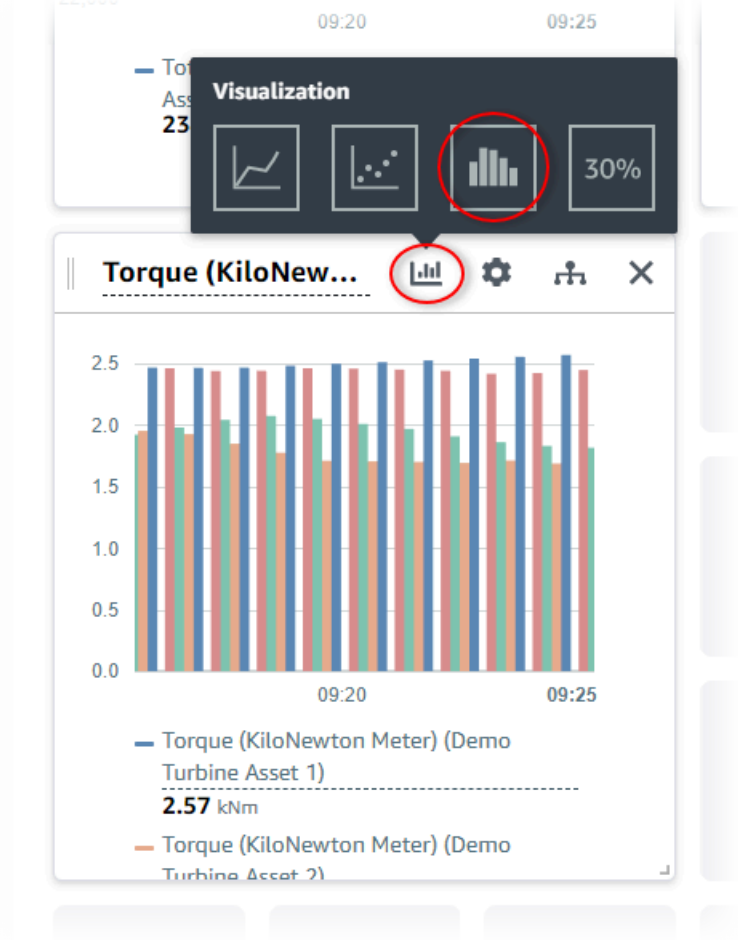


5. Demo Turbine Asset 2、3、4 (この順) の各風速の新しい視覚化にWind Speedを追加します。

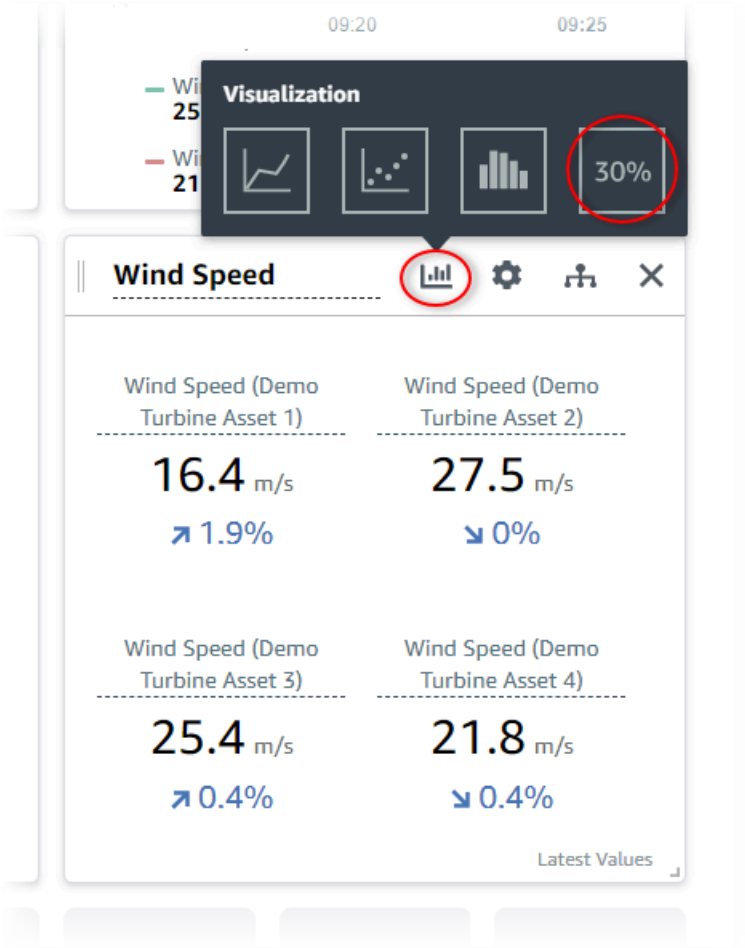
   Wind Speed 視覚化は、次のスクリーンショットのようになります。



6. 風力タービンの Torque (KiloNewton Meter) プロパティに対してステップ 4 と 5 を繰り返して、風力タービントルクの視覚化を作成します。
7. Torque (KiloNewton Meter) の視覚化の視覚化タイプアイコンを選択し、棒グラフのアイコンを選択します。

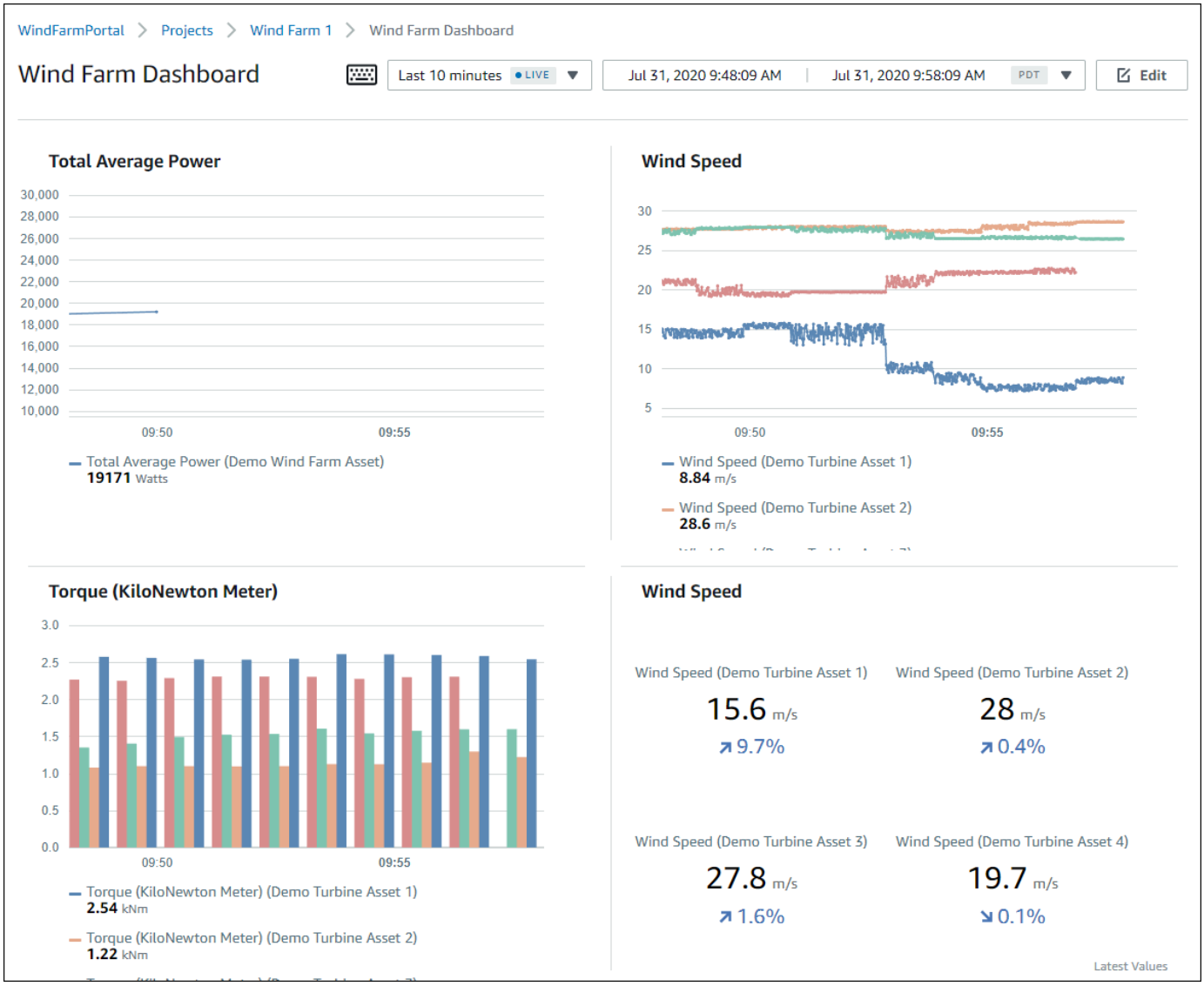


8. 風力タービンの Wind Direction プロパティに対してステップ 4 と 5 を繰り返して、風向の視覚化を作成します。
9. Wind Direction の視覚化の視覚化型アイコンを選択し、次に、KPI グラフアイコン (30%) を選択します。



10. (オプション) 必要に応じて、各視覚化のタイトル、凡例タイトル、型、サイズ、場所を変更します。
11. 右上の [ダッシュボードの保存] を選択してダッシュボードに保存します。

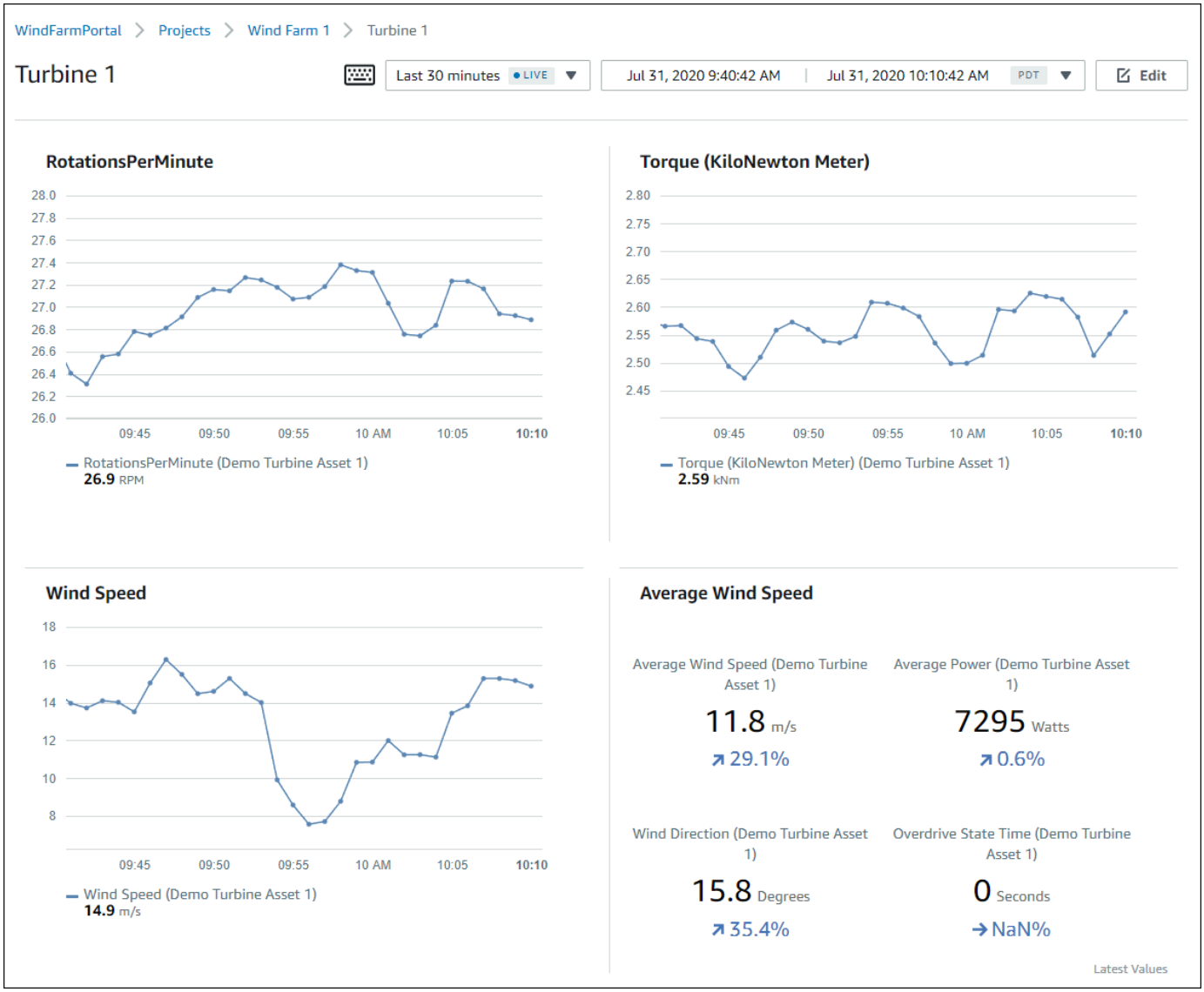
    ダッシュボードは、次のスクリーンショットのようになります。



12. (オプション) 風力タービンアセットごとに追加のダッシュボードを作成します。

    ベストプラクティスとして、プロジェクトビューワーが個々のアセットに関する問題を調査できるように、アセットごとにダッシュボードを作成することをお勧めします。各視覚化には最大 5 つのアセットしか追加できないため、多くのシナリオで階層アセットに対して複数のダッシュボードを作成する必要があります。

    デモ風力タービンのダッシュボードは、次のスクリーンショットのようになります。



13. (オプション) タイムラインを変更するか、視覚化のデータポイントを選択して、ダッシュボードのデータを調べます。詳細は、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[Viewing dashboards] (ダッシュボードを表示する) を参照してください。

ステップ 5: ポータルを調べる

この手順では、 AWS IoT SiteWise ポータル管理者よりも権限が少ないユーザーとしてポータルを探索できます。

ポータルを探索し、チュートリアルを終了するには

- (オプション) プロジェクトに他のユーザーを所有者またはビューワーとして追加した場合、これらのユーザーとしてポータルにサインインすることができます。これにより、ポータル管理者よりも少ないアクセス許可を持つユーザーとして、ポータルを探索することができます。

  > **Important**
  >
  > ポータルにサインインしたユーザーごとに課金されます。詳細については、AWS IoT SiteWise 料金を参照してください。

  他のユーザーとしてポータルを探索するには、次のようにします。

  a. ポータルの左下にある [ログアウト] を選択して、ウェブアプリケーションを終了します。

  b. IAM Identity Center アプリケーションポータルの右上にある [サインアウト] を選択して、IAM Identity Center ユーザーからサインアウトします。

  c. プロジェクト所有者またはプロジェクトビューワーとして割り当てた IAM Identity Center ユーザーとしてポータルにサインインします。詳細については、「ステップ 2: ポータルにサインインする」を参照してください。

これで、このチュートリアルは終了です。 SiteWise Monitor でデモ用風力発電所の探索が終わったら、次の手順に従ってリソースをクリーンアップします。

ステップ 6: チュートリアル終了後にリソースをクリーンアップする

チュートリアルを完了したら、リソースをクリーンアップできます。ユーザーがポータルにサインインしない場合、 AWS IoT SiteWise に対して料金は発生しませんが、ポータルと AWS IAM アイデンティティセンターディレクトリ ユーザーを削除することはできます。デモ風力発電施設アセットは、デモの作成時に選択した期間の終了時に削除されます。デモを手動で削除することもできます。詳細については、「AWS IoT SiteWise デモを削除する。」を参照してください。

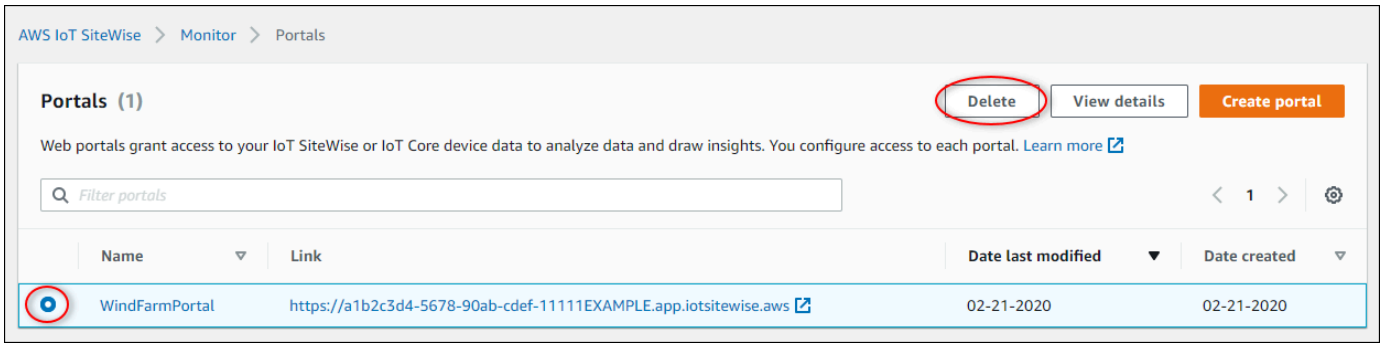
ポータルと IAM Identity Center ユーザーを削除するには、次の手順に従います。

ポータルを削除するには

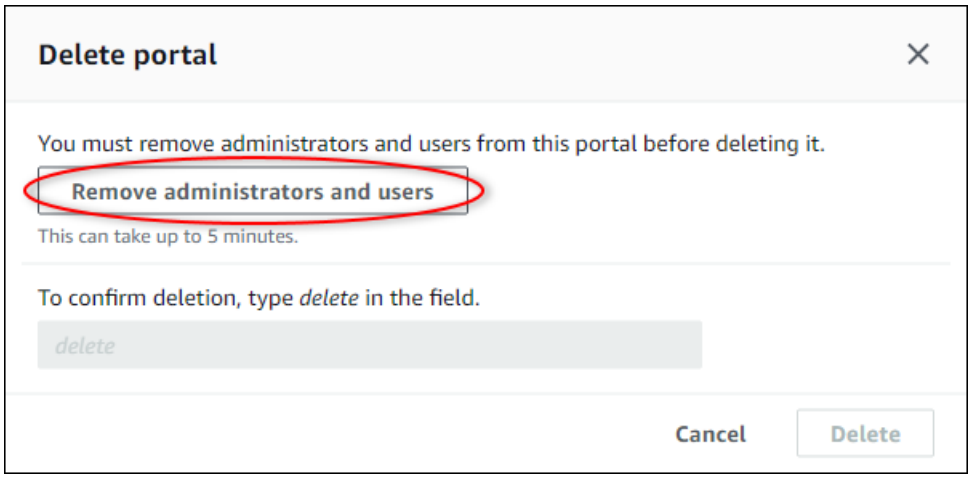
1. AWS IoT SiteWise コンソールに移動します。
2. 左のナビゲーションペインで、[ポータル] を選択します。

3. ポータルを選択しWindFarmPortal、[削除] を選択します。

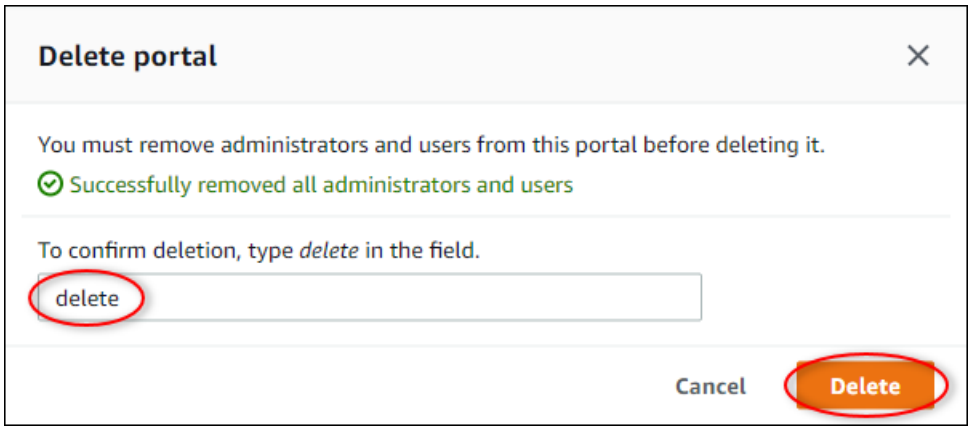
   ポータルまたはプロジェクトを削除しても、削除されたプロジェクトに関連付けられているアセットは影響を受けません。



4. [ポータルの削除] ダイアログボックスで、[管理者とユーザーの削除] を選択します。

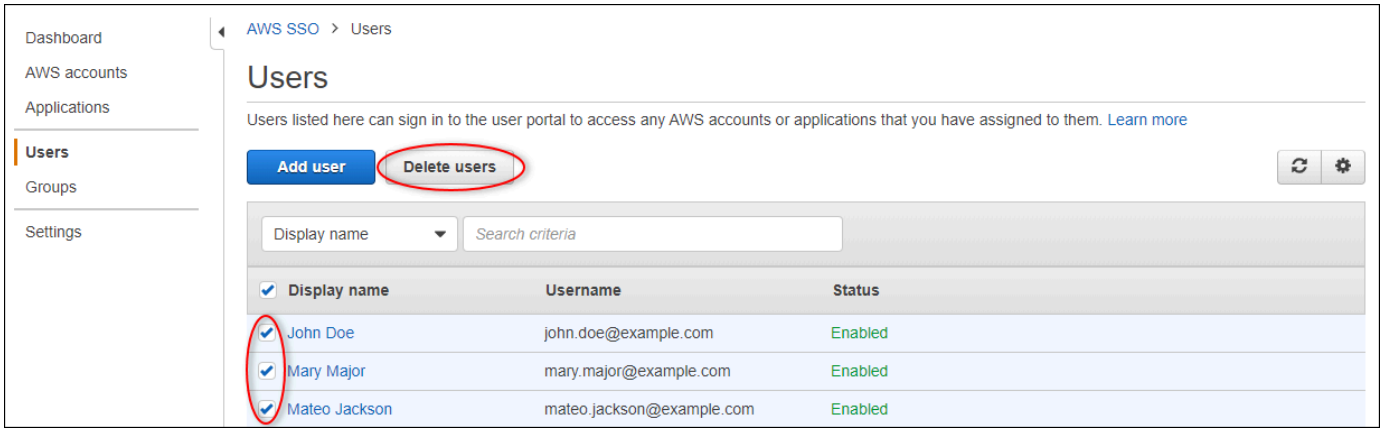


5. **`delete`** と入力して削除を確認し、[削除] を選択します。

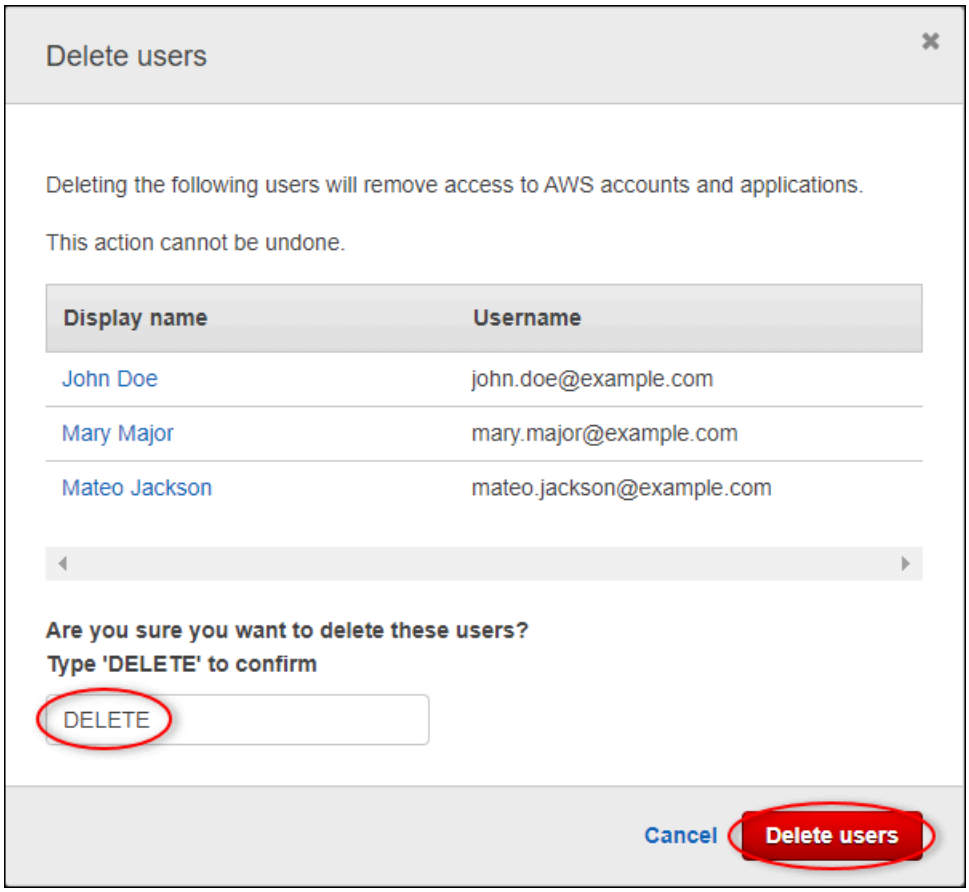


IAM Identity Center ユーザーを削除するには

1. IAM Identity Center コンソールに移動します。
2. 左のナビゲーションペインで、[ユーザー] を選択します。
3. 削除する各ユーザーのチェックボックスをオンにし、[ユーザーの削除] を選択します。



4. [ユーザーの削除] ダイアログボックスで、**DELETE** と入力し、[ユーザーの削除] を選択します。



プロパティ値の更新を Amazon DynamoDB に公開する

このチュートリアルでは、Amazon DynamoDB を使用してデータを保存する便利な方法を紹介します。これにより、API を繰り返しクエリしなくても過去のアセットデータに簡単にアクセスできます。 AWS IoT SiteWise このチュートリアルを完了すると、風力発電所全体の風速と風向のライブマップなど、アセットデータを使用するカスタムソフトウェアを作成できます。カスタムソフトウェアソリューションを実装せずにデータを監視して視覚化したい場合は、を参照してください。によるデータの監視 AWS IoT SiteWise Monitor

このチュートリアルでは、 AWS IoT SiteWise 風力発電所のサンプルデータセットを提供するデモを基にしています。風力発電施設のデモでは、プロパティ値の更新が AWS IoT Core ルールを介して、作成したDynamoDB テーブルにデータを送信するように設定されています。プロパティ値の更新を有効にすると、データを MQTT AWS IoT SiteWise AWS IoT Core メッセージで送信します。次に、メッセージの内容に応じて DynamoDB AWS IoT アクションなどのアクションを実行するコアルールを定義します。詳細については、「AWS 他のサービスとのやり取り」を参照してください。

トピック

前提条件

このチュートリアルを完了するには、以下が必要です。

- アカウント AWS 。アカウントをお持ちでない場合は、「のセットアップ AWS アカウント」を参照してください。
- Windows、macOS、Linux、または Unix を実行してにアクセスする開発用コンピュータ。 AWS Management Console詳細については、「AWS Management Consoleの開始方法」を参照してください。
- 管理者権限を持つ IAM ユーザー。

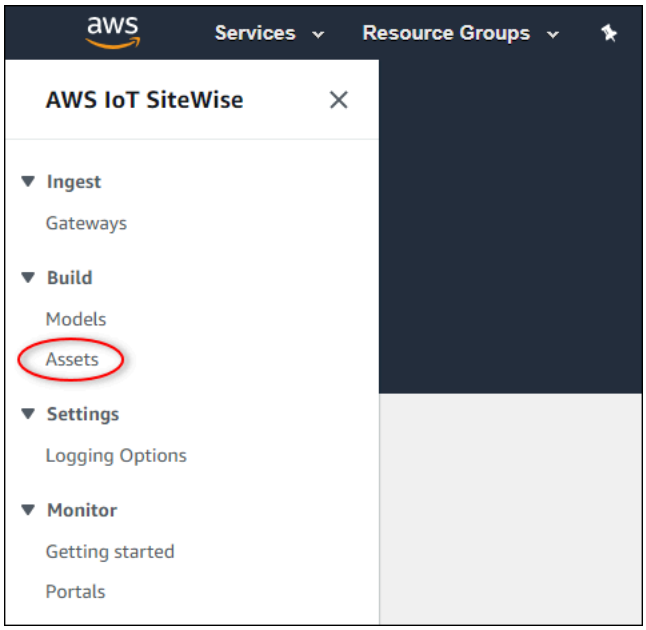
- AWS IoT SiteWise 実行中の風力発電所のデモ。デモをセットアップすると、 AWS IoT SiteWise 風力発電所を表すモデルとアセットを定義し、それらにデータをストリーミングします。詳細については、「AWS IoT SiteWise デモを使う」を参照してください。

ステップ 1: AWS IoT SiteWise プロパティ値の更新を公開するように設定する

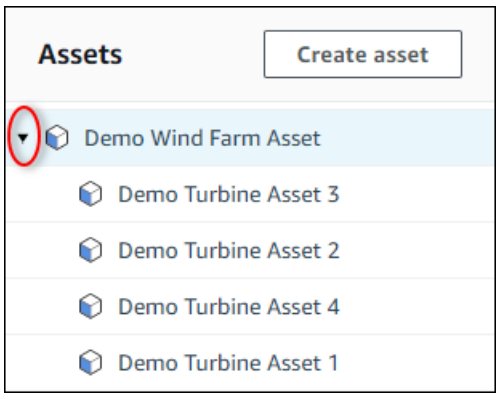
この手順では、デモタービンアセットの [Wind Speed] プロパティでプロパティ値の通知を有効にします。プロパティ値通知を有効にすると、MQTT メッセージ内の各値更新が Core AWS IoT SiteWise AWS IoT に公開されます。

アセットプロパティでプロパティ値の更新に関する通知を有効にするには

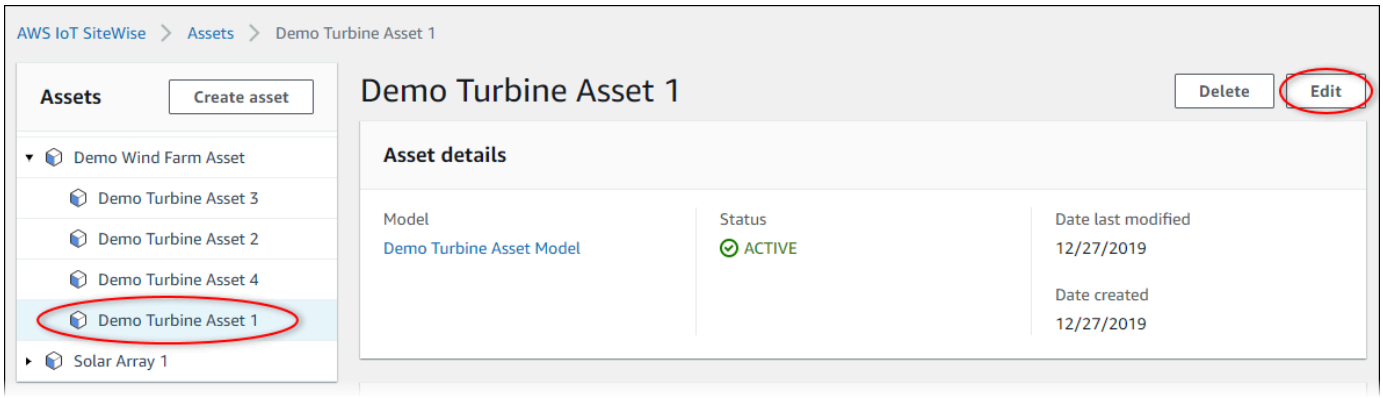
1. AWS IoT SiteWise コンソールにサインインします。
2. AWS IoT SiteWise サポートされているエンドポイントとクォータを確認し、 AWS IoT SiteWise 必要に応じてリージョンを切り替えます AWS 。デモを実施しているリージョンに切り替えます。 AWS IoT SiteWise
3. 左側のナビゲーションペインで [アセット] を選択します。



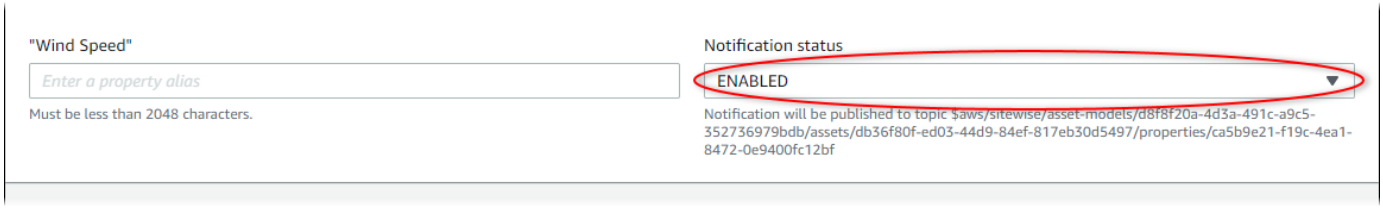
4. Demo Wind Farm Asset の横にある矢印を選択して、風力発電施設のアセットの階層を展開します。



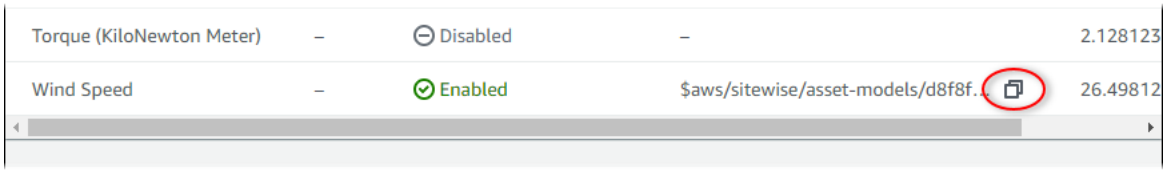
5. Demo Turbine を選択し、[編集] を選択します。



6. Wind Speedプロパティの [通知ステータス] を [已启用] に更新します。



7. ページの下部にある [Save asset (アセットを保存)] を選択します。
8. 各 Demo Turbine Asset に対し、手順 5 ~ 7 を繰り返します。
9. デモタービン (Demo Turbine Asset 1など) を選択します。
10. [測定] を選択します。
11. [Wind Speed] プロパティの横にあるコピーアイコンを選択して、通知トピックをクリップボードにコピーします。保存した通知トピックは、このチュートリアルの後半で使用します。通知トピックは、1 つの Turbine から記録するだけで大丈夫です。



通知トピックは、次の例のようになります。

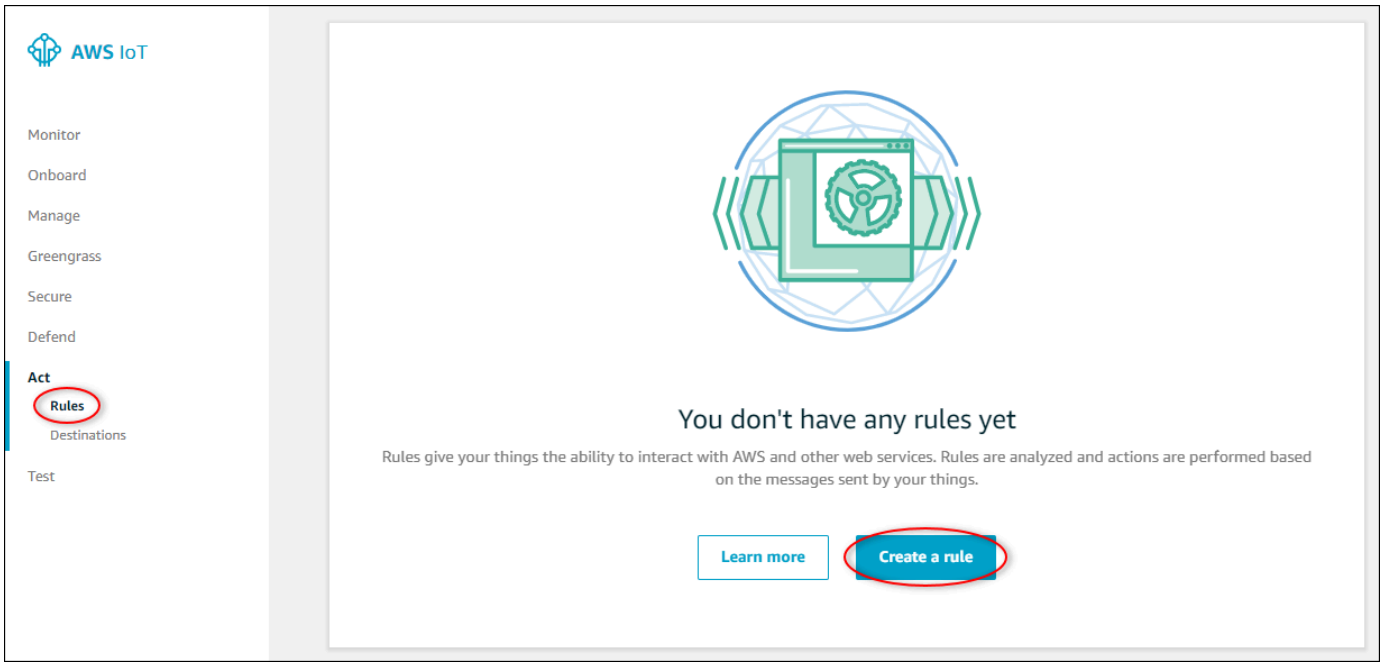
```
$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/
assets/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE/properties/a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE
```

ステップ 2: AWS IoT Core でルールを作成する

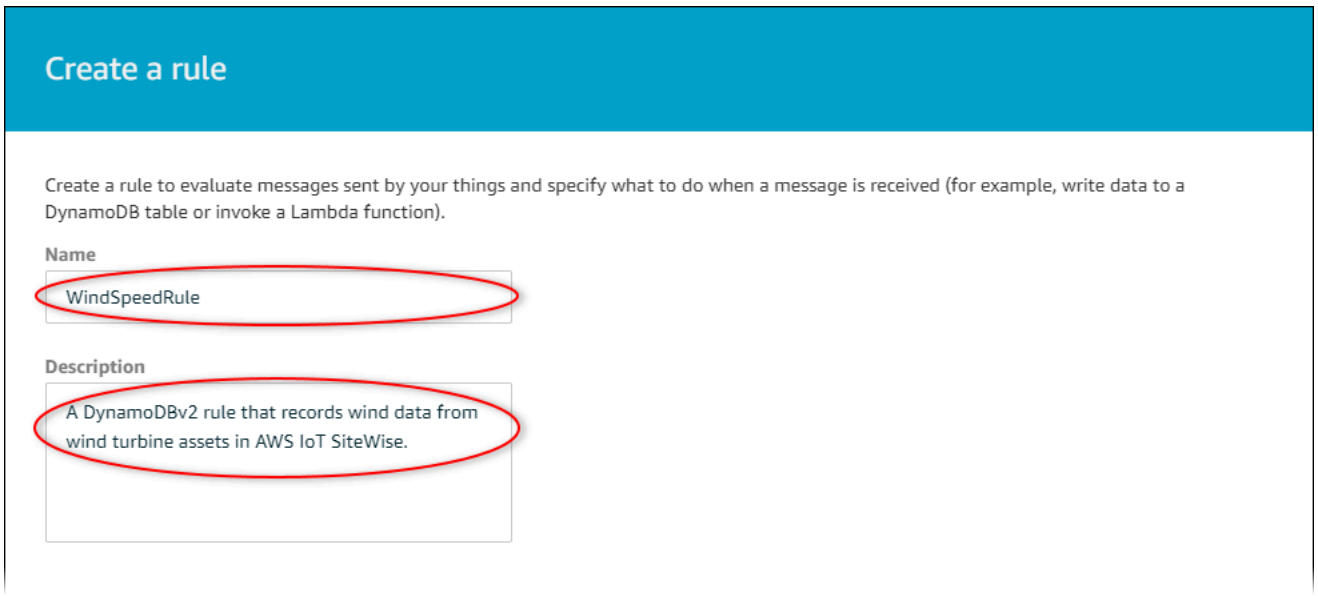
このプロシージャでは、プロパティ値の通知メッセージを解析し、Amazon DynamoDB テーブルにデータを挿入するルールを AWS IoT Core で作成します。 AWS IoT コアルールは MQTT メッセージを解析し、各メッセージの内容とトピックに基づいてアクションを実行します。次に、このチュートリアルの一部として作成する DynamoDB テーブルにデータを挿入する DynamoDB アクションを含むルールを作成します。

DynamoDB アクションを使用したルールの作成

1. AWS IoT コンソールに移動します。[今すぐ始める] ボタンが表示された場合はそれをクリックします。
2. 左側のナビゲーションペインで [Act (アクト)] を選択し、[ルール] を選択します。



3. [ルールはまだ作成されていません] ダイアログボックスが表示された場合は、[ルールの作成] を選択します。それ以外の場合は、[作成] を選択します。
4. ルールの名前と説明を入力します。



5. このチュートリアルの前の手順で保存した通知トピックを見つけます。

```
$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/
assets/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE/properties/a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE
```

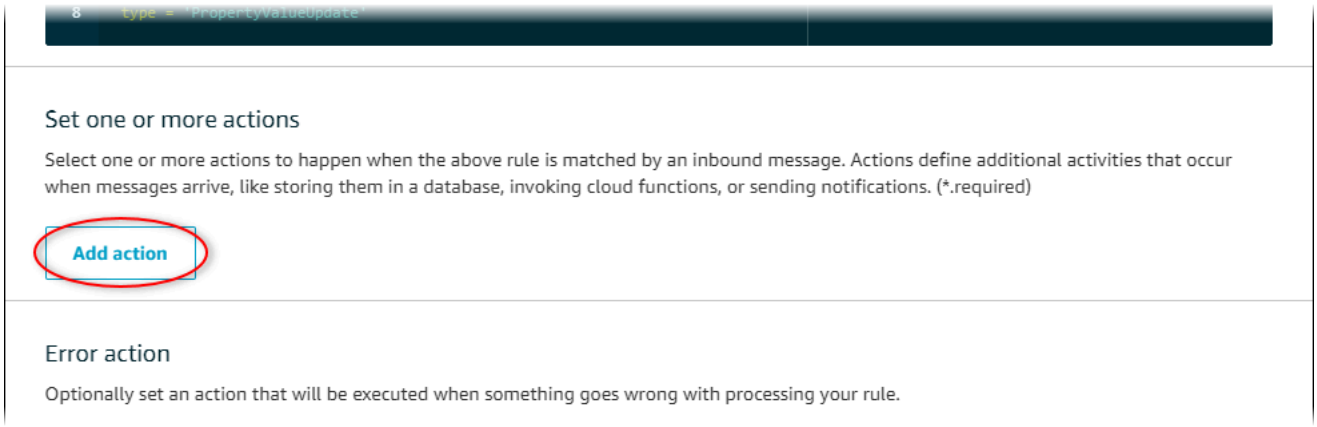
トピック内のアセット ID (後の IDassets/) をに置き換えます+。これにより、すべてのデモ風力タービンアセットの風速プロパティが選択されます。+ トピックフィルターは、トピック内の 1 つのレベルのすべてのノードを受け入れます。トピックは次の例のようになるはずです。

```
$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/assets/+/
properties/a1b2c3d4-5678-90ab-cdef-33333EXAMPLE
```

6. 次のルールクエリステートメントを入力します。FROM セクションのトピックを通知トピックに置き換えます。

```
SELECT
  payload.assetId AS asset,
  (SELECT VALUE (value.doubleValue) FROM payload.values) AS windspeed,
  timestamp() AS timestamp
FROM
  '$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/assets/+/
properties/a1b2c3d4-5678-90ab-cdef-33333EXAMPLE'
WHERE
  type = 'PropertyValueUpdate'
```

7. [1 つ以上のアクションを設定する] で、[アクションの追加] を選択します。



8. [アクションを選択] ページで、[DynamoDB テーブル (DynamoDBv2) の複数列にメッセージを分割する] を選択します。

9. ページの下部にある [アクションの設定] を選択します。
10. [配置操作] ページで [創建新資源] を選択します。

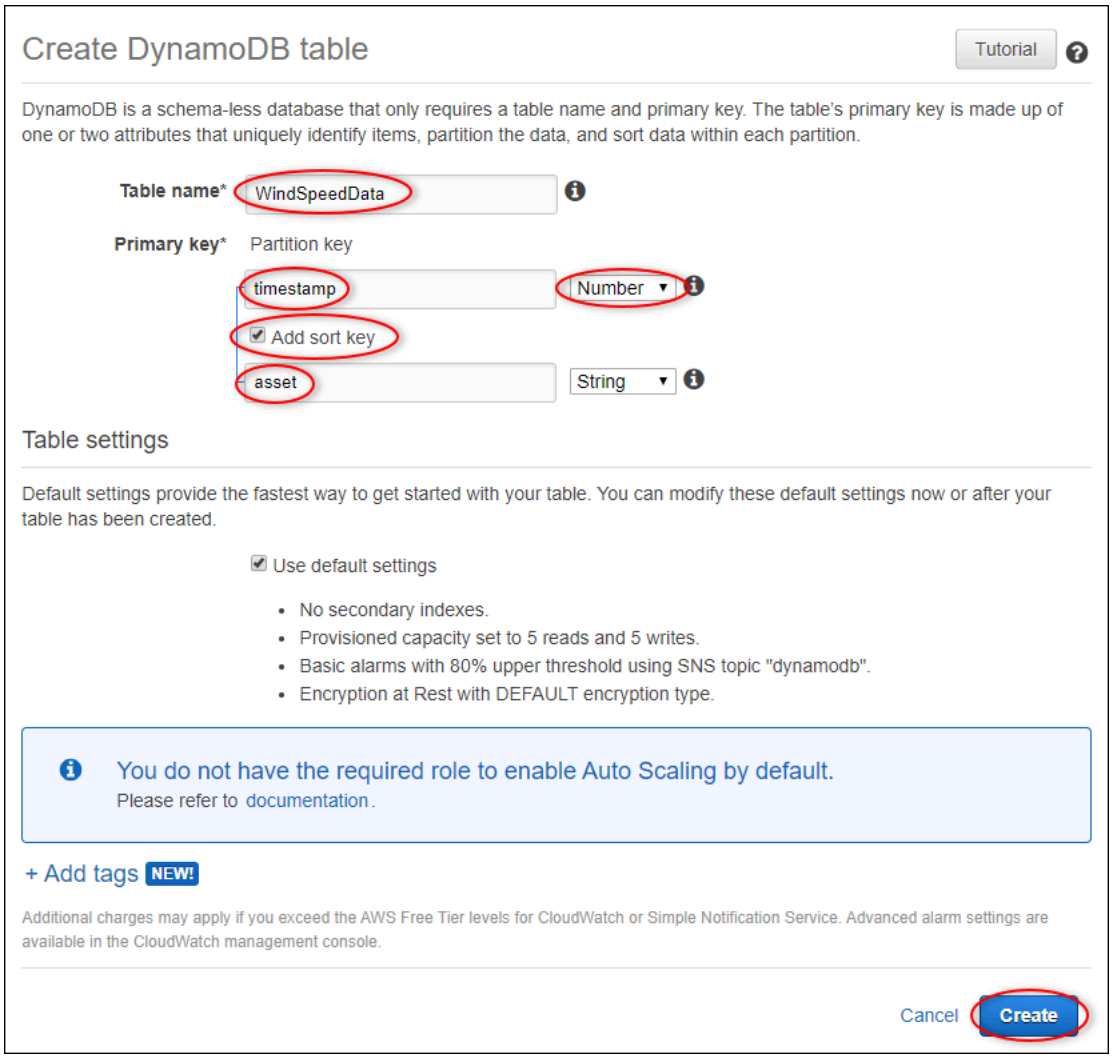
    新しいタブで DynamoDB コンソールが開きます。次の手順を実行している間は、[ルールアクション] タブを開いたままにしておきます。

ステップ 3: DynamoDB テーブルを作成する

このプロシージャでは、ルールアクションから風速データを受け取る Amazon DynamoDB テーブルを作成します。

DynamoDB テーブルを作成するには

1. DynamoDB コンソールのダッシュボードで、[テーブルの作成] を選択します。
2. テーブル名を入力します。



3. プライマリキーのページで、以下の操作を行います。

   a. パーティションキーとして **timestamp** を入力します。

   b. [数値] の種類を選択します。

   c. [ソートキーの追加] のチェックボックスをオンにします。

   d. ソートキーとして **asset** と入力し、デフォルトのソートキーの種類は文字列のままにします。

4. [作成] を選択します。

   テーブルが作成されているという通知が消えたら、テーブルの準備ができたことを示します。

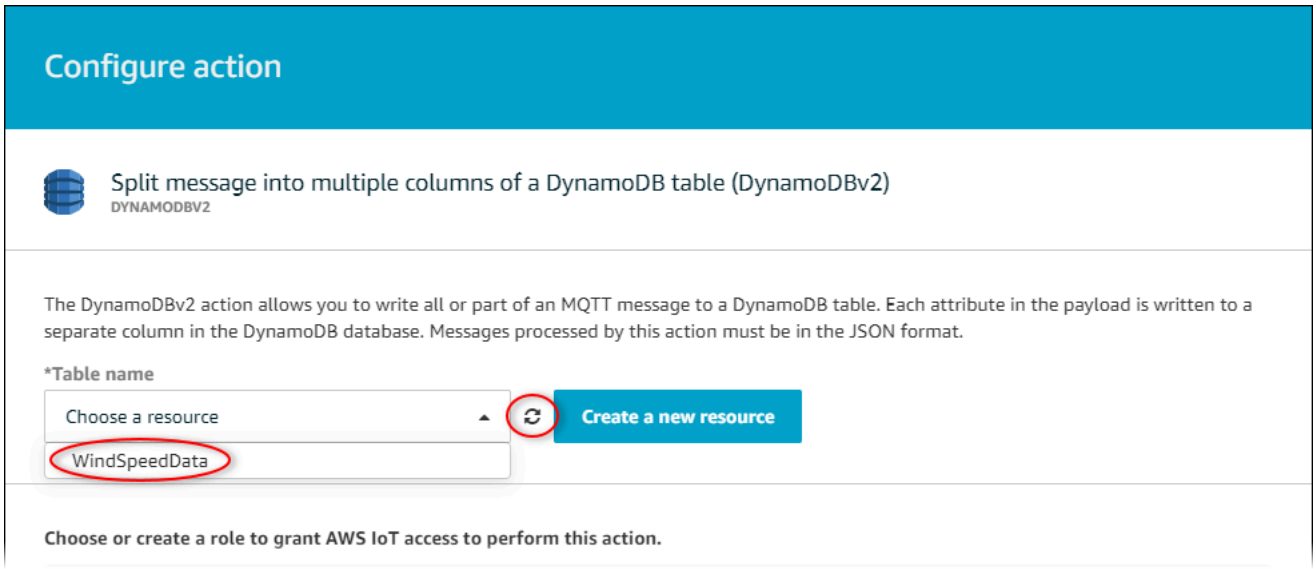
5. アクションの設定のページのタブに戻ります。次のステップを実行している間は、 DynamoDB のタブを開いたままにしておきます。

ステップ 4: DynamoDB ルールアクションを設定する

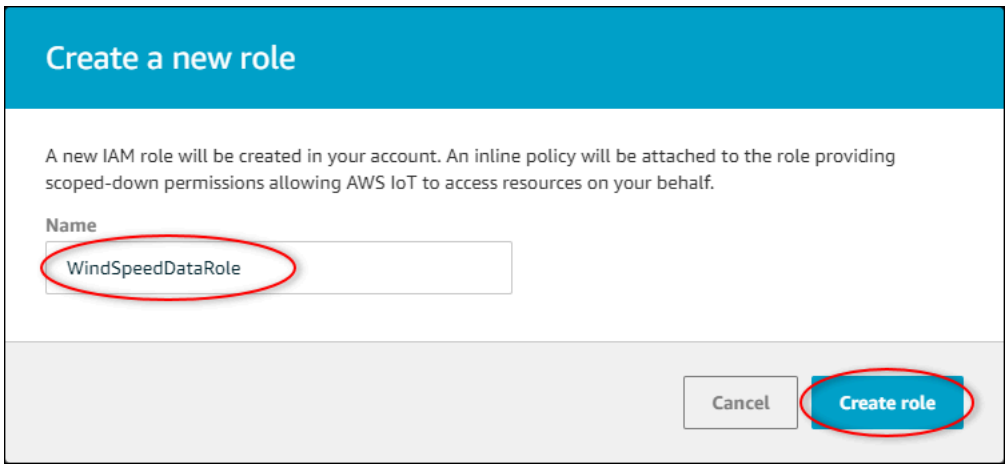
このプロシージャでは、プロパティ値の更新から新しい DynamoDB テーブルにデータを挿入するように Amazon DynamoDB ルールアクションを設定します。

DynamoDB ルールアクションを構成するには

1. [アクションの設定] のページで、[テーブル名] リストを更新し、新しい DynamoDB テーブルを選択します。



2. [ロールの作成] を選択して、 AWS IoT Core にルールアクションを実行するためのアクセス権を付与する IAM ロールを作成します。
3. ロール名を入力し、[ロールの作成] を選択します。



4. [添加操作] を選択します。
5. ページの下部にある [ルールの作成] を選択して、ルールの作成を終了します。

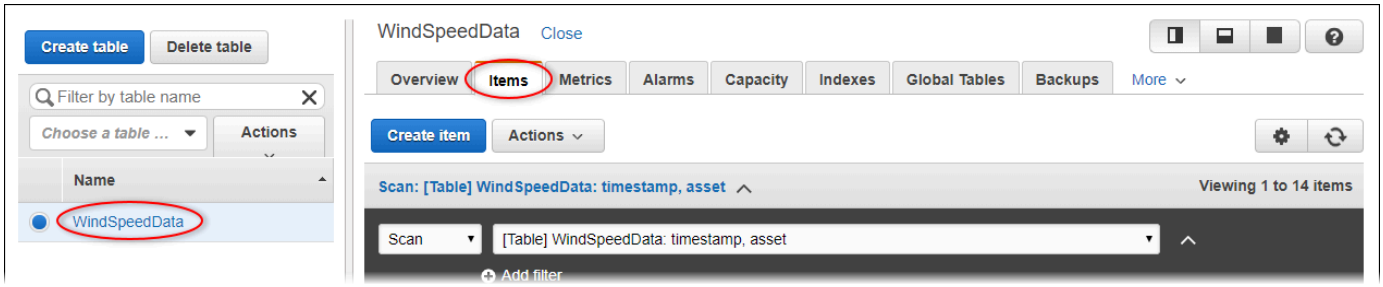
   デモのアセットデータが DynamoDB テーブルに表示され始めます。

ステップ 5: DynamoDB 内のデータを探索する

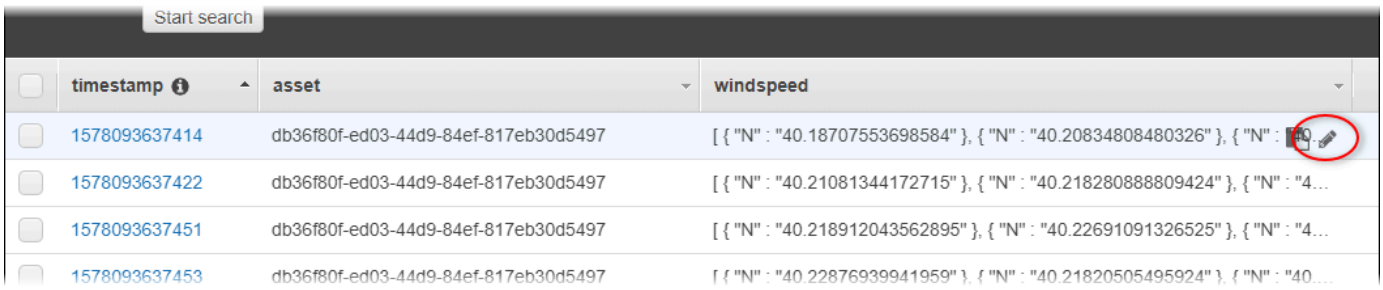
このプロシージャでは、新しい Amazon DynamoDB テーブルにあるデモアセットの風速データを調べます。

DynamoDB でアセットデータを検索する

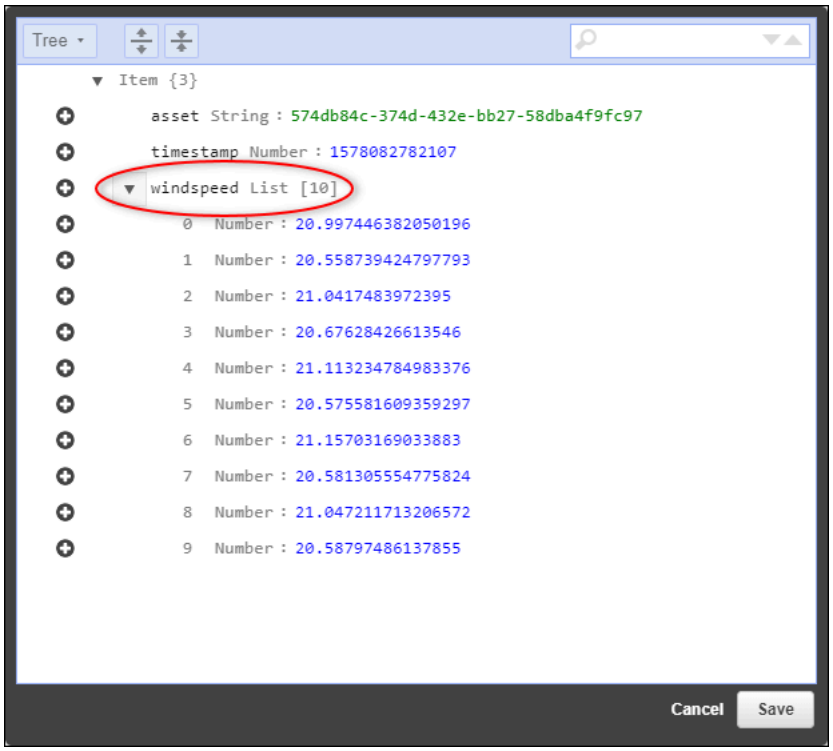
1. DynamoDB テーブルを開いた状態でタブに戻ります。
2. 前に作成したテーブルで、[項目] のタブを選択して、テーブル内のデータを表示します。テーブルに行が表示されない場合は、ページを更新します。数分経過しても行が表示されない場合は、ルールのトラブルシューティング を参照してください。



3. テーブルの行で、編集アイコンを選択してデータを展開します。



4. windspeed 構造の横にある矢印を選択して、風速データポイントのリストを展開します。各リストには、 AWS IoT SiteWise 風力発電所のデモから送信された風速データポイントのバッチが反映されています。独自のルールアクションを設定する場合は、別のデータ形式を使用することもできます。詳細については、「アセットプロパティ通知メッセージのクエリ」を参照してください。



チュートリアルを完了したら、追加料金が発生しないように、ルールを無効化または削除し、DynamoDB テーブルを削除します。リソースをクリーンアップする方法については、を参照してください。ステップ 6: チュートリアル終了後にリソースをクリーンアップする

ステップ 6: チュートリアル終了後にリソースをクリーンアップする

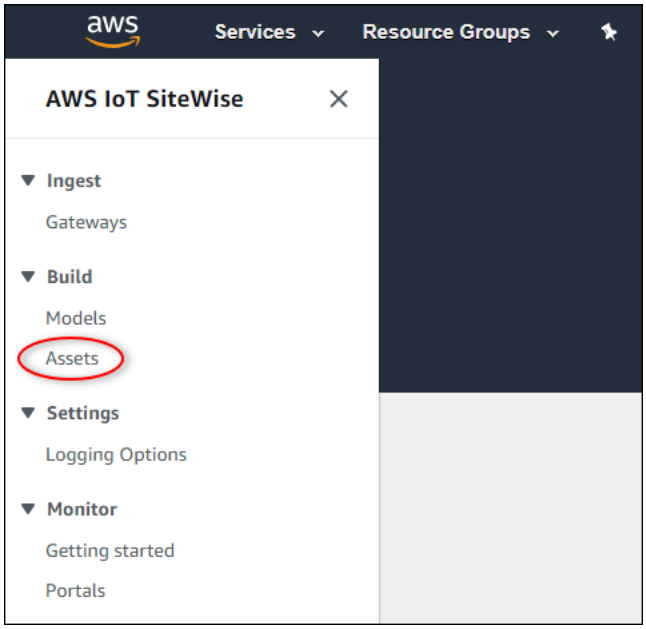
チュートリアルを完了したら、追加料金が発生しないようにリソースをクリーンアップします。デモ風力発電所のアセットは、デモを作成したときに選択した期間の終了時に削除されます。デモは手動

で削除することもできます。詳細については、「AWS IoT SiteWise デモを削除する。」を参照してください。

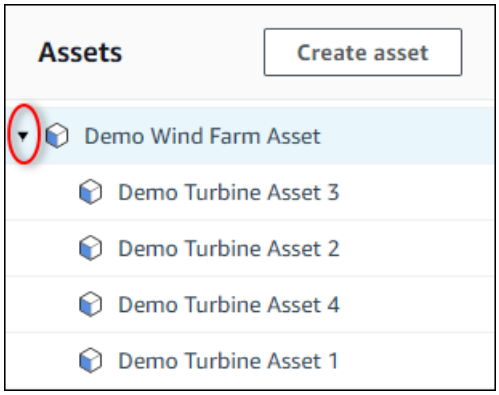
次の手順を使用して、プロパティ値の更新通知を無効にし (デモを削除していない場合)、 AWS IoT ルールを無効または削除し、DynamoDB テーブルを削除します。

アセットプロパティでプロパティ値の更新に関する通知を無効にするには

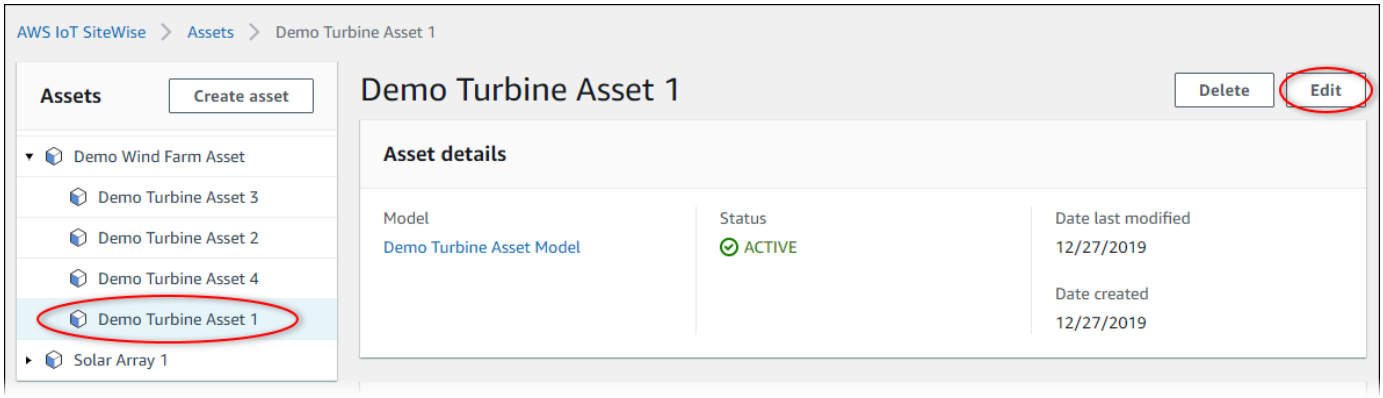
1. AWS IoT SiteWise コンソールに移動します。
2. 左側のナビゲーションペインで [アセット] を選択します。



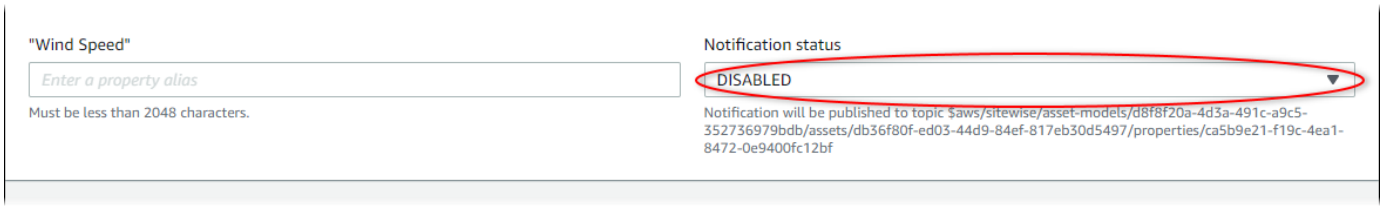
3. Demo Wind Farm Asset の横にある矢印を選択して、風力発電施設のアセットの階層を展開します。



4. Demo Turbine を選択し、[編集] を選択します。



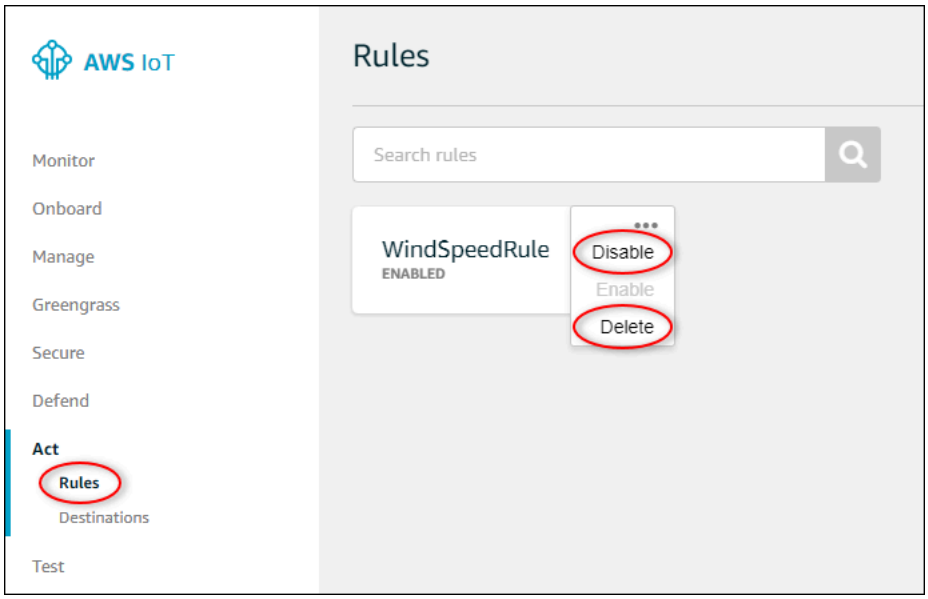
5. [Wind Speed] プロパティの [通知ステータス] を [已禁用] に更新します。



6. ページの下部にある [アセットを保存] を選択します。
7. 各 Demo Turbine Asset に対し、手順 4 ~ 6 を繰り返します。

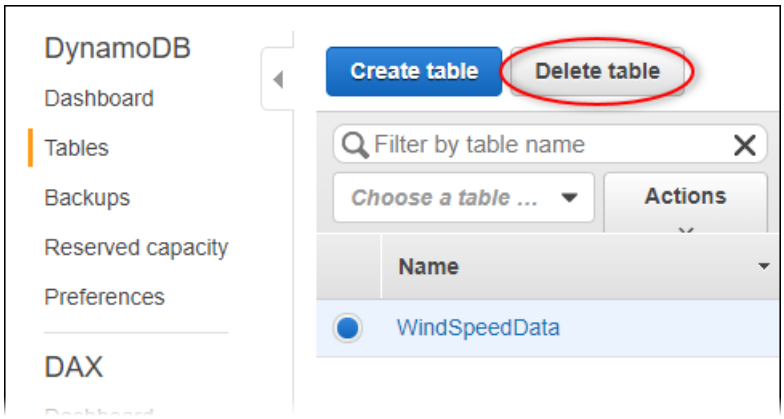
でルールを無効化または削除するには AWS IoT Core

1. AWS IoT コンソールに移動します。
2. 左側のナビゲーションペインで [Act (アクト)] を選択し、[ルール] を選択します。
3. ルールのメニューを選択し、[無効化] または [削除] を選択します。

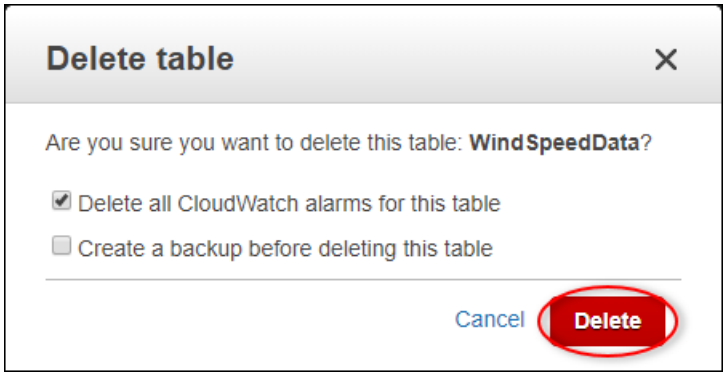


DynamoDB テーブルを削除するには

1. [DynamoDB console] (DynamoDB コンソール) に移動します。
2. 左のナビゲーションペインで、[テーブル] を選択します。
3. 先ほど作成したテーブル、WindSpeedData を選択します。
4. [テーブルの削除] を選択します。



5. [テーブルの削除] ダイアログで、[削除] を選択します。



へのデータの取り込み AWS IoT SiteWise

AWS IoT SiteWise は、産業データを効率的に収集し、対応するアセットと関連付けるように設計されており、産業運用のさまざまな側面を表します。このドキュメントでは、 へのデータの取り込みの実用的な側面に焦点を当て AWS IoT SiteWise、さまざまな産業ユースケースに合わせた複数の方法を提供しています。仮想産業オペレーションを構築する手順については、産業用アセットのモデリング を参照してください。

産業データは、次のいずれかのオプション AWS IoT SiteWise を使用して に送信できます。

- AWS IoT SiteWise Edge – SiteWise Edge ゲートウェイを AWS IoT SiteWise とデータサーバー間の仲介として使用します。 AWS IoT SiteWise は、 SiteWise エッジゲートウェイをセットアップ AWS IoT Greengrass するために実行できる任意のプラットフォームにデプロイできる AWS IoT Greengrass コンポーネントを提供します。このオプションは、OPC-UA サーバープロトコルとのリンクをサポートしています。
- AWS IoT SiteWise API – AWS IoT SiteWise API を使用して、他のソースからデータをアップロードします。ストリーミング BatchPutAssetPropertyValue API を数秒以内の取り込みに使用するか、バッチ指向 CreateBulkImportJob API を使用して、より大きなバッチでのコスト効率の高い取り込みを容易にします。
- AWS IoT コアルール – AWS IoT Core ルールを使用して、 AWS IoT モノまたは別の AWS サービスによって発行された MQTT メッセージからデータをアップロードします。
- AWS IoT Events actions – の特定のイベントによってトリガーされるAWS IoT Events アクションを使用します AWS IoT Events。この方法は、データのアップロードがイベントの発生に関連付けられているシナリオに適しています。
- AWS IoT Greengrass ストリームマネージャー – AWS IoT Greengrass ストリームマネージャーを使用して、エッジデバイスを使用してローカルデータソースからデータをアップロードします。このオプションは、オンプレミスまたはエッジロケーションからデータが送信される状況に対応します。

これらのメソッドは、さまざまなソースからのデータを管理するための幅広いソリューションを提供します。各オプションの詳細を調べて、 AWS IoT SiteWise が提供するデータインジェスト機能を包括的に理解してください。

データストリームの管理

での資産モデルや資産の作成に取り掛かる前に AWS IoT SiteWise、まず産業機器からプラットフォームに情報を直接送信するようにデータソースを設定することから始めましょう。 AWS IoT SiteWise 未加工データを収集するデータストリームを自動的に生成するように設計されています。各データストリームは固有のエイリアスで識別されるため、各データの出所を簡単に追跡できます。

たとえば、風力発電所が AWS IoT SiteWise Edge ゲートウェイを使用して、気温、プロペラの回転速度、および出力の時系列データに関するデータを OPC-UA サーバーからに送信するとします。AWS IoT SiteWise`server1-windfarm/3/turbine/7/temperature`データストリームエイリアスは、風力発電所 #3 のタービン #7 からの温度値を識別します。 `server1`OPC-UA データソースの名前です。`server1`プレフィックスは、このサーバーから送信されるすべてのデータストリームに使用され、ソース別にデータを整理するのに役立ちます。

アセットモデルとアセットを作成したら、各データストリームを特定のアセットプロパティに関連付けて、流入するデータを整理します。 AWS IoT SiteWise この関連付けにより、データを収集するだけでなく、アセットの構造に従って処理することもできます。必要に応じて、データストリームとアセットプロパティの間のリンクを削除することもできます。

現在、データストリームは測定値にのみ関連付けることができます。測定値とは、タイムスタンプ付きの温度値やタイムスタンプ付きの回転数 (RPM) 値など、デバイスの raw センサーデータストリームを表す一種のアセットプロパティです。

これらの測定値によって指標や変換が定義されると、受信データによって特定の計算が実行されます。アセットプロパティは一度に 1 つのデータストリームにしかリンクできないことに注意してください。

> **Note**
>
> アセットプロパティは、同時に複数のデータストリームに関連付けることはできません。

AWS IoT SiteWise Amazon リソースネーム (ARN) `TimeSeries` リソースを使用してストレージ料金を決定します。詳細については、「AWS IoT SiteWise の料金」を参照してください。

以下のセクションでは、 AWS IoT SiteWise コンソールまたは API を使用してデータストリームを管理する方法を示します。

トピック

- データストリームの管理

データストリームの管理

データストリームを管理するにあたっては、以下を実行しておきます。

> **Note**
>
> 2021 年 11 月 24 AWS IoT SiteWise 日以降を初めて使用する場合は、このセクションをスキップできます。 AWS IoT SiteWise この日より前に使用を開始したお客様は、 AWS IoT SiteWise アセットモデルやアセットなしでデータを取り込むことができるようにサービス設定を構成する必要があります。

- 次の例に示すようなアクセス許可を IAM ロールが持っていることを確認してください。

  Example IAM ユーザーポリシー

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PutAssetPropertyValuesAssetPropertyOnly",
            "Effect": "Allow",
            "Action": "iotsitewise:BatchPutAssetPropertyValue",
            "Resource": "arn:aws:iotsitewise:*:*:asset/*"
        },
        {
            "Sid": "PutAssetPropertyValuesPropertyAliasAllowed",
            "Effect": "Allow",
            "Action": "iotsitewise:BatchPutAssetPropertyValue",
            "Resource": "arn:aws:iotsitewise:*:*:time-series/*"
        }
    ]
}
```

> **Important**
>
> データストリームにデータを取り込む前に、次のことを行ってください。

- データストリームの識別にプロパティエイリアスを使用する場合は、`time-series` リソースを認可する必要があります。
- 関連するアセットプロパティを含むアセットを識別するためにアセット ID を使用する場合、`asset` リソースは認可されていなければなりません。

IAM ポリシーの設定の詳細については、[IAM User Guide] (IAM ユーザーガイド) の [Managing IAM policies] (IAM ポリシーの管理) を参照してください。

- AWS IoT SiteWise アセットのプロパティに関連付けられていないデータストリームを受け付けるようにデータ取り込み設定を行います。

トピック

- データ取り込み設定の設定
- データストリームの管理

データ取り込み設定の設定

Console

AWS IoT SiteWise コンソールを使用して AWS IoT SiteWise 、アセットプロパティに関連付けられていないデータストリームを受け入れるように設定します。

データ取り込みの設定を行うには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Settings] (設定) の下にある [Data ingestion] (データの取り込み) を選択します。
3. [Data ingestion] (データの取り込み) ページで、[Edit] (編集) を選択します。
4. [Disassociated data ingestion] (関連付けのないデータの取り込み) セクションで、[Enable data ingestion for data streams not associated with asset properties] (アセットプロパティに関連しないデータストリームのデータインジェストを可能にする) を選択します。

> **Important**
>
> AWS IoT SiteWise アセットプロパティに関連付けられていないデータストリームを受け入れるように設定した後は、この設定をオフにすることはできません。

5. [保存] を選択します。
6. [Enable disassociated data ingestion] (分離されたデータの取り込みを有効にする) で、[Update] (更新) を選択します。[関連付けのないデータインジェスト] のステータスが「アクティブ」になります。このプロセスには数分かかることがあります。

AWS CLI

PutStorageConfigurationAPI オペレーションを使用して、アセットプロパティに関連付けられていないデータストリームを受け入れるように設定します AWS IoT SiteWise 。次の項では、 AWS CLIを使用します。

データ取り込みの設定を行うには (AWS CLI)。

1. AWS IoT SiteWise アセットプロパティに関連付けられていないデータストリームを受信するように設定するには、以下のコマンドを実行します。

> **Important**
>
> AWS IoT SiteWise アセットプロパティに関連付けられていないデータストリームを受け入れるように設定した後は、この設定をオフにすることはできません。

```
aws iotsitewise put-storage-configuration \
                                  -\-storage-type SITEWISE_DEFAULT_STORAGE \
                                  -\-disassociated-data-storage ENABLED
```

`storageType` が `MULTI_LAYER_STORAGE` を使用するように設定できます。詳細については、「ストレージの管理」を参照してください。

Example レスポンス

```
{
```

```
            "storageType": "SITEWISE_DEFAULT_STORAGE",
            "disassociatedDataStorage": "ENABLED",
            "configurationStatus": {
                "state": "UPDATE_IN_PROGRESS"
            }
        }
```

このプロセスには数分かかることがあります。

2. ストレージの設定情報を取得するには、次のコマンドを実行します。

```
aws iotsitewise describe-storage-configuration
```

Example レスポンス

```
{
            "storageType": "SITEWISE_DEFAULT_STORAGE",
            "disassociatedDataStorage": "ENABLED",
            "configurationStatus": {
                "state": "ACTIVE"
            },
            "lastUpdateDate": "2021-11-16T15:54:14-07:00"
        }
```

データストリームの管理

AWS IoT SiteWise コンソール AWS CLIまたはを使用してデータストリームを管理します。

Console

AWS IoT SiteWise コンソールを使用してデータストリームを管理します。

データストリームを管理するには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Data streams] (データストリーム) を選択します。
3. (オプション) タグを追加または更新するには、編集するデータストリームを選択し、[タグの管理] を選択します。

[タグの編集] ページで、[タグの追加] を選択します。[キー] フィールドに、使用するタグの名前を入力します。

[保存] を選択します。

4. (オプション) [Data stream] (データストリーム) 表では、次の方法でデータストリームをフィルタリングすることができます。

   - 最初のドロップダウンメニューで、エイリアスプレフィックスまたはアセット ID を選択します。
     - [Alias prefix] (エイリアスプレフィックス) - データストリームのエイリアスのプレフィックス。ターゲットデータストリームにエイリアスのプレフィックスがある場合、このオプションを選択することができます。
     - [Asset ID] (アセット ID) - アセットプロパティが作成されたアセットの ID。ターゲットデータストリームがアセットプロパティと関連付けられている場合、このオプションを選択することができます。
   - 2 番目のドロップダウンメニューで、[すべてのデータストリーム]、[関連するデータストリーム]、または [関連付けられていないデータストリーム] を選択します。
     - [All data streams] (すべてのデータストリーム) - アセットプロパティに関連する、または関連しないデータストリーム。
     - [Associated data streams] (関連するデータストリーム) - アセットプロパティに関連するデータストリーム。
     - Disassociated data streams (関連付けられていないデータストリーム) - アセットプロパティと関連付けられていないデータストリーム。
5. 管理しているデータストリームを選択します。 AWS IoT SiteWise 選択したデータストリームがページ下部のグラフに表示されます。10 個以上選択した場合、グラフは最初に選択した 10 個のみが表示されます。
6. (オプション) 以下の方法でグラフを設定します。

   a. [集計機能] で次のいずれかを選択してください。

      - [Data point count] (データ点数コンピューティング) - 現在の時間間隔における指定された変数のデータポイントの合計数を返します。
      - [Average] (平均) – 現在の時間間隔における指定された変数値の平均を返します。
      - [Sum] (合計) – 現在の時間間隔における指定された変数値の合計を返します。
      - [Minimum] (最小値) – 現在の時間間隔における指定された変数の最小値を返します。

- [Maximum] (最大値) – 現在の時間間隔における指定された変数の最大値を返します。

詳細については、「数式での集計関数の使用」を参照してください。

b. [時間範囲] で次のいずれかを選択します。

- [Last 1 hour] (過去 1 時間) - グラフは過去 1 時間のデータを集計して表示します。
- [Last 2 hour] (過去 2 時間) - グラフは過去 2 時間のデータを集計して表示します。
- [Last 3 hour] (過去 3 時間) - グラフは過去 3 時間のデータを集計して表示します。
- [Last 4 hour] (過去 4 時間) - グラフは過去 4 時間のデータを集計して表示します。

c. [時間間隔] で次のいずれかを選択します。

- [1 minute] (1 分) - 指定した時間範囲のデータを 1 分ごとに集計します。
- [1 hour] (1 時間) - 指定した時間範囲のデータを 1 時間ごとに集計します。

7. [Manage data streams] (データストリームの管理) を選択します。
8. [データストリームの関連付けの更新] セクションの [測定名] 列で、次のいずれかを実行します。

- データストリームが測定値と関連付けられている場合、閉じるアイコンを選択して関連付けを削除します。
- データストリームが測定値に関連付けられていない場合、[Choose measurement] (測定値の選択) を選択します。

9. [Choose measurement] (測定値の選択) の表で、対象アセットに移動し、関連付ける測定を選択します。
10. (オプション) [Update asset property aliases] (アセットプロパティエイリアスの更新) セクションで、各測定に対して一意のエイリアスを入力します。
11. [Update (更新)] を選択します。

[ステータス] 列には、次のいずれかの値が表示されます。

- [Pending] (保留中) - データストリームの関連付けまたはアセットプロパティのエイリアスを更新しているところです。
- [Submit] (送信) - 関連付けまたはアセットプロパティのエイリアスへの変更が保存されます。
- エラー — AWS IoT SiteWise 測定のデータストリームの関連付けまたはエイリアスを更新するリクエストを処理できませんでした。

- [Success] (成功) - データストリームのアソシエーションまたは測定のエイリアスを正常に更新しました。

AWS CLI

次の API オペレーションを使用してデータストリームを管理してください。コード例では、AWS CLIを使用しています。

- AssociateTimeSeriesToAssetProperty— データストリーム (時系列) をアセットプロパティに関連付けます。
- DisassociateTimeSeriesFromAssetProperty— データストリームとアセットプロパティの関連付けを解除します。
- DeleteTimeSeries— データストリームを削除します。
- DescribeTimeSeries— データストリームに関する情報を取得します。
- ListTimeSeries— ページ分割されたデータストリームのリストを取得します。

AssociateTimeSeriesToAssetProperty

データストリームとアセットプロパティを関連付けるには、次のコマンドを実行します。

> **Important**
>
> 指定されたアセットプロパティは、現在どのデータストリームにも関連付けられてはならない。

- *data-stream-alias*関連付けるデータストリームのエイリアスに置き換えます。
- *asset-ID* をアセットプロパティが作成されたアセットの ID に置き換えてください。
- *property-ID* をアセットプロパティの ID に置き換えてください。

```
aws iotsitewise associate-time-series-to-asset-property \
                        --alias data-stream-alias \
                        --assetId asset-ID \
                        --propertyId property-ID
```

DisassociateTimeSeriesFromAssetProperty

データストリームとアセットプロパティの関連付けを解除するには、次のコマンドを実行します。

- *data-stream-alias*関連付けを解除するデータストリームのエイリアスに置き換えます。
- *asset-ID* をアセットプロパティが作成されたアセットの ID に置き換えてください。
- *property-ID* をアセットプロパティの ID に置き換えてください。

```
aws iotsitewise disassociate-time-series-from-asset-property \
                        --alias data-stream-alias \
                        --assetId asset-ID \
                        --propertyId property-ID
```

DeleteTimeSeries

データストリームを削除するには、次のコマンドを実行します。

*data-stream-alias*削除するデータストリームのエイリアスに置き換えます。

```
aws iotsitewise delete-time-series --alias data-stream-alias
```

データストリームを特定するには、次のいずれかを行います。

- データストリームがアセットプロパティと関連付けられていない場合、データストリームの alias を指定します。
- データストリームがアセットプロパティと関連付けられている場合、次のいずれかを指定します。
  - データストリームの alias。
  - アセットプロパティを識別する assetId、propertyId です。

DescribeTimeSeries

DescribeTimeSeriesAPI オペレーションを使用して、データストリームを正常に関連付けたか、関連付けを解除したかを確認します。

データストリームの情報を取得するには、次のコマンドを実行します。

```
aws iotsitewise describe-time-series --alias data-stream-alias
```

データストリームを特定するには、次のいずれかを行います。

- データストリームがアセットプロパティと関連付けられていない場合、データストリームの `alias` を指定します。
- データストリームがアセットプロパティと関連付けられている場合、次のいずれかを指定します。
  - データストリームの `alias`。
  - アセットプロパティを識別する `assetId`、`propertyId` です。

ListTimeSeries

`ListTimeSeries`API オペレーションを使用して、データストリームを正常に削除したかどうかを確認します。

データストリームのページ割りされたリストを取得するには、次のコマンドを実行します。

```
aws iotsitewise list-time-series
```

API を使用してデータを取り込む AWS IoT SiteWise

AWS IoT SiteWise API を使用して、タイムスタンプ付きの産業データをアセットの属性および測定プロパティに送信します。API は timestamp-quality-value (TQV) 構造を含むペイロードを受け入れます。

BatchPutAssetPropertyValueオペレーションを使用してデータをアップロードします。この操作では、複数のデータエントリを一度にアップロードして複数のデバイスからデータを収集し、そのすべてを 1 回のリクエストで送信できます。

> **Important**
>
> BatchPutAssetPropertyValueこの操作には以下のクォータが適用されます。
>
> - 1 リクエストあたりのエントリ数: 最大 10。
> - 1 エントリあたりのプロパティ値 (TQV データポイント) 数: 最大 10。
> - AWS IoT SiteWise 過去 7 日以上、またはfuture の 10 分を超えるタイムスタンプのデータはすべて拒否されます。

これらのクォータの詳細については、API BatchPutAssetPropertyValueリファレンスのを参照してください。AWS IoT SiteWise

アセットプロパティを識別するには、次のいずれかを指定します。

- データの送信先となるアセットプロパティの AND。`assetId` `propertyId`
- データストリームのエイリアスである `propertyAlias` (例えば、`/company/windfarm/3/turbine/7/temperature`)。このオプションを使用するには、最初にアセットプロパティのエイリアスを設定する必要があります。プロパティエイリアスを設定するには、を参照してくださいアセットプロパティへの産業データストリームのマッピング。

次の例は、JSON ファイルに格納されているペイロードから風力タービンの温度と 1 分あたりの回転数 (RPM) の読み取り値を送信する方法を示しています。

```
aws iotsitewise batch-put-asset-property-value --cli-input-json file://batch-put-payload.json
```

`batch-put-payload.json` のペイロードの例には、次のコンテンツが含まれています。

```
{
  "entries": [
    {
      "entryId": "unique entry ID",
      "propertyAlias": "/company/windfarm/3/turbine/7/temperature",
      "propertyValues": [
        {
          "value": {
            "integerValue": 38
          },
          "timestamp": {
            "timeInSeconds": 1575691200
          }
        }
      ]
    },
    {
      "entryId": "unique entry ID",
      "propertyAlias": "/company/windfarm/3/turbine/7/rpm",
```

```
      "propertyValues": [
        {
          "value": {
            "doubleValue": 15.09
          },
          "timestamp": {
            "timeInSeconds": 1575691200
          },
          "quality": "GOOD"
        }
      ]
    }
  ]
}
```

ペイロードの各エントリには、一意の文字列として定義できる `entryId` が含まれています。リクエストのエントリが失敗した場合、各エラーには、対応するリクエストの `entryId` が含まれるため、再試行するリクエストを確認できます。

リスト内の各構造体は、a、a `timestamp`、およびオプションで a `value` を含む timestamp-quality-value (TQV) `propertyValues` 構造です。`quality`

- `value` - 設定中のプロパティの型に応じて、次のいずれかのフィールドを含む構造。
  - `booleanValue`
  - `doubleValue`
  - `integerValue`
  - `stringValue`
- `timestamp` エポック時間からの現在の UNIX エポック時刻 (秒単位) を含む構造。`timeInSeconds`時間的に正確なデータがある場合は、`offsetInNanos`timestamp構造にキーを設定することもできます。 AWS IoT SiteWise 過去 7 日以上経過した、またはfuture の 10 分を超えるタイムスタンプのデータポイントをすべて拒否します。
- `quality` -(オプション) 次の品質の文字列のいずれか。
  - `GOOD` - (デフォルト) データはいずれの問題による影響も受けません。
  - `BAD` - データはセンサーの障害などの問題による影響を受けます。
  - `UNCERTAIN` - データはセンサーの不正確さなどの問題による影響を受けます。

  AWS IoT SiteWise 計算におけるデータ品質の処理方法の詳細については、「数式におけるデータ品質」を参照してください。

ルールを使ったデータの取り込み AWS IoT Core

のルールを使用して AWS IoT Thing AWS IoT SiteWise AWS や他のサービスからにデータを送信します。 AWS IoT Coreルールは MQTT メッセージを変換し、 AWS サービスとやり取りするためのアクションを実行します。 AWS IoT SiteWise ルールアクションは API BatchPutAssetPropertyValueからメッセージデータをオペレーションに転送します。 AWS IoT SiteWise 詳しくは、[AWS IoT Developer Guide] (デベロッパーガイド) の[Rules] (ルール) [AWS IoT SiteWise action] (アクション) をご覧ください。

デバイスシャドウからデータを取り込むルールを設定するのに必要な手順を説明するチュートリアルに従うには、を参照してください。モノからのデータの取り込み AWS IoT

AWS IoT SiteWise 他のサービスからデータを送信することもできます。 AWS 詳細については、「AWS 他のサービスとのやり取り」を参照してください。

トピック

- AWS IoT 必要なアクセス権の付与
- AWS IoT SiteWise ルールアクションの設定
- 基本的な取り込みによるコストの削減

AWS IoT 必要なアクセス権の付与

IAM ロールを使用して、 AWS 各ルールがアクセスできるリソースを制御します。ルールを作成する前に、 AWS ルールが必要なリソースに対してアクションを実行することを許可するポリシーを含む IAM ロールを作成する必要があります。 AWS IoT ルールを実行するとこのロールを引き継ぎます。

AWS IoT コンソールでルールアクションを作成する場合、ルートアセットを選択して、選択したアセット階層にアクセスできるロールを作成できます。ルールのロールを手動で定義する方法について詳しくは、『AWS IoT 開発者ガイド』の「必要なアクセス権の付与 AWS IoT」と「ロール権限の譲渡」を参照してください。

AWS IoT SiteWise ルールアクションでは、`iotsitewise:BatchPutAssetPropertyValue`ルールがデータを送信するアセットプロパティへのアクセスを許可するロールを定義する必要があります。セキュリティを強化するために、 AWS IoT SiteWise `Condition`プロパティにアセット階層パスを指定できます。

次の信頼ポリシーの例では、特定のアセットとその子へのアクセスを許可します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iotsitewise:BatchPutAssetPropertyValue",
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "iotsitewise:assetHierarchyPath": [
            "/root node asset ID",
            "/root node asset ID/*"
          ]
        }
      }
    }
  ]
}
```

Conditionをポリシーから削除して、すべてのアセットへのアクセスを許可してください。次の信頼ポリシーの例では、現在のリージョン内のすべてのアセットへのアクセスを許可します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iotsitewise:BatchPutAssetPropertyValue",
      "Resource": "*"
    }
  ]
}
```

AWS IoT SiteWise ルールアクションの設定

AWS IoT SiteWise ルールアクションは、ルールを開始した MQTT メッセージのデータを内のアセットプロパティに送信します。 AWS IoT SiteWise複数のデータエントリを異なるアセットプロパティに同時にアップロードして、デバイスのすべてのセンサーの更新を 1 つのメッセージで送信できます。また、各データ入力に一度に複数のデータポイントをアップロードすることもできます。

> **Note**
>
> AWS IoT SiteWise ルールアクションを使用してにデータを送信する場合、`BatchPutAssetPropertyValue`データは操作のすべての要件を満たしている必要があります。たとえば、現在の UNIX エポック時間から 7 日以上前ののタイムスタンプをデータで使用することはできません。詳細については、「AWS IoT SiteWise API を使用したデータの取り込み」を参照してください。

ルールアクションの各データエントリについて、アセットプロパティを識別し、そのアセットプロパティの各データポイントのタイムスタンプ、品質、および値を指定します。ルールアクションには、すべてのパラメータに対する文字列を指定します。

エントリ内のアセットのプロパティを識別するには、以下のいずれかを指定します。

- データを送信するアセットプロパティの [アセット ID] (`assetId`) と [プロパティ ID] (`propertyId`)。アセット ID とプロパティ ID は、を使用して確認できます AWS IoT SiteWise コンソール。アセット ID がわかっている場合は、を使用してプロパティ ID を検索できます。 AWS CLI DescribeAsset
- [Property alias (プロパティエイリアス)] (`propertyAlias`)。これは、データストリームのエイリアス (`/company/windfarm/3/turbine/7/temperature` など) です。このオプションを使用するには、最初にアセットプロパティのエイリアスを設定する必要があります。プロパティのエイリアスを設定する方法については、「アセットプロパティへの産業データストリームのマッピング」を参照してください。

各エントリのタイムスタンプには、機器から報告されたタイムスタンプまたはから提供されたタイムスタンプを使用してください。 AWS IoT Coreタイムスタンプは 2 つのパラメータがあります。

- [Time in seconds] (時間 (秒)) (`timeInSeconds`) - センサーまたは機器がデータを報告した UNIX エポック時間 (秒)。
- [Offset in nanos] (オフセット (ナノ秒)) (`offsetInNanos`) -(オプション) 時間 (秒) からのオフセット (ナノ秒)。

> **Important**
>
> タイムスタンプが文字列であったり、小数部があったり、秒単位でない場合、 AWS IoT SiteWise はリクエストを拒否します。タイムスタンプを秒とナノ秒のオフセットに変換す

る必要があります。 AWS IoT ルールエンジンの機能を使用してタイムスタンプを変換します。詳細については、次を参照してください。

- 正確な時間を報告しないデバイスのタイムスタンプを取得する。
- 文字列形式のタイムスタンプを変換する

アクションのいくつかのパラメータに置換テンプレートを使用して、コンピューティングを実行したり、関数を呼び出したり、メッセージペイロードから値を引き出したりすることができます。詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Substitution templates] (置換テンプレート) を参照してください。

> **Note**
>
> 置換テンプレート内の式は SELECT ステートメントとは別に評価されるため、AS 句を使用して作成されたエイリアスは置換テンプレートを使用して参照することはできません。サポートされている関数と演算子に加えて、元のペイロードに存在する情報のみを参照できます。

トピック

- 正確な時間を報告しないデバイスのタイムスタンプを取得する。
- 文字列形式のタイムスタンプを変換する
- ナノ秒精度のタイムスタンプ文字列を変換する。
- ルール設定の例。
- ルールアクションのトラブルシューティング

正確な時間を報告しないデバイスのタイムスタンプを取得する。

センサーや機器が正確な時間データを報告しない場合は、timestamp () AWS IoT を使用してルールエンジンから現在の Unix エポック時間を取得します。この関数は、ミリ秒単位で時間を出力するので、時間を秒単位、オフセットをナノ秒単位に変換する必要があります。そのためには、次の変換を使用します。

- [Time in seconds (時間 (秒))] (`timeInSeconds`) には、**`${floor(timestamp() / 1E3)}`** を使用して、時間をミリ秒から秒に変換します。

- [Offset in nanos (オフセット (ナノ秒))] (`offsetInNanos`) には、**`${(timestamp() % 1E3) * 1E6}`** を使用して、タイムスタンプオフセット (ナノ秒) をコンピューティングします。

文字列形式のタイムスタンプを変換する

センサーまたは機器が時間データを文字列形式 (例:) で報告する場合は、time_to_epoch (String, String`2020-03-03T14:57:14.699Z`) を使用してください。この関数は、タイムスタンプとフォーマットパターンをパラメータとして入力し、時刻をミリ秒単位で出力します。次に、時間を秒単位で、オフセットをナノ秒単位で変換する必要があります。そのためには、次の変換を使用します。

- [Time in seconds] (秒単位の時間) (`timeInSeconds`) の場合、**`${floor(time_to_epoch("2020-03-03T14:57:14.699Z", "yyyy-MM-dd'T'HH:mm:ss'Z'") / 1E3)}`** でタイムスタンプ文字列をミリ秒に変換し、さらに秒に変換します。
- [Offset in nanos](オフセット (ナノ秒)) (`offsetInNanos`) には、**`${(time_to_epoch("2020-03-03T14:57:14.699Z", "yyyy-MM-dd'T'HH:mm:ss'Z'") % 1E3) * 1E6}`** を使用して、タイムスタンプ文字列オフセット (ナノ秒) をコンピューティングします。

> **Note**
>
> `time_to_epoch` 関数は、最大ミリ秒精度のタイムスタンプ文字列をサポートします。文字列をマイクロ秒またはナノ秒の精度で変換するには、 AWS Lambda タイムスタンプを数値に変換するためにルールが呼び出す関数を設定します。詳細については、「ナノ秒精度のタイムスタンプ文字列を変換する。」を参照してください。

ナノ秒精度のタイムスタンプ文字列を変換する。

デバイスがナノ秒精度のタイムスタンプ情報を文字列形式 (`2020-03-03T14:57:14.699728491Z` など) で送信する場合、次のステップでルールアクションを設定します。タイムスタンプを文字列の時間 (秒)、オフセット (nanos**`timeInSeconds`**) () AWS Lambda に変換する関数を作成できます。`offsetInNanos`次に、ルールアクションパラメータで aws_lambda (functionArn、inputJson) を使用してその Lambda 関数を呼び出し、その出力をルールで使用します。

> **Note**
>
> このセクションでは、次のリソースの作成方法を熟知していることを前提とした高度な手順について説明します。
>
> - Lambda 関数。詳細については、[AWS Lambda Developer Guide] (デベロッパーガイド) の[Create a Lambda function with the console] (コンソールでLambda 関数を作成する) または[Using Lambda with the AWS CLI] (Lambda を使用する) を参照してください。
> - AWS IoT ルールとルールアクション。 AWS IoT SiteWise 詳細については、「ルールを使ったデータの取り込み AWS IoT Core」を参照してください。

AWS IoT SiteWise タイムスタンプ文字列を解析するルールアクションを作成するには

1. 次のプロパティを持つ Lambda 関数を作成します。

   - [Function name] (関数名) - わかりやすい関数名を使用します (例: **`ConvertNanosecondTimestampFromString`**)。
   - ランタイム — Python 3.11 (**`python3.11`**) などの Python 3 ランタイムを使用してください。
   - 権限 — 基本的な Lambda 権限を持つロールを作成します () AWS LambdaBasicExecutionRole。
   - レイヤー — Lambda AWS関数が使用できるように SDKPandas-Python311 レイヤーを追加します。numpy
   - [Function code] (関数コード) - 次の関数コードを使用します。この関数コードでは、`timestamp` という文字列引数を消費し、そのタイムスタンプの `timeInSeconds` と `offsetInNanos` 値を出力します。

     ```
     import json
     import math
     import numpy

     # Converts a timestamp string into timeInSeconds and offsetInNanos in Unix epoch
      time.
     # The input timestamp string can have up to nanosecond precision.
     def lambda_handler(event, context):
         timestamp_str = event['timestamp']
         # Parse the timestamp string as nanoseconds since Unix epoch.
         nanoseconds = numpy.datetime64(timestamp_str, 'ns').item()
         time_in_seconds = math.floor(nanoseconds / 1E9)
     ```

```
    # Slice to avoid precision issues.
    offset_in_nanos = int(str(nanoseconds)[-9:])
    return {
        'timeInSeconds': time_in_seconds,
        'offsetInNanos': offset_in_nanos
    }
```

この Lambda 関数は、から datetime64 を使用して ISO 8601 形式のタイムスタンプ文字列を入力します。 NumPy

> **Note**
>
> タイムスタンプ文字列が ISO 8601 形式でない場合、タイムスタンプ形式を定義した pandas で解決策を実装することが可能です。詳しくはpandas.to_datetimeを参照してください。

2. AWS IoT SiteWise ルールに合わせてアクションを設定するときは、「秒単位の時間 ()」と「ナノ秒単位のオフセット ()」(**timeInSeconds**) に次の代替テンプレートを使用します。offsetInNanosこれらの置換テンプレートは、メッセージペイロードに timestamp のタイムスタンプ文字列が含まれていることを前提としています。この aws_lambda 関数は 2 番目のパラメータに JSON 構造消費するため、必要に応じて以下の置換テンプレートを変更できます。

   - [Time in seconds (時間 (秒))] (timeInSeconds) には、次の置換テンプレートを使用します。

     ```
     ${aws_lambda('arn:aws:lambda:region:account-
     id:function:ConvertNanosecondTimestampFromString', {'timestamp':
      timestamp}).timeInSeconds}
     ```

   - Offset in nanos (オフセット (ナノ秒))] (offsetInNanos) には、次の置換テンプレートを使用します。

     ```
     ${aws_lambda('arn:aws:lambda:region:account-
     id:function:ConvertNanosecondTimestampFromString', {'timestamp':
      timestamp}).offsetInNanos}
     ```

   各パラメータについて、*###### account-id ############### ID* に置き換えてください。 AWS Lambda 関数に別の名前を使用した場合は、それも変更してください。

3. AWS IoT その権限で関数を呼び出すための権限を付与します。`lambda:InvokeFunction`詳細については、「aws_lambda (functionArn, inputJson) 」を参照してください。
4. ルールをテスト (たとえば AWS IoT MQTT テストクライアントを使用) し、 AWS IoT SiteWise 送信したデータを受信することを確認します。

   ルールが期待どおりに動作しない場合は、「AWS IoT SiteWise ルールアクションのトラブルシューティング」を参照してください。

> **Note**
>
> このソリューションは、タイムスタンプ文字列ごとに Lambda 関数を 2 回呼び出します。ルールが各ペイロードで同じタイムスタンプを持つ複数のデータポイントを処理する場合、別のルールを作成して Lambda 関数呼び出し回数を減らすことができます。
> そのためには、Lambda を呼び出し、タイムスタンプ文字列を `timeInSeconds` と `offsetInNanos` に変換して元のペイロードをパブリッシュする再パブリッシュアクションを含むルールを作成します。次に、 AWS IoT SiteWise 変換されたペイロードを使用するルールアクションを含むルールを作成します。このアプローチでは、ルールが Lambda を呼び出す回数を減らしますが、 AWS IoT 実行するルールアクションの数は増やします。このソリューションをユースケースに適用する場合は、各サービスの価格を考慮してください。

ルール設定の例。

このセクションには、アクションを含むルールを作成するためのルール設定の例が含まれています。AWS IoT SiteWise

Example プロパティエイリアスをメッセージトピックとして使用するルールアクションの例

次の例では、アセットのプロパティを識別するプロパティエイリアスとして (topic () から) AWS IoT SiteWise トピックを使用するアクションを含むルールを作成します。この例を使用して、すべての風力発電所のすべての風力タービンにダブルタイプのデータを取り込むための 1 つのルールを定義します。この例では、すべてのタービンアセットのプロパティにプロパティエイリアスを定義する必要があります。整数型データを取り込むには、同様のルールをもう一つ定義する必要があります。

```
aws iot create-topic-rule \
  --rule-name SiteWiseWindFarmRule \
  --topic-rule-payload file://sitewise-rule-payload.json
```

sitewise-rule-payload.json のペイロードの例には、次のコンテンツが含まれています。

```
{
  "sql": "SELECT * FROM '/company/windfarm/+/turbine/+/+' WHERE type = 'double'",
  "description": "Sends data to the wind turbine asset property with the same alias as
 the topic",
  "ruleDisabled": false,
  "awsIotSqlVersion": "2016-03-23",
  "actions": [
    {
      "iotSiteWise": {
        "putAssetPropertyValueEntries": [
          {
            "propertyAlias": "${topic()}",
            "propertyValues": [
              {
                "timestamp": {
                  "timeInSeconds": "${timeInSeconds}"
                },
                "value": {
                  "doubleValue": "${value}"
                }
              }
            ]
          }
        ],
        "roleArn": "arn:aws:iam::account-id:role/MySiteWiseActionRole"
      }
    }
  ]
}
```

このルールアクションでは、データを取り込むトピックとして次のメッセージを風力タービンのプロパティエイリアス (例:/company/windfarm/3/turbine/7/temperature) に送信します。

```
{
  "type": "double",
  "value": "38.3",
  "timeInSeconds": "1581368533"
}
```

Example timestamp () を使用して時刻を決定するルールアクションの例

次の例では、ID でアセットプロパティを識別し、timestamp () AWS IoT SiteWise を使用して現在の時刻を決定するアクションを含むルールを作成します。

```
aws iot create-topic-rule \
  --rule-name SiteWiseAssetPropertyRule \
  --topic-rule-payload file://sitewise-rule-payload.json
```

`sitewise-rule-payload.json` のペイロードの例には、次のコンテンツが含まれています。

```
{
  "sql": "SELECT * FROM 'my/asset/property/topic'",
  "description": "Sends device data to an asset property",
  "ruleDisabled": false,
  "awsIotSqlVersion": "2016-03-23",
  "actions": [
    {
      "iotSiteWise": {
        "putAssetPropertyValueEntries": [
          {
            "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
            "propertyId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
            "propertyValues": [
              {
                "timestamp": {
                  "timeInSeconds": "${floor(timestamp() / 1E3)}",
                  "offsetInNanos": "${(timestamp() % 1E3) * 1E6}"
                },
                "value": {
                  "doubleValue": "${value}"
                }
              }
            ]
          }
        ],
        "roleArn": "arn:aws:iam::account-id:role/MySiteWiseActionRole"
      }
    }
  ]
}
```

このルールアクションでは、次のメッセージをに送信してデータを取り込みます。my/asset/property/topic

```
{
  "type": "double",
  "value": "38.3"
}
```

ルールアクションのトラブルシューティング

AWS IoT SiteWise でルールアクションをトラブルシューティングするには AWS IoT Core、CloudWatch ルールにログを設定するか、再公開エラーアクションを設定します。詳細については、「AWS IoT SiteWise ルールアクションのトラブルシューティング」を参照してください。

基本的な取り込みによるコストの削減

AWS IoT Core にはベーシックインジェストと呼ばれる機能があり、 AWS IoT Core これを使用してメッセージングコストを発生させずにデータを送信できます。AWS IoT基本的な取り込みは、取り込みパスからパブリッシュ/サブスクライブのメッセージブローカーを除去して、大量のデータ取り込みワークロードのデータフローを最適化します。メッセージをルーティングするルールがわかっている場合は、基本的な取り込みを使用できます。

基本的な取り込みを使用するには、特別なトピック、$aws/rules/*rule-name* を使用して、特定のルールに直接メッセージを送信します。たとえば、SiteWiseWindFarmRule という名前のルールにメッセージを送信するには、トピック $aws/rules/SiteWiseWindFarmRule にメッセージを送信します。

ルールアクションで、トピック (10 進数) を含む置換テンプレートが使用されている場合、$aws/rules/*rule-name*/*original-topic* などの基本的な取り込みの特別なトピックの最後に元のトピックを渡すことができます。たとえば、前のセクションの風力発電施設プロパティエイリアスの例で基本的な取り込みを使用するには、次のトピックにメッセージを送信できます。

```
$aws/rules/SiteWiseWindFarmRule//company/windfarm/3/turbine/7/temperature
```

> **Note**
>
> 上記の例には、ルールアクションに表示されるトピックから Basic Ingest プレフィックス (//) AWS IoT が削除されるため、2 つ目のスラッシュ ($aws/rules/*rule-name*/) が使

用されています。この例では、ルールはトピック `/company/windfarm/3/turbine/7/temperature` を受信します。

詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Reducing messaging costs with basic ingest] (ベーシックインジェストでメッセージングコストを削減) を参照してください。

からのデータの取り込み AWS IoT Events

を使用すると AWS IoT Events、IoT AWS フリート向けの複雑なイベント監視アプリケーションをクラウドで構築できます。 AWS IoT SiteWise イベントが発生すると、IoT SiteWise アクションを使用してアセットプロパティにデータを送信します。 AWS IoT Events

AWS IoT Events AWS クラウド内の IoT デバイスおよびシステム用のイベント監視アプリケーションの開発を効率化するように設計されています。を使用すると AWS IoT Events、次のことが可能になります。

- IoT フリート全体の変化、異常、または特定の条件を検出して対応します。
- 運用効率を高め、IoT エコシステムの積極的な管理を可能にします。

AWS IoT SiteWise AWS IoT SiteWise AWS IoT Events アクションを通じてと統合することで機能が拡張され、特定のイベントに応じてアセットのプロパティを自動的に更新できるようになります。AWS IoT SiteWise このやりとりにより、データの取り込みと管理が簡単になります。また、実行可能なインサイトも得られます。

詳細については、[AWS IoT Events Developer Guide] (デベロッパーガイド) の次のトピックをご覧ください。

- とは何ですか? AWS IoT Events
- AWS IoT Events アクション
- IoT SiteWise アクション

AWS IoT Greengrass ストリームマネージャーを使用する

AWS IoT Greengrass ストリームマネージャーは、 AWS ローカルソースからクラウドへのデータストリームの転送を容易にする統合機能です。データフローを管理する中間層として機能し、エッジで

動作するデバイスが、送信前にデータを収集して保存し AWS IoT SiteWise、さらに分析や処理を行えるようにします。

データ送信先を追加するには、 AWS IoT SiteWise コンソールでローカルソースを設定します。AWS IoT Greengrass カスタムソリューションでストリームマネージャーを使用してデータを取り込むこともできます。 AWS IoT SiteWise

> **Note**
>
> OPC-UA ソースからデータを取り込むには、上で動作する AWS IoT SiteWise Edge ゲートウェイを設定します。 AWS IoT Greengrass詳細については、「SiteWise Edge ゲートウェイを使用する」を参照してください。

ローカルソースデータの送信先を設定する方法の詳細については、を参照してください。データソースの設定

AWS IoT Greengrass カスタムソリューションで Stream Manager を使用してデータを取り込む方法の詳細については、『AWS IoT Greengrass Version 2 開発者ガイド』の以下のトピックを参照してください。

- とは何ですか AWS IoT Greengrass?
- [Manage data streams on the AWS IoT Greengrass core] (コアでのデータストリームの管理)。
- AWS IoT SiteWise 資産プロパティへのデータのエクスポート

API を使用してデータを取り込む CreateBulkImportJob

`CreateBulkImportJob`API を使用して Amazon S3 から大量のデータをインポートします。Amazon S3 のデータは、CSV 形式で保存されている必要があります。データファイルには、次の列を含めることができます。

> **Note**
>
> アセットプロパティを識別するには、次のいずれかを指定します。
>
> - データの送信先のアセットプロパティの `ASSET_ID` および `PROPERTY_ID`。
> - データストリームのエイリアスである `ALIAS` (例えば、`/company/windfarm/3/turbine/7/temperature`)。このオプションを使用するには、最初にアセットプロパ

ティのエイリアスを設定する必要があります。プロパティのエイリアスを設定する方法については、「the section called “アセットプロパティへの産業データストリームのマッピング”」を参照してください。

- ALIAS – OPC-UA サーバのデータストリームのパスなど、プロパティを識別するエイリアス (例: /company/windfarm/3/turbine/7/temperature)。詳細については、「アセットプロパティへの産業データストリームのマッピング」を参照してください。
- ASSET_ID – アセットの ID。
- PROPERTY_ID – アセットプロパティの ID。
- DATA_TYPE – プロパティのデータ型。次のいずれかを指定できます。
  - STRING - 最大 1024 バイトの文字列。
  - INTEGER - [-2,147,483,648, 2,147,483,647] の範囲を持つ 32 ビットの符合付き整数。
  - DOUBLE - [-10 ^ 100、10 ^ 100] の範囲および IEEE 754 倍精度の浮動小数点数。
  - BOOLEAN – true または false
- TIMESTAMP_SECONDS – データポイントのタイムスタンプ (Unix エポック時間形式)。
- TIMESTAMP_NANO_OFFSET – TIMESTAMP_SECONDS から変換されたナノ秒オフセット。
- QUALITY – (オプション) アセットプロパティ値の品質。値には次のいずれかを指定できます。
  - GOOD - (デフォルト) データはいずれの問題による影響も受けません。
  - BAD - データはセンサーの障害などの問題による影響を受けます。
  - UNCERTAIN - データはセンサーの不正確さなどの問題による影響を受けます。

  AWS IoT SiteWise 計算におけるデータ品質の処理方法の詳細については、「数式におけるデータ品質」を参照してください。
- VALUE – アセットプロパティの値。

Example .csv 形式のデータファイル

```
asset_id,property_id,DOUBLE,1635201373,0,GOOD,1.0
asset_id,property_id,DOUBLE,1635201374,0,GOOD,2.0
asset_id,property_id,DOUBLE,1635201375,0,GOOD,3.0
```

```
unmodeled_alias1,DOUBLE,1635201373,0,GOOD,1.0
unmodeled_alias1,DOUBLE,1635201374,0,GOOD,2.0
```

```
unmodeled_alias1,DOUBLE,1635201375,0,GOOD,3.0
unmodeled_alias1,DOUBLE,1635201376,0,GOOD,4.0
unmodeled_alias1,DOUBLE,1635201377,0,GOOD,5.0
unmodeled_alias1,DOUBLE,1635201378,0,GOOD,6.0
unmodeled_alias1,DOUBLE,1635201379,0,GOOD,7.0
unmodeled_alias1,DOUBLE,1635201380,0,GOOD,8.0
unmodeled_alias1,DOUBLE,1635201381,0,GOOD,9.0
unmodeled_alias1,DOUBLE,1635201382,0,GOOD,10.0
```

AWS IoT SiteWise 一括インポートジョブを作成し、既存のジョブに関する情報を取得するための、次の API オペレーションを提供します。

- CreateBulkImportJob— 新しい一括インポートジョブを作成します。
- DescribeBulkImportJob— 一括インポートジョブに関する情報を取得します。
- ListBulkImportJob— すべての一括インポートジョブの概要をページ分割したリストを取得します。

一括インポートジョブ (AWS CLI) を作成する

CreateBulkImportJobAPI オペレーションを使用して、Amazon S3 AWS IoT SiteWiseからにデータを転送します。CreateBulkImportJobAPI を使用すると、費用対効果の高い方法でデータを小さなバッチで取り込むことができます。次の例では AWS CLIを使用しています。

> **Important**
>
> 一括インポートジョブを作成する前に、 AWS IoT SiteWise AWS IoT SiteWise ウォーム階層またはコールド階層を有効にする必要があります。詳細については、「ストレージ設定の構成」を参照してください。
> 一括インポートは、履歴データをに保存するように設計されています AWS IoT SiteWise。AWS IoT SiteWise AWS IoT SiteWise ウォームティアやコールドティアでは計算や通知は開始されません。

以下のコマンドを実行します。*file-name* の部分は、一括インポートジョブの設定を含むファイルの名前に置き換えます。

```
aws iotsitewise create-bulk-import-job --cli-input-json file://file-name.json
```

Example ジョブ設定を一括インポートします。

構成設定の例を以下に示します。

- *adaptive-ingestion-flag* を `true` または `false` に置き換えます。
  - に設定すると`false`、一括インポートジョブは履歴データをに取り込みます AWS IoT SiteWise。
  - に設定すると`true`、一括インポートジョブは次の処理を行います。
    - 新しいデータをに取り込みます。 AWS IoT SiteWise
    - メトリクスとトランスフォームを計算し、7 日以内のタイムスタンプが付いたデータの通知をサポートします。
- ウォーム階層ストレージに取り込んだ後に S3 `true` データバケットからデータを削除するには、*delete-files-after-import-flag* をに置き換えます。 AWS IoT SiteWise
- *error-bucket* の部分は、この一括インポートジョブに関連するエラーの送信先である Amazon S3 バケットの名前に置き換えます。
- この一括インポートジョブに関連するエラーの送信先の Amazon S3 バケットのプレフィックスに置き換えます*error-bucket-prefix*。

  Amazon S3 では、このプレフィックスをフォルダ名として使用して、バケット内のデータを整理します。Amazon S3 バケット内の各オブジェクトには、バケット内の一意の識別子であるキーが含まれます。バケット内のすべてのオブジェクトは、厳密に 1 個のキーを持ちます。このプレフィックスは、スラッシュ (/) で終わる必要があります。詳細は、『Amazon Simple Storage Service ユーザーガイド』の「プレフィックスを使用してオブジェクトを整理する」を参照してください。
- *data-bucket* の部分は、データのインポート元である Amazon S3 バケットの名前に置き換えます。
- データを含む Amazon S3 *data-bucket-key*オブジェクトのキーに置き換えます。すべてのオブジェクトには、一意の識別子であるキーがあります。すべてのオブジェクトは、厳密に 1 個のキーを持ちます。
- データを含む Amazon S3 オブジェクトの特定のバージョンを識別するには、バージョン ID *data-bucket-version-id*に置き換えます。このパラメータはオプションです。
- *column-name* の部分は、.csv ファイルに指定されている列名に置き換えます。
- *job-name* の部分は、一括インポートジョブを識別する一意の名前に置き換えます。
- Amazon S3 AWS IoT SiteWise データの読み取りを許可する IAM *job-role-arn*ロールに置き換えてください。

Note

次の例に示すようなアクセス許可をロールが持っていることを確認してください。*data-bucket* は、データを含む Amazon S3 バケットの名前に置き換えてください。また、*error-bucket* は、この一括インポートジョブに関連するエラーの送信先となる Amazon S3 バケットの名前に置き換えてください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:GetObject",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::data-bucket",
                "arn:aws:s3:::data-bucket/*",
            ],
            "Effect": "Allow"
        },
      {
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetBucketLocation"
            ],
            "Resource": [
                "arn:aws:s3:::error-bucket",
                "arn:aws:s3:::error-bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

```
{
   "adaptiveIngestion": adaptive-ingestion-flag,
   "deleteFilesAfterImport": delete-files-after-import-flag,
   "errorReportLocation": {
```

```
      "bucket": "error-bucket",
      "prefix": "error-bucket-prefix"
   },
   "files": [
      {
         "bucket": "data-bucket",
         "key": "data-bucket-key",
         "versionId": "data-bucket-version-id"
      }
   ],
   "jobConfiguration": {
      "fileFormat": {
         "csv": {
            "columnNames": [ "column-name" ]
         }
      }
   },
   "jobName": "job-name",
   "jobRoleArn": "job-role-arn"
}
```

Example レスポンス

```
{
   "jobId":"f8c031d0-01d1-4b94-90b1-afe8bb93b7e5",
   "jobStatus":"PENDING",
   "jobName":"myBulkImportJob"
}
```

一括インポートジョブ の説明の取得 (AWS CLI)

DescribeBulkImportJobAPI オペレーションを使用して、一括インポートジョブに関する情報を取得します。以下の例ではを使用しています AWS CLI。

job-ID の部分は、取得する一括インポートジョブの ID に置き換えます。

```
aws iotsitewise describe-bulk-import-job --job-id job-ID
```

Example レスポンス

```
{
   "files":[
```

```
      {
         "bucket":"test-bucket",
         "key":"100Tags12Hours.csv"
      },
      {
         "bucket":"test-bucket",
         "key":"BulkImportData1MB.csv"
      },
      {
         "bucket":"test-bucket",
         "key":"UnmodeledBulkImportData1MB.csv"
      }
   ],
   "errorReportLocation":{
      "prefix":"errors/",
      "bucket":"test-error-bucket"
   },
   "jobConfiguration":{
      "fileFormat":{
         "csv":{
            "columnNames":[
               "ALIAS",
               "DATA_TYPE",
               "TIMESTAMP_SECONDS",
               "TIMESTAMP_NANO_OFFSET",
               "QUALITY",
               "VALUE"
            ]
         }
      }
   },
   "jobCreationDate":1645745176.498,
   "jobStatus":"COMPLETED",
   "jobName":"myBulkImportJob",
   "jobLastUpdateDate":1645745279.968,
   "jobRoleArn":"arn:aws:iam::123456789012:role/DemoRole",
   "jobId":"f8c031d0-01d1-4b94-90b1-afe8bb93b7e5"
}
```

一括インポートジョブの一覧の取得 (AWS CLI)

ListBulkImportJobsAPI オペレーションを使用して、すべての一括インポートジョブの概要をページ分割したリストを取得します。以下の例ではを使用しています。 AWS CLI

```
aws iotsitewise list-bulk-import-jobs --filter COMPLETED
```

Example レスポンス

```
{
   "jobSummaries":[
      {
         "id":"bdbbfa52-d775-4952-b816-13ba1c7cb9da",
         "name":"myBulkImportJob",
         "status":"COMPLETED"
      },
      {
         "id":"15ffc641-dbd8-40c6-9983-5cb3b0bc3e6b",
         "name":"myBulkImportJob2",
         "status":"RUNNING"
      }
   ]
}
```

SiteWise Edge ゲートウェイを使用する

AWS IoT SiteWise エッジゲートウェイは、産業機器とを仲介する役割を果たします。 AWS IoT SiteWise SiteWise AWS IoT Greengrass V2 エッジゲートウェイはその上で稼働し、オンプレミスでのデータ収集と処理をサポートします。を使用して AWS OpsHub SiteWise Edge ゲートウェイを管理し、現場での運用を監視できます。 AWS IoT SiteWise

ローカルデバイスの監視ポータルを使用して、 SiteWise 施設内のデータをローカルで監視できます。詳細については、「エッジでのポータルの有効化」を参照してください。

トピック

- SiteWise エッジゲートウェイの要件
- SiteWise Edge ゲートウェイの作成
- ローカルデバイスに SiteWise Edge ゲートウェイソフトウェアをインストールする
- エッジデータ処理を有効にする。
- エッジでのデータ処理
- AWS IoT SiteWise パブリッシャーコンポーネントの設定
- データソースの設定
- SiteWise Edge ゲートウェイへのパートナーデータソースの追加
- パックを使用する。
- SiteWise Edge ゲートウェイの管理
- 産業用 SiteWise エッジでの Edge の実行
- SiteWise Edge ゲートウェイのアセットのフィルタリング
- エッジでの AWS IoT SiteWise API の使用
- SiteWise Edge ゲートウェイのバックアップと復元
- SiteWise Edge ゲートウェイのセットアップ (AWS IoT Greengrass Version 1)

SiteWise エッジゲートウェイの要件

AWS IoT SiteWise エッジゲートウェイは、オンプレミスでのデータ収集、処理、公開をサポートする AWS IoT Greengrass コンポーネントのセット AWS IoT Greengrass V2 として で実行されます。

で実行される SiteWise Edge ゲートウェイを設定するには AWS IoT Greengrass V2、 でゲートウェイを作成し、 SiteWise Edge ゲートウェイソフトウェア AWS クラウド を実行してローカルデバイスをセットアップする必要があります。

要件

SiteWise Edge ゲートウェイソフトウェアをインストールして実行するには、ローカルデバイスが次の要件を満たしている必要があります。

- AWS IoT Greengrass V2 Core ソフトウェアバージョン v2.3.0 以降をサポートします。詳細については、[AWS IoT Greengrass Version 2 Developer Guide] (デベロッパーガイド) の[Requirements] (必要条件) を参照してください。
- 対応プラットフォームは次のいずれかです。
  - OS: Ubuntu 20.04 以降

    アーキテクチャ: x86_64 (AMD64) または ARMv8 (Aarch64)
  - OS: Red Hat Enterprise Linux (RHEL) 8

    アーキテクチャ: x86_64 (AMD64) または ARMv8 (Aarch64)
  - OS: Amazon Linux 2

    アーキテクチャ: x86_64 (AMD64) または ARMv8 (Aarch64)
  - OS: Debian 11

    アーキテクチャ: x86_64 (AMD64) または ARMv8 (Aarch64)
  - OS: Windows Server 2019 以降

    Architecture: x86_64 (AMD64)

  > **Note**
  >
  > ARM プラットフォームは、Data Collection Pack を備えた SiteWise Edge ゲートウェイのみをサポートします。Data Processing Pack はサポートされていません。
- 最小 4 GB の RAM。
- SiteWise Edge ゲートウェイソフトウェアで使用できる最小 10 GB のディスク容量。
- を使用してエッジでデータを処理する場合は AWS IoT SiteWise、ローカルデバイスも次の要件を満たしている必要があります。

  - x86 64 bit クアッドコアプロセッサ搭載。
  - 16 GB 以上の RAM があること。
  - Windows を使用している場合、RAM に 32 GB 以上が必要です。
  - 256 GB 以上の空きディスク容量があること。
- 最小ディスク容量とコンピューティング容量の要件は、実装とユースケースに固有のさまざまな要因によって異なります。
  - 断続的なインターネット接続のためにデータをキャッシュするために必要なディスク容量は、次の要因によって異なります。
    - アップロードされたデータストリームの数
    - 1 秒あたりのデータストリームごとのデータポイント
    - 各データポイントのサイズ
    - 通信速度
    - 予想されるネットワークのダウンタイム
  - データのポーリングとアップロードに必要なコンピューティング性能は、次の要因によって異なります。
    - アップロードされたデータストリームの数
    - 1 秒あたりのデータストリームごとのデータポイント
- 次の S3 バケットにアクセスするようにローカルデバイスを設定します: `iot-sitewise-gateway-<region>-748875242063`。
- 次のポートにアクセスできるようにローカルデバイスを設定します。
  - ローカルデバイスは、ポート 443 でネットワークインバウンドトラフィックを許可する必要があります。
  - ローカルデバイスは、ポート 443 および 8883 でアウトバウンドトラフィックを許可する必要があります。

    必要なアウトバウンドサービスエンドポイントの完全なリストについては、AWS IoT SiteWise「エッジゲートウェイの必須サービスエンドポイント」を参照してください。
  - 次のポートは、 での使用のために予約されています AWS IoT SiteWise: 80、443、3001、4569、4572、8000、8081、8082、8084、8085、8445、8086、9000、9500、11080 および 50010。トラフィック用の予約ポートを使用すると、接続が切断されることがあります。

> **Note**
>
> AWS IoT Greengrass V2 ストリームマネージャーコンポーネントには独自の要件があります。詳細については、「 AWS IoT Greengrass Version 2 デベロッパーガイド」の「設定」を参照してください。

- Java Runtime Environment (JRE) バージョン 11 以降。デバイスの PATH環境変数で Java が利用可能である必要があります。Java を使用してカスタムコンポーネントを開発するには、Java Development Kit (JDK) をインストールする必要があります。Amazon Corretto または OpenJDKを使用することをお勧めします。

SiteWise Edge ゲートウェイを使用するには、次のアクセス許可が必要です。

> **Note**
>
> AWS IoT SiteWise コンソールを使用して SiteWise Edge ゲートウェイを作成する場合、これらのアクセス許可が自動的に追加されます。

- SiteWise Edge ゲートウェイの IAM ロールでは、 AWS IoT Greengrass V2 デバイスで SiteWise Edge ゲートウェイを使用してアセットモデルデータとアセットデータを処理できる必要があります。

  このロールを担うことで、次のサービスを利用することができます: `credentials.iot.amazonaws.com`。

  許可の詳細

  ロールには、次のアクセス許可が必要です。

  - `iotsitewise` - プリンシパルがエッジでアセットモデルデータとアセットデータを取得できるようにする。
  - `iot` – AWS IoT Greengrass V2 デバイスが とやり取りできるようにします AWS IoT。
  - `logs` – AWS IoT Greengrass V2 デバイスが Amazon CloudWatch Logs にログを送信できるようにします。
  - `s3` — AWS IoT Greengrass V2 デバイスが Amazon S3 からカスタムコンポーネントアーティファクトをダウンロードできるようにします。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:BatchPutAssetPropertyValue",
                "iotsitewise:List*",
                "iotsitewise:Describe*",
                "iotsitewise:Get*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iot:DescribeCertificate",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams",
                "s3:GetBucketLocation",
                "s3:GetObject",
                "iot:Connect",
                "iot:Publish",
                "iot:Subscribe",
                "iot:Receive",
                "iot:DescribeEndpoint"
            ],
            "Resource": "*"
        }
    ]
}
```

SiteWise Edge ゲートウェイの作成

AWS IoT SiteWise コンソールを使用して SiteWise Edge ゲートウェイを作成できます。この手順では、独自のハードウェアにインストールするセルフホスト型 SiteWise Edge ゲートウェイを作成する方法について説明します。 Industrial SiteWise Edge で実行される Edge ゲートウェイの作成については、「」を参照してください産業用 SiteWise エッジでの Edge の実行。

SiteWise Edge ゲートウェイを作成する

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、エッジゲートウェイ を選択します。
3. [Create gateway (ゲートウェイの作成)] を選択します。
4. デプロイタイプ で、セルフホスト型ゲートウェイ を選択します。
5. SiteWise Edge ゲートウェイの名前を入力するか、 によって生成された名前を使用します AWS IoT SiteWise。
6. Greengrass デバイス OS で、この SiteWise Edge ゲートウェイをインストールするデバイスのオペレーティングシステムを選択します。

   > **Note**
   >
   > Data Processing Pack は x86 プラットフォームでのみ使用できます。

7. (オプション) エッジでデータを処理および整理するには、エッジ機能 で、データ処理パック を選択します。

   > **Note**
   >
   > 企業ディレクトリ内のユーザーグループにこの SiteWise Edge ゲートウェイへのアクセスを許可するには、「」を参照してください。 エッジ機能の設定。

8. (オプション) 詳細設定 で、次の操作を行います。

   - [Greengrass core device] (Greengrass コアデバイス) の場合、次のいずれかを選択します。
     - デフォルト設定 — デフォルト設定を使用して、 で Greengrass コアデバイス AWS を自動的に作成します AWS IoT Greengrass V2。
       1. Greengrass コアデバイスの名前を入力するか、 によって生成された名前を使用します AWS IoT SiteWise。
     - 詳細設定 — 既存の Greengrass コアデバイスを使用する場合、または手動で作成する場合は、このオプションを選択します。
       1. Greengrass コアデバイスを選択するか、[Create Greengrass core device] (Greengrass コアデバイスの作成) を選択して、 AWS IoT Greengrass V2 コンソールで作成し

ます。詳細については、「 AWS IoT Greengrass Version 2 デベロッパーガイド」のAWS IoT Greengrass V2 「コアデバイスのセットアップ」を参照してください。

9. [Create gateway (ゲートウェイの作成)] を選択します。
10. SiteWise 「エッジゲートウェイインストーラの生成」ダイアログボックスで、「 の生成とダウンロード」を選択します。 は、ローカルデバイスの設定に使用できるインストーラ AWS IoT SiteWise を自動的に生成します。

> **Important**
>
> インストーラーファイルは、必ず安全な場所に保存してください。このファイルは後で使用します。

Edge SiteWise ゲートウェイを作成したので、データソース を追加し、パブリッシャーコンポーネント を設定し、 SiteWise Edge ゲートウェイがデータを受信して AWS クラウドに送信します。

ローカルデバイスに SiteWise Edge ゲートウェイソフトウェアをインストールする

SiteWise Edge ゲートウェイを作成したら、ローカルデバイスに SiteWise Edge ゲートウェイソフトウェアをインストールする必要があります。 SiteWise Edge ゲートウェイソフトウェアは、Linux または Windows サーバーのオペレーティングシステムがインストールされているローカルデバイスにインストールできます。

> **Important**
>
> ローカルデバイスがインターネットに接続されていることを確認します。

Linux

次の手順では、SSH を使用してローカルデバイスに接続します。または、USB フラッシュドライブまたはその他のツールを使用して、インストーラファイルをローカルデバイスに転送することもできます。SSH を使用しない場合は、以下の「ステップ 2: SiteWise Edge ゲートウェイソフトウェアをインストールする」に進みます。

SSH の前提条件

SSH を使用してデバイスに接続する前に、次の前提条件を満たしていることを確認してください。

- デバイスの IP アドレスを取得します。
- デバイスに接続するためのユーザー名を取得します。
- 必要に応じてローカルコンピューターに SSH クライアントをインストールする

  ローカルコンピュータには、デフォルトで SSH クライアントがインストールされている場合があります。これは、コマンドラインで sshと入力することで確認できます。ご使用のコンピュータでこのコマンドが認識されない場合、SSH クライアントをインストールできます。

  - Linux および MacOS - OpenSSH をダウンロードしてインストールします。詳細については、https://www.openssh.com を参照してください。

ステップ 1: インストーラを SiteWise Edge ゲートウェイデバイスにコピーする

次の手順では、SSH クライアントを使用してローカルデバイスに接続する方法について説明します。

1. デバイスに接続するには、コンピュータのターミナルウィンドウで次のコマンドを実行し、*username* と *IP* を昇格されたプリンシパルと IP アドレスを持つユーザー名に置き換えます。

   ```
   ssh username@IP
   ```

2. が AWS IoT SiteWise 生成したインストーラーファイルを SiteWise Edge ゲートウェイデバイスに転送するには、次のコマンドを実行します。

   > **Note**
   >
   > - を、インストーラファイルの保存に使用したコンピュータ上のパスとインストーラファイルの名前*path-to-saved-installer*に置き換えます。
   > - *IP-address* をローカルデバイスの IP アドレスに置き換えます。
   > - を、インストーラファイルの受信に使用するローカルデバイスのパス*directory-to-receive-installer*に置き換えます。

```
scp path-to-saved-installer.sh user-name@IP-address:directory-to-receive-
installer
```

ステップ 2: SiteWise Edge ゲートウェイソフトウェアをインストールする

次の手順では、 SiteWise エッジゲートウェイデバイスのターミナルウィンドウでコマンドを実行します。

1. インストーラーファイルに実行アクセス許可を与える。

```
chmod +x path-to-installer.sh
```

2. インストーラーを実行します。

```
sudo ./path-to-installer.sh
```

Windows server

前提条件

SiteWise Edge ゲートウェイソフトウェアをインストールするには、次の前提条件が必要です。

- Windows Server 2019 以降がインストールされている
- 管理者権限
- PowerShell バージョン 5.1 以降がインストールされている
- SiteWise プロビジョニング先の Windows Server にダウンロードされたエッジゲートウェイインストーラ

ステップ 1: 管理者 PowerShell として を実行する

1. SiteWise Edge ゲートウェイをインストールする Windows サーバーで、管理者としてログインします。
2. Windows の検索バーPowerShellに と入力します。
3. 検索結果で、Windows PowerShell アプリのコンテキスト (右クリック) メニューを開きます。[管理者として実行] を選択します。

ステップ 2: SiteWise Edge ゲートウェイソフトウェアをインストールする

SiteWise Edge Gateway デバイスのターミナルウィンドウで次のコマンドを実行します。

1. SiteWise Edge ゲートウェイインストーラのブロックを解除します。

```
unblock-file path-to-installer.ps1
```

2. インストーラーを実行します。

```
./path-to-installer.ps1
```

> **Note**
>
> システム上でスクリプト実行が無効になっている場合は、スクリプト実行ポリシーを RemoteSignedに変更します。
>
> ```
> Set-ExecutionPolicy RemoteSigned
> ```

エッジデータ処理を有効にする。

AWS IoT SiteWise Edge を使用して、機器データをローカルで収集、保存、整理、モニタリングできます。 SiteWise Edge を使用すると、産業データをモデル化し、 SiteWise Monitor を使用して、運用スタッフがデータをローカルで視覚化するためのダッシュボードを作成できます。データをローカルで処理して に送信するか AWS クラウド、 AWS IoT SiteWise API を使用してオンプレミスで処理できます。

AWS IoT SiteWise Edge を使用すると、raw データをローカルで処理し、集約されたデータのみをに送信して AWS クラウド 、帯域幅の使用量とクラウドストレージコストを最適化できます。

> **Note**
>
> - AWS IoT SiteWise は、エッジデータを SiteWise エッジゲートウェイに最大 30 日間保持します。データの保持期間は、デバイスの使用可能なディスク容量によって異なります。
> - SiteWise Edge ゲートウェイが から 30 AWS クラウド 日間切断されている場合、Data Processing Pack は自動的に無効になります。

エッジ機能の設定。

AWS IoT SiteWise には、 SiteWise エッジゲートウェイがデータの収集と処理方法を決定するために使用できる以下のパックが用意されています。Edge SiteWise ゲートウェイのエッジ機能を有効にするには、パックを選択します。

- データ収集パックを使用すると、 SiteWise エッジゲートウェイは複数の OPC-UA サーバーからデータを収集し、エッジから にデータをエクスポートできます AWS クラウド。 SiteWise Edge ゲートウェイにデータソースを追加すると、アクティブになります。
- Data Processing Pack を使用すると、 SiteWise エッジゲートウェイはエッジで機器データを処理できます。例えば、アセットモデルを使用して、メトリクスや変換をコンピューティングすることができます。アセットモデルとアセットについては、産業用アセットのモデリング をご参照ください。

> **Note**
>
> - Data Processing Pack は x86 プラットフォームでのみ使用できます。
> - Data Processing Pack はネットワークプロキシをサポートしていません。

エッジ機能を設定するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、エッジゲートウェイ を選択します。
3. SiteWise エッジ機能をアクティブ化するエッジゲートウェイを選択します。
4. Edge 機能セクションで、編集 を選択します。
5. Edge 機能セクションで、データ処理パックを有効にする (追加料金が発生します) を選択します。
6. (オプション) Edge LDAP 接続セクションでは、企業ディレクトリ内のユーザーグループにこの SiteWise Edge ゲートウェイへのアクセスを許可できます。ユーザーグループは、Lightweight Directory Access Protocol (LDAP) 認証情報を使用して SiteWise Edge ゲートウェイにアクセスできます。その後、 AWS OpsHub アプリケーション、 AWS IoT SiteWise API AWS IoT SiteWise オペレーション、またはその他のツールを使用して SiteWise Edge ゲートウェイを管理できます。詳細については、「 SiteWise Edge ゲートウェイの管理」を参照してください。

> **Note**
>
> Linux または Windows の認証情報を使用して SiteWise Edge ゲートウェイにアクセスすることもできます。詳細については、「Linux オペレーティングシステムの認証情報を使用した SiteWise Edge ゲートウェイへのアクセス」を参照してください。

a. アクティブ化された を選択します。

b. [Provider name] (プロバイダー名) には、LDAP プロバイダーの名前を入力します。

c. ホスト名または IP アドレス に、LDAP サーバーのホスト名または IP アドレスを入力します。

d. [Port] (ポート) には、ポート番号を入力します。

e. [Base distinguished name (DN)] (ベース識エイリアス (DN)) には、ベースの識エイリアス (DN) を入力します。

   次の属性タイプがサポートされています: commonName (CN)、localityName (L)、stateOrProvinceName (ST)、organizationName (O)、 organizationalUnitName (OU)、countryName (C)、streetAddress (STREET)、domainComponent (DC)、userid (UID)。

f. [Admin group DN] (Admin グループの DN) には、DN を入力します。

g. [User group DN] (ユーザーグループ DN) には、DN を入力します。

7. [保存] を選択します。

Edge ゲートウェイでエッジ機能をアクティブ化したので、 SiteWise エッジ用にアセットモデルを設定する必要があります。アセットモデルのエッジ設定は、アセットプロパティがコンピューティングされる場所を指定します。すべてのプロパティをエッジでコンピューティングすることもできますし、アセットモデルのプロパティを個別に設定することも可能です。アセットモデルのプロパティには、メトリクス 、変換 、および測定値 が含まれます。

アセットプロパティの詳細については、the section called “データのプロパティを定義する。” を参照してください。

アセットモデルを作成したら、次はエッジに合わせた設定を行います。エッジ用のアセットモデルの構成については、the section called “アセットモデルを作成する (コンソール) ” を参照してください。

> **Note**
>
> アセットモデルとダッシュボードは、10 分ごとに AWS クラウド と SiteWise Edge ゲートウェイ間で自動的に同期されます。ローカル SiteWise Edge ゲートウェイアプリケーションから手動で同期することもできます。

エッジでのデータ処理

エッジで SiteWise エッジゲートウェイデータを処理できるようにするには、エッジ用にアセットモデルを設定する必要があります。アセットモデルのエッジ設定は、アセットプロパティがコンピューティングされる場所を指定します。エッジのすべてのプロパティを計算して結果を に送信するか AWS クラウド、各アセットプロパティを個別に計算する場所をカスタマイズするかを選択できます。詳細については、「エッジデータ処理を有効にする。」を参照してください。

アセットプロパティには、メトリクス、変換、測定値が含まれます。

- メトリクスはアセットの一定期間のデータを集計したものです。既存のメトリクスデータを使用して新しいメトリクスを計算できます。 AWS IoT SiteWise デフォルトでは、 はメトリクスを AWS クラウドに送信し、 AWS クラウド上の長期ストレージ AWS IoT SiteWise コンピューティングメトリクスを取得します。エッジでメトリクスを計算するようにアセットモデルを設定できます。AWS IoT SiteWise は処理された結果を AWS クラウドに送信します。
- 変換は、アセットのプロパティのデータポイントをあるフォームから別のフォームにマッピングする数式です。変換は入力データとしてメトリクスを使用することができ、その入力と同じ場所でコンピューティングされ、保存されなければなりません。エッジで計算するようにメトリクス入力を設定すると、 はエッジで関連する変換 AWS IoT SiteWise も計算します。
- 測定値は、デバイスが収集し、デフォルトで AWS クラウドに送信する raw データとしてフォーマットされます。このデータをローカルデバイスに保存するように、アセットモデルを設定することができます。

アセットプロパティの詳細については、the section called “データのプロパティを定義する。” を参照してください。

アセットモデルを作成したら、次はエッジに合わせた設定を行います。エッジ用のアセットモデルの構成については、the section called “アセットモデルを作成する (コンソール) ” を参照してください。

> **Note**
>
> アセットモデルとダッシュボードは、 AWS クラウドと SiteWise Edge ゲートウェイ間で 10 分ごとに自動的に同期されます。から手動で同期することもできます SiteWise Edge ゲートウェイの管理。

AWS IoT SiteWise REST APIs と AWS Command Line Interface （ AWS CLI) を使用して、 SiteWise エッジでエッジゲートウェイにデータをクエリできます。 SiteWise Edge ゲートウェイにエッジのデータについてクエリを実行する前に、次の前提条件を満たす必要があります。

- REST APIに認証情報を設定する必要があります。認証情報の設定については、the section called “ SiteWise Edge ゲートウェイの管理” を参照してください。
- SDK エンドポイントは、 SiteWise エッジゲートウェイの IP アドレスを指している必要があります。詳細は、お使いのSDK のドキュメントでご確認ください。例えば、[AWS SDK for Java 2.x Developer Guide] (デベロッパーガイド) の [Specifying Custom Endpoints] (カスタムエンドポイントの指定) を参照してください。
- SiteWise Edge ゲートウェイ証明書を登録する必要があります。 SiteWise Edge ゲートウェイ証明書の登録の詳細については、 SDK のドキュメントを参照してください。例えば、[AWS SDK for Java 2.x Developer Guide] (デベロッパーガイド) の [Registering Certificate Bundles in Node.js] (Node.jsで証明書バンドルを登録する) を参照してください。

を使用したデータのクエリの詳細については AWS IoT SiteWise、「」を参照してくださいからのデータのクエリ AWS IoT SiteWise。

AWS IoT SiteWise パブリッシャーコンポーネントの設定

AWS IoT SiteWise Edge ゲートウェイを作成してソフトウェアをインストールしたら、 SiteWise Edge ゲートウェイがデータを AWS クラウドにエクスポートできるように Publisher コンポーネントを設定します。詳細については、「 AWS IoT Greengrass Version 2 デベロッパーガイドAWS IoT SiteWise 」の「パブリッシャー」を参照してください。

Console

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、エッジゲートウェイ を選択します。

3. パブリッシャーを設定する SiteWise Edge ゲートウェイを選択します。
4. パブリッシャー設定セクションで、編集を選択します。
5. 発行順序 で、次のいずれかを選択します。

   - 最も古いデータを最初に公開する – SiteWise Edge ゲートウェイは、デフォルトで最も古いデータを最初にクラウドに公開します。
   - 最新のデータを最初に公開する – SiteWise Edge ゲートウェイは、最新のデータを最初にクラウドに公開します。

6. （オプション) SiteWise Edge ゲートウェイでデータを圧縮したくない場合は、データのアップロード時に圧縮を有効にするの選択を解除します。
7. （オプション) 古いデータを公開しない場合は、有効期限が切れたデータを除外を選択し、次の操作を行います。

   - カットオフ期間 には、値を入力し、単位を選択します。カットオフ期間は 5 分から 7 日の間でなければなりません。例えば、カットオフ期間が 3 日の場合、3 日以上経過したデータはクラウドに公開されません。

8. （オプション) ローカルデバイスでのデータの処理方法に関するカスタム設定を設定するには、ローカルストレージ設定を選択し、以下を実行します。

   a. [保持期間] には、数値を入力して単位を選択します。保持期間は 1 分から 30 日の間で、ローテーション期間と同じかそれ以上でなければなりません。例えば、保持期間が 14 日の場合、 SiteWise エッジゲートウェイは 14 日間保存された後、指定されたカットオフ期間より古いエッジにあるデータをすべて削除します。

   b. [ローテーション期間] には、数値を入力して単位を選択します。ローテーション期間は 1 分以上、保持期間以下である必要があります。例えば、ローテーション期間が 2 日間で、 SiteWise エッジゲートウェイがカットオフ期間より古いデータをバッチアップして 1 つのファイルに保存します。 SiteWise Edge ゲートウェイは、データのバッチを 2 日に 1 回、次のローカルディレクトリに転送します: `/greengrass/v2/work/aws.iot.SiteWiseEdgePublisher/exports`。

   c. ストレージ容量 には、1 以上の値を入力します。ストレージ容量が 2 GB の場合、2 GB を超えるデータがローカルに保存されると、 SiteWise エッジゲートウェイはデータの削除を開始します。

9. [保存] を選択します。

AWS CLI

UpdateGatewayCapabilityConfiguration API を使用してパブリッシャーを設定できます。`capabilityNamespace` パラメータを `iotsitewise:publisher:2` に設定します。

パブリッシャーには、カスタマイズできる以下の設定パラメータが用意されています。

`SiteWisePublisherConfiguration`

`publishingOrder`

データがクラウドに公開される順序。このパラメータの値は、次のいずれかになります。

- `TIME_ORDER` (古い順にデータを公開) – デフォルトでは、データは古い順にクラウドに公開されます。
- `RECENT_DATA` (新しい順にデータを最初) – データは新しい順にクラウドに公開されます。

`dropPolicy`

(オプション) クラウドに公開するデータを制御するポリシー。

`cutoffAge`

カットオフ期間より前のデータはクラウドに公開されません。カットオフ期間は 5 分から 7 日の間でなければなりません。

dカットオフ期間を指定すると `m`と `h`、を使用できます。`m` は分数、hは時数、dは日数を表します。

`exportPolicy`

(オプション) エッジのデータストレージを管理するポリシー。このポリシーは、カットオフ期間より前のデータに適用されます。

`retentionPeriod`

SiteWise Edge ゲートウェイは、指定された保持期間に保存されると、カットオフ期間より前のエッジのデータをローカルストレージから削除します。保持期間は 1 分から 30 日の間で、ローテーション期間と同じかそれ以上でなければなりません。

d保持期間を指定すると、`m`と `h`、を使用できます。`m` は分数、hは時数、dは日数を表します。

`rotationPeriod`

カットオフ期間より前のデータをバッチ処理して 1 つのファイルとして保存する時間間隔。 SiteWise Edge ゲートウェイは、各ローテーション期間の終了時に 1 つのバッチのデータを次のローカルディレクトリに転送します: `/greengrass/v2/work/aws.iot.SiteWiseEdgePublisher/exports`。ローテーション期間は 1 分より大きく、保持期間と同じかそれ以上でなければなりません。

ローテーション期間を指定すると、mと h、dを使用できます。m は分数、hは時数、dは日数を表します。

`exportSizeLimitGB`

ローカルに保存されるデータの最大許容サイズ (GB 単位)。このクォータを超えると、 SiteWise エッジゲートウェイは、ローカルに保存されているデータのサイズがクォータ以下になるまで、最も古いデータの削除を開始します。このパラメータの値は、1 以上である必要があります。

Example パブリッシャー設定：

パブリッシャーの名前空間: `iotsitewise:publisher:2`

```
{
"SiteWisePublisherConfiguration": {
    "publishingOrder": "TIME_ORDER",
    "dropPolicy": {
        "cutoffAge": "7d",
        "exportPolicy": {
            "retentionPeriod": "7d",
            "rotationPeriod": "6h",
            "exportLocation": "/greengrass/v2/work/aws.iot.SiteWiseEdgePublisher/
exports",
            "exportSizeLimitGB": 10
        }
    }
}
}
```

データソースの設定

AWS IoT SiteWise Edge ゲートウェイを設定したら、 SiteWise Edge ゲートウェイがローカル産業機器から にデータを取り込むことができるようにデータソースを設定できます AWS IoT SiteWise。各ソースは、 SiteWise エッジゲートウェイが接続して産業用データストリームを取得する OPC-UA サーバーなどのローカルサーバーを表します。 SiteWise Edge ゲートウェイの設定の詳細については、「」を参照してくださいAWS IoT Greengrass V1 SiteWise Edge ゲートウェイの設定。

> **Note**
>
> AWS IoT SiteWise は、ソースを追加または編集するたびに SiteWise Edge ゲートウェイを再起動します。 SiteWise Edge ゲートウェイは、再起動中にデータを取り込みません。SiteWise Edge ゲートウェイを再起動する時間は、 SiteWise エッジゲートウェイのソースのタグの数によって異なります。再起動時間の範囲は、数秒 (タグ数が少ない SiteWise エッジゲートウェイの場合) から数分 (タグ数が多い SiteWise エッジゲートウェイの場合) です。

ソースを作成したら、データストリームをアセットプロパティに関連付けることができます。アセットの作成および使用方法の詳細については、産業用アセットのモデリング と アセットプロパティへの産業データストリームのマッピング を参照してください。

CloudWatch メトリクスを表示して、データソースが に接続されていることを確認できます AWS IoT SiteWise。詳細については、「AWS IoT Greengrass Version 2 ゲートウェイメトリクス」を参照してください。

現在、 は、次のデータソースプロトコル AWS IoT SiteWise をサポートしています。

- OPC-UA – 産業オートメーション用の machine-to-machine (M2M) 通信プロトコル。

> **Note**
>
> SiteWise AWS IoT Greengrass V2 現在、 で実行されているエッジゲートウェイは、Modbus TCP およびイーサネット IP ソースをサポートしていません。

トピック

- OPC-UA ソースを設定する。
- データソース認証の設定

- ソースサーバーのデータの保存先を選択する。

OPC-UA ソースを設定する。

AWS IoT SiteWise コンソールまたは SiteWise Edge ゲートウェイ機能を使用して、ローカル OPC-UA サーバーを表す OPC-UA ソースを定義して SiteWise Edge ゲートウェイに追加できます。

トピック

- OPC-UA ソースの設定する (コンソール)。
- OPC-UA ソースの設定 (CLI)。
- OPC-UA ソースサーバーが SiteWise Edge ゲートウェイを信頼できるようにする
- OPC-UA でデータ取り込み範囲をフィルタリングする。
- OPC-UA ノードフィルターの使用

OPC-UA ソースの設定する (コンソール)。

AWS IoT SiteWise コンソールを使用して OPC-UA ソースを設定するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Gateways] を選択します。
3. Edge SiteWise ゲートウェイを選択して OPC-UA ソースを追加します。
4. [Add data source] (データソースを追加する) を選択する。
5. ソースの名前を入力します。
6. データソースサーバーの [Local endpoint (ローカルエンドポイント)] を入力します。エンドポイントには、IP アドレスまたはホスト名を指定することができます。また、ローカルエンドポイントにポート番号を追加することもできます。例えば、ローカルエンドポイントは次のようになります。 **`opc.tcp://203.0.113.0:49320`**

> **Note**
>
> SiteWise Edge ゲートウェイに `Deployment type` Industrial Edge デバイスの - 新しいがあり、Edge アプリケーションと同じ Industrial Edge Device で実行されている AWS IoT SiteWise Edge OPC UA Server アプリケーションからデータを取り込む場合は、「」と入力します**`opc.tcp://ie-opcua:48010`**。

7. (オプション) 選択用のノード ID に、ノードフィルターを追加して、 に取り込まれるデータストリームを制限します AWS クラウド。デフォルトでは、 SiteWise Edge ゲートウェイはサーバーのルートノードを使用してすべてのデータストリームを取り込みます。ノードフィルターを使用すると、 でモデル化するデータへのパスのみを含めることで、 SiteWise エッジゲートウェイの起動時間と CPU 使用率を削減できます AWS IoT SiteWise。デフォルトでは、 SiteWise エッジゲートウェイは、 で始まるパスを除くすべての OPC-UA パスをアップロードします/Server/。OPC-UA ノードフィルターを定義するには、ノードパス、* および ** のワイルドカード文字を使用できます。詳細については、「OPC-UA ノードフィルターの使用」を参照してください。
8. 送信先 で、ソースデータの送信先を選択します。

   - AWS IoT SiteWise リアルタイム – データを AWS IoT SiteWise ストレージに直接送信するには、これを選択します。データをリアルタイムで取り込み、モニタリングし、エッジでデータを処理します。
   - AWS IoT SiteWise Amazon S3 を使用してバッファリング — Parquet 形式でデータを Amazon S3 に送信し、 AWS IoT SiteWise ストレージにインポートします。データをバッチで取り込み、履歴データを費用対効果の高い方法で保存するには、このオプションを選択します。任意の Amazon S3 バケットの場所と、Amazon S3 にデータをアップロードする頻度を設定できます。への取り込み後にデータを処理する方法を選択することもできます AWS IoT SiteWise。データと Amazon S3 の両方 SiteWiseで使用できるようにするか、Amazon S3 からデータを自動的に削除するかを選択できます。 Amazon S3
     - Amazon S3 バケットはステージングおよびバッファリングメカニズムであり、parquet 形式のファイルをサポートしています。
     - 「ストレージ に AWS IoT SiteWise データをインポートする」チェックボックスをオンにすると、データは最初に Amazon S3 にアップロードされ、次にストレージにアップロードされます AWS IoT SiteWise 。
       - Amazon S3 からデータを削除するチェックボックスをオンにすると、データは SiteWise ストレージにインポートされた後に Amazon S3 から削除されます。
       - チェックボックスをオフにすると、Amazon S3 からデータを削除すると、データは Amazon S3 と SiteWise ストレージの両方に保存されます。
     - チェックボックスをオフにすると、データをストレージ に AWS IoT SiteWise インポートすると、データは Amazon S3 にのみ保存されます。 SiteWise ストレージにインポートされません。

AWS IoT SiteWise が提供するさまざまなストレージオプションの詳細については、ストレージの管理「」を参照してください。料金オプションの詳細については、AWS IoT SiteWise「 の料金」を参照してください。

- AWS IoT Greengrass ストリームマネージャー – AWS IoT Greengrass ストリームマネージャーを使用して、 の AWS クラウド チャネル AWS IoT Analytics、Amazon Kinesis Data Streams のストリーム、 のアセットプロパティ AWS IoT SiteWise、または Amazon Simple Storage Service (Amazon S3) のオブジェクトにデータを送信します。詳細については、「AWS IoT Greengrass Version 2 デベロッパーガイド」の AWS IoT Greengrass 「Core のデータストリームの管理」を参照してください。

  AWS IoT Greengrass ストリームの名前を入力します。

データソースを設定するときに、選択用のノード ID を使用してデータフローの送信先を決定します。

- Amazon S3 を使用して同じデータがAWS IoT SiteWise リアルタイムとバッファリングの両方に発行される場合は、両方の送信先に発行する 2 つのデータソースを追加する必要があります。 AWS IoT SiteWise Amazon S3
- データを分割して、その一部がAWS IoT SiteWise リアルタイム に発行されるようにし、もう 1 つの部分が AWS IoT SiteWise Amazon S3 を使用してバッファリングされるようにするには、次のデータエイリアスをフィルタリングする必要があります。

```
/Alias01/Data1
/Alias02/Data1
/Alias03/Data1
/Alias03/Data2
```

例えば、`/**/Data1`ノードフィルター、AWS IoT SiteWise リアルタイム を指すデータソース、および Amazon S3 を使用して`/**/Data2`バッファリングされた を指す別のデータソースを追加できます。 AWS IoT SiteWise Amazon S3

9. 詳細設定ペインでは、次の操作を実行できます。

   a. ソースサーバーと SiteWise Edge ゲートウェイ間で転送中の接続とデータのメッセージセキュリティモードを選択します。OPC-UA セキュリティポリシーとメッセージセキュリティモードの組み合わせのフィールドです。OPC-UA サーバーに指定したセキュリティポリシーとメッセージセキュリティモードと同じものを選択します。

b. ソースで認証が必要な場合は、認証設定リストから AWS Secrets Manager シークレットを選択します。 SiteWise Edge ゲートウェイは、このデータソースに接続するときに、このシークレットの認証情報を使用します。シークレットをデータソース認証に使用するには、SiteWise エッジゲートウェイの AWS IoT Greengrass コンポーネントにシークレットをアタッチする必要があります。詳細については、「the section called “データソース認証の設定”」を参照してください。

> **Tip**
>
> データサーバーに、[Allow anonymous login (匿名ログインを許可)] というオプションがある場合があります。このオプションが [はい] の場合、ソースは認証を必要としません。

c. (オプション) [Data stream prefix] (データストリームのプレフィックス) を入力します。SiteWise Edge ゲートウェイは、このソースからのすべてのデータストリームにこのプレフィックスを追加します。データストリームのプレフィックスを使用して、異なるソースから同じ名前を持つデータストリームを区別します。各データストリームは、アカウント内で一意の名前を持つ必要があります。

d. (オプション) プロパティグループ で、新しいグループの追加 を選択します。

   i. プロパティグループの「名前」を入力します。

   ii. [Properties] (プロパティ) の場合。

      1. ノードパス には、OPC-UA ノードフィルターを追加して、 にアップロードされる OPC-UA パスを制限します AWS IoT SiteWise。形式は、選択 のノード ID に似ています。

   iii. [Group settings] (グループ設定) については、次のようにします。

      1. データ品質設定 で、 AWS IoT SiteWise コレクターが取り込むデータ品質のタイプを選択します。

      2. スキャンモード設定 では、次の標準サブスクリプションプロパティを設定します。

         - スキャンモード では、次のいずれかを選択します。スキャンモードの詳細については、「」を参照してくださいthe section called “OPC-UA によるデータ取り込み範囲のフィルタリング。”。
            - すべてのデータポイントを送信するには、サブスクライブを選択し、以下を設定します。

- データ変更トリガー — データ変更アラートを開始する条件。
- サブスクリプションキューのサイズ — モニタリング対象項目の通知がキューに入れられる特定のメトリクスの OPC–UA サーバー上のキューの深さ。
- サブスクリプションの発行間隔 — サブスクリプションの作成時に指定された発行サイクルの間隔 (ミリ秒単位) 。
- スナップショット間隔 – AWS IoT SiteWise Edge が安定したデータストリームを取り込むようにするためのスナップショット頻度のタイムアウト設定。
- スキャンレート — SiteWise Edge ゲートウェイが registers. を読み取るレート。 は、 SiteWise Edge ゲートウェイの最小許容スキャンレート AWS IoT SiteWise を自動的に計算します。

- 特定の間隔でデータポイントを送信するには、ポーリングを選択してスキャンレートを入力します。

3. Subscibe のスキャンモードを選択した場合は、ソースの Deadband タイプと関連する設定を設定します。これにより、ソースが に送信するデータと AWS IoT SiteWise、破棄するデータが制御されます。デッドバンド設定の詳細は、the section called “OPC-UA によるデータ取り込み範囲のフィルタリング。” を参照してください。

10. [保存] を選択します。

OPC-UA ソースの設定 (CLI)。

Edge ゲートウェイの OPC-UA SiteWise データソースは、 を使用して定義できます AWS CLI。これを行うには、OPC-UA 機能設定 JSON ファイルを作成し、 update-gateway-capability-configuration コマンドを使用して SiteWise Edge ゲートウェイ設定を更新します。すべての OPC-UA ソースを 1 つの機能構成で定義する必要があります。

この機能には次の名前空間があります。

- `iotsitewise:opcuacollector:2`

リクエストの構文

```
{
  "sources": [
    {
      "name": "string",
```

```
      "endpoint": {
        "certificateTrust": {
          "type": "TrustAny" | "X509",
          "certificateBody": "string",
          "certificateChain": "string",
        },
        "endpointUri": "string",
        "securityPolicy": "NONE" | "BASIC128_RSA15" | "BASIC256" | "BASIC256_SHA256" |
 "AES128_SHA256_RSAOAEP" | "AES256_SHA256_RSAPSS",
        "messageSecurityMode": "NONE" | "SIGN" | "SIGN_AND_ENCRYPT",
        "identityProvider": {
          "type": "Anonymous" | "Username",
          "usernameSecretArn": "string"
        },
        "nodeFilterRules": [
          {
            "action": "INCLUDE",
            "definition": {
              "type": "OpcUaRootPath",
              "rootPath": "string"
            }
          }
        ]
      },
      "measurementDataStreamPrefix": "string"
     "destination": {
        "type": "StreamManager",
        "streamName": "string",
        "streamBufferSize": integer
      },
      "propertyGroups": [
        {
          "name": "string",
          "nodeFilterRuleDefinitions": [
            {
              "type": "OpcUaRootPath",
              "rootPath": "string"
            }
          ],
          "deadband": {
            "type": "PERCENT" | "ABSOLUTE",
            "value": double,
            "eguMin": double,
            "eguMax": double,
```

```
            "timeoutMilliseconds": integer
          },
          "scanMode": {
            "type": "EXCEPTION" | "POLL",
            "rate": integer
          },
          "dataQuality": {
            "allowGoodQuality": true | false,
            "allowBadQuality": true | false,
            "allowUncertainQuality": true | false
          },
          "subscription": {
            "dataChangeTrigger": "STATUS" | "STATUS_VALUE" | "STATUS_VALUE_TIMESTAMP",
            "queueSize": integer,
            "publishingIntervalMilliseconds": integer,
            "snapshotFrequencyMilliseconds": integer
          }
        }
      ]
    }
  ]
}
```

リクエスト本文

ソース

OPC-UA ソース定義構造のリスト。それぞれに次の情報が含まれています。

name

ソースの一意のわかりやすい名前。

エンドポイント

次の情報を含むエンドポイント構造。

certificateTrust

次の情報を含む証明書信頼ポリシー構造。

type

ソースに対する証明書信頼モード。以下のうちのひとつを選択します。

- **`TrustAny`** – SiteWise Edge ゲートウェイは、OPC-UA ソースに接続するときに証明書を信頼します。

- X509 – SiteWise Edge ゲートウェイは、OPC-UA ソースに接続するときに X.509 証明書を信頼します。このオプションを選択する場合は、certificateTrust で certificateBody を定義する必要があります。certificateTrust で certificateChain を定義することもできます。

certificateBody

(オプション) X.509 証明書の本文。

このフィールドは、certificateTrust で type に X509 を選択する場合に必須です。

certificateChain

(オプション) X.509 証明書の信頼チェーン。

このフィールドは、certificateTrust で type に X509 を選択した場合のみ使用されます。

endpointUri

OPC-UA ソースのローカルエンドポイント。たとえば、ローカルエンドポイントは opc.tcp://203.0.113.0:49320 のようになります。

securityPolicy

OPC-UA のソースから読み込むメッセージを保護するために使用するセキュリティポリシーです。以下のうちのひとつを選択します。

- NONE – SiteWise Edge ゲートウェイは、OPC-UA ソースからのメッセージを保護しません。別のセキュリティポリシーを選択することをお勧めします。このオプションを選択する場合は、messageSecurityMode に NONE を選択する必要もあります。
- BASIC256_SHA256 - Basic256Sha256のセキュリティポリシーです。
- AES128_SHA256_RSAOAEP - Aes128_Sha256_RsaOaepのセキュリティポリシーです。
- AES256_SHA256_RSAPSS - Aes256_Sha256_RsaPssのセキュリティポリシーです。
- BASIC128_RSA15 - (非推奨) OPC-UA 仕様において、Basic128Rsa15 セキュリティポリシーは安全でないと判断されたため、非推奨とする。別のセキュリティポリシーを選択することをお勧めします。詳しくは、[Basic128Rsa15]をご覧ください。
- BASIC256 - (非推奨) OPC-UA 仕様において、Basic256 セキュリティポリシーは安全でないと判断されたため、非推奨とする。別のセキュリティポリシーを選択することをお勧めします。詳しくは、[Basic256]をご覧ください。

> **Important**
>
> NONE 以外のセキュリティポリシーを選択した場合、messageSecurityMode は SIGN または SIGN_AND_ENCRYPT を選択する必要があります。Edge SiteWise ゲートウェイを信頼するようにソースサーバーを設定する必要があります。詳細については、「OPC-UA ソースサーバーが SiteWise Edge ゲートウェイを信頼できるようにする」を参照してください。

メッセージSecurityMode

OPC-UA ソースへの接続を保護するために使用するメッセージセキュリティモード。以下のうちのひとつを選択します。

- NONE – SiteWise Edge ゲートウェイは OPC-UA ソースへの接続を保護しません。他のメッセージセキュリティモードを選択することをお勧めします。このオプションを選択する場合は、securityPolicy に NONE を選択する必要もあります。
- SIGN – SiteWise Edge ゲートウェイと OPC-UA ソース間で転送中のデータは署名されていますが、暗号化されていません。
- SIGN_AND_ENCRYPT - ゲートウェイと OPC-UA ソース間で転送中のデータは署名され、暗号化されます。

> **Important**
>
> メッセージセキュリティモードを NONE 以外を選択した場合、securityPolicy は NONE以外を選択する必要があります。Edge SiteWise ゲートウェイを信頼するようにソースサーバーを設定する必要があります。詳細については、「OPC-UA ソースサーバーが SiteWise Edge ゲートウェイを信頼できるようにする」を参照してください。

identityProvider

次の情報を含む ID プロバイダ構造。

type

ソースで必要な認証資格情報のタイプ。以下のうちのひとつを選択します。

- Anonymous - ソースは接続に認証を必要としません。

- `Username` - ソースに接続するには、ユーザー名とパスワードが必要です。このオプションを選択する場合は、`identityProvider` で `usernameSecretArn` を定義する必要があります。

ユーザー名SecretArn

(オプション) AWS Secrets Manager シークレットの ARN。 SiteWise Edge ゲートウェイは、このソースに接続するときに、このシークレットの認証情報を使用します。シークレットをソース認証に使用するには、 SiteWise エッジゲートウェイの IoT SiteWise コネクタにシークレットをアタッチする必要があります。詳細については、「データソース認証の設定」を参照してください。

このフィールドは、`identityProvider` で `type` に `Username` を選択する場合に必須です。

ノードFilterRules

AWS クラウドに送信する OPC-UA データストリームパスを定義するノードフィルタールール構造のリスト。ノードフィルターを使用すると、 でモデル化するデータへのパスのみを含めることで、 SiteWise エッジゲートウェイの起動時間と CPU 使用率を減らすことができます AWS IoT SiteWise。デフォルトでは、 SiteWise エッジゲートウェイは で始まるパスを除くすべての OPC-UA パスをアップロードします`/Server/`。OPC-UA ノードフィルターを定義するには、ノードパス、`*` および `**` のワイルドカード文字を使用できます。詳細については、「OPC-UA ノードフィルターの使用」を参照してください。

リスト内の各構造には、次の情報が含まれている必要があります。

アクション

このノードフィルタルールのアクション。以下のオプションを選択できます。

- `INCLUDE` – SiteWise Edge ゲートウェイには、このルールに一致するデータストリームのみが含まれます。

定義

次の情報を含むノードフィルタルール構造。

type

このルールのノードフィルタパスのタイプ。以下のオプションを選択できます。

- `OpcUaRootPath` – SiteWise Edge ゲートウェイは、OPC-UA パス階層のルートに対してこのノードフィルターパスを評価します。

rootPath

OPC-UA パス階層のルートに対して評価するノードフィルタパス。このパスは / から始まる必要があります。

測定DataStreamプレフィックス

ソースからのすべてのデータストリームの前に付加する文字列。 SiteWise Edge ゲートウェイは、このソースからのすべてのデータストリームにこのプレフィックスを追加します。データストリームのプレフィックスを使用して、異なるソースから同じ名前を持つデータストリームを区別します。各データストリームは、アカウント内で一意の名前を持つ必要があります。

propertyGroups

(オプション) プロトコルが要求する deadband と scanMode を定義するプロパティグループのリストです。

name

プロパティグループの名前です。これは一意な識別子である必要があります。

デッドバンド

次の情報を含む deadband 構造体。

type

対応するデッドバンドの種類。有効な値は、ABSOLUTE および PERCENT です。

value

デッドバンドの値。type が ABSOLUTE のとき、この値は単位なしの 2 倍となります。type が PERCENT の場合、1 と 100 の間の 2 倍の値となります。

eguMin

(オプション) PERCENT　デッドバンドを使用する場合の工学単位の最小値。OPC-UA サーバに工学単位が設定されていない場合に設定します。

eguMax

(オプション) PERCENT デッドバンドを使用する場合の工学単位の最大値。OPC-UA サーバに工学単位が設定されていない場合に設定します。

timeoutMilliseconds

タイムアウトまでの時間をミリ秒単位で指定する。最小値は 100 です。

scanMode

次の情報を含む scanMode 構造体。

type

scanMode の対応型。有効な値は、POLL および EXCEPTION です。

レート

スキャンモードのサンプリング間隔。

ノードFilterRule定義

(オプション) プロパティグループに含めるノードパスのリスト。プロパティグループの重複があってはいけません。このフィールドに値を指定しなかった場合、グループにはルート下のすべてのパスが含まれ、追加のプロパティグループを作成することはできません。nodeFilterRuleDefinitions 構造体には、以下の情報が含まれています。

type

OpcUaRootPathは、サポートされている唯一のタイプです。これは、rootPathの値が OPC-UA ブラウジングスペースのルートからの相対パスであることの指定です。

rootPath

プロパティグループに含めるパス (ルートからの相対パス) を指定したカンマ区切りのリスト。

機能の設定例

次の例では、JSON ファイルに保存されているペイロードから OPC-UA SiteWise Edge ゲートウェイ機能設定を定義します。

```
aws iotsitewise update-gateway-capability-configuration \
--capability-namespace "iotsitewise:opcuacollector:2" \
--capability-configuration file://opc-ua-configuration.json
```

Example : OPC-UA ソースの設定

次の opc-ua-configuration.json ファイルは、、基本的で安全な OPC-UA ソース設定を定義しています。

```
{
    "sources": [
```

```
        {
            "name": "Wind Farm #1",
            "endpoint": {
                "certificateTrust": {
                    "type": "TrustAny"
                },
                "endpointUri": "opc.tcp://203.0.113.0:49320",
                "securityPolicy": "NONE",
                "messageSecurityMode": "NONE",
                "identityProvider": {
                    "type": "Anonymous"
                },
                "nodeFilterRules": []
            },
            "measurementDataStreamPrefix": ""
        }
    ]
}
```

Example : プロパティグループが定義されたOPC-UAソース構成。

次の opc-ua-configuration.json ファイルは、プロパティグループを定義した基本的な安全でない OPC-UA のソース構成を定義している。

```
{
    "sources": [
        {
            "name": "source1",
            "endpoint": {
                "certificateTrust": {
                    "type": "TrustAny"
                },
                "endpointUri": "opc.tcp://10.0.0.9:49320",
                "securityPolicy": "NONE",
                "messageSecurityMode": "NONE",
                "identityProvider": {
                    "type": "Anonymous"
                },
                "nodeFilterRules": [
                    {
                        "action": "INCLUDE",
                        "definition": {
                            "type": "OpcUaRootPath",
```

```
                        "rootPath": "/Utilities/Tank"
                    }
                }
            ]
        },
        "measurementDataStreamPrefix": "propertyGroups",
        "propertyGroups": [
            {
                "name": "Deadband_Abs_5",
                "nodeFilterRuleDefinitions": [
                    {
                        "type": "OpcUaRootPath",
                        "rootPath": "/Utilities/Tank/Temperature/TT-001"
                    },
                    {
                        "type": "OpcUaRootPath",
                        "rootPath": "/Utilities/Tank/Temperature/TT-002"
                    }
                ],
                "deadband": {
                    "type":"ABSOLUTE",
                    "value": 5.0,
                    "timeoutMilliseconds": 120000
                }
            },
            {
                "name": "Polling_10s",
                "nodeFilterRuleDefinitions": [
                    {
                        "type": "OpcUaRootPath",
                        "rootPath": "/Utilities/Tank/Pressure/PT-001"
                    }
                ],
                "scanMode": {
                    "type": "POLL",
                    "rate": 10000
                }
            },
            {
                "name": "Percent_Deadband_Timeout_90s",
                "nodeFilterRuleDefinitions": [
                    {
                        "type": "OpcUaRootPath",
                        "rootPath": "/Utilities/Tank/Flow/FT-*"
```

```
                        }
                    ],
                    "deadband": {
                        "type":"PERCENT",
                        "value": 5.0,
                        "eguMin": -100,
                        "eguMax": 100,
                        "timeoutMilliseconds": 90000
                    }
                }
            ]
        }
    ]
}
```

Example : OPC-UA プロパティ付きソース構成。

次の opc-ua-configuration.json の JSON 例は、次のプロパティを使用して OPC-UA ソース設定を定義します。

- すべての証明書を信頼します。
- メッセージの保護に BASIC256 セキュリティポリシーを使用します。
- SIGN_AND_ENCRYPT モードを使用して接続をセキュリティ保護します。
- Secrets Manager シークレットに保存されている認証資格情報を使用します。
- パスが /WindFarm/2/WindTurbine/ で始まるものを除き、データストリームを除外します。
- この「Wind Farm #2」と別のエリアの「Wind Farm #2」を区別するために、すべてのデータストリームパスの先頭に /Washington を追加します。

```
{
    "sources": [
        {
            "name": "Wind Farm #2",
            "endpoint": {
                "certificateTrust": {
                    "type": "TrustAny"
                },
                "endpointUri": "opc.tcp://203.0.113.1:49320",
                "securityPolicy": "BASIC256",
                "messageSecurityMode": "SIGN_AND_ENCRYPT",
                "identityProvider": {
```

```
                    "type": "Username",
                    "usernameSecretArn":
 "arn:aws:secretsmanager:region:123456789012:secret:greengrass-windfarm2-auth-1ABCDE"
                },
                "nodeFilterRules": [
                  {
                      "action": "INCLUDE",
                      "definition": {
                          "type": "OpcUaRootPath",
                          "rootPath": "/WindFarm/2/WindTurbine/"
                      }
                  }
                ]
            },
            "measurementDataStreamPrefix": "/Washington"
        }
    ]
}
```

Example : 証明書の信頼を持つ OPC-UA ソース設定

次の opc-ua-configuration.json の JSON 例は、次のプロパティを使用して OPC-UA ソース設定を定義します。

- 指定された X.509 証明書を信頼します。
- メッセージの保護に BASIC256 セキュリティポリシーを使用します。
- SIGN_AND_ENCRYPT モードを使用して接続をセキュリティ保護します。

```
{
    "sources": [
        {
            "name": "Wind Farm #3",
            "endpoint": {
                "certificateTrust": {
                    "type": "X509",
                    "certificateBody": "-----BEGIN CERTIFICATE-----
          MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w
 0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZ
 WF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIw
 EAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5
 jb20wHhcNMTEwNDI1MjA0NTIxWhcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
```

```
MCVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
WF6b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
HzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
ITxOUSQv7c7ugFFDzQGBzZswY6786m86gpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4nUhVVxYUntneD9+h8Mg9q6q+auN
KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FkbFFBjvSfpJIlJ00zbhNYS5f6Guo
EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTbNYiytVbZPQUQ5Yaxu2jXnimvw
3rrszlaEXAMPLE=
        -----END CERTIFICATE-----",
                "certificateChain": "-----BEGIN CERTIFICATE-----
        MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w
0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZ
WF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIw
EAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5
jb20wHhcNMTEwNDI1MjA0NTIxWhcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
MCVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
WF6b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
HzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
ITxOUSQv7c7ugFFDzQGBzZswY6786m86gpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4nUhVVxYUntneD9+h8Mg9q6q+auN
KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FkbFFBjvSfpJIlJ00zbhNYS5f6Guo
EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTbNYiytVbZPQUQ5Yaxu2jXnimvw
3rrszlaEXAMPLE=
        -----END CERTIFICATE-----"
              },
              "endpointUri": "opc.tcp://203.0.113.2:49320",
              "securityPolicy": "BASIC256",
              "messageSecurityMode": "SIGN_AND_ENCRYPT",
              "identityProvider": {
                  "type": "Anonymous"
              },
              "nodeFilterRules": []
            },
          "measurementDataStreamPrefix": ""

        }
    ]
}
```

OPC-UA ソースサーバーが SiteWise Edge ゲートウェイを信頼できるようにする

OPC-UA ソースを設定するときに None `messageSecurityMode`以外の を選択した場合は、ソースサーバーが AWS IoT SiteWise Edge ゲートウェイを信頼できるようにする必要があります。SiteWise Edge ゲートウェイは、ソースサーバーに必要な証明書を生成します。プロセスはソースサーバーによって異なります。詳細については、サーバーのドキュメントを参照してください。

次の手順では、基本的な手順の概要を説明します。

OPC-UA サーバーが SiteWise Edge ゲートウェイを信頼できるようにするには

1. OPC-UA サーバーを設定するためのインターフェイスを開きます。
2. OPC-UA サーバー管理者のユーザー名とパスワードを入力してください。
3. インターフェイスで [Trusted Clients (信頼済みクライアント)] を見つけて、[AWS IoT SiteWise Gateway Client (ゲートウェイクライアント)] を選択します。
4. [Trust (信頼)] を選択します。

OPC-UA クライアント証明書のエクスポート

一部の OPC-UA サーバーは、 SiteWise エッジゲートウェイを信頼するために OPC-UA クライアント証明書ファイルにアクセスする必要があります。これが OPC-UA サーバーに適用される場合は、次の手順を使用して SiteWise Edge ゲートウェイから OPC-UA クライアント証明書をエクスポートできます。その後、OPC-UA サーバーに証明書をインポートできます。

ソースの OPC-UA クライアント証明書ファイルをエクスポートするには

1. 次のコマンドを実行して、証明書ファイルを含むディレクトリに変更します。*sitewise-work* を *aws.iot.SiteWiseEdgeCollectorOpcua* Greengrass 作業フォルダのローカルストレージパスに置き換え、*source-name* をデータソースの名前に置き換えます。

   デフォルトでは、Greengrass 作業フォルダは Linux の場合は */greengrass/v2/work/aws.iot.SiteWiseEdgeCollectorOpcua*、Windows の場合は *C:/greengrass/v2/work/aws.iot.SiteWiseEdgeCollectorOpcua* です。

   ```
   cd /sitewise-work/source-name/opcua-certificate-store
   ```

2. このソースの SiteWise エッジゲートウェイの OPC-UA クライアント証明書は `aws-iot-opcua-client.pfx` ファイルにあります。

次のコマンドを実行して、証明書を aws-iot-opcua-client-certificate.pem という名前の .pem ファイルにエクスポートします。

```
keytool -exportcert -v -alias aws-iot-opcua-client -keystore aws-iot-opcua-client.pfx -storepass amazon -storetype PKCS12 -rfc > aws-iot-opcua-client-certificate.pem
```

3. 証明書ファイル を SiteWise Edge ゲートウェイaws-iot-opcua-client-certificate.pemから OPC-UA サーバーに転送します。

   これを行うには、scp プログラムなどの一般的なソフトウェアを使用して、SSH プロトコルを使用したファイルを転送することができます。詳しくは[Wikipedia] (ウィキペディア) の[Secure copy] (セキュアコピー) をご覧ください。

   > **Note**
   >
   > SiteWise Edge ゲートウェイが Amazon Elastic Compute Cloud (Amazon EC2) で実行されていて、初めて接続する場合は、接続するための前提条件を設定する必要があります。詳細については、「Amazon EC2 ユーザーガイド」の「Linux インスタンスに接続する」を参照してください。 Amazon EC2

4. Edge ゲートウェイを信頼するには、証明書ファイル を OPC-UA SiteWise サーバーaws-iot-opcua-client-certificate.pemにインポートします。使用するソースサーバーによりステップは異なります。サーバーのドキュメントを参照してください。

OPC-UA でデータ取り込み範囲をフィルタリングする。

OPC-UA ソースでのデータ取り込みは、スキャンモードとデッドバンド範囲を使用することで制御することができます。これらの機能により、取り込むデータの種類と、サーバーと SiteWise Edge ゲートウェイがこの情報を交換する方法とタイミングを制御できます。

品質に基づいてデータを収集またはフィルタリングする

OPC-UA ソースから収集するデータを制御するために、データ品質設定を設定できます。データソースは、送信時に品質評価をメタデータとして含めます。以下のオプションのいずれかまたはすべてを選択できます。

- Good

- `Bad`
- `Uncertain`

スキャンモードでデータ収集頻度を制御する。

OPC-UA のスキャンモードを設定することで、OPC-UA ソースからのデータ収集方法を制御することができます。サブスクリプションモードとポーリングモードを選択することができます。

- サブスクリプションモード – OPC-UA ソースは、スキャンレートで定義された頻度で SiteWise Edge ゲートウェイに送信するデータを収集します。サーバーは、値が変更されたときにのみデータを送信するため、これは SiteWise Edge ゲートウェイがデータを受信する最大頻度です。
- ポーリングモード – SiteWise Edge ゲートウェイは、スキャンレートで定義された設定された頻度で OPC-UA ソースをポーリングします。サーバーは、値が変更されたかどうかに関係なくデータを送信するため、 SiteWise エッジゲートウェイは常にこの間隔でデータを受信します。

> **Note**
>
> ポーリングモードオプションは、このソースのデッドバンド設定をオーバーライドします。

OPC-UA のデータ取り込みをデッドバンド幅でフィルタリングする。

OPC-UA ソースプロパティグループにデッドバンドを適用して、特定のデータを AWS クラウドに送信する代わりにフィルタリングして破棄できます。デッドバンドは、OPC-UA ソースから受信するデータ値の変動を予測するウィンドウを指定します。値がこのウィンドウ内にある場合、OPC-UA サーバーはそれを AWS クラウドに送信しません。デッドバンドフィルタリングを使用すると、処理して AWS クラウドに送信するデータの量を減らすことができます。 SiteWise Edge ゲートウェイの OPC-UA ソースを設定する方法については、「」を参照してくださいthe section called “データソースの設定”。

> **Note**
>
> サーバーは、デッドバンドで指定されたウィンドウ内にあるすべてのデータを削除します。この破棄されたデータは復元できません。

デッドバンドの種類

OPC-UA サーバのプロパティグループには、2 種類のデッドバンドを指定することができます。これにより、 AWS クラウドに送信されるデータの量と破棄されるデータの量を選択できます。

- パーセンテージ - 測定値の予想変動率のパーセンテージを使用してウィンドウを指定します。サーバーはこのパーセンテージから正確なウィンドウを計算し、 を超えるデータを AWS クラウドに送信します。例えば、華氏 -100 度から華氏 +100 度の範囲のセンサーで 2% のデッドバンド値を指定すると、値が華氏 4 度以上変化したときにデータを AWS クラウドに送信するようにサーバーに指示します。

  > **Note**
  >
  > ソースサーバーが工学的単位を定義していない場合、このウィンドウに最小および最大のデッドバンド値をオプションで指定できます。工学単位範囲が指定されていない場合、OPC-UA サーバは測定データ型のフルレンジをデフォルトとします。

- 絶対 - 正確な単位を使用してウィンドウを指定します。例えば、あるセンサーのデッドバンドを 2 と指定すると、その値が 2 単位以上変化したときに AWS クラウドにデータを送信するようにサーバーに指示する。通常の運用で定期的に変動が予想されるようなダイナミックな環境では、絶対デッドバンドを使用することができます。

デッドバンドタイムアウト

オプションで、デッドバンドのタイムアウト設定を行うことができます。このタイムアウト後は、現在の測定値が想定されるデッドバンド変動の範囲内であっても OPC-UA サーバは送信します。タイムアウト設定を使用すると、値が定義されたデッドバンドウィンドウを超えない場合でも、 AWS IoT SiteWise が常に安定したデータストリームを取り込むようにできます。

OPC-UA ノードフィルターの使用

SiteWise Edge ゲートウェイの OPC-UA データソースを定義するときに、ノードフィルターを定義できます。ノードフィルターを使用すると、 SiteWise エッジゲートウェイがクラウドに送信するデータストリームパスを制限できます。ノードフィルターを使用すると、 でモデル化するデータへのパスのみを含めることで、 SiteWise エッジゲートウェイの起動時間と CPU 使用率を削減できます AWS IoT SiteWise。デフォルトでは、 SiteWise エッジゲートウェイは で始まるパスを除くすべての OPC-UA パスをアップロードします`/Server/`。ノードフィルターで * および ** ワイルドカード文字を使用すると、1 つのフィルターに複数のデータストリームパスを含めることができま

す。 SiteWise Edge ゲートウェイの OPC-UA ソースを設定する方法については、「」を参照してください[データソースの設定]。

> **Note**
>
> AWS IoT SiteWise は、ソースを追加または編集するたびに SiteWise Edge ゲートウェイを再起動します。 SiteWise Edge ゲートウェイは、再起動中にデータを取り込みません。 SiteWise Edge ゲートウェイを再起動する時間は、 SiteWise エッジゲートウェイのソースのタグの数によって異なります。再起動時間の範囲は、数秒 (タグ数が少ない SiteWise エッジゲートウェイの場合) から数分 (タグ数が多い SiteWise エッジゲートウェイの場合) です。

次の表に、OPC-UA データソースのフィルターに使用できるワイルドカードを示します。

OPC-UA ノードフィルターワイルドカード

ワイルドカード	説明
*	データストリームパスの単一レベルと一致します。
**	データストリームパスの複数のレベルに一致します。

> **Note**
>
> 広範なフィルターを使用してソースを設定し、後でソースを変更してより制限の厳しいフィルターを使用する場合、 は新しいフィルターと一致しないデータの保存を AWS IoT SiteWise 停止します。

Example ノードフィルターを使用したシナリオの例

次の架空のデータストリームについて考えてみます。

- `/WA/Factory 1/Line 1/PLC1`
- `/WA/Factory 1/Line 1/PLC2`
- `/WA/Factory 1/Line 2/Counter1`

- `/WA/Factory 1/Line 2/PLC1`
- `/OR/Factory 1/Line 1/PLC1`
- `/OR/Factory 1/Line 2/Counter2`

以前のデータストリームを使用して、ノードフィルターを定義して、OPC-UA ソースから含めるデータを制限できます。

- この例のすべてのノードを選択するには、`/` または `/**/`を使用します。`**` ワイルドカード文字を使用して、複数のディレクトリまたはフォルダを含めることができます。
- すべての PLC データストリームを選択するには、`/*/*/*/PLC*` または `/**/PLC*` を使用します。
- この例のすべてのカウンタを選択するには、`/**/Counter*` または `/*/*/*/Counter*`を使用します。
- `Line 2` からすべてのカウンターを選択するには、`/**/Line 2/Counter*` を使用します。

データソース認証の設定

OPC-UA サーバーで接続に認証情報が必要な場合は、 AWS Secrets Manager を使用してシークレットを作成し、 SiteWise エッジゲートウェイにデプロイできます。 は、デバイス上のシークレットを AWS Secrets Manager 暗号化して、ユーザー名とパスワードを使用するまで安全に保つことができます。 AWS IoT Greengrass シークレットマネージャーコンポーネントの詳細については、「 AWS IoT Greengrass Version 2 デベロッパーガイド」の「シークレットマネージャー」を参照してください。

Secrets Manager シークレットへのアクセスの管理については、以下を参照してください。

- AWS Secrets Manager シークレット に対するアクセス許可を持つユーザー。
- アカウント 内でリクエストが許可または拒否されるかどうかを決定します。

ステップ 1: ソース認証シークレットを作成する

AWS Secrets Manager を使用して、データソースの認証シークレットを作成できます。シークレットで、データソースの認証詳細を含む **`username`** および **`password`** のキーと値のペアを定義します。

シークレットを作成する (コンソール)

1. AWS Secrets Manager コンソールに移動します。
2. [新しいシークレットを保存] を選択します。
3. [シークレットのタイプ] で、[その他のシークレットのタイプ] を選択します。
4. [Key/value pairs] (キー/値ペア) で、次のように実行します。
    1. 最初の入力ボックスに「」と入力**username**し、2 番目の入力ボックスにユーザー名を入力します。
    2. [Add row] (行の追加) を選択します。
    3. 最初の入力ボックスに「」と入力**password**し、2 番目の入力ボックスにパスワードを入力します。
5. 暗号化キー で、aws/secretsmanager を選択し、次へ を選択します。
6. [新しいシークレットの保存] ページで、シークレット名を入力します。
7. (オプション) このシークレットの識別に役立つ説明を入力し、[次へ] を選択します。
8. (オプション) [新規シークレットを保存] ページで [自動ローテーション] をオンにします。詳細については、『AWS Secrets Manager ユーザーガイド』の「シークレットのローテーション」を参照してください。
9. ローテーションスケジュールを指定します。
10. このシークレットをローテーションできる Lambda 関数を選択し、[次へ] を選択します。
11. シークレットの設定を確認し、[保存] を選択します。

SiteWise Edge ゲートウェイが とやり取りすることを許可するには AWS Secrets Manager、SiteWise Edge ゲートウェイの IAM ロールで `secretsmanager:GetSecretValue`アクションを許可する必要があります。Greengrass コアデバイスを使用して IAM ポリシーを検索できます。IAM ポリシーの更新の詳細については、「 AWS Identity and Access Management ユーザーガイド」の「IAM ポリシーの編集」を参照してください。

Example ポリシー

`secret-arn` の部分をステップ 2 で作成したシークレットの Amazon リソースネーム (ARN) に置き換えます。シークレットの ARN を取得する方法の詳細については、『AWS Secrets Manager ユーザーガイド』の「AWS Secrets Managerからのシークレットの取得」を参照してください。

```
{
```

```
"Version":"2012-10-17",
"Statement":[
  {
     "Action":[
        "secretsmanager:GetSecretValue"
     ],
     "Effect":"Allow",
     "Resource":[
        "secret-arn"
     ]
  }
]
}
```

ステップ 2: SiteWise Edge ゲートウェイデバイスにシークレットをデプロイする

AWS IoT SiteWise コンソールを使用して、 SiteWise Edge ゲートウェイにシークレットをデプロイできます。

シークレットを更新する (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Gateways] を選択します。
3. Gateways リストから、ターゲット Edge SiteWise ゲートウェイを選択します。
4. ゲートウェイ設定セクションで、Greengrass コアデバイスリンクを選択して、 SiteWise エッジゲートウェイに関連付けられた AWS IoT Greengrass コアを開きます。
5. ナビゲーションペインで、[デプロイ] を選択します。
6. ターゲットデプロイを選択し、[修正] を選択します。
7. [Specify target] (ターゲットの指定) ページで [Next] (次へ) を選択します。
8. [コンポーネントの選択] ページの [公開コンポーネント] セクションで、「選択したコンポーネントのみを表示」をオフにします。
9. aws.greengrass.SecretManager コンポーネントを検索して選択し、次へ を選択します。
10. 選択したコンポーネントリストから aws.greengrass.SecretManager コンポーネントを選択し、コンポーネント の設定 を選択します。
11. [マージする設定] フィールドで、次の JSON オブジェクトを追加します。

> **Note**
>
> *secret-arn* の部分を、前のステップで作成したシークレットの ARN に置き換えます。シークレットの ARN を取得する方法の詳細については、『AWS Secrets Manager ユーザーガイド』の「AWS Secrets Managerからのシークレットの取得」を参照してください。

```
{
"cloudSecrets":[
  {
     "arn":"secret-arn"
  }
]
}
```

12. [確認] を選択します。
13. [次へ] をクリックします。
14. [詳細設定の設定] ページで、[次へ] を選択します。
15. デプロイの設定を確認し、[デプロイ] を選択します。

ステップ 3: 認証設定を追加する

AWS IoT SiteWise コンソールを使用して、認証設定を SiteWise Edge ゲートウェイに追加できます。

認証設定を追加する (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. Gateways リストから、ターゲット Edge SiteWise ゲートウェイを選択します。
3. データソースのリストから、ターゲットデータソースを選択し、[編集] を選択します。
4. [データソースの追加] ページで [詳細設定] を選択します。
5. [認証設定] で、前のステップでデプロイしたシークレットを選択します。
6. [保存] を選択します。

ソースサーバーのデータの保存先を選択する。

データはエッジから AWS IoT SiteWise にリアルタイムでエクスポートされるか、Amazon S3 を使用してバッチでエクスポートされます。ストリームを使用して別のコンポーネントに AWS IoT Greengrass ストリームを送信することもできます。

- AWS IoT SiteWise リアルタイム – データを AWS IoT SiteWise ストレージに直接送信するには、これを選択します。データをリアルタイムで取り込み、モニタリングし、エッジでデータを処理します。
- AWS IoT SiteWise Amazon S3 を使用してバッファリング — Parquet 形式でデータを Amazon S3 に送信し、 AWS IoT SiteWise ストレージにインポートします。データをバッチで取り込み、履歴データを費用対効果の高い方法で保存するには、このオプションを選択します。任意の Amazon S3 バケットの場所と、Amazon S3 にデータをアップロードする頻度を設定できます。への取り込み後にデータを処理する方法を選択することもできます AWS IoT SiteWise。データを SiteWise と Amazon S3 の両方で利用できるようにすることも、Amazon S3 から自動的に削除することもできます。
  - Amazon S3 バケットはステージングおよびバッファリングメカニズムであり、parquet 形式のファイルをサポートしています。
  - 「ストレージ に AWS IoT SiteWise データをインポートする」チェックボックスをオンにすると、データは最初に Amazon S3 にアップロードされ、次にストレージにアップロードされます AWS IoT SiteWise 。
    - チェックボックスをオンにすると、Amazon S3 からデータを削除すると、データはストレージに SiteWiseインポートされた後に Amazon S3 から削除されます。
    - チェックボックスをオフにすると、Amazon S3 からデータを削除すると、データは Amazon S3 と SiteWise ストレージの両方に保存されます。
  - チェックボックスをオフにすると、データをストレージ に AWS IoT SiteWise インポートすると、データは Amazon S3 にのみ保存されます。ストレージには SiteWiseインポートされません。

  AWS IoT SiteWise が提供するさまざまなストレージオプションの詳細については、ストレージの管理「」を参照してください。料金オプションの詳細については、AWS IoT SiteWise 「 の料金」を参照してください。
- AWS IoT Greengrass ストリームマネージャー – AWS IoT Greengrass ストリームマネージャーを使用して、 の AWS クラウド チャネル AWS IoT Analytics、Amazon Kinesis Data Streams のストリーム、 のアセットプロパティ AWS IoT SiteWise、または Amazon Simple Storage Service (Amazon S3) のオブジェクトにデータを送信します。詳細については、「 AWS IoT Greengrass

Version 2 デベロッパーガイド」の AWS IoT Greengrass 「 Core のデータストリームの管理」を参照してください。

次の例は、必要なデータストリームのメッセージ構造を示している。すべての項目が必須です。

```
{
    "assetId": "string",
    "propertyAlias": "string",
    "propertyId": "string",
    "propertyValues": [
      {
         "quality": "string",
         "timestamp": {
            "offsetInNanos": number,
            "timeInSeconds": number
         },
         "value": {
            "booleanValue": boolean,
            "doubleValue": number,
            "integerValue": number,
            "stringValue": "string"
         }
      }
    ]
}
```

> **Note**
>
> データストリームメッセージは、その構造propertyAliasに (assetId と propertyId) または を含める必要があります。

`assetId`

(オプション) 更新するアセットの ID。

`propertyAlias`

(オプション) OPC-UA サーバーのデータストリームパスなど、 プロパティを識別するエイリアス。例:

```
/company/windfarm/3/turbine/7/temperature
```

詳細については、「 AWS IoT SiteWise ユーザーガイド」の「産業用データストリームをアセットプロパティにマッピングする」を参照してください。

`propertyId`

(オプション) このエントリのアセットプロパティの ID。

`propertyValues`

(必須) アップロードするプロパティ値のリスト。最大 10 個の`propertyValues`配列要素を指定できます。

`quality`

(オプション) アセットプロパティ値の品質。

`timestamp`

(必須) アセットプロパティ値のタイムスタンプ。

`offsetInNanos`

(オプション) からのナノ秒オフセット`timeInSeconds`。

`timeInSeconds`

(必須) Unix エポック形式の秒単位のタイムスタンプ日付。分数ナノ秒のデータは、`offsetInNanos` で提供されます。

`value`

(必須) アセットプロパティの値。

> **Note**
>
> `value` フィールドには、次のいずれかの値しか存在できません。

`booleanValue`

(オプション) ブール型のアセットプロパティデータ (`true` または `false`) 。

`doubleValue`

(オプション) double (浮動小数点数) 型のアセットプロパティデータ。

`integerValue`

(オプション) 整数型 (整数) のアセットプロパティデータ。

`stringValue`

(オプション) 文字列型 (文字のシーケンス) のアセットプロパティデータ。

SiteWise Edge ゲートウェイへのパートナーデータソースの追加

AWS IoT SiteWise Edge ゲートウェイを使用する場合、パートナーデータソースを SiteWise Edge ゲートウェイに接続し、 SiteWise エッジゲートウェイとAWSクラウドのパートナーからデータを受信できます。これらのパートナーデータソースは、AWS IoT Greengrass と AWS とパートナーがパートナーシップで開発されるコンポーネントです。パートナーデータソースを追加すると、 AWS IoT SiteWiseはこのコンポーネントを作成し、 SiteWise それを Edge ゲートウェイにデプロイします。

パートナーデータソースを追加するには、次の手順を実行します。

- パートナーデータソースの追加
- パートナーのウェブポータルに移動し、 SiteWise Edge ゲートウェイに接続するようにパートナーデータソースを設定します。

トピック

- セキュリティ
- パートナーデータソースの追加
- SiteWise Edge ゲートウェイで Docker をセットアップする
- SiteWise エッジゲートウェイパートナーデータソース

セキュリティ

以下では、AWS と顧客、当社パートナー間の責任分担モデルの一環として、セキュリティの責任分担について説明します。

顧客の責任

- パートナーの審査。
- パートナーに与えるネットワークアクセス権の設定。

AWS の責任

- パートナーが必要とするものを除き、顧客の AWS クラウドリソースからのパートナーの隔離。この場合は、AWS IoT SiteWise インジェスト。
- パートナーソリューションを SiteWise Edge ゲートウェイマシンリソース (CPU、メモリ、ファイルシステム) の妥当な使用に制限します。

パートナーの責任

- 安全なデフォルトの使用。
- パッチやその他の適切な更新による長期にわたるソリューションの安全維持。
- 顧客データの機密保持。

パートナーデータソースの追加

パートナーデータソースを SiteWise Edge ゲートウェイに接続するには、データソースとして追加します。データソースとして追加すると、 AWS IoT SiteWiseはプライベートAWS IoT Greengrassコンポーネントを SiteWise Edge ゲートウェイにデプロイします。

前提条件

パートナーデータソースを追加するには、次の手順を実行します。

- パートナーのアカウントを作成します。
- アカウントをバインドします。

パートナーデータソースを使用して SiteWise Edge ゲートウェイを作成するには

新しい SiteWise Edge ゲートウェイを作成する場合は、「」の手順を完了します SiteWise Edge ゲートウェイの作成。 SiteWise Edge ゲートウェイを作成したら、「」の手順に従ってパートナーデータソースパートナーデータソースを既存の SiteWise Edge ゲートウェイに追加するにはを追加します。

パートナーデータソースを既存の SiteWise Edge ゲートウェイに追加するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Gateways] を選択します。
3. パートナーデータソースを接続する SiteWise エッジゲートウェイを選択します。

4. [データソース] で、[データソースを追加] を選択します。
5. ソースタイプ で、 SiteWise エッジゲートウェイの接続先のパートナーを選択します。

> **Note**
>
> 現在、 は唯一の利用可能なパートナーデータソース EasyEdge です。 EasyEdge データソースを初めて追加するときは、EasyEdge アカウント を作成する必要があります。

6. ソースの名前を入力します。
7. パートナーにデータソースへのアクセス権を付与するには、[承認] を選択します。
8. AWS IoT SiteWise が AWS IoT SiteWise パブリッシャーコンポーネントと AWS IoT SiteWise プロセッサコンポーネント (データ処理パックが有効になっている場合) を更新できるようにするには、[コンポーネントの更新] を選択します。
9. [保存] を選択します。

SiteWise Edge ゲートウェイで Docker をセットアップする

パートナーデータソースを追加するには、ローカルデバイスに Docker Engine 1.9.1 以降がインストールされている必要があります。

> **Note**
>
> バージョン 20.10 は、 SiteWise エッジゲートウェイソフトウェアで動作することが検証されている最新バージョンです。

Docker がインストールされていることを確認する

Docker がインストールされていることを確認するには、 SiteWise エッジゲートウェイに接続されているターミナルから次のコマンドを実行します。

```
docker info
```

コマンドから `docker is not recognized` の結果が返された場合、または古いバージョンの Docker がインストールされている場合は、続行する前に Docker Engine をインストールしてください。

Docker をセットアップする

Docker コンテナコンポーネントを実行するシステムユーザーには、ルート権限または管理者権限が必要です。権限がない場合は、ルート権限または管理者権限を持たないユーザーとして実行されるように Docker を設定する必要があります。

Linux デバイスの場合は、`ggc_user` ユーザーを `docker` グループに追加することで、`sudo` なしで Docker コマンドを呼び出すことができます。

`ggc_user`、または Docker コンテナコンポーネントの実行に使用する ルート以外のユーザーを `docker` グループに追加するには、次のコマンドを実行します。

```
sudo usermod -aG docker ggc_user
```

詳細については、「Docker Engine インストール後の Linux 手順」を参照してください。

SiteWise エッジゲートウェイパートナーデータソース

以下の情報を使用してパートナーデータソースを設定します。

EasyEdge

ポータル:

https://studio.easyedge.io/

EasyEdge ドキュメント :

EasyEdge の AWS

EasyEdge requirements – ファイアウォールの設定に必要なエンドポイントやポートなど、EasyEdge 要件に関する情報。注: このドキュメントにアクセスするには EasyEdge アカウントが必要です。

パックを使用する。

AWS IoT SiteWise エッジゲートウェイは、さまざまなパックを使用してデータを収集および処理する方法を決定します。

現在、次のパックが利用できます。

- データ収集パック – このパックを使用して産業データを収集し、 AWS クラウドの送信先にルーティングします。デフォルトでは、このパックは SiteWise Edge ゲートウェイで自動的に有効になります。
- データ処理パック — このパックを使用して、 SiteWise エッジゲートウェイとエッジ設定のアセットモデルおよびアセットとの通信を有効にします。エッジ設定を使用すると、オンサイトでコンピューティング、処理するアセットデータを制御できます。その後、データを AWS IoT SiteWise または他の AWS サービスに送信できます。データ処理パックの詳細については、the section called “エッジデータ処理を有効にする。” を参照してください。

パックのアップグレード

> **Important**
>
> データ処理パックのバージョン 2.0.x またはそれ以前からバージョン 2.1.x にアップグレードすると、ローカルに保存されている測定値のデータが失われます。

SiteWise エッジゲートウェイは、さまざまなパックを使用して、データの収集と処理の方法を決定します。 AWS IoT SiteWise コンソールを使用してパックをアップグレードできます。

パックをアップグレードする (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Gateways] を選択します。
3. Gateways リストで、アップグレードするパックを含む SiteWise Edge ゲートウェイを選択します。
4. ゲートウェイ設定セクションで、利用可能なソフトウェア更新を選択します。
5. 「ソフトウェアバージョンの編集」ページの「ゲートウェイコンポーネントの更新」セクションで、次の操作を行います。
   - OPC-UA コレクター を更新するには、バージョンを選択し、デプロイ を選択します。
   - パブリッシャー を更新するには、バージョンを選択し、デプロイ を選択します。
   - データ処理パック を更新するには、バージョンを選択し、デプロイ を選択します。
6. 新しいバージョンのデプロイが完了したら、「完了」を選択します。

パックのアップグレードで問題が発生した場合は、「SiteWise Edge ゲートウェイにパックをデプロイできない」を参照してください。

SiteWise Edge ゲートウェイの管理

AWS IoT SiteWise コンソールと API オペレーションを使用して、 AWS IoT SiteWise エッジゲートウェイを管理できます。AWS OpsHubAWS IoT SiteWise for Windows アプリケーションを使用して、ローカルデバイスから SiteWise Edge ゲートウェイの一部の側面を管理することもできます。

for AWS OpsHub AWS IoT SiteWise アプリケーションを使用して、ローカルデバイスのディスク使用量をモニタリングすることを強くお勧めします。また、 `Gateway.AvailableDiskSpace`および `Gateway.UsedPercentageDiskSpace` Amazon CloudWatch メトリクスをモニタリングし、ディスク容量が不足したときに通知を受け取るアラームを作成することもできます。Amazon CloudWatch アラームの詳細については、「静的しきい値 に基づいて CloudWatch アラームを作成する」を参照してください。

今後のデータを保存するために、端末に十分な空き容量があることを確認してください。ローカルデバイスで容量が不足しようとすると、サービスは最も古いタイムスタンプを持つ少量のデータを自動的に削除し、今後のデータ用のスペースを確保します。

サービスによってデータが削除されたかどうかをチェックするには、次の操作を行います。

1. for AWS OpsHub AWS IoT SiteWise アプリケーションにサインインします。
2. [設定] を選択します。
3. [Logs] (ログ) の場合は、時間範囲を指定し、[Download] (ダウンロード) を選択します。
4. ログファイルを解凍します。
5. ログファイルに次のメッセージが含まれている場合、サービスはデータを削除しました。SiteWise エッジゲートウェイストレージの容量不足を防ぐために、データバイト##削除されました。

AWS IoT SiteWise コンソールを使用した SiteWise Edge ゲートウェイの管理

AWS IoT SiteWise コンソールを使用して、 AWS アカウント内のすべての SiteWise Edge ゲートウェイを設定、更新、モニタリングできます。

SiteWise Edge ゲートウェイを表示するには、AWS IoT SiteWise コンソールの Edge Gateway ページに移動します。特定のゲートウェイのエッジゲートウェイの詳細ページにアクセスするには、エッジゲートウェイの名前を選択します。

Edge ゲートウェイの詳細ページの概要タブから、次の操作を実行できます。

- データソースセクションで、データソース設定を更新し、追加のデータソースを設定します。
- Open CloudWatch metrics を選択して、 CloudWatch メトリクスコンソールでデータソースごとに取り込まれたデータポイントの数を表示します。
- Edge 機能セクションで、編集をクリックして SiteWise Edge ゲートウェイにデータパックを追加します。
- ゲートウェイ設定セクションで、 SiteWise エッジゲートウェイの接続ステータスを表示します。
- パブリッシャー設定セクションで、 SiteWise エッジゲートウェイの同期ステータスとパ AWS IoT SiteWise ブリッシャーコンポーネントの設定を表示します。

Edge ゲートウェイの詳細ページの「更新」タブで、エッジゲートウェイにデプロイされている現在のコンポーネントとパックのバージョンを確認できます。また、新しいバージョンが利用可能になったときにデプロイする場所でもあります。

AWS OpsHub の を使用した SiteWise Edge ゲートウェイの管理 AWS IoT SiteWise

AWS OpsHub for AWS IoT SiteWise アプリケーションを使用して、 SiteWise エッジゲートウェイを管理およびモニタリングします。このアプリケーションは、次のモニタリング、管理オプションを提供します。

- [Overview] (概要) では、次のことができます。
  - SiteWise Edge ゲートウェイの詳細を表示すると、 SiteWise エッジゲートウェイのデバイスデータに関するインサイトの取得、問題の特定、 SiteWise エッジゲートウェイのパフォーマンスの向上に役立ちます。
  - エッジのローカルサーバーと機器からのデータをモニタリングする SiteWise Monitor ポータルを表示します。詳しくは、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[What is AWS IoT SiteWise Monitor] (説明) をご覧ください。
- Health の下に、 SiteWise エッジゲートウェイのデータを表示するダッシュボードがあります。プロセスエンジニアなどのドメインエキスパートは、ダッシュボードを使用して SiteWise Edge ゲートウェイの動作の概要を確認できます。

- アセット で、ローカルデバイスにデプロイされたアセットと、アセットプロパティに対して収集または計算された最後の値を表示します。
- [設定] では、次のことができます。
  - Data Processing Pack がインストールされている場合は、 SiteWise エッジゲートウェイの設定情報を表示し、リソースを クラウドと同期します AWS 。
  - 他のツールを使用して、 SiteWise エッジゲートウェイへのアクセスに使用できる認証ファイルをダウンロードします。
  - SiteWise Edge ゲートウェイのトラブルシューティングに使用できるログをダウンロードします。
  - SiteWise Edge ゲートウェイにデプロイされた AWS IoT SiteWise コンポーネントを表示します。

> **Important**
>
> AWS OpsHub に を使用するには、以下が必要です AWS IoT SiteWise。
>
> - ローカルデバイスと AWS OpsHub for AWS IoT SiteWise アプリケーションが同じネットワークに接続されている必要があります。
> - データ処理パックを有効にする必要があります。

を使用して SiteWise Edge ゲートウェイを管理するには AWS OpsHub

1. AWS OpsHubAWS IoT SiteWise for Windows アプリケーションをダウンロードしてインストールします。
2. アプリケーションを開きます。
3. ゲートウェイにローカル認証情報を設定していない場合は、 の手順に従ってローカルオペレーティングシステムの認証情報を使用した SiteWise Edge ゲートウェイへのアクセス設定します。
4. Linux または Lightweight Directory Access Protocol (LDAP) の認証情報を使用して SiteWise Edge ゲートウェイにサインインできます。 SiteWise Edge ゲートウェイにサインインするには、次のいずれかを実行します。

Linux

1. ホスト名または IP アドレス に、ローカルデバイスのホスト名または IP アドレスを入力します。
2. [Authentication] (認証) は、Linuxを選択します。
3. [User name] (ユーザーネーム) には、お使いの Linux OS のユーザー名を入力してください。
4. [Password] (パスワード) には、お使いの Linux OS のパスワードを入力します。
5. [Sign in (サインイン)] を選択します。

LDAP

1. ホスト名または IP アドレス に、ローカルデバイスのホスト名または IP アドレスを入力します。
2. [Authentication] (認証) は、LDAPを選択します。
3. [User name] (ユーザーネーム) には、LDAP のユーザー名を入力します。
4. [Password] (パスワード) には、LDAP パスワードを入力します。
5. [Sign in (サインイン)] を選択します。

ローカルオペレーティングシステムの認証情報を使用した SiteWise Edge ゲートウェイへのアクセス

Lightweight Directory Access Protocol (LDAP) に加えて、Linux または Windows の認証情報を使用して SiteWise Edge ゲートウェイにアクセスできます。

> **Important**
>
> Linux 認証情報を使用して SiteWise Edge ゲートウェイにアクセスするには、 SiteWise エッジゲートウェイのデータ処理パックをアクティブ化する必要があります。

Linux オペレーティングシステムの認証情報を使用した SiteWise Edge ゲートウェイへのアクセス

次のステップは、Ubuntu が搭載された端末を使用することを想定しています。別の Linux ディストリビューションをお使いの場合は、お使いのデバイスの関連ドキュメントを参照してください。

ユーザープールを作成するには。

1. 管理者グループを作成するには、次のコマンドを実行します。

   ```
   sudo groupadd --system SWE_ADMIN_GROUP
   ```

   SWE_ADMIN_GROUP グループ内のユーザーは、 SiteWise エッジゲートウェイの管理者アクセスを許可できます。
2. ユーザーグループを作成するには、次のコマンドを実行します。

   ```
   sudo groupadd --system SWE_USER_GROUP
   ```

   SWE_USER_GROUP グループのユーザーは、 SiteWise Edge ゲートウェイへの読み取り専用アクセスを許可できます。
3. 管理者グループにユーザーを追加するには、次のコマンドを実行します。*user-name* と *password* の部分は、追加するユーザーのユーザー名とパスワードに置き換えます。

   ```
   sudo useradd -p $(openssl passwd -1 password) user-name
   ```
4. ユーザーを SWE_ADMIN_GROUP または SWE_USER_GROUP に追加するには、*user-name* を、前のステップで追加したユーザー名に置き換えます。

   ```
   sudo usermod -a -G SWE_ADMIN_GROUP user-name
   ```

これで、ユーザー名とパスワードを使用して、 AWS OpsHub for AWS IoT SiteWise アプリケーションの SiteWise Edge ゲートウェイにサインインできます。

Windows 認証情報を使用した SiteWise Edge ゲートウェイへのアクセス

次のステップは、Windows が搭載されたデバイスを使用することを想定しています。

> **Important**
>
> セキュリティは、 AWS とユーザー間で共有される責任です。12 文字以上、大文字、小文字、数字、記号を組み合わせた強力なパスワードポリシーを作成します。また、Windows ファイアウォールのルールを設定して、ポート 443 では受信トラフィックを許可し、その他のすべてのポートでは受信トラフィックをブロックします。

Windows Server ユーザープールを作成する

1. 管理者 PowerShell として を実行します。

   a. SiteWise Edge Gateway をインストールする Windows サーバーで、管理者としてログインします。

   b. Windows の検索バーPowerShellに と入力します。

   c. 検索結果で、Windows PowerShell アプリを右クリックします。[管理者として実行] を選択します。

2. 管理者グループを作成するには、次のコマンドを実行します。

   ```
   net localgroup SWE_ADMIN_GROUP /add
   ```

   SiteWise Edge ゲートウェイの管理者アクセスを許可するには、 SWE_ADMIN_GROUPグループのユーザーである必要があります。

3. ユーザーグループを作成するには、次のコマンドを実行します。

   ```
   net localgroup SWE_USER_GROUP /add
   ```

   SiteWise Edge ゲートウェイへの読み取り専用アクセスを許可するには、 SWE_USER_GROUPグループのユーザーである必要があります。

4. ユーザーを追加するには、次のコマンドを実行します。*user-name* と *password* の部分は、作成するユーザーのユーザー名とパスワードに置き換えます。

   ```
   net user user-name password /add
   ```

5. 管理者グループにユーザーを追加するには、次のコマンドを実行します。*user-name* は、追加したいユーザー名に置き換えてください。

   ```
   net localgroup SWE_ADMIN_GROUP user-name /add
   ```

これで、ユーザー名とパスワードを使用して、 AWS OpsHub for AWS IoT SiteWise アプリケーションの SiteWise Edge ゲートウェイにサインインできます。

SiteWise Edge ゲートウェイ証明書の管理

SiteWise エッジゲートウェイデバイスでは、 SiteWise Monitor や Grafana などのサードパーティーアプリケーションを使用できます。これらのアプリケーションには、 サービスへの TLS 接続が必要です。 SiteWise Edge ゲートウェイは現在、自己署名証明書を使用しています。 SiteWise Monitor ポータルなどのアプリケーションをブラウザで開くと、信頼できない証明書に関する警告が表示されることがあります。

以下は、 AWS IoT SiteWise アプリケーション AWS OpsHub 用の から信頼された証明書をダウンロードする方法を示しています。

1. アプリケーションにサインインします。
2. [設定] を選択します。
3. [Authentication] (認証) で[Download certificate] (証明書のダウンロード) を選択します。

以下では、Google Chrome または を使用していることを前提としています FireFox。他のブラウザをお使いの場合は、お使いのブラウザの関連ドキュメントを参照してください。前のステップでダウンロードした証明書をブラウザに追加するには、次のいずれかを実行します。

- Google Chrome を使用する場合は、[Google Chrome Enterprise Help documentation] (Google Chrome Enterprise ヘルプドキュメント) にある[Set up certificates] (証明書を設定する) に従ってください。
- Firefox を使用している場合は、[Oracle documentation] (Oracleドキュメント) の[To Load the Certificate into the Mozilla or Firefox Browser] (Mozilla または Firefox のブラウザーに証明書を読み込むには) に従ってください。

SiteWise Edge ゲートウェイコンポーネントパックのバージョンの変更

AWS IoT SiteWise コンソールを使用して、 SiteWise エッジゲートウェイのコンポーネントパックのバージョンを変更できます。

SiteWise Edge ゲートウェイコンポーネントパックのバージョンを変更するには

1. AWS IoT SiteWise コンソールに移動します。
2. 左のナビゲーションペインで [ゲートウェイ] を選択します。
3. パックバージョンを変更する SiteWise Edge ゲートウェイを選択します。

4. ゲートウェイ設定 で、ソフトウェアバージョンの表示 を選択します。
5. 「ソフトウェアバージョンの編集」ページで、バージョンを更新するパックについて、デプロイするバージョンを選択し、「デプロイ」を選択します。
6. [完了] をクリックします。

産業用 SiteWise エッジでの Edge の実行

デバイス上で Edge ゲートウェイを実行する AWS アカウント ことで、産業用 SiteWise エッジデバイスから にデータを取り込むことができます。これを行うには、デプロイターゲットを TAK 産業用 SiteWise エッジデバイス - 新しい にして Edge ゲートウェイリソースを作成し、設定ファイルをダウンロードし、TAK 産業用エッジ管理 (IEM) ポータルを介して TAK アプリにアップロードします。必須の TAK リソースの設定方法など、 Industrial Edge で AWS IoT SiteWise Edge を実行する方法の詳細については、 ドキュメントの「What is Industrial Edge?」を参照してください。

> **Note**
>
> TAK は AWS IoT SiteWise Edge のベンダーまたはサプライヤーではありません。TAK Industrial Edge Marketplace は独立したマーケットプレイスです。

トピック

前提条件

Edge を TAK Industrial Edge AWS IoT SiteWise で実行するには、以下が必要です。

- Digital Exchange Platform アカウント
- 産業用 Edge Hub (iehub) アカウント
- 産業用エッジマネジメント (IEM) インスタンス

- 産業用エッジデバイス (IED) または 産業用エッジ仮想デバイス (IEvD) のいずれか
- TAK Industrial Edge デバイスのデプロイターゲットにアクセスします。アクセスを取得するには、 AWS IoT SiteWise コンソールに移動し、アクセスのリクエストを選択します。

セキュリティ

、 のお客様 AWS、 パートナー間の責任共有モデルの一環として、セキュリティのさまざまな側面を担当するのは誰であるかを以下に示します。

顧客の責任

- パートナーの審査。
- パートナーに与えるネットワークアクセス権の設定。
- AWS IoT SiteWise Edge を実行しているデバイスを物理的に保護します。

AWS 責任

- カスタマー AWS クラウドリソースからパートナーを分離します。

パートナーの責任

- 安全なデフォルトの使用。
- パッチやその他の適切な更新による長期にわたるソリューションの安全維持。
- 顧客データの機密保持。
- パートナーマーケットプレイスで利用可能な他のアプリケーションを審査します。

この機能のプレビュー段階では、パートナーデバイスに AWS IoT SiteWise キャッシュする顧客データに、パートナーマーケットプレイスを通じてインストールされたパートナーやその他のアプリケーションからアクセスできます。

設定ファイルを作成する

適切な TAK アカウントと IEM インスタンスを取得したら、デプロイタイプ の SiteWise Edge ゲートウェイを作成できます。

設定ファイルを作成するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、エッジゲートウェイ を選択します。

3. [Create gateway (ゲートウェイの作成)] を選択します。
4. デプロイタイプ で、産業用エッジデバイス - 新しい を選択します。
5. SiteWise Edge ゲートウェイの名前を入力するか、 によって生成された名前を使用します AWS IoT SiteWise。
6. (オプション) 詳細設定 で、次の操作を行います。
   - AWS IoT Core モノの名前を入力するか、 によって生成された名前を使用します AWS IoT SiteWise。
7. [Create gateway (ゲートウェイの作成)] を選択します。
8. SiteWise 「エッジゲートウェイ設定ファイルの生成」ダイアログボックスで「 の生成とダウンロード」を選択します。 は、 AWS IoT SiteWise エッジアプリケーションの設定に使用する設定ファイル AWS IoT SiteWise を自動的に生成します。

> **Important**
> 設定ファイルは必ず安全な場所に保存してください。このファイルは後で使用します。

Edge SiteWise ゲートウェイを作成したので、次の手順を実行して SiteWise Edge ゲートウェイの設定を完了します。

1. データソースを追加する
2. パブリッシャーコンポーネントを設定する

設定ファイルと SiteWise エッジゲートウェイを設定したら、「Industrial AWS IoT SiteWise Edge Marketplace」からエッジアプリケーションをダウンロードし、「Industrial Edge Management (IEM) ポータル」を使用してインストールします。次に、TAK Industrial Edge Management (IEM) ポータルから TAK Industrial Edge デバイスにアクセスし、 SiteWise Edge ゲートウェイをインストールするデバイスに設定ファイルをアップロードします。

トラブルシューティング

産業用 SiteWise エッジデバイスの Edge ゲートウェイのトラブルシューティングを行うには、 産業用エッジ管理 (IEM) または TAK 産業用エッジデバイス (IED) ポータルからアプリケーションのログにアクセスできます。詳細については、 ドキュメントの「Downloading Logs」を参照してください。

「SESSION_TAKEN_OVER」または「com.aws.greengrass.mqttclient」と表示されます。MqttClient: スプーラー経由でメッセージを発行できませんでした。再試行します。」 ログ内の

を含む警告`SESSION_TAKEN_OVER`または の口グ`com.aws.greengrass.mqttclient.MqttClient: Failed to publish the message via Spooler and will retry.` に を含むエラーが表示された場合は`/greengrass/v2/logs/greengrass.log`、複数のデバイス上の複数の SiteWise Edge ゲートウェイに同じ設定ファイルを使用しようとしている可能性があります。各 SiteWise Edge ゲートウェイには、 に接続するための固有の設定ファイルが必要です AWS アカウント。

「com.aws.greengrass.deployment.IotJobsHelper: No deployment job found.」と表示されます。または「デプロイ結果はすでに報告されています」 ログ内の

ログ`Deployment result already reported.` に `com.aws.greengrass.deployment.IotJobsHelper: No deployment job found.` または が表示されている場合は`/greengrass/v2/logs/greengrass.log`、同じ設定ファイルを再利用しようとしている可能性があります。

複数のソリューションがあります。

- 設定ファイルを再利用する場合は、次の手順を実行します。
  1. AWS IoT SiteWise コンソールに移動します。
  2. ナビゲーションペインで、[Gateways] を選択します。
  3. 再利用する SiteWise Edge ゲートウェイを選択します。
  4. [更新] タブを選択します。
  5. 別のパブリッシャーバージョンを選択し、デプロイを選択します。
- 「」の手順に従って設定ファイルを作成する、新しい設定ファイルを作成します。

ログに「Config ファイルには AWS_REGION」がありません。

TAK ログ`Config file missing AWS_REGION` に が表示された場合、設定ファイルの JSON が破損しています。新しい設定ファイルを作成する必要があります。「」の手順に従って設定ファイルを作成する、新しい設定ファイルを作成します。

お問い合わせ

- アプリケーションへのアクセスをリクエストする場合は、AWS IoT SiteWise コンソールに移動し、アクセスのリクエストを選択します。
- アプリケーションのトラブルシューティングのサポートが必要な場合は、AWS IoT SiteWise コンソール に移動し、 SiteWise エッジゲートウェイの詳細ページに移動し、サポート を選択します。

SiteWise Edge ゲートウェイのアセットのフィルタリング

エッジフィルタリングを使用すると、アセットの一部のみを特定の SiteWise Edge ゲートウェイに送信してデータ処理に使用することで、アセットをより効率的に管理できます。アセットがツリー構造または親子構造になっている場合は、 SiteWise Edge ゲートウェイの IAM ロールにアタッチされた IAM ポリシーを設定して、ツリーのルート、つまり親、およびその子のみを特定の Edge ゲートウェイに送信できるようにすることができます。 SiteWise

> **Note**
>
> 既存のアセットをツリー構造に整理する場合は、構造を作成した後に、構造に追加した既存の各アセットに移動して [編集] を選択し、[保存] を選択して、 AWS IoT SiteWise 新しい構造が認識されることを確認します。

エッジフィルタリングのセットアップ

SiteWise Edge ゲートウェイの IAM ロールに次の IAM ポリシーを追加し、*< root-asset-id >* を SiteWise Edge ゲートウェイに送信するルートアセットの ID に置き換えて、Edge ゲートウェイにエッジフィルタリングを設定します。 SiteWise

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "iotsitewise:DescribeAsset",
                "iotsitewise:ListAssociatedAssets"
            ],
```

```
            "Resource": "arn:aws:iotsitewise:*:*:asset/*",
            "Condition": {
                "StringNotLike": {
                    "iotsitewise:assetHierarchyPath": "/<root-asset-id>*"
                }
            }
        }
    ]
}
```

SiteWise Edge ゲートウェイに現在削除したいアセットがある場合は、 SiteWise Edge ゲートウェイにログインし、次のコマンドを実行して、 AWS IoT SiteWise キャッシュを削除して SiteWise Edge ゲートウェイを強制的に同期させます。

```
sudo rm /greengrass/v2/work/aws.iot.SiteWiseEdgeProcessor/sync-app/
sync_resource_bundles/edge.json
```

エッジでの AWS IoT SiteWise API の使用

エッジ固有の API と共に利用可能な AWS IoT SiteWise API のサブセットを使用して、エッジ上のアセットモデルとそのアセットを操作できます。アセットモデルは、エッジで実行するように設定する必要があります。詳細については、「エッジでのデータ処理」を参照してください。

これらの API を使用して、アセットモデルとアセットに関するデータを収集したり、デプロイされたポータルとダッシュボードメトリックスを監視したり、エッジで収集されたアセットデータを取得したりできます。ネットワークでの AWS IoT SiteWise とのインタラクションのための、Web API コールを必要としない一元的なホストの機能が提供されます。

トピック

- AWS IoT SiteWise エッジデバイスで使用できるすべての API
- AWS IoT SiteWise エッジデバイス用のエッジ専用 API
- チュートリアル: SiteWise Edge ゲートウェイでのアセットモデルのリストの取得

AWS IoT SiteWise エッジデバイスで使用できるすべての API

エッジ上のデバイスの操作では、さまざまな API を使用して AWS IoT SiteWise を操作したり、デバイスにローカルにタスクを実行したりできます。

使用可能な AWS IoT SiteWise API

エッジデバイスでは次の AWS IoT SiteWise API を使用できます。

- ListAssetModels
- DescribeAssetModel
- ListAssets
- DescribeAsset
- DescribeAssetProperty
- ListAssociatedAssets
- GetAssetPropertyAggregates
- GetAssetPropertyValue
- GetAssetPropertyValueHistory
- ListDashboards
- ListPortals
- ListProjectAssets
- ListProjects
- DescribeDashboard
- DescribePortal
- DescribeProject

使用可能なエッジ専用の API

次の API はエッジ上のデバイスでローカルに使用されます。

- Authenticate – この API を使用して、API コールに使用する SigV4 の一時認証情報を取得できます。

AWS IoT SiteWise エッジデバイス用のエッジ専用 API

エッジで使用できる AWS IoT SiteWise API の他に、エッジ特有の API もあります。以下では、そうしたエッジ特有の API を説明します。

Authenticate

SiteWise Edge ゲートウェイから認証情報を取得します。LDAP または Linux ユーザープールを使用してローカルユーザーを追加するか、システムに接続する必要があります。ユーザーの追加の詳細については、「LDAP」または「Linux ユーザープール」を参照してください。

リクエストの構文

```
POST /authenticate HTTP/1.1
Content-type: application/json
{
  "username": "string",
  "password": "string",
  "authMechanism": "string"
}
```

URI リクエストのパラメータ

リクエストでは URI パラメータを使用しません。

リクエスト本文

リクエストは以下の JSON 形式のデータを受け入れます。

username

リクエスト呼び出しの検証に使用するユーザー名。

型: 文字列

必須: はい

password

認証情報をリクエストしているユーザーのパスワード。

型: 文字列

必須: はい

authMechanism

ホストでこのユーザーを検証するための認証方法。

型: 文字列

有効な値: `ldap`, `linux`, `winnt`

必須: はい

レスポンスの構文

```
HTTP/1.1 200
Content-type: application/json
{
  "accessKeyId": "string",
  "secretAccessKey": "string",
  "sessionToken": "string",
  "region": "edge"
}
```

レスポンス要素

アクションが成功すると、サービスは HTTP 200 レスポンスを返します。

以下のデータが JSON 形式で返されます。

accessKeyId

仮のセキュリティ認証情報を識別するアクセスキー ID。

長さ制限: 最小長は 16 です。最大長は 128 です。

パターン: `[\w]*`

secretAccessKey

リクエストへの署名に使用できるシークレットアクセスキー。

型: 文字列

sessionToken

仮の認証情報を使用するために、ユーザーがサービス API に渡す必要があるトークン。

型: 文字列

region

API コールのターゲットにするリージョン。

型: 定数 - edge

エラー

IllegalArgumentException

提供された本文ドキュメントの形式に誤りがあるため、リクエストが拒否されました。エラーメッセージは、それぞれのエラーの説明です。

HTTP ステータスコード : 400

AccessDeniedException

ユーザーには、現在の ID プロバイダーに基づく有効な認証情報がありません。このエラーメッセージは認証メカニズムの説明です。

HTTP ステータスコード: 403

TooManyRequestsException

リクエストが認証試行回数の上限に達しました。このエラーメッセージには、新たに認証を試みるまでの待ち時間が含まれます。

HTTP ステータスコード: 429

チュートリアル: SiteWise Edge ゲートウェイでのアセットモデルのリストの取得

エッジ固有の API と共に利用可能な AWS IoT SiteWise API のサブセットを使用して、エッジ上のアセットモデルとそのアセットを操作できます。このチュートリアルでは、AWS IoT SiteWiseエッジゲートウェイへの一時的な認証情報の取得と SiteWise 、エッジゲートウェイ上のアセットモデルのリストの取得について説明します。

前提条件

このチュートリアルの手順では、さまざまなツールを利用できます。そうしたツールを使用するには、対応する前提条件がインストールされていることを確認してください。

このチュートリアルを完了するには、以下が必要です。

- A がデプロイされて、SiteWise エッジゲートウェイの要件 が稼働中
- ポート 443 経由で同じネットワーク内の SiteWise Edge ゲートウェイにアクセスします。
- OpenSSL のインストール
- （ AWS OpsHub の場合は AWS IoT SiteWise) AWS OpsHubAWS IoT SiteWiseアプリケーション用の
- (curl) curl のインストール
- (Python) urllib3 のインストール
- (パイソン) Python3 のインストール
- (Python) Boto3 のインストール
- (Python) BotoCoreがインストールされている

ステップ 1: SiteWise Edge ゲートウェイサービスの署名付き証明書を取得する

SiteWise Edge ゲートウェイで利用可能な APIs への TLS 接続を確立するには、信頼できる証明書が必要です。この証明書は、OpenSSL または AWS OpsHubの を使用して生成できますAWS IoT SiteWise。

OpenSSL

> **Note**
>
> このコマンドを実行するには、OpenSSL がインストールされている必要があります。

ターミナルを開き、次のコマンドを実行して、 SiteWise エッジゲートウェイから署名付き証明書を取得します。を SiteWise Edge ゲートウェイの IP <sitewise_gateway_ip>に置き換えます。

```
openssl s_client -connect <sitewise_gateway_ip>:443 </dev/null 2>/dev/null | openssl
 x509 -outform PEM > GatewayCert.pem
```

AWS OpsHub for AWS IoT SiteWise

AWS IoT SiteWise には AWS OpsHub を使用できます。詳細については、「SiteWise Edge ゲートウェイの管理」を参照してください。

このチュートリアルでは、ダウンロードした SiteWise Edge ゲートウェイ証明書への絶対パスを使用します。次のコマンドを実行して、証明書の完全なパスをエクスポートします。<absolute_path_to_certificate> の部分はその証明書へのパスに置き換えます。

```
export PATH_TO_CERTIFICATE='<absolute_path_to_certificate>'
```

ステップ 2: SiteWise Edge ゲートウェイのホスト名を取得する

> **Note**
>
> このコマンドを実行するには、OpenSSL がインストールされている必要があります。

チュートリアルを完了するには、 SiteWise エッジゲートウェイのホスト名が必要です。 SiteWise Edge ゲートウェイのホスト名を取得するには、以下を実行し、 を Edge ゲートウェイの IP SiteWise <sitewise_gateway_ip>に置き換えます。

```
openssl s_client -connect <sitewise_gateway_ip>:443 </dev/null 2>/dev/null | grep -Po
 'CN = \K.*'| head -1
```

次のコマンドを実行して、後で使用するためにホスト名をエクスポートし、 を SiteWise Edge ゲートウェイのホスト名<your_edge_gateway_hostname>に置き換えます。

```
export GATEWAY_HOSTNAME='<your_edge_gateway_hostname>'
```

ステップ 3: SiteWise Edge ゲートウェイの一時的な認証情報を取得する

署名付き証明書と SiteWise Edge ゲートウェイのホスト名を取得したら、ゲートウェイで APIs を実行できるように、一時的な認証情報を取得する必要があります。これらの認証情報は、 APIs を使用して、 AWS OpsHubの を通じて、AWS IoT SiteWiseまたは SiteWise Edge ゲートウェイから直接取得できます。

> **Important**
>
> 認証情報は 4 時間ごとに期限切れになるため、 SiteWise エッジゲートウェイで APIs を使用する直前に認証情報を取得する必要があります。認証情報を 4 時間以上キャッシュしないでください。

for を使用して一時的な認証情報AWS OpsHubを取得する AWS IoT SiteWise

> **Note**
> AWS OpsHub for AWS IoT SiteWise アプリケーションをインストールしておく必要があります。

AWS OpsHub for AWS IoT SiteWise アプリケーションを使用して仮の認証情報を取得するには、以下の操作を行います。

1. アプリケーションにログインします。
2. [設定] を選択します。
3. [認証] で [認証情報のコピー] を選択します。
4. 環境に合ったオプションを展開し、[コピー] を選択します。
5. 後で使用するため認証情報を保存しておきます。

SiteWise Edge ゲートウェイ API を使用して一時的な認証情報を取得する

SiteWise Edge ゲートウェイ API を使用して Python スクリプトまたは curl を使用できる一時的な認証情報を取得するには、まず SiteWise Edge ゲートウェイのユーザー名とパスワードが必要です。 SiteWise Edge ゲートウェイは SigV4 認証と認可を使用します。ユーザーの追加の詳細については、「LDAP」または「Linux ユーザープール」を参照してください。これらの認証情報は、以下のステップで AWS IoT SiteWise APIsを使用するために必要な SiteWise Edge ゲートウェイのローカル認証情報を取得するために使用します。

Python

> **Note**
> urllib3 と Python3 がインストールされている必要があります。

Python を使用して認証情報を取得する

1. get_credentials.py というファイルを作成し、そのファイルに次のコードをコピーします。

```
'''
```

```
The following demonstrates how to get the credentials from the SiteWise Edge
 gateway. You will need to add local users or connect your system to LDAP/AD
https://docs.aws.amazon.com/iot-sitewise/latest/userguide/manage-gateways-
ggv2.html#create-user-pool

Example usage:
    python3 get_credentials.py -e https://<gateway_hostname> -c
 <path_to_certificate> -u '<gateway_username>' -p '<gateway_password>' -m
 '<method>'
'''
import urllib3
import json
import urllib.parse
import sys
import os
import getopt

"""
This function retrieves the AWS IoT SiteWise Edge gateway credentials.
"""
def get_credentials(endpoint,certificatePath, user, password, method):
    http = urllib3.PoolManager(cert_reqs='CERT_REQUIRED', ca_certs=
 certificatePath)
    encoded_body = json.dumps({
        "username": user,
        "password": password,
        "authMechanism": method,
    })

    url =  urllib.parse.urljoin(endpoint, "/authenticate")

    response = http.request('POST', url,
        headers={'Content-Type': 'application/json'},
        body=encoded_body)

    if response.status != 200:
        raise Exception(f'Failed to authenticate! Response status
 {response.status}')

    auth_data = json.loads(response.data.decode('utf-8'))

    accessKeyId = auth_data["accessKeyId"]
    secretAccessKey = auth_data["secretAccessKey"]
    sessionToken = auth_data["sessionToken"]
```

```
    region = "edge"

    return accessKeyId, secretAccessKey, sessionToken, region

def print_help():
    print('Usage:')
    print(f'{os.path.basename(__file__)} -e <endpoint> -c <path/to/certificate>
 -u <user> -p <password> -m <method> -a <alias>')
    print('')
    print('-e, --endpoint   edge gateway endpoint. Usually the Edge gateway
 hostname.')
    print('-c, --cert_path path to downloaded gateway certificate')
    print('-u, --user       Edge user')
    print('-p, --password   Edge password')
    print('-m, --method     (Optional) Authentication method (linux, winnt,
 ldap), default is linux')
    sys.exit()


def parse_args(argv):
    endpoint = ""
    certificatePath = None
    user = None
    password = None
    method = "linux"

    try:
        opts, args = getopt.getopt(argv, "he:c:u:p:m:",
 ["endpoint=","cert_path=", "user=", "password=", "method="])
    except getopt.GetoptError:
        print_help()

    for opt, arg in opts:
        if opt == '-h':
            print_help()
        elif opt in ("-e", "--endpoint"):
            endpoint = arg
        elif opt in ("-u", "--user"):
            user = arg
        elif opt in ("-p", "--password"):
            password = arg
        elif opt in ("-m", "--method"):
            method = arg.lower()
        elif opt in ("-c", "--cert_path"):
```

```
            certificatePath = arg

    if method not in ['ldap', 'linux', 'winnt']:
        print("not valid method parameter, required are ldap, linux, winnt")
        print_help()

    if (user == None or password == None):
        print("To authenticate against edge user, password have to be passed
 together, and the region has to be set to 'edge'")
        print_help()

    if(endpoint == ""):
        print("You must provide a valid and reachable gateway hostname")
        print_help()

    return endpoint,certificatePath, user, password, method


def main(argv):
    # get the command line args
    endpoint, certificatePath, user, password, method = parse_args(argv)

    accessKeyId, secretAccessKey, sessionToken, region=get_credentials(endpoint,
 certificatePath, user, password, method)

    print("Copy and paste the following credentials into the shell, they are
 valid for 4 hours:")
    print(f"export AWS_ACCESS_KEY_ID={accessKeyId}")
    print(f"export AWS_SECRET_ACCESS_KEY={secretAccessKey}")
    print(f"export AWS_SESSION_TOKEN={sessionToken}")
    print(f"export AWS_REGION={region}")
    print()




if __name__ == "__main__":
   main(sys.argv[1:])
```

2. ターミナルから get_credentials.py を実行します。<gateway_username> および <gateway_password> の部分は作成した認証情報に置き換えます。

```
python3 get_credentials.py -e https://$GATEWAY_HOSTNAME -c $PATH_TO_CERTIFICATE
 -u '<gateway_username>' -p '<gateway_password>' -m 'linux'
```

curl

> **Note**
>
> curl がインストールされている必要があります。

curl を使用して認証情報を取得する

1. ターミナルから次のコマンドを実行します。<gateway_username> および <gateway_password> の部分は作成した認証情報に置き換えます。

   ```
   curl --cacert $PATH_TO_CERTIFICATE --location \
   -X POST https://$GATEWAY_HOSTNAME:443/authenticate \
   --header 'Content-Type: application/json' \
   --data-raw '{
       "username": "<gateway_username>",
       "password": "<gateway_password>",
       "authMechanism": "linux"
   }'
   ```

   レスポンスは次のようになります。

   ```
   {
       "username": "sweuser",
       "accessKeyId": "<accessKeyId>",
       "secretAccessKey": "<secretAccessKey>",
       "sessionToken": "<sessionToken>",
       "sessionExpiryTime": "2022-11-17T04:51:40.927095Z",
       "authMechanism": "linux",
       "role": "edge-user"
   }
   ```

2. ターミナルから次のコマンドを実行します。

   ```
   export AWS_ACCESS_KEY_ID=<accessKeyId>
   ```

```
export AWS_SECRET_ACCESS_KEY=<secretAccessKey>
export AWS_SESSION_TOKEN=<sessionToken>
export AWS_REGION=edge
```

ステップ 4: SiteWise Edge ゲートウェイでアセットモデルのリストを取得する

Edge ゲートウェイの署名付き証明書、 SiteWise エッジ SiteWise ゲートウェイのホスト名、および一時的な認証情報を取得したら、 `ListAssetModels` API を使用して SiteWise 、エッジゲートウェイ上のアセットモデルのリストを取得できます。

Python

> **Note**
>
> Python3、Boto3、および BotoCoreがインストールされている必要があります。

Python を使用してアセットモデルのリストを取得する

1. list_asset_model.py というファイルを作成し、そのファイルに次のコードをコピーします。

```
import json
import boto3
import botocore
import os

# create the client using the credentials
client = boto3.client("iotsitewise",
    endpoint_url= "https://"+ os.getenv("GATEWAY_HOSTNAME"),
    region_name=os.getenv("AWS_REGION"),
    aws_access_key_id=os.getenv("AWS_ACCESS_KEY_ID"),
    aws_secret_access_key=os.getenv("AWS_SECRET_ACCESS_KEY"),
    aws_session_token=os.getenv("AWS_SESSION_TOKEN"),
    verify=os.getenv("PATH_TO_CERTIFICATE"),
    config=botocore.config.Config(inject_host_prefix=False))

# call the api using local credentials
response = client.list_asset_models()
print(response)
```

2. ターミナルから list_asset_model.py を実行します。

```
python3 list_asset_model.py
```

curl

> **Note**
>
> curl がインストールされている必要があります。

curl を使用してアセットモデルのリストを取得する

ターミナルから、次のコマンドを実行します。

```
curl \
  --request GET https://$GATEWAY_HOSTNAME:443/asset-models \
  --cacert $PATH_TO_CERTIFICATE \
  --aws-sigv4 "aws:amz:edge:iotsitewise" \
  --user "$AWS_ACCESS_KEY_ID:$AWS_SECRET_ACCESS_KEY" \
  -H "x-amz-security-token:$AWS_SESSION_TOKEN"
```

レスポンスは次のようになります。

```
{
    "assetModelSummaries": [
        {
            "arn": "arn:aws:iotsitewise:{region}:{account-id}:asset-model/{asset-
model-id}",
            "creationDate": 1.669245291E9,
            "description": "This is a small example asset model",
            "id": "{asset-model-id}",
            "lastUpdateDate": 1.669249038E9,
            "name": "Some Metrics Model",
            "status": {
                "error": null,
                "state": "ACTIVE"
            }
        },
        .
        .
```

```
        .
    ],
    "nextToken": null
}
```

SiteWise Edge ゲートウェイのバックアップと復元

このトピックでは、 SiteWise エッジゲートウェイを復元し、メトリクスデータをバックアップする方法について説明します。同じマシンで SiteWise Edge ゲートウェイの破損に関する問題が発生していて、問題のトラブルシューティングが必要な場合は、SiteWise 「Edge ゲートウェイの問題のトラブルシューティング」の AWS IoT SiteWise ドキュメントを参照してください。

> **Note**
>
> このトピックで説明するガイダンスは、 AWS IoT Greengrass V2 バージョン 2.1.0 以降にインストールされている SiteWise Edge ゲートウェイを対象としています。

メトリクスデータの日次バックアップ

データを新しいマシンに転送または復元する場合は、バックアップを作成しておくことが大切です。データをバックアップしておくことで、転送または復元中に運用データが失われるリスクを大幅に軽減できます。

influxdb フォルダーのパスは次のとおりです。

Linux

```
/greengrass/v2/work/aws.iot.SiteWiseEdgeProcessor/influxdb
```

Windows

```
C:\greengrass\v2\work\aws.iot.SiteWiseEdgeProcessor\influxdb
```

フォルダー全体とその下にあるすべてのものをバックアップすることをお勧めします。

メトリクスデータは、1.0 SiteWise Edge から外部ハードドライブまたは AWS クラウドに定期的にバックアップすることをお勧めします。

SiteWise Edge ゲートウェイを復元する

SiteWise Edge ゲートウェイを復元するには、次の手順に従います。

1. SiteWise Edge ゲートウェイの作成時にダウンロードしたインストールスクリプトを使用して、新しいマシンで SiteWise Edge ゲートウェイを復元します。Edge SiteWise ゲートウェイをセットアップするには、ローカルデバイスへの Edge ゲートウェイソフトウェアのインストールの手順を参照してください。 SiteWise

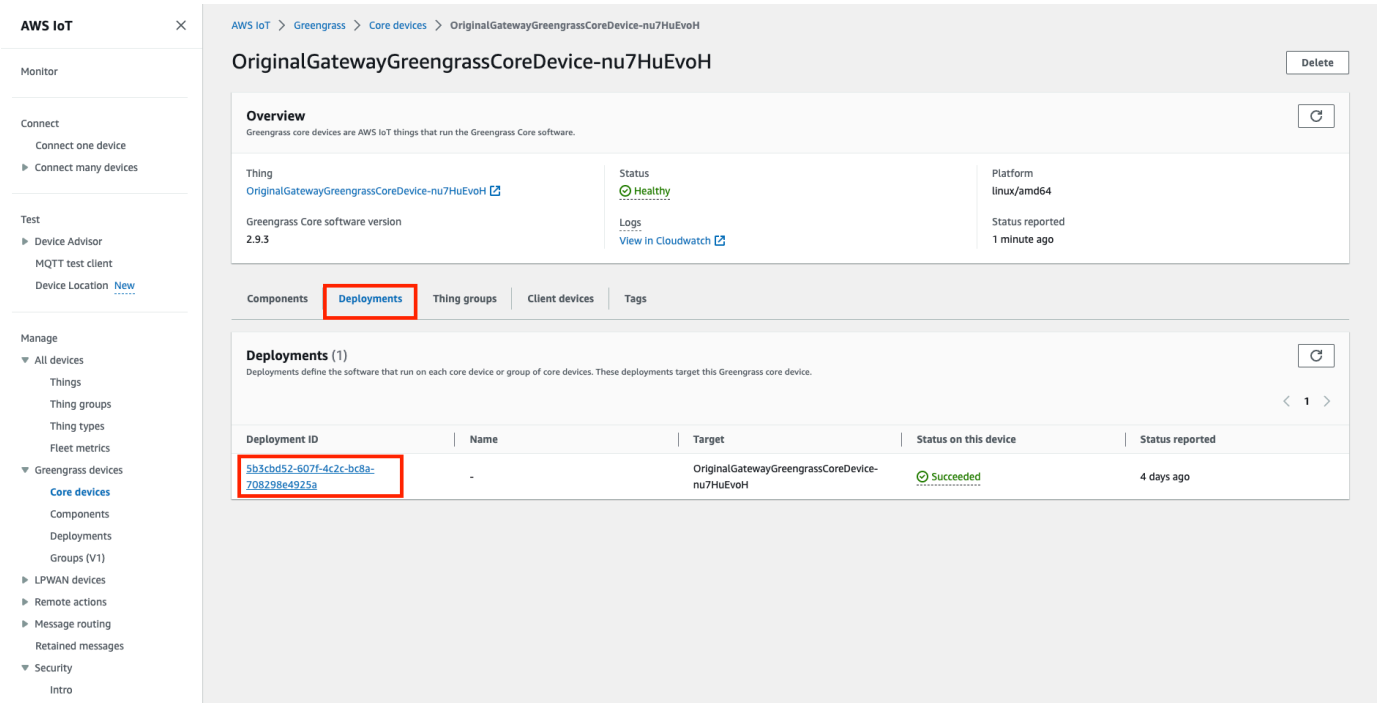
   インストールスクリプトを紛失したり、見つからない場合は、AWS カスタマーサポート にご連絡ください。

2. SiteWise Edge ゲートウェイがインストールされたら、AWS IoT Greengrass コンソール にログインします。

3. コンポーネントを再デプロイするには、[管理] に移動し、[AWS IoT Greengrass デバイス] で [コアデバイス] を選択します。

4. AWS IoT Greengrass コアデバイステーブルで、 SiteWise エッジゲートウェイに対応するコアデバイスを選択します。

5. デバイスページが開いたら、[デプロイ] タブを開いてデプロイ ID を選択します。選択した ID で [デプロイ] ページが開きます。



6. [デプロイ] ページが表示されたら、右上の [アクション] ボタンを押して [修正] オプションを選択し、新しいデプロイを開始します。デプロイの設定をします。デプロイをそのまま残しておきたい場合は、[レビュー] と [デプロイ」に進みます。
7. デプロイステータスが Completed になるまで待ちます。

> **Note**
>
> また、 SiteWise エッジ上のすべてのコンポーネントが完全にセットアップされて実行されるまでに数分かかります。

AWS IoT SiteWise データの復元

以下の手順に従って、新しいマシンにデータを復元できます。

1. influxdb フォルダーを新しいマシンにコピーします。
2. ターミナルで次のコマンドを実行して、 SiteWise EdgeProcessor コンポーネントを停止します。

   Linux

   ```
   sudo /greengrass/v2/bin/greengrass-cli component stop -n
   aws.iot.SiteWiseEdgeProcessor
   ```

   Windows

   ```
   C:\greengrass\v2\bin\greengrass-cli component stop -n
   aws.iot.SiteWiseEdgeProcesso
   ```

3. データをバックアップしたパスを見つけて、以下のコマンドを実行します。

   Linux

   ```
   sudo yes | sudo cp -rf <influxdb_backup_path> /greengrass/v2/work/
   aws.iot.SiteWiseEdgeProcessor/influxdb
   ```

   PowerShell

   ```
   Copy-Item -Recurse -Force <influxdb_backup_path>\* C:\greengrass
   \v2\work\aws.iot.SiteWiseEdgeProcessor\
   ```

Windows

```
robocopy <influxdb_backup_path> C:\greengrass\v2\work
\aws.iot.SiteWiseEdgeProcessor\ /E
```

4. SiteWiseEdgeProcessor コンポーネントを再起動します。

   Linux

   ```
   sudo /greengrass/v2/bin/greengrass-cli component restart -n
   aws.iot.SiteWiseEdgeProcessor
   ```

   Windows

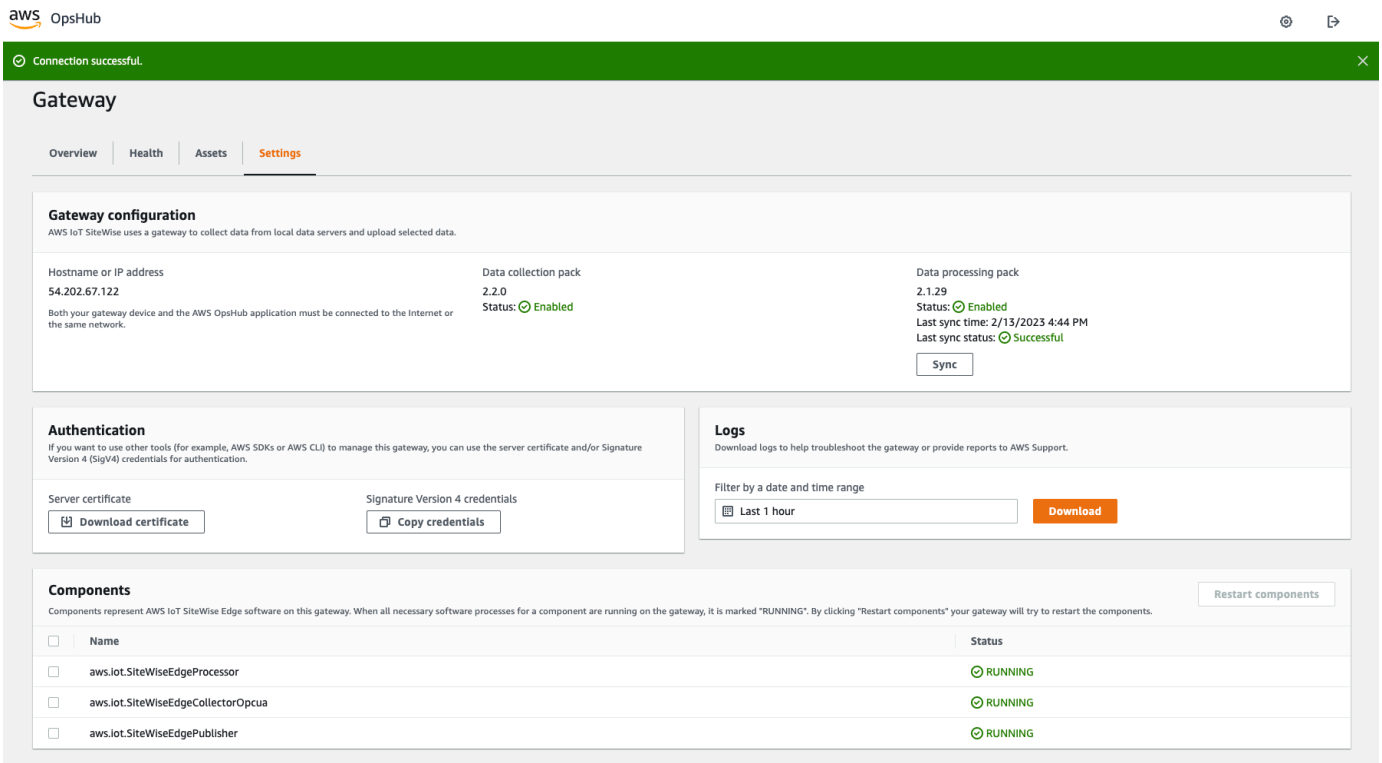
   ```
   C:\greengrass\v2\bin\greengrass-cli component restart -n
   aws.iot.SiteWiseEdgeProcessor
   ```

バックアップと復元が正常に完了したことを検証する

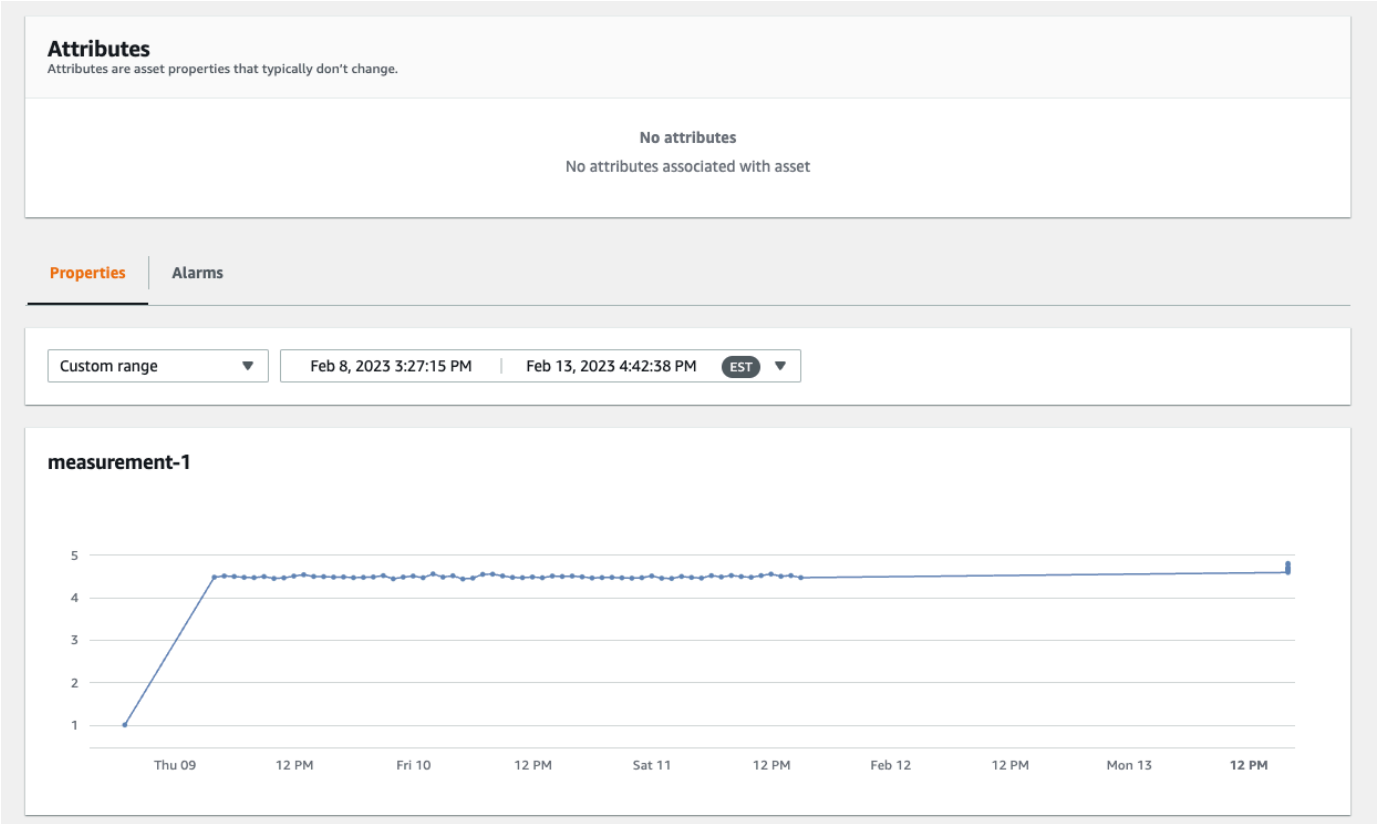
この手順を使用して、バックアップデータと SiteWise Edge ゲートウェイの復元を検証します。

> **Note**
>
> この手順では、 AWS OpsHub に をインストールする必要があります AWS IoT SiteWise。詳細については、「 の を使用した SiteWise Edge ゲートウェイの管理 AWS OpsHub AWS IoT SiteWise」を参照してください。

1. AWS OpsHub で を開きます AWS IoT SiteWise。
2. SiteWise エッジゲートウェイ設定ページで、コンポーネントテーブルにリストされている各コンポーネントのステータスを確認します。ステータスの色が緑色で、表示に RUNNING と表示されていることを確認します。



3. ポータルダッシュボードで過去のデータを検証し、過去のデータと新しいデータの両方が正しく設定されていることを確認します。過去のデータと新しいデータの間にはダウンタイムがあります。その場合は、データポイントが収集されていない期間が表示されます。



Edge ゲートウェイのバックアップまたは復元で問題が発生した場合は、次のトラブルシューティングトピック SiteWise 「Edge AWS IoT SiteWise ゲートウェイのトラブルシューティング」を参照してください。

SiteWise Edge ゲートウェイのセットアップ (AWS IoT Greengrass Version 1)

> **Note**
>
> SiteWise で実行されているエッジゲートウェイ AWS IoT Greengrass V1 は、2021 年 7 月 29 日より前にこの機能の使用を開始した場合にのみ使用できます。それ以外の場合は、 で実行されている SiteWise Edge ゲートウェイを設定します AWS IoT Greengrass V2。

SiteWise Edge ゲートウェイ AWS IoT SiteWise を使用して産業データを に送信し、産業機器からデータをアップロードできます。 SiteWise Edge ゲートウェイは、 AWS IoT SiteWise とデータ産業機器の間の仲介として機能します。 は、 SiteWise Edge ゲートウェイをセットアップ AWS IoT

Greengrass するために実行できる任意のデバイスにデプロイできる AWS IoT Greengrass コンポーネント AWS IoT SiteWise を提供します。 は、OPC-UA サーバープロトコルとのリンク AWS IoT SiteWise をサポートしています。

で実行される AWS IoT SiteWise Edge ゲートウェイがある場合は AWS IoT Greengrass V1、SiteWise Edge ゲートウェイを にアップグレードできます AWS IoT Greengrass V2。詳細については、 SiteWise 「エッジゲートウェイを から にアップグレード AWS IoT Greengrass V1 する手順 AWS IoT Greengrass V2」を参照してください。

トピック

- AWS IoT Greengrass V1 SiteWise Edge ゲートウェイデバイスの選択
- AWS IoT Greengrass V1 SiteWise Edge ゲートウェイの設定
- AWS IoT Greengrass V1 SiteWise Edge ゲートウェイでのデータソースの設定

AWS IoT Greengrass V1 SiteWise Edge ゲートウェイデバイスの選択

産業オペレーションに最適なローカルデバイスを選択します。 SiteWise Edge ゲートウェイは、 を実行できる任意のデバイスで設定できます AWS IoT Greengrass。すべてのローカルデバイスは、次の要件を満たしている必要があります。

- AWS IoT Greengrass Core ソフトウェア v1.10.2 以降をサポートします。詳細については、[AWS IoT Greengrass Version 1 Developer Guide] (デベロッパーガイド) の[Supported platforms and requirements] (対応プラットフォームと要件) を参照してください。
- RAM が 4 GB 以上ある。
- 10 GB 以上の空きディスク容量があること。
- Java 8 仮想マシン (JVM) をサポートしていること。

を使用してエッジでデータを処理する場合は AWS IoT SiteWise、ローカルデバイスも次の要件を満たしている必要があります。

- x86 64 bit クアッドコアプロセッサ搭載。
- 16 GB 以上の RAM があること。
- Windows を使用している場合、RAM に 32 GB 以上が必要です。
- 256 GB 以上の空きディスク容量があること。

断続的なインターネット接続のためにデータをキャッシュするために必要なディスク容量は、次の要因によって異なります。

- アップロードされたデータストリームの数
- 1 秒あたりのデータストリームごとのデータポイント
- 各データポイントのサイズ
- 通信速度
- 予想されるネットワークのダウンタイム

データのポーリングとアップロードに必要なコンピューティング性能は、次の要因によって異なります。

- アップロードされたデータストリームの数
- 1 秒あたりのデータストリームごとのデータポイント

AWS IoT Greengrass V1 SiteWise Edge ゲートウェイの設定

AWS IoT SiteWise Edge ゲートウェイは、産業機器と の間の仲介として機能します AWS IoT SiteWise。 SiteWise Edge ゲートウェイソフトウェアは、 を実行できる任意のデバイスにデプロイできます AWS IoT Greengrass。詳細については、「AWS IoT Greengrass V1 SiteWise Edge ゲートウェイデバイスの選択」を参照してください。

Edge ゲートウェイのデータ処理パックを使用して AWS IoT SiteWise 、 SiteWise がエッジデバイスでローカルにデータを処理できるようにします。これは、 SiteWise エッジゲートウェイを に追加するときに行います AWS IoT SiteWise。エッジでのデータ処理については、the section called “エッジデータ処理を有効にする。” を参照してください。

> **Note**
>
> ローカルネットワークおよび企業ネットワークへの IT 管理アクセス権を持つユーザーと一緒に次のステップを実行することをお勧めします。これらのステップでは、産業機器とファイアウォール設定を構成する権限に精通している人が必要になる場合があります。

トピック

- SiteWise Edge ゲートウェイ環境のセットアップ

- IAMポリシーとロールを作成する。
- AWS IoT Greengrass グループの設定
- AWS IoT SiteWise コネクタの設定
- Edge SiteWise ゲートウェイを に追加する AWS IoT SiteWise

SiteWise Edge ゲートウェイ環境のセットアップ

この手順では、 で使用する SiteWise Edge ゲートウェイをインストール AWS IoT Greengrass して設定します AWS IoT SiteWise。

> **Note**
>
> このセクションでは、apt コマンドを使用してパッケージをインストールする手順を説明します。これは、Ubuntu などを実行しているシステムに適用されます。同様のシステムを使用していない場合は、使用しているディストリビューションのドキュメントを参照し、推奨されるパッケージインストーラを使用してください。

SiteWise Edge ゲートウェイをセットアップするには

1. 必要に応じて、 SiteWise エッジゲートウェイの BIOS 設定を次のように変更します。
   a. 該当する場合、潜在的な停電後に SiteWise Edge ゲートウェイが自動的に再起動することを確認します。
   b. 該当する場合は、 SiteWise エッジゲートウェイが休止またはスリープ状態にならないようにしてください。
2. SiteWise Edge ゲートウェイがインターネットに接続されていることを確認します。
3. (オプション) マウス、キーボード、モニターを使用せずに SiteWise Edge ゲートウェイを使用するには、次の手順を実行して SiteWise Edge ゲートウェイsshで を設定します。
   a. SSH パッケージをまだインストールしていない場合は、次のコマンドを実行します。

      ```
      sudo apt install ssh
      ```

   b. 以下のコマンドを実行します。

      ```
      service ssh status
      ```

c. SSH サーバーが実行されていることを確認するには、出力で `Active: active (running)` を検索します。

d. Q を押して終了します。

次のコマンドを実行して、SSH を使用して別のコンピュータから SiteWise Edge ゲートウェイに接続します。*`username`* をユーザーログインに置き換え、*`IP`* を SiteWise Edge ゲートウェイの IP アドレスに置き換えます。

```
ssh username@IP
```

`-p` *`port-number`* 引数を使用して、デフォルトのポート 22 以外のポートに接続できます。

4. AWS IoT Greengrass Core ソフトウェア v1.10.2 以降をダウンロードしてインストールし、SiteWise エッジゲートウェイの AWS IoT Greengrass グループを作成します。これを行うには、[AWS IoT Greengrass Developer Guide] (デベロッパーガイド) の [Getting started with AWS IoT Greengrass] (始めるにあたって) の指示に従ってください。

   すぐに開始するには、AWS IoT Greengrass デバイスセットアップスクリプトを実行することをお勧めします。要件とプロセスをより詳細に確認 AWS IoT Greengrass したい場合は、モジュール 1 とモジュール 2 の手順を実行して をセットアップできます AWS IoT Greengrass。

   > **Important**
   >
   > AWS IoT SiteWise がサポートされているAWS リージョンを確認します。のリージョンを選択するときは AWS IoT Greengrass、そのリージョンが もサポートしていることを確認してください AWS IoT SiteWise。それ以外の場合は、 SiteWise エッジゲートウェイを に接続できません AWS IoT SiteWise。

   次のステップに進む前に、 SiteWise エッジゲートウェイに AWS IoT Greengrass Core ソフトウェアがインストールされている必要があります。

5. 次のコマンドを実行して Java 8 をインストールします。

```
sudo apt update
      sudo apt install openjdk-8-jre
```

このガイドの後半でインストールする SiteWise Edge ゲートウェイソフトウェアは、Java 8 ランタイムを使用します。

6. Java が正常にインストールされたことを確認するには、次のコマンドを実行します。

```
java -version
```

7. AWS IoT Greengrass Core ソフトウェアは`java8`ディレクトリを引き受けます。次のコマンドを実行して、Java インストールをその `java8` ディレクトリにリンクします。

```
sudo ln -s /usr/bin/java /usr/bin/java8
```

8. 次のコマンドを実行して`/var/sitewise`データディレクトリを作成し、そのディレクトリの`ggc_user`アクセス許可を付与します。 は、このディレクトリにデータ AWS IoT SiteWise を保存します。この手順の前 AWS IoT Greengrass 半で をセットアップ`ggc_user`したときに を作成しました。

```
sudo mkdir /var/sitewise
      sudo chown ggc_user /var/sitewise
      sudo chmod 700 /var/sitewise
```

`/var/sitewise` は、 が AWS IoT SiteWise 使用するデフォルトのディレクトリです。ディレクトリパスはカスタマイズできますが (例えば、 を `/var/sitewise`に置き換えます`/var/`*`custom`*`/`*`path`*`/`) 、これを行うには SiteWise Edge ゲートウェイの作成後に追加のステップが必要です。詳細については、AWS IoT SiteWise コネクタの設定 のステップ 6 を参照してください。

9. 必要に応じて、以下のエンドポイントとポートをローカルネットワークの許可リストに追加するように IT 管理者に依頼してください。

   - ポート: 443、8443、および 8883

> **Important**
>
> すべてのネットワーク通信にポート 443 のみを使用するように AWS IoT Greengrass Core を設定できます。詳細については、[AWS IoT Greengrass Developer Guide] (デベロッパーガイド) の[Connect on port 443 or through a network proxy] (ポート443またはネットワークプロキシ経由での接続) を参照してください。

- SiteWise Edge ゲートウェイ (ポート 443) の IP アドレス。IP アドレスを取得するには、`ip address` または `ifconfig` コマンドを実行し、`inet` 値 (`203.0.113.0` など) を書き留めます。
- AWS IoT SiteWise データエンドポイント: `data.iotsitewise.`*`region`*`.amazonaws.com` (ポート 443)。
- SiteWise Edge ゲートウェイが使用する次の AWS エンドポイント。これらは /*`greengrass-root`*/config/config.json ファイルにあります。*`greengrass-root`* を AWS IoT Greengrass インストールのルートに置き換えます。
  - ggHost: `greengrass-ats.iot.`*`region`*`.amazonaws.com` (ポート 443、8443、および 8883)。
  - iotHost: *`prefix`*`-ats.iot.`*`region`*`.amazonaws.com` (ポート 443、8443、および 8883)。

  詳細については、「AWS IoT Greengrass エンドポイントとクォータ」を参照してください。

10. AWS IoT Greengrass Core ソフトウェアがまだ実行されていない場合は、次のコマンドを実行して AWS IoT Greengrass Core ソフトウェアを起動します。*`greengrass-root`* をインストールのルートに置き換えます AWS IoT Greengrass 。デフォルトの *`greengrass-root`* は /`greengrass` です。

    ```
    cd /greengrass-root/ggc/core
          sudo ./greengrassd start
    ```

    次のメッセージが表示されます。`Greengrass successfully started with PID:` *`some-PID-number`*

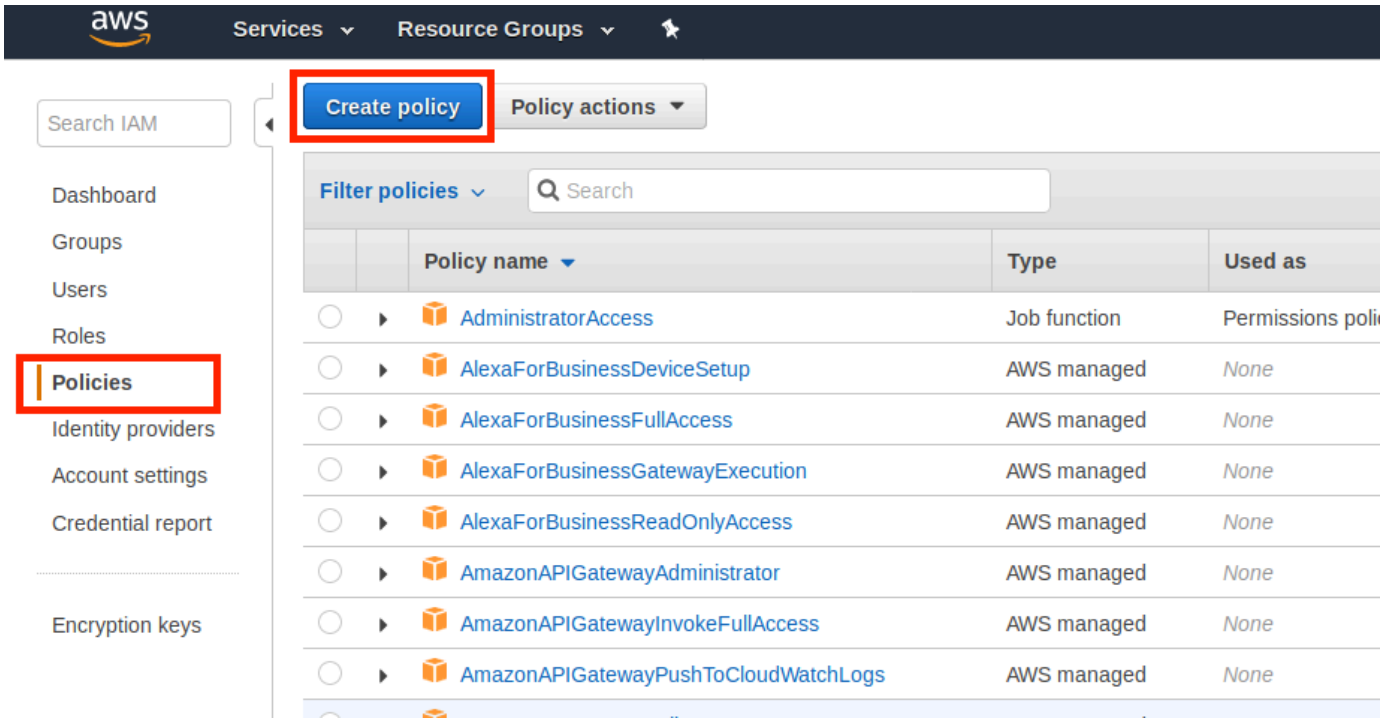
11. SiteWise Edge ゲートウェイがオンになったときに自動的に起動するように AWS IoT Greengrass Core ソフトウェアを設定します。 SiteWise Edge ゲートウェイのオペレーティングシステムのドキュメントを参照してください。

IAMポリシーとロールを作成する。

Edge ゲートウェイが AWS IoT SiteWise ユーザーに代わって にアクセスできるようにするには、AWS Identity and Access Management (IAM) SiteWise ポリシーとロールを作成する必要があります。

IAM ポリシーとロールを作成するには。

1. [IAM console] (IAM コンソール) に入ります。
2. ナビゲーションペインで ポリシーを選択してから ポリシーの作成を選択します。



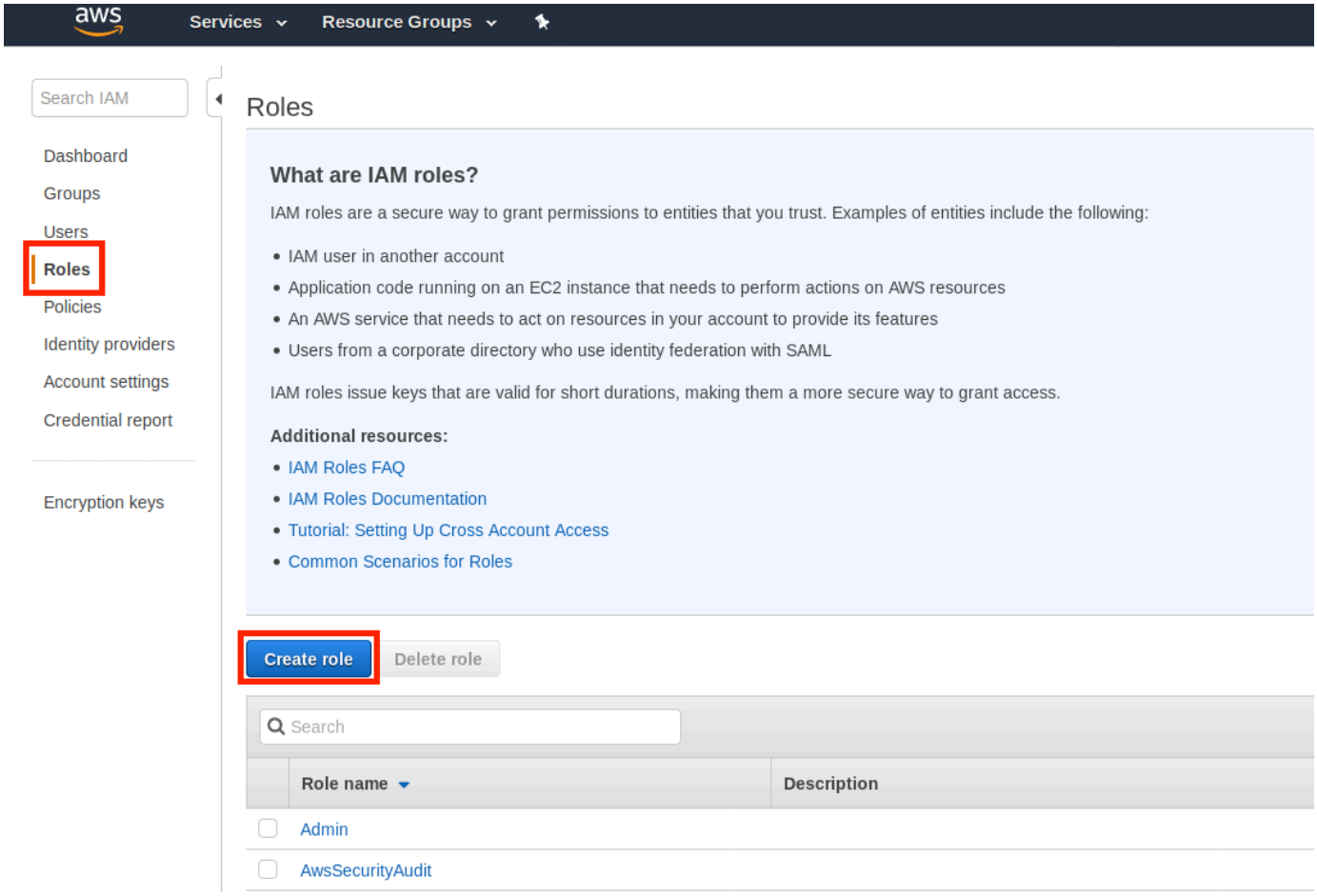
3. [JSON] タブで、ポリシーフィールドの現在の内容を削除し、以下のポリシーをフィールドに貼り付けます。

```
{
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": "iotsitewise:BatchPutAssetPropertyValue",
            "Resource": "*"
          }
        ]
      }
```

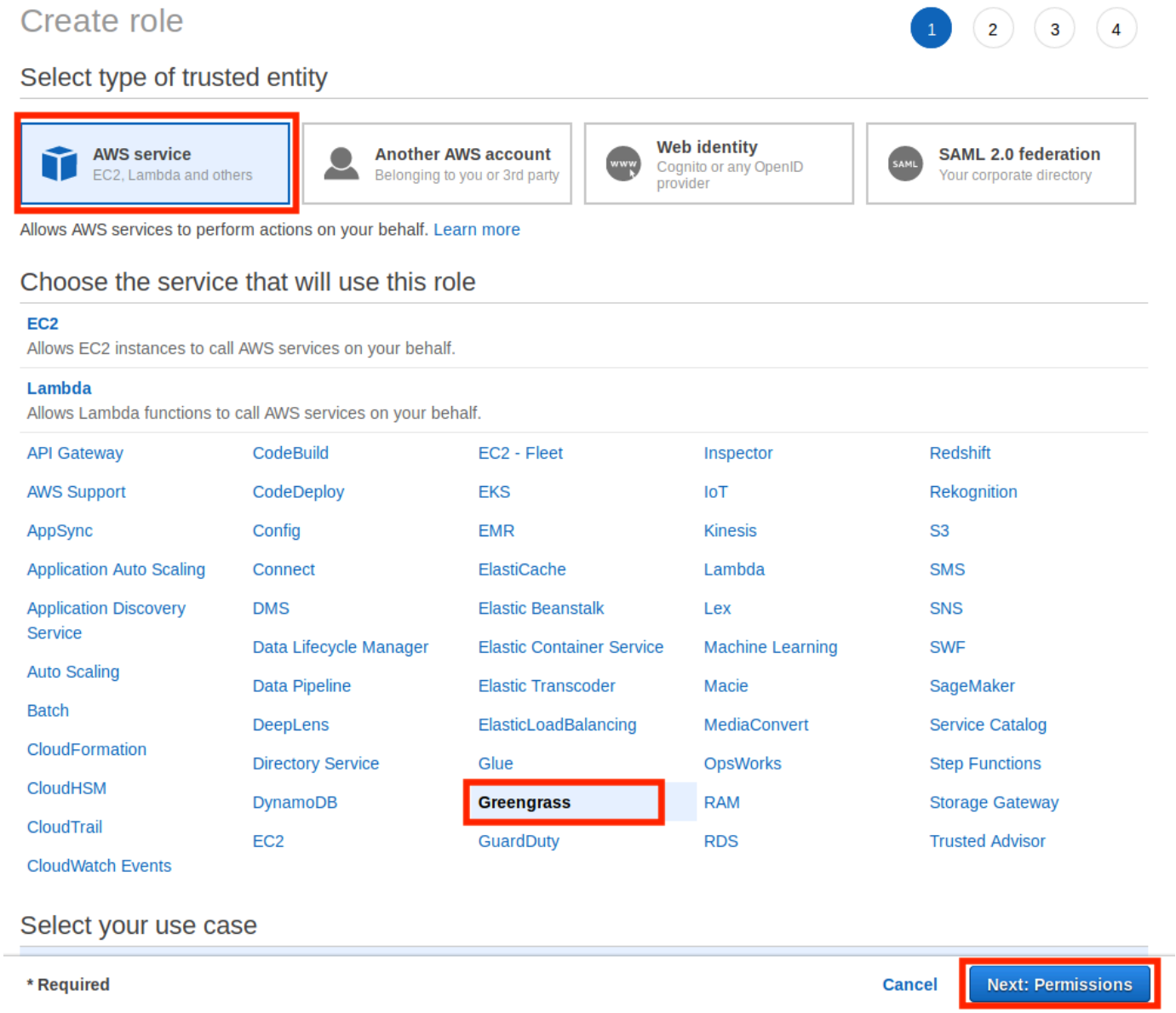
> **Note**
>
> セキュリティを向上させるには、 `Condition`プロパティで AWS IoT SiteWise アセット階層パスを指定できます。次の例は、アセット階層パスを指定する信頼ポリシーです。

```
{
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": "iotsitewise:BatchPutAssetPropertyValue",
            "Resource": "*",
            "Condition": {
              "StringLike": {
                "iotsitewise:assetHierarchyPath": [
                  "/root node asset ID",
                  "/root node asset ID/*"
                ]
              }
            }
          }
        ]
      }
```

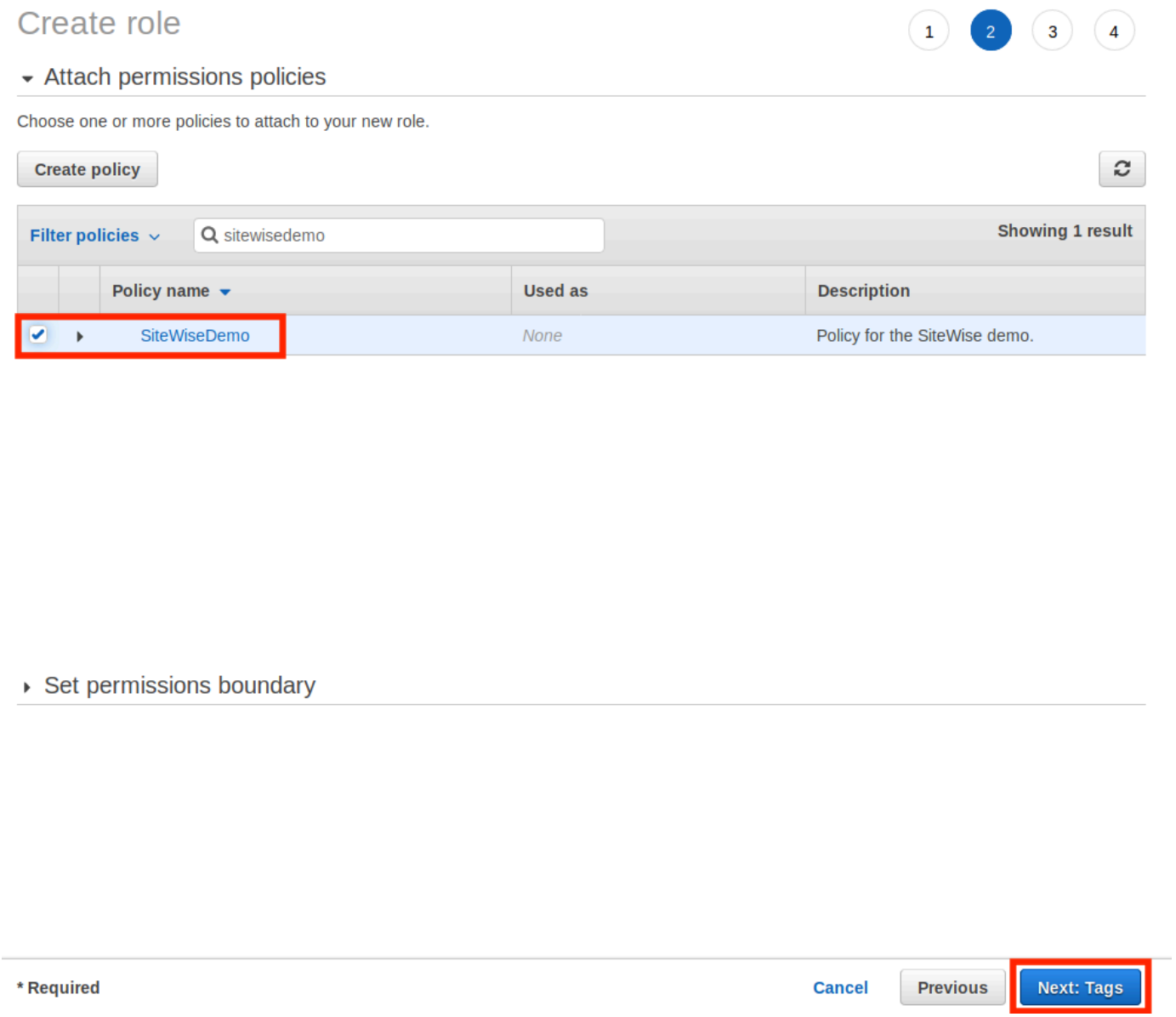
4. [ポリシーの確認] を選択します。
5. ポリシーの名前と説明を入力し、[ポリシーの作成] を選択します。
6. ナビゲーションペインで ロール を選択してから、ロールを作成する を選択します。



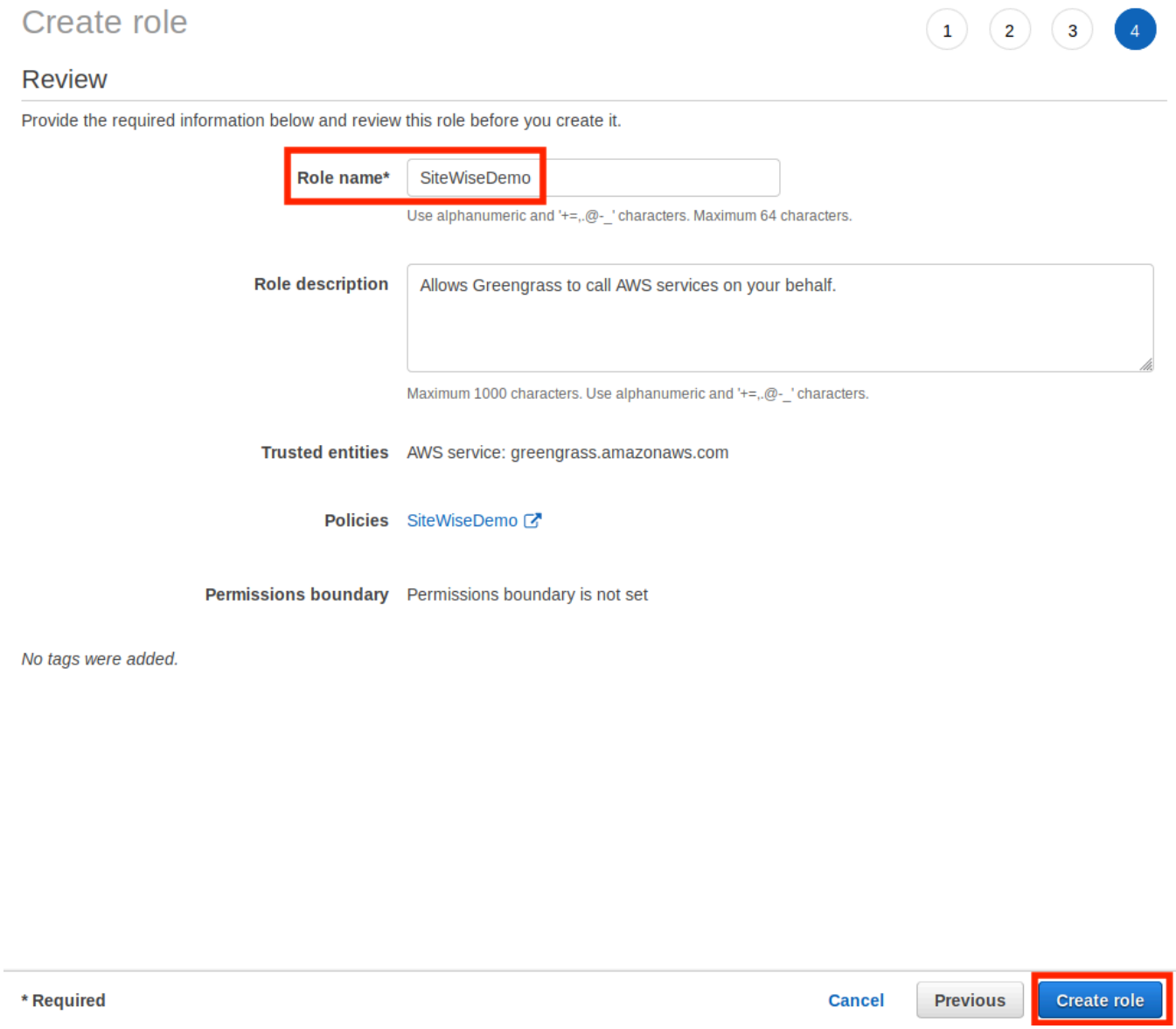
7. [Select type of trusted entity] (信頼されたエンティティの種類を選択) の下で、[AWS service] (サービス) を選択します。[このロールを使用するサービスを選択] で、[Greengrass] を選択し、[次の手順: アクセス許可] を選択します。



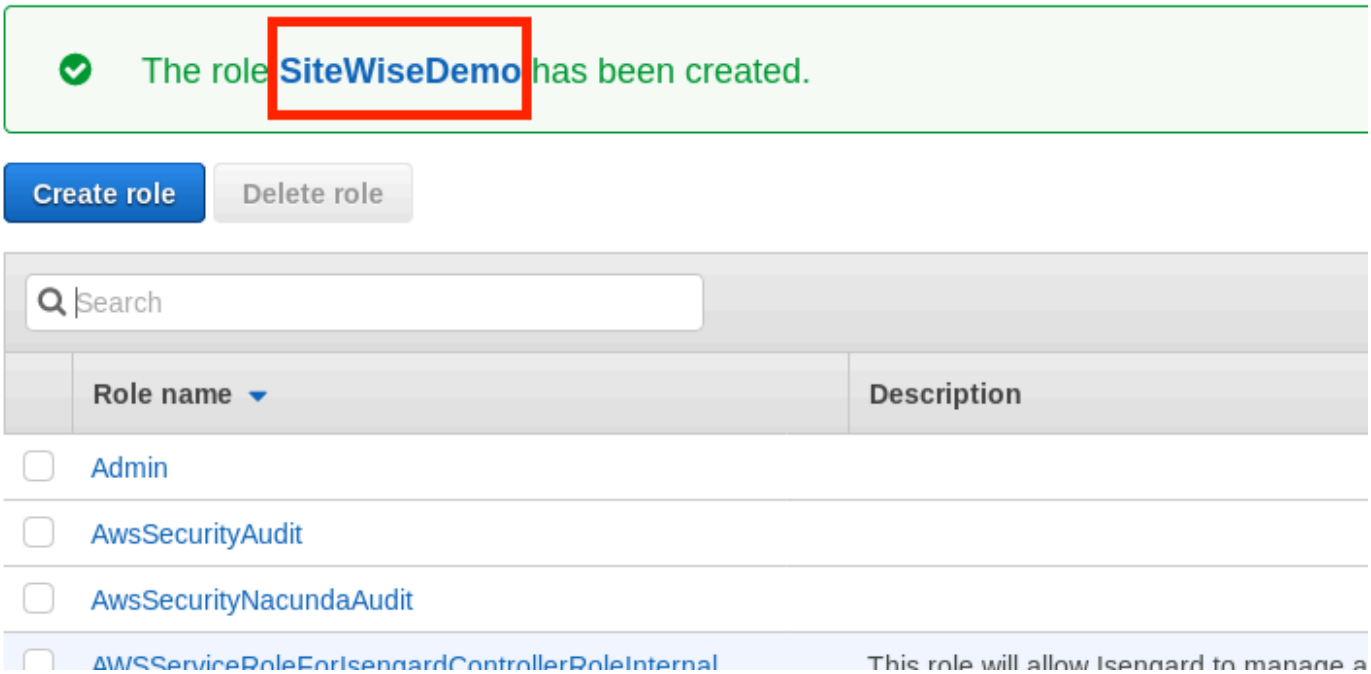
8. 作成したポリシー を検索し、チェックボックスをオンにして、[Next: Tags] (次のステップ: タグ) を選択します。



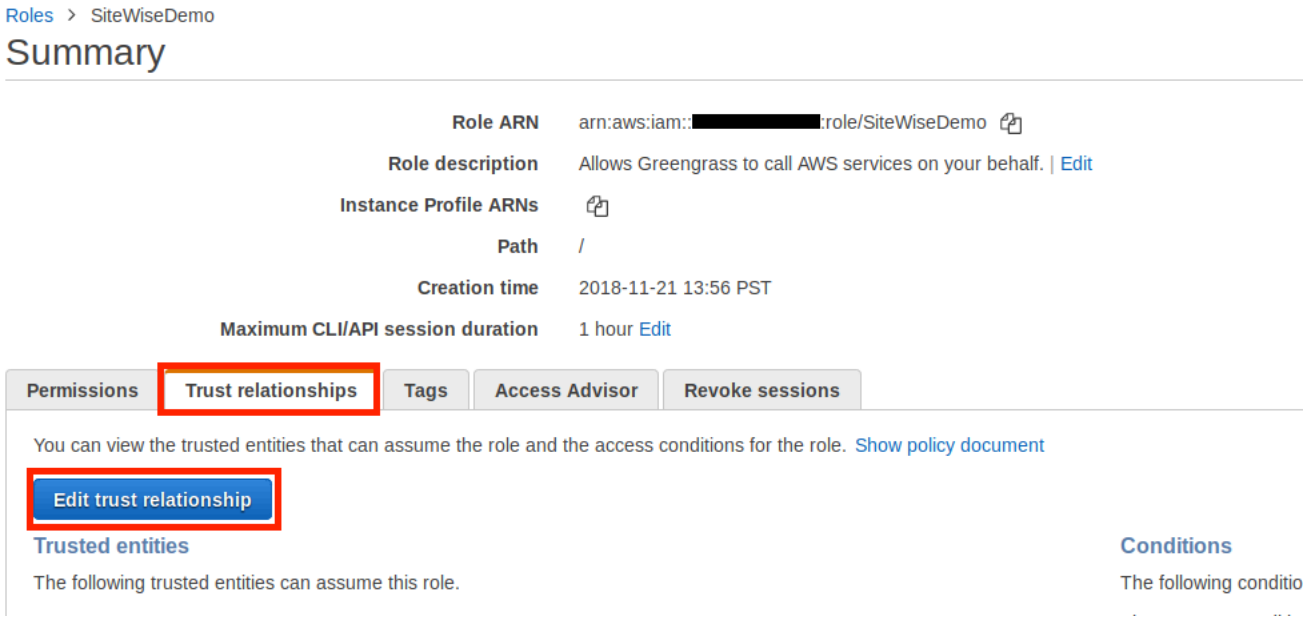
9. (オプション) ロールにタグを追加し、[Next: Review (次のステップ: 確認)] を選択します。
10. ロールの名前と説明を入力し、[Create role (ロールの作成)] を選択します。



11. 緑色のバナーで、新しいロールへのリンクを選択します。検索フィールドを使用してロールを検索することもできます。



12. [Trust relationships] タブを選択し、続いて [Edit trust relationship] を選択します。



13. ポリシーフィールドの現在の内容を次のように置き換え、[Update Trust Policy (信頼ポリシーの更新)] を選択します。

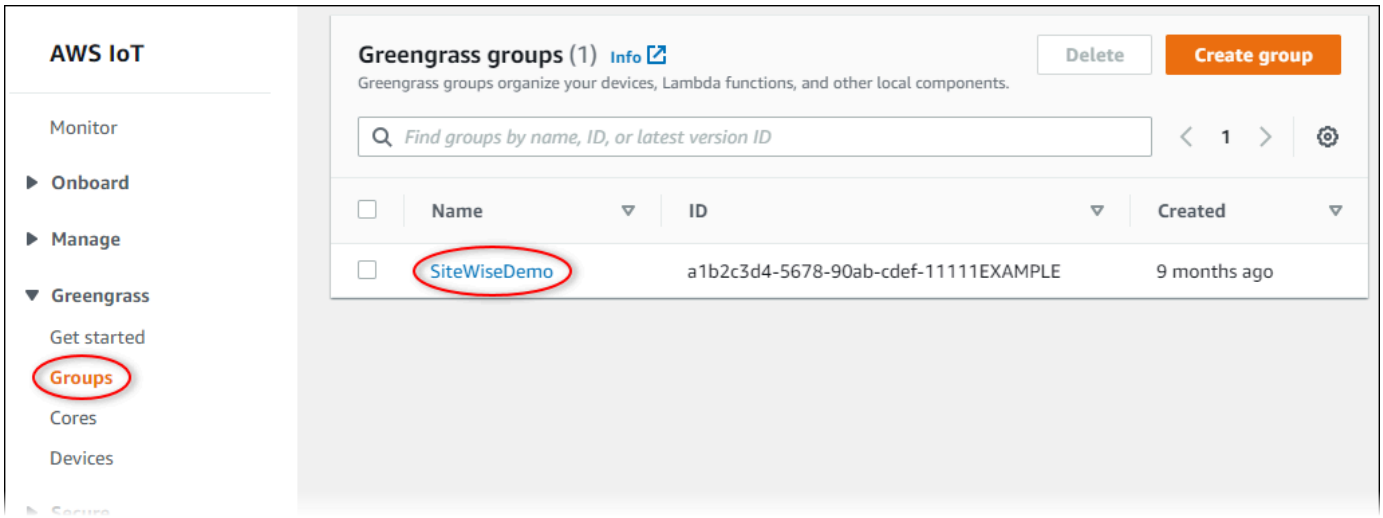
```
{
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
```

```
            "Principal": {
              "Service": "greengrass.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      }
```

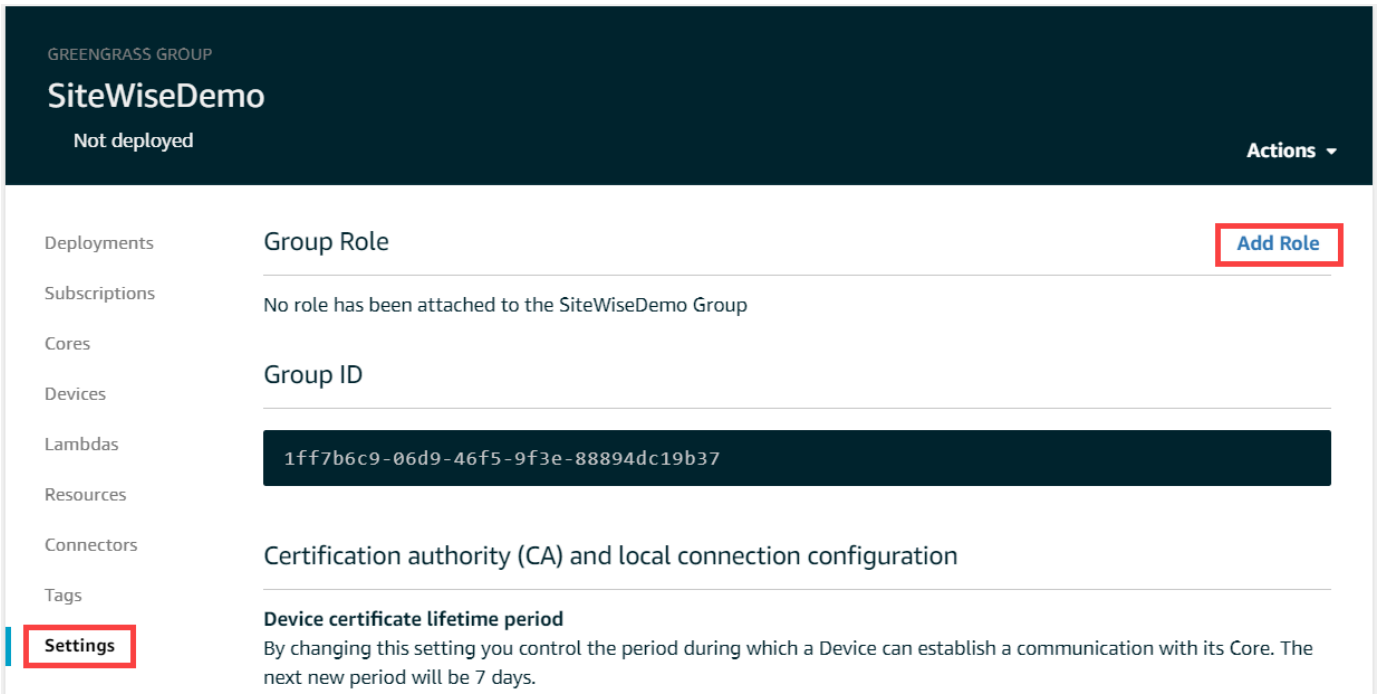
AWS IoT Greengrass グループの設定

グループに IAM ロールをアタッチし、ストリームマネージャーを有効にするには

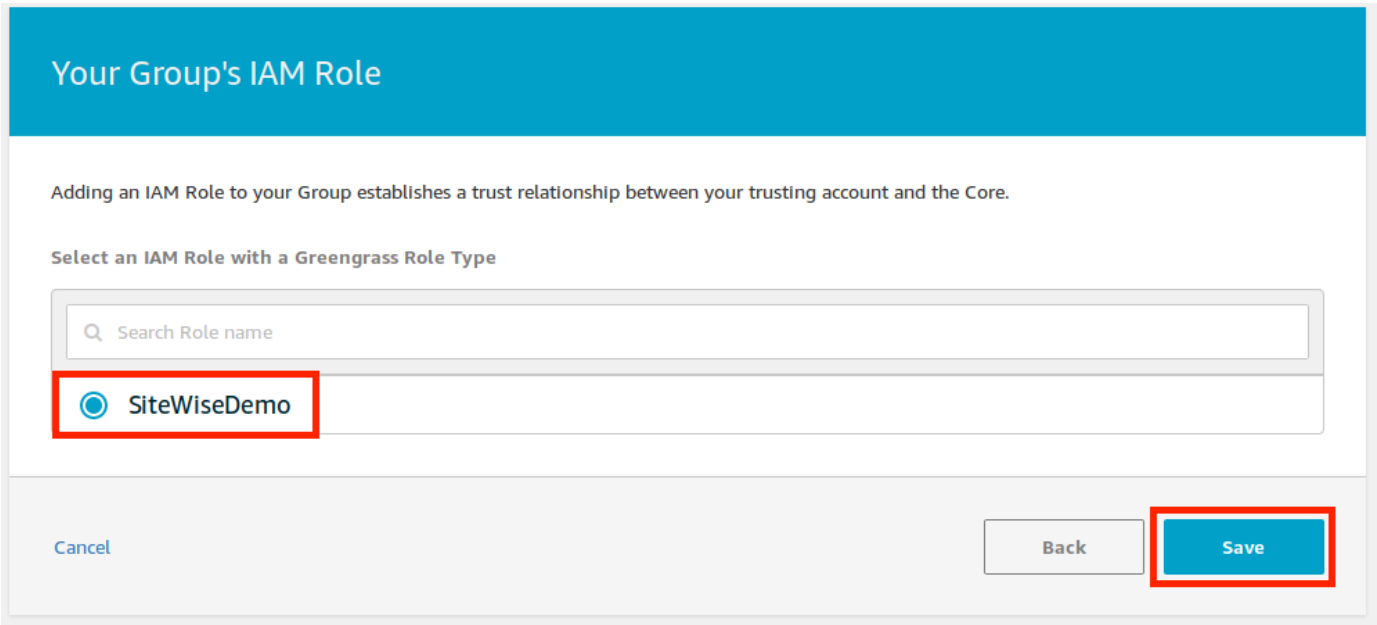
1. AWS IoT Greengrass コンソールに移動します。
2. 左のナビゲーションペインの、[Greengrass] で、[グループ] を選択し、SiteWise Edge ゲートウェイ環境のセットアップ で作成したグループを選択します。



3. 左側のナビゲーションペインで [設定] を選択します。[グループのロール] セクションで、[ロールの追加] を選択します。



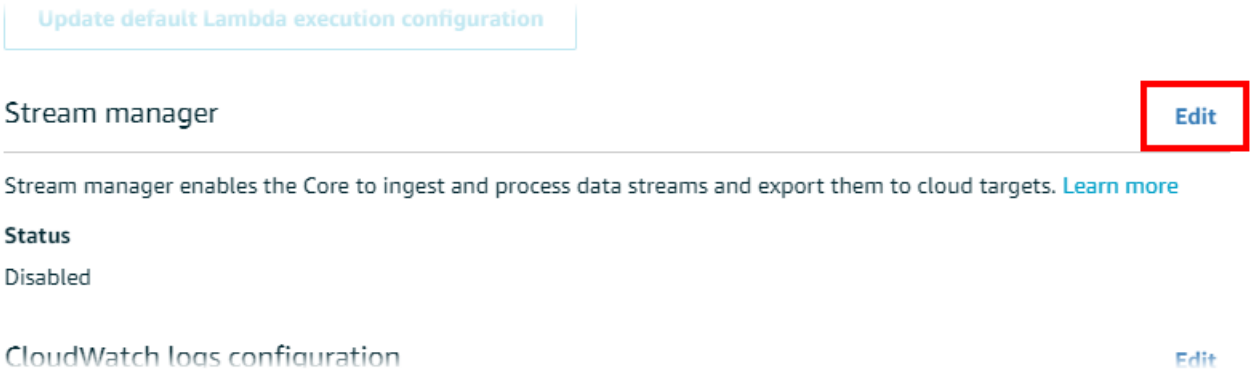
4. IAMポリシーとロールを作成する。で作成したロールを選択し、[Save (保存)] を選択します。



5. [設定] ページの [Stream manager (ストリームマネージャー)] セクションで [編集] を選択します。

   ストリームマネージャーは、 AWS IoT Greengrass Core AWS IoT Greengrass がデータを AWS Cloud. SiteWise Edge ゲートウェイにストリーミングできるようにする の機能です。エッジゲートウェイでは、ストリームマネージャーを有効にする必要があります。詳細については、「

AWS IoT Greengrass Version 1 デベロッパーガイド」の AWS IoT Greengrass 「Core でデータストリームを管理する」を参照してください。



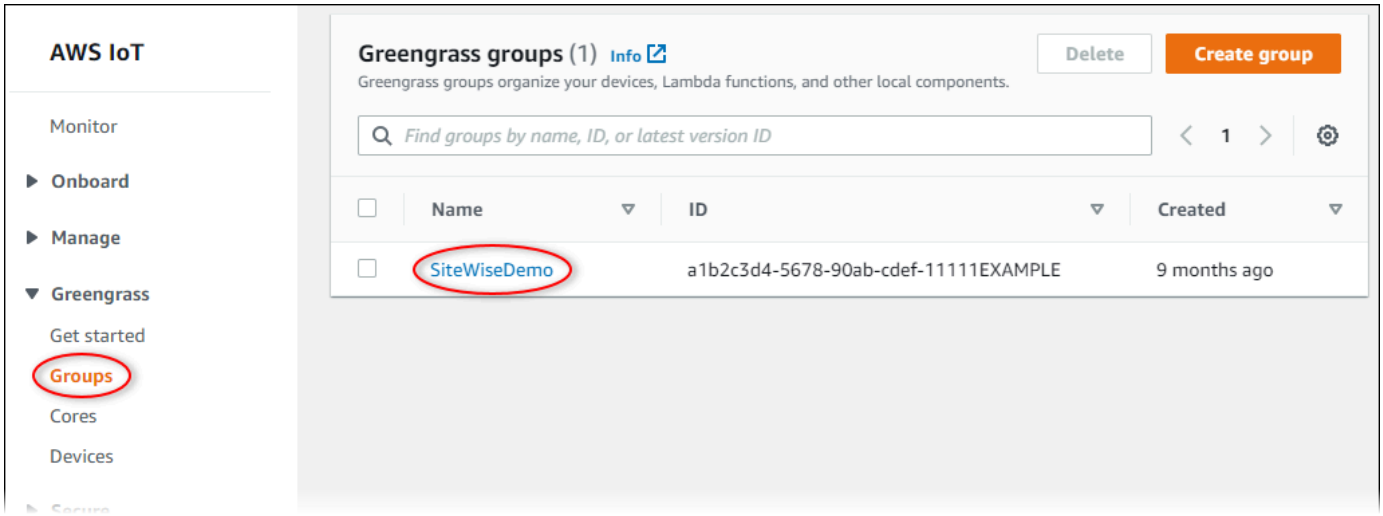
6. [Enabled (有効化)] を選択し、[保存] を選択します。
7. 左上隅で、[Services (サービス)] を選択して次のステップの準備をします。

AWS IoT SiteWise コネクタの設定

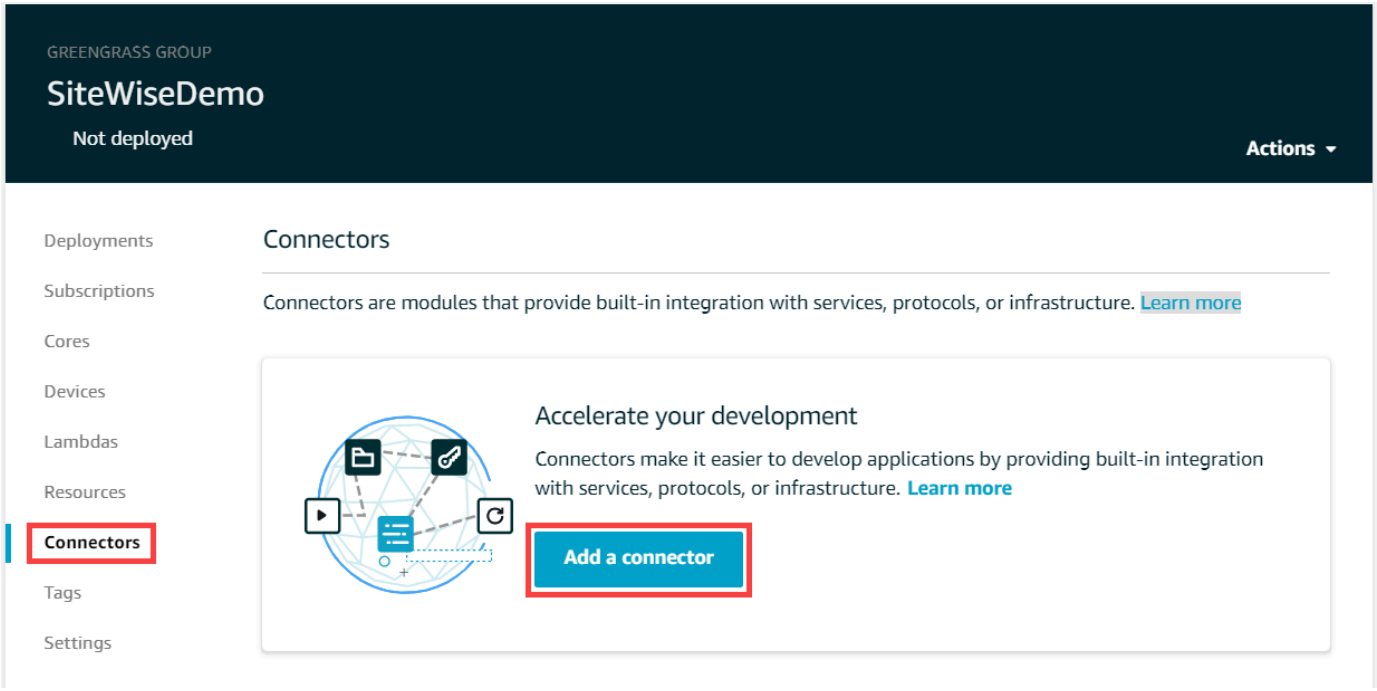
この手順では、Greengrass グループに AWS IoT SiteWise コネクタを設定します。コンポーネントは、一般的なエッジシナリオの開発ライフサイクルを高速化する構築済みのモジュールです。詳細については、[AWS IoT Greengrass Version 1 Developer Guide] (デベロッパーガイド) の[AWS IoT Greengrass connectors] (コネクター) を参照してください。

AWS IoT SiteWise コネクタを設定するには

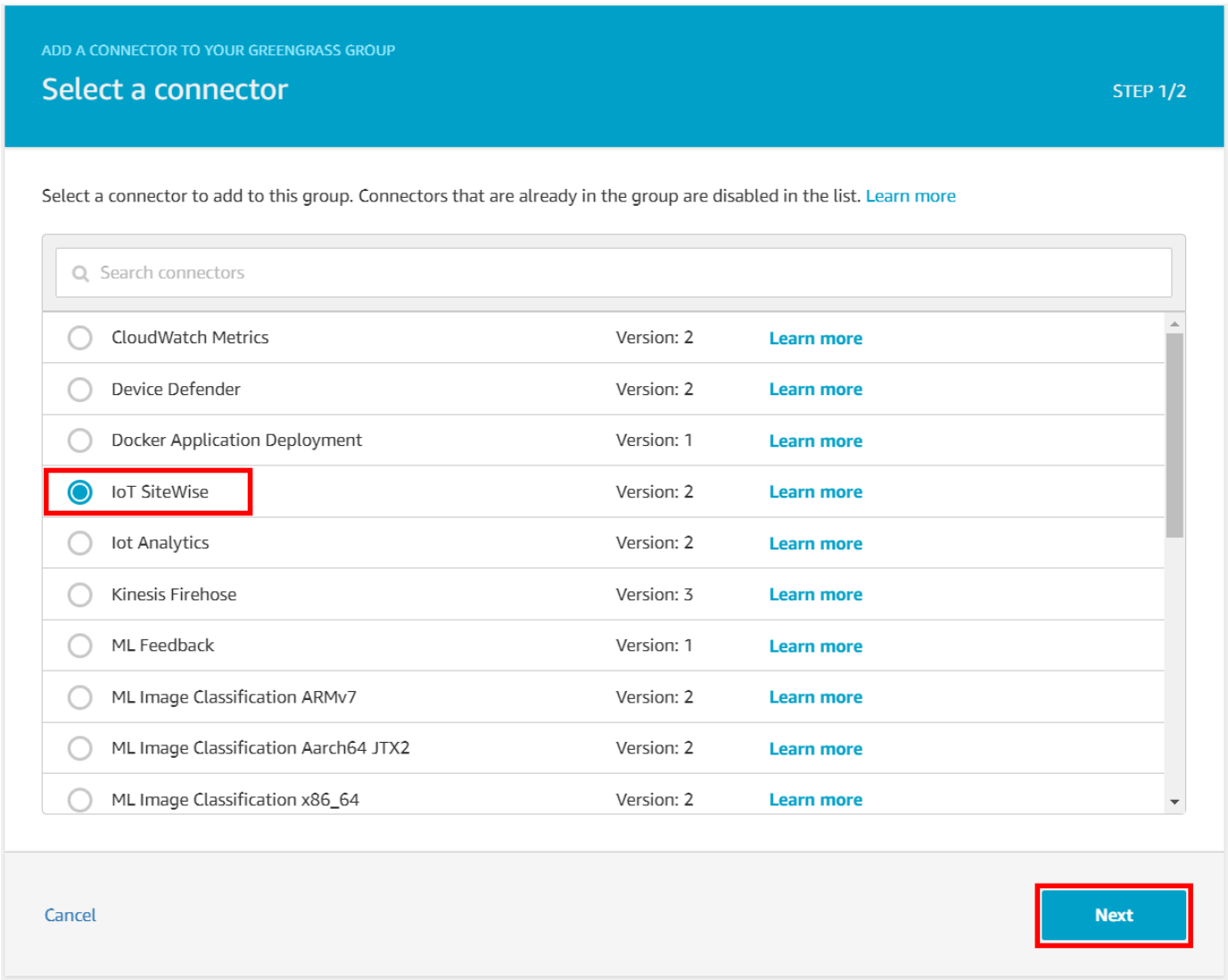
1. AWS IoT Greengrass コンソールに移動します。
2. 左のナビゲーションペインの、[Greengrass] で、[グループ] を選択し、SiteWise Edge ゲートウェイ環境のセットアップ で作成したグループを選択します。



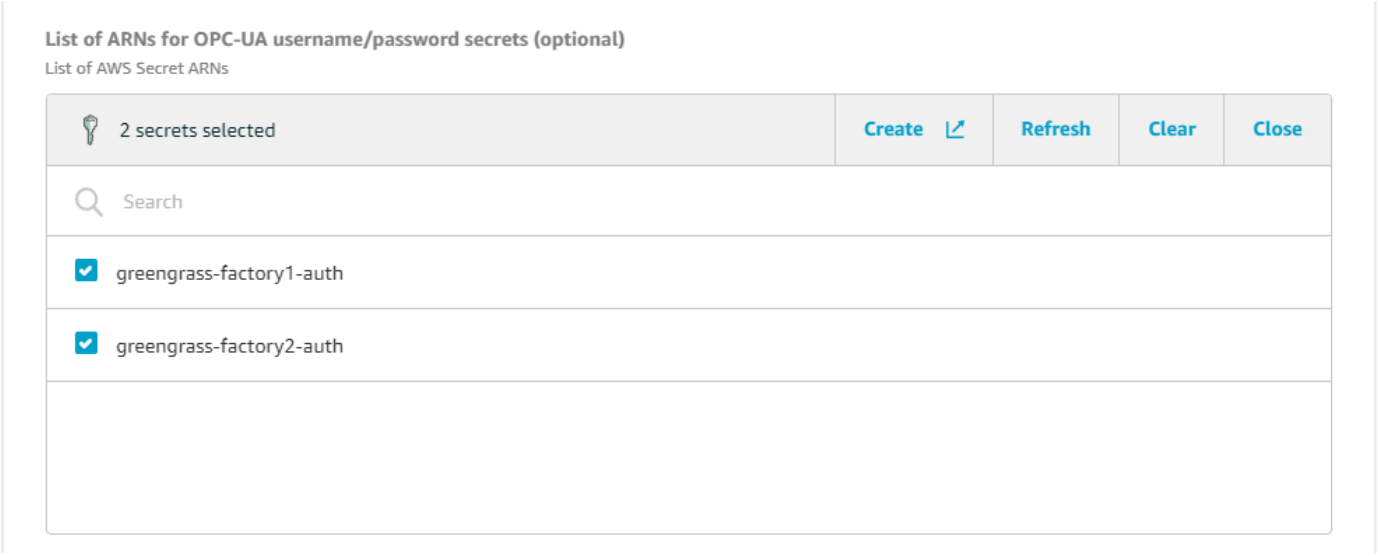
3. 左側のナビゲーションページで、[Connectors (コネクタ)] を選択します。[Connectors (コネクタ)] ページで、[Add a connector (コネクタの追加)] を選択します。



4. リストから IoT SiteWise を選択し、次へ を選択します。



5. サーバーで認証が必要な場合は、サーバーのユーザー名とパスワードを使用して AWS Secrets Manager シークレットを作成できます。そして、それぞれの秘密を Greengrass グループにアタッチし、[List of ARNs for username/password secrets] (ユーザーネーム/パスワードの秘密の ARN リスト) で選択することができます。シークレットを作成および設定する方法の詳細については、「ソース認証の設定」を参照してください。後でコネクタにシークレットを追加することもできます。



6. とは異なるパスで SiteWise Edge ゲートウェイを設定する場合は`/var/sitewise`、ローカルストレージパス にそのパスを入力します。
7. (オプション) コネクタの最大ディスクバッファサイズを入力します。 AWS IoT Greengrass コアが AWS クラウドへの接続を失った場合、コネクタは正常に接続できるようになるまでデータをキャッシュします。キャッシュサイズが最大ディスクバッファサイズを超えると、コネクタはキューから最も古いデータを破棄します。
8. [追加] を選択します。
9. ページの右上隅の [Actions (アクション)] メニューで、[Deploy (デプロイ)] を選択します。
10. [Automatic detection (自動検出)] を選択して、デプロイを開始します。

    デプロイが失敗した場合は、再度[デプロイ] を選択します。デプロイが失敗し続ける場合は、「AWS IoT Greengrass デプロイのトラブルシューティング」を参照してください。

Edge SiteWise ゲートウェイを に追加する AWS IoT SiteWise

この手順では、 SiteWise エッジゲートウェイの Greengrass グループを に追加します AWS IoT SiteWise。 SiteWise Edge ゲートウェイを に登録すると AWS IoT SiteWise、サービスはデータソース設定を SiteWise Edge ゲートウェイにデプロイできます。

SiteWise Edge ゲートウェイを に追加するには AWS IoT SiteWise

1. AWS IoT SiteWise コンソールに移動します。
2. [Add gateway (ゲートウェイの追加)] を選択します。
3. SiteWise ゲートウェイの追加ページで、次の操作を行います。

a. SiteWise Edge ゲートウェイの名前を入力します。Edge SiteWise ゲートウェイの場所を名前に含めることを検討して、簡単に識別できるようにします。

b. [Greengrass group ID] (Greengrassグループ ID) は、先ほど作成したGreengrassグループを選択します。

Example

c. (オプション) [Edge capabilities] (Edge機能) については、[Data processing pack] (データ処理パック) を選択します。これにより、 SiteWise エッジゲートウェイと、エッジ用に設定されたアセットモデルおよびアセットとの通信が可能になります。詳細については、「the section called “エッジデータ処理を有効にする。”」を参照してください。

> **Important**
>
> データ処理パックを SiteWise Edge ゲートウェイに追加する場合は、 AWS IoT Greengrass グループで SiteWise Edge コネクタを設定してデプロイする必要があります。次のステップに進んでください。

d. [Add gateway (ゲートウェイの追加)] を選択します。

4. SiteWise Edge ゲートウェイにデータ処理パックを追加する場合は、 AWS IoT Greengrass グループで AWS IoT SiteWise データ処理コネクタを設定してデプロイします。「」の手順に従ってthe section called “AWS IoT SiteWise コネクタの設定”、 AWS IoT SiteWise データ処理コネクタを設定します。

   a. AWS IoT Greengrass コンソールでコネクタを選択する で、「データ処理AWS IoT SiteWise 」を選択します。
   b. ローカルストレージパス には、 SiteWise エッジゲートウェイへのパスを入力します。
   c. [追加] を選択します。
   d. 右上の[Actions] (アクション) メニューから[Deploy] (デプロイ) を選択し、[Automatic detection] (自動検出) を選択してデプロイを開始します。

SiteWise Edge ゲートウェイがデプロイされたら、 SiteWise エッジゲートウェイがデータを取り込む産業機器ごとにソースを追加できます。詳細については、「データソースの設定」を参照してください。

Amazon CloudWatch メトリクスを表示して、 SiteWise エッジゲートウェイが に接続していることを確認できます AWS IoT SiteWise。詳細については、「AWS IoT Greengrass Version 1 ゲートウェイメトリクス」を参照してください。

AWS IoT Greengrass V1 SiteWise Edge ゲートウェイでのデータソースの設定

AWS IoT SiteWise Edge ゲートウェイを設定したら、 SiteWise エッジゲートウェイがローカル産業機器から にデータを取り込むことができるようにデータソースを設定できます AWS IoT SiteWise。各ソースは、 SiteWise エッジゲートウェイが接続して産業用データストリームを取得する OPC-UA サーバーなどのローカルサーバーを表します。 SiteWise Edge ゲートウェイの設定の詳細については、「」を参照してくださいAWS IoT Greengrass V1 SiteWise Edge ゲートウェイの設定。

> **Note**
>
> AWS IoT SiteWise は、ソースを追加または編集するたびに SiteWise Edge ゲートウェイを再起動します。 SiteWise Edge ゲートウェイは、再起動中にデータを取り込みません。 SiteWise Edge ゲートウェイを再起動する時間は、 SiteWise Edge ゲートウェイのソースのタグの数によって異なります。再起動時間の範囲は、数秒 (タグ数が少ない SiteWise エッジゲートウェイの場合) から数分 (タグ数が多い SiteWise エッジゲートウェイの場合) です。

ソースを作成したら、データストリームをアセットプロパティに関連付けることができます。アセットの作成および使用方法の詳細については、産業用アセットのモデリング と アセットプロパティへの産業データストリームのマッピング を参照してください。

CloudWatch メトリクスを表示して、データソースが に接続されていることを確認できます AWS IoT SiteWise。詳細については、「AWS IoT Greengrass Version 1 ゲートウェイメトリクス」を参照してください。

現在、 は、次のデータソースプロトコル AWS IoT SiteWise をサポートしています。

- OPC-UA – 産業オートメーション用の machine-to-machine (M2M) 通信プロトコル。
- [Modbus TCP] - プログラマブルロジックコントローラ (PLC) とのインターフェースに使用されるデータ通信プロトコル。
- [Ethernet/IP (EIP)] (イーサネット/IP (EIP)) - 制御用通信プロトコル (CIP/Common Industrial Protocol) を標準イーサネットに適応させた産業用ネットワークプロトコルです。

> **Note**
>
> SiteWise AWS IoT Greengrass V2 現在、 で実行されているエッジゲートウェイは、Modbus TCP およびイーサネット IP ソースをサポートしていません。

トピック

- Modbus TCP ソースを設定する。
- イーサネット/IP (EIP) ソースを設定する。
- ソース認証の設定
- コネクタのアップグレード

Modbus TCP ソースを設定する。

AWS IoT SiteWise コンソールまたは AWS IoT SiteWise Edge ゲートウェイ機能を使用して、Modbus TCP ソースを定義し、 SiteWise Edge ゲートウェイに追加できます。このソースは、ローカルの Modbus TCP サーバーを表します。

> **Note**
>
> - SiteWise AWS IoT Greengrass V2 現在、 で実行されているエッジゲートウェイは Modbus TCP ソースをサポートしていません。
> - Modbus TCP ソースを使用するには、 AWS IoT SiteWise コネクタをインストールする必要があります。

Modbus TCP ソースを使用して、 SiteWise エッジゲートウェイで受信されたデータ型をソースから別のデータ型に変換できます。ソースデータ型は、デスティネーションデータに選択できるデータ型を決定します。また、Modbus TCPのソースを使用してバイトを交換することも可能です。次の表は、互換性のあるソースデータ型、デスティネーションデータ型、およびスワップモードの詳細についてです。

スワップ・モードの詳細については、Modbus メッセージのエンコーディングに関する記事[How Real (Floating Point) and 32-bit Data is Encoded in Modbus RTU Messages] (Modbus RTU メッセージにおけるリアル (浮動小数点) および 32 ビットデータのエンコード方法) を参照してください。

出典データ型	対応するデスティネーションデータ型。	互換性のあるスワップモード。	互換性のあるコネクタのバージョン。
ASCII	文字列	NoSwap	2
UTF8	文字列	NoSwap	2
ISO8859	文字列	NoSwap	2
Int16	整数、 2 倍、文字列	noSwap	1 および 2
Int32	整数、 2 倍、文字列	noSwap 、byteWordSwap、byteSwap 、wordSwap	1 および 2
浮動小数点数	2 倍、文字列	noSwap 、byteWordSwap、byteSwap 、wordSwap	1 および 2

出典データ型	対応するデスティネーションデータ型。	互換性のあるスワップモード。	互換性のあるコネクタのバージョン。
ブール値	ブール値	noSwap	1 および 2
16 進ダンプ	文字列	noSwap	1 および 2

トピック

- Modbus TCP ソースを設定する (コンソール)。
- Modbus TCP ソース (CLI) を設定する。

Modbus TCP ソースを設定する (コンソール)。

Modbus TCP ソースを設定するには。

1. AWS IoT SiteWise コンソールに移動します。
2. 左のナビゲーションペインで [ゲートウェイ] を選択します。
3. ソースを作成する SiteWise Edge ゲートウェイで、 の管理 を選択し、詳細の表示 を選択します。
4. 右上隅の [New source (新しいソース)] を選択します。
5. [Protocol options] (プロトコルオプション) は、[Modbus TCP] を選択します。
6. [Modbus TCP source configuration] (Modbus TCP ソース設定) の場合、ソースの [Name] (名前) を入力します。
7. [IP address] (IP アドレス) には、データソースサーバーの IP アドレスを入力します。
8. (オプション) ソースサーバーの[Port] (ポート) および[Unit ID] (ユニット ID) を入力します。
9. (オプション) [Minimum inter-request duration] (最小要求間隔) には、サーバーに送信される後続の要求間の時間間隔を入力します。 SiteWise Edge ゲートウェイは、デバイスと登録数に基づいて最小許容間隔を自動的に計算します。
10. [Property groups] (プロパティグループ) の場合は、[Name] (名前) を入力します。
11. [Properties] (プロパティ) の場合。
    a. [Tag] (タグ) には、登録セットのプロパティエイリアスを入力します。例えば **`TT-001`** です。

b. [Register address] (登録アドレス) には、登録セットを開始する登録のアドレスを入力します。

c. [Source data type] (ソースデータ型) では、データを変換する Modbus TCP データ型を選択します。デフォルトは[Hex dump] (16 進ダンプ) です。

> **Note**
>
> 選択したソースデータ型によって、選択できるデータサイズ、デスティネーションデータ型、スワップモードが決まります。詳細については、「the section called “Modbus TCP ソースを設定する。”」を参照してください。

d. [Data size] (データサイズ) には、[Register address] (登録アドレス) から読み出す場合の登録の数を入力します。これは、このソースに選択したソースデータ型によって決まります。

e. 送信先データ型 で、 AWS IoT SiteWise データを変換するデータ型を選択します。デフォルト値は [String] (文字列) です。デスティネーション型は、このソースに選択したソースデータ型と互換性がある必要があります。詳細については、「the section called “Modbus TCP ソースを設定する。”」を参照してください。

f. [Swap model] (スワップモード) では、登録セットからデータを読み出すために使用するデータスワップモードを選択します。スワップ・モードは、このソースに選択したソース・データ型と互換性がある必要があります。詳細については、「the section called “Modbus TCP ソースを設定する。”」を参照してください。

12. スキャンレート では、 SiteWise エッジゲートウェイが registers を読み取る速度を更新します。 は SiteWise 、エッジゲートウェイの最小許容スキャンレート AWS IoT SiteWise を自動的に計算します。

13. (オプション) [Destination] (デスティネーション) では、ソースデータのデスティネーションを選択します。デフォルトでは、ソースはデータを に送信します AWS IoT SiteWise。代わりに、AWS IoT Greengrass ストリームを使用してローカルの送信先または AWS クラウドにデータをエクスポートできます。

> **Note**
>
> を使用してエッジでこのソースからのデータを処理する場合は、ソースデータの送信先 AWS IoT SiteWise として を選択する必要があります AWS IoT SiteWise。エッジでのデータ処理については、the section called “エッジデータ処理を有効にする。” を参照してください。

データを別の送信先に送る場合。

a. [Destination options] (デスティネーションオプション) で、[Other destinations] (その他のデスティネーション) を選択します。

b. Greengrass ストリーム名 には、 AWS IoT Greengrass ストリームの正確な名前を入力します。

> **Note**
>
> すでに作成したストリーミングを使用することもできますし、新たに AWS IoT Greengrass ストリーミングを作成してデータをエクスポートすることも可能です。既存のストリームを使用する場合は、正確なストリーム名を入力しないと、新しいストリームが作成されます。
> AWS IoT Greengrass ストリームの操作の詳細については、「 デ AWS IoT Greengrass ベロッパーガイド」の「データストリームの管理」を参照してください。

14. [Add VOD source] (VOD ソースを追加) をクリックします。

AWS IoT SiteWise は SiteWise Edge ゲートウェイ設定を AWS IoT Greengrass コアにデプロイします。デプロイを手動で起動する必要はありません。

Modbus TCP ソース (CLI) を設定する。

Edge ゲートウェイ機能で Modbus TCP SiteWise データソースを定義できます。すべての Modbus TCP ソースを 1 つの機能構成で定義する必要があります。

> **Note**
>
> Modbus TCP ソースを使用するには、 AWS IoT SiteWise コネクタをインストールする必要があります。

この機能には、次のバージョンがあります。

Version	名前空間
1	`iotsitewise:modbuscollector:1`

Modbus TCP 機能設定パラメータ

機能設定で Modbus TCP ソースを定義する場合は、`capabilityConfiguration` JSON ドキュメントで次の情報を指定する必要があります。

ソース

Modbus TCP ソース定義構造のリスト。それぞれに次の情報が含まれています。

name

ソースの一意のわかりやすい名前。

measurementDataStreamプレフィックス

(オプション) ソースからのすべてのデータストリームの前に付加する文字列。 SiteWise Edge ゲートウェイは、このソースからのすべてのデータストリームにこのプレフィックスを追加します。データストリームのプレフィックスを使用して、異なるソースから同じ名前を持つデータストリームを区別します。各データストリームは、アカウント内で一意の名前を持つ必要があります。

送信先

次の情報を含むデスティネーション構造体。

type

デスティネーションの型。

streamName

AWS IoT Greengrass ストリームの名前。

streamBufferSize

ストリームバッファのサイズ。

エンドポイント

次の情報を含むエンドポイント構造。

ipAddress

Modbus TCP ソースの IP アドレスです。

port

(オプション) Modbus TCP ソースのポートです。

unitId

(オプション) UnitID。このデフォルト値は 1 です。

minimumInterRequest期間

各リクエスト間の最小継続時間をミリ秒単位で指定します。

propertyGroups

プロトコルで要求されたタグ定義を行うプロパティグループのリスト。

name

プロパティグループの名前です。これは一意な識別子である必要があります。

tagPathDefinitions

ソース内の測定位置。例えば、バイトやワードの順番、アドレス、変換型などです。各 `MeasurementPathDefinition` の構造はコネクタで定義される。

scanMode

ソースのスキャンモード動作と設定可能なパラメータを定義します。

イーサネット/IP (EIP) ソースを設定する。

AWS IoT SiteWise コンソールまたは SiteWise エッジゲートウェイ機能を使用して、イーサネット IP ソースを定義し、 SiteWise エッジゲートウェイに追加できます。このソースは、ローカルイーサネット IP サーバを表します。

> **Note**
>
> - SiteWise AWS IoT Greengrass V2 現在、 で実行されているエッジゲートウェイはイーサネット IP ソースをサポートしていません。

- イーサネット IP ソースを使用するには、 AWS IoT SiteWise コネクタをインストールする必要があります。

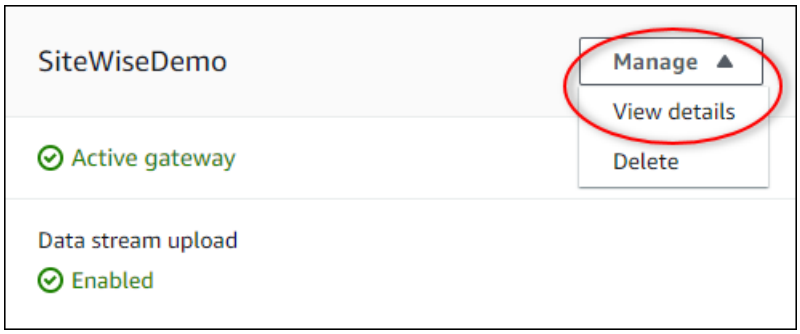
トピック

- イーサネット/IPソースを設定する (コンソール)。
- Ethernet/IPソースを設定する (CLI)。

イーサネット/IPソースを設定する (コンソール)。

イーサネット/IP ソースを設定するには

1. AWS IoT SiteWise コンソールに移動します。
2. 左のナビゲーションペインで [ゲートウェイ] を選択します。
3. ソースを作成する SiteWise Edge ゲートウェイで、 の管理 を選択し、詳細の表示 を選択します。



4. 右上隅の [New source (新しいソース)] を選択します。
5. [Protocol options] (プロトコルオプション) は、[Ethernet/IP (EIP)] (イーサネット/IP (EIP)) を選択します。
6. EtherNet/IP ソース設定 には、ソースの名前を入力します。
7. [IP address] (IP アドレス) には、データソースサーバーの IP アドレスを入力します。
8. (オプション) ソースサーバーの[Port] (ポート) を入力します。
9. [Minimum inter-request duration] (最小リクエスト間隔) には、サーバーに送信される後続のリクエスト間の時間間隔を入力します。 SiteWise Edge ゲートウェイは、デバイスと登録数に基づいて最小許容間隔を自動的に計算します。
10. [Property groups] (プロパティグループ) の場合は、[Name] (名前) を入力します。
11. [Properties] (プロパティ) の場合。

a. [Tag] (タグ) には、登録・セットのプロパティ・エイリアスを入力します。例えば **`boiler.inlet.temperature.value`** です。

b. 送信先データ型 で、 AWS IoT SiteWise データを変換するデータ型を選択します。デフォルト値は [String] (文字列) です。

12. スキャンレート では、 SiteWise エッジゲートウェイが registers を読み取る速度を更新します。 は SiteWise 、エッジゲートウェイの最小許容スキャンレート AWS IoT SiteWise を自動的に計算します。

13. (オプション) [Destination] (デスティネーション) では、ソースデータのデスティネーションを選択します。デフォルトでは、ソースはデータを に送信します AWS IoT SiteWise。代わりに、AWS IoT Greengrass ストリームを使用してローカルの送信先または AWS クラウドにデータをエクスポートできます。

> **Note**
>
> を使用してエッジでこのソースからのデータを処理する場合は、ソースデータの送信先 AWS IoT SiteWise として を選択する必要があります AWS IoT SiteWise。エッジでのデータ処理については、the section called “エッジデータ処理を有効にする。” を参照してください。

データを別の送信先に送る場合。

a. [Destination options] (デスティネーションオプション) で、[Other destinations] (その他のデスティネーション) を選択します。

b. Greengrass ストリーム名 には、 AWS IoT Greengrass ストリームの正確な名前を入力します。

> **Note**
>
> すでに作成したストリーミングを使用することもできますし、新たに AWS IoT Greengrass ストリーミングを作成してデータをエクスポートすることも可能です。既存のストリームを使用する場合は、正確なストリーム名を入力しないと、新しいストリームが作成されます。

AWS IoT Greengrass ストリームの操作の詳細については、「 AWS IoT Greengrass デベロッパーガイド」の「データストリームの管理」を参照してください。

14. [Add VOD source] (VOD ソースを追加) をクリックします。

    AWS IoT SiteWise は SiteWise Edge ゲートウェイ設定を AWS IoT Greengrass コアにデプロイします。デプロイを手動で起動する必要はありません。

Ethernet/IPソースを設定する (CLI)。

Edge ゲートウェイ機能で EIP SiteWise データソースを定義できます。すべての EIP ソースを 1 つの機能構成で定義する必要があります。

> **Note**
>
> イーサネット IP ソースを使用するには、 AWS IoT SiteWise コネクタをインストールする必要があります。

この機能には、次のバージョンがあります。

Version	名前空間
1	`iotsitewise:eipcollector:1`

EIP 機能設定パラメータ

機能設定で EIP ソースを定義する場合は、`capabilityConfiguration` JSON ドキュメントで次の情報を指定する必要があります。

ソース

EIP ソース定義構造のリスト。それぞれに次の情報が含まれています。

name

ソースの一意のわかりやすい名前。最大 256 文字まで可能です。

destinationPathPrefix

(オプション) ソースからのすべてのデータストリームの前に付加する文字列。 SiteWise Edge ゲートウェイは、このソースからのすべてのデータストリームにこのプレフィックスを追加します。データストリームのプレフィックスを使用して、異なるソースから同じ名前を持つデータストリームを区別します。各データストリームは、アカウント内で一意の名前を持つ必要があります。

送信先

次の情報を含むデスティネーション構造体。

type

デスティネーションの型。

streamName

AWS IoT Greengrass ストリームの名前。

streamBufferSize

ストリームバッファのサイズ。

エンドポイント

次の情報を含むエンドポイント構造。

ipAddress

EIP ソースの IP アドレスです。

port

(オプション) EIP ソースのポートです。許容値は 1 から 65535 までの数字です。

minimumInterRequest期間

(オプション) 各リクエスト間の最小継続時間をミリ秒単位で指定します。

propertyGroups

プロトコルで要求されたタグ定義を行うプロパティグループのリスト。各ソースは 1 つのプロパティグループを持つことができます。

name

プロパティグループの名前です。これは 256 文字以内の一意の識別子である必要があります。

tagPathDefinitions

イーサネット/IPデバイスから収集するデータと、それをどのように変換して出力するかを指定する構造体のリストです。

type

`tagPathDefinition` のタイプ。例えば `EIPTagPath` です。

パス

`tagPathDefinition` のパス。パス内の各タグは、最大 40 文字で、文字またはアンダースコアで始まります。タグには連続したアンダースコアや末尾のアンダースコアを含めることはできません。パスには、任意の値 がプレフィックスとして付けられます destinationPathPrefix。

dstDataType

タグデータを出力するためのデータ型。許容される値は、`integer`、`double`、`string`、および `boolean` です。

scanMode

ソースのスキャンモード動作と設定可能なパラメータを定義します。

type

スキャンモード動作の型。許容される値は `POLL` です。

レート

コネクタがEthernet/IPソースからタグを読み込む速度をミリ秒単位で指定します。

ソース認証の設定

OPC-UA サーバーの接続に認証資格情報が必要な場合、 AWS Secrets Managerのソースごとに、シークレットでユーザー名とパスワードを定義できます。次に、Greengrass グループと IoT SiteWise コネクタにシークレットを追加して、 SiteWise エッジゲートウェイでシークレットを使用できるようにします。詳細については、「 AWS IoT Greengrass Version 1 デベロッパーガイド」のAWS IoT Greengrass 「コアにシークレットをデプロイする」を参照してください。

SiteWise Edge ゲートウェイでシークレットが使用可能になったら、ソースを設定するときにシークレットを選択できます。次に、 SiteWise Edge ゲートウェイは、ソースに接続するときにシークレットの認証情報を使用します。詳細については、「データソースの設定」を参照してください。

トピック

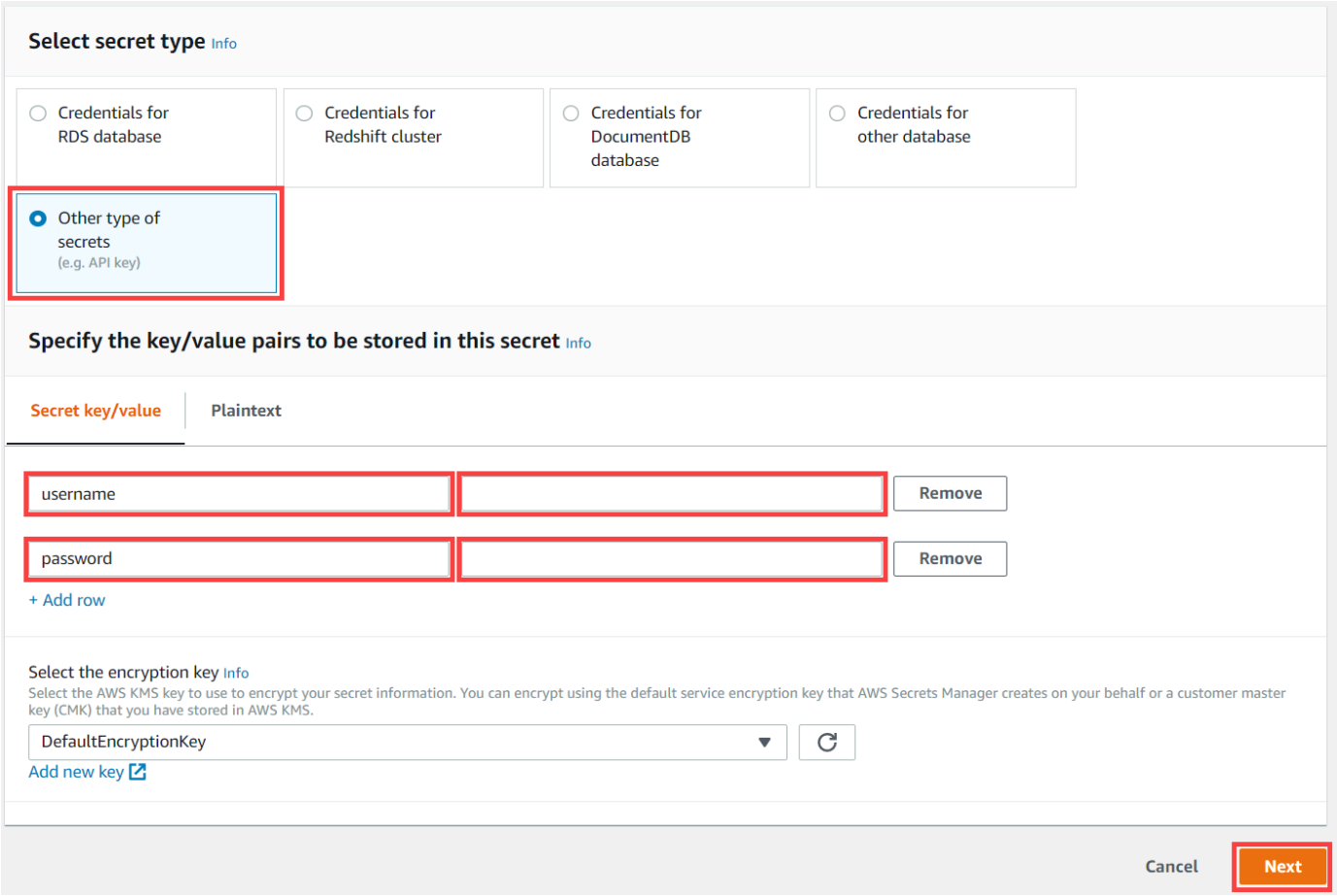
- ソース認証シークレットの作成
- Greengrass グループにシークレットを追加する。
- IoT SiteWise コネクタへのシークレットの追加

ソース認証シークレットの作成

このステップでは、Secrets Manager でソースの認証シークレットを作成します。シークレットで、ソースの認証詳細を含む **username** と **password** キーと値のペアを定義します。

ソース認証シークレットを作成するには

1. [Secrets Manager console] (Secrets Managerのコンソール) に移動します。
2. [新しいシークレットを保存] を選択します。
3. [Select secret type] (シークレットの種類を選択) で、[Other type of secrets] (他の種類のシークレット) を選択します。
4. **username** と OPC-UA サーバーの認証値の **password** のキーと値のペアを入力し、[Next (次へ)] を選択します。



5. **`greengrass-factory1-auth`** など、`greengrass-` から始まる[Secret name] (シークレット名) を入力してください。

> **Important**
>
> シークレットにアクセスするには、デフォルトの AWS IoT Greengrass サービスロールの `greengrass-` プレフィックスを使用する必要があります。このプレフィックスなしでシークレットに名前を付ける場合は、シークレットにアクセスするための AWS IoT Greengrass カスタムアクセス許可を付与する必要があります。詳細については、「 AWS IoT Greengrass Version 1 デベロッパーガイド」の「シークレット値 AWS IoT Greengrass の取得を許可する」を参照してください。

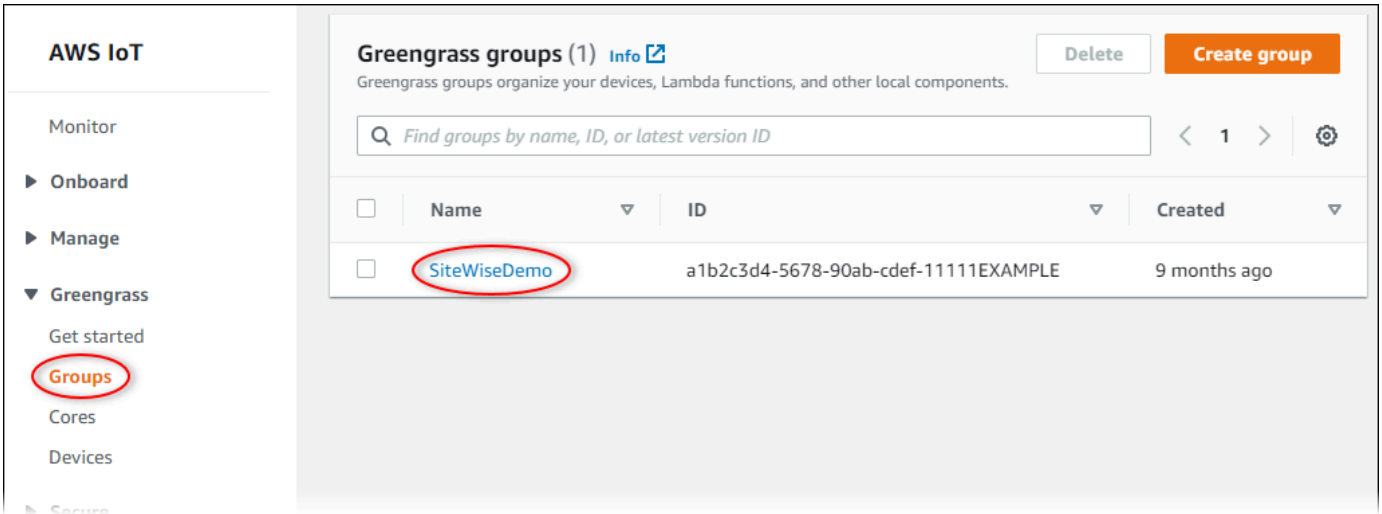
6. 説明を入力し、[Next (次へ)] を選択します。
7. (オプション) [Configure automatic rotation (自動ローテーションの設定)] ページでシークレットの自動ローテーションを設定します。自動ローテーションを設定する場合、シークレットがローテーションされるたびに Greengrass グループを再デプロイする必要があります。
8. [Configure automatic rotation (自動回転の設定)] ページで、[Next (次へ)] を選択します。
9. 新しいシークレットを確認し、[Store (保存)] を選択します。

Greengrass グループにシークレットを追加する。

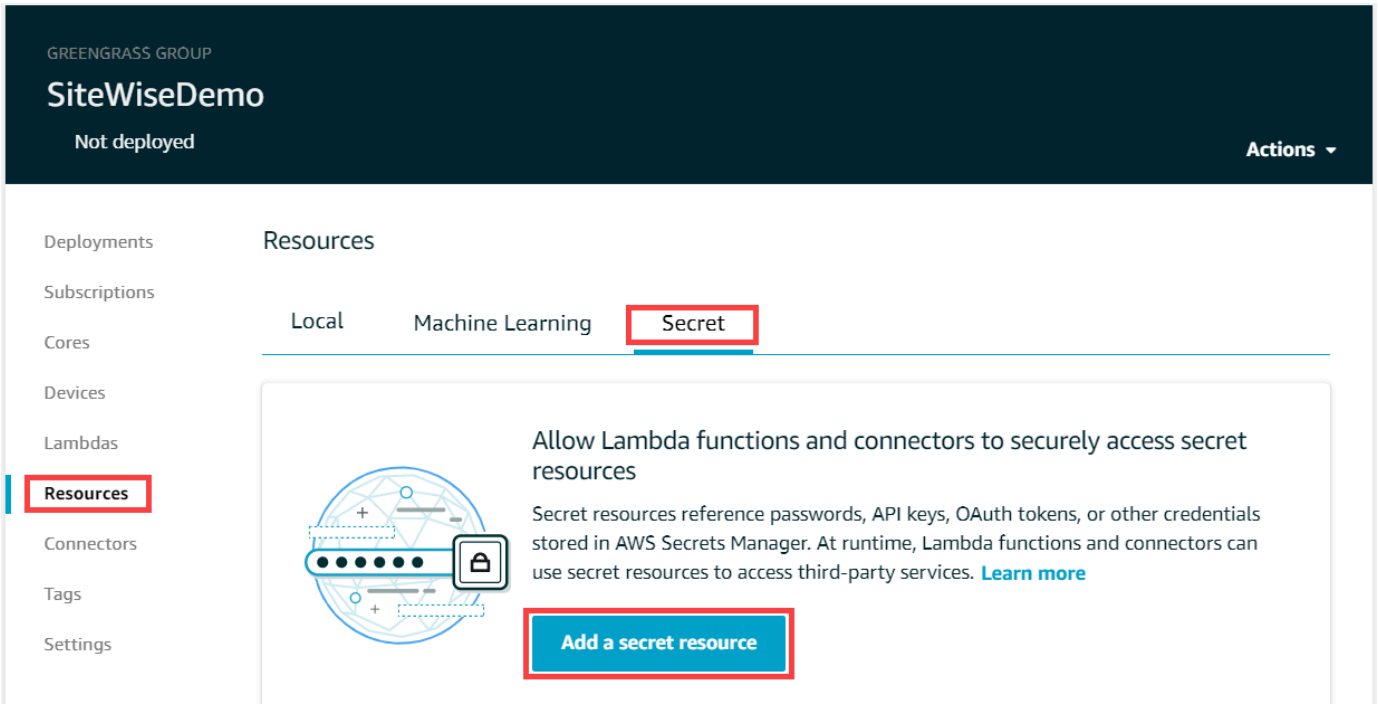
この手順では、ソース認証シークレットを AWS IoT Greengrass グループに追加して、IoT SiteWise コネクタで使用できるようにします。

Greengrass のグループにシークレットを追加するには。

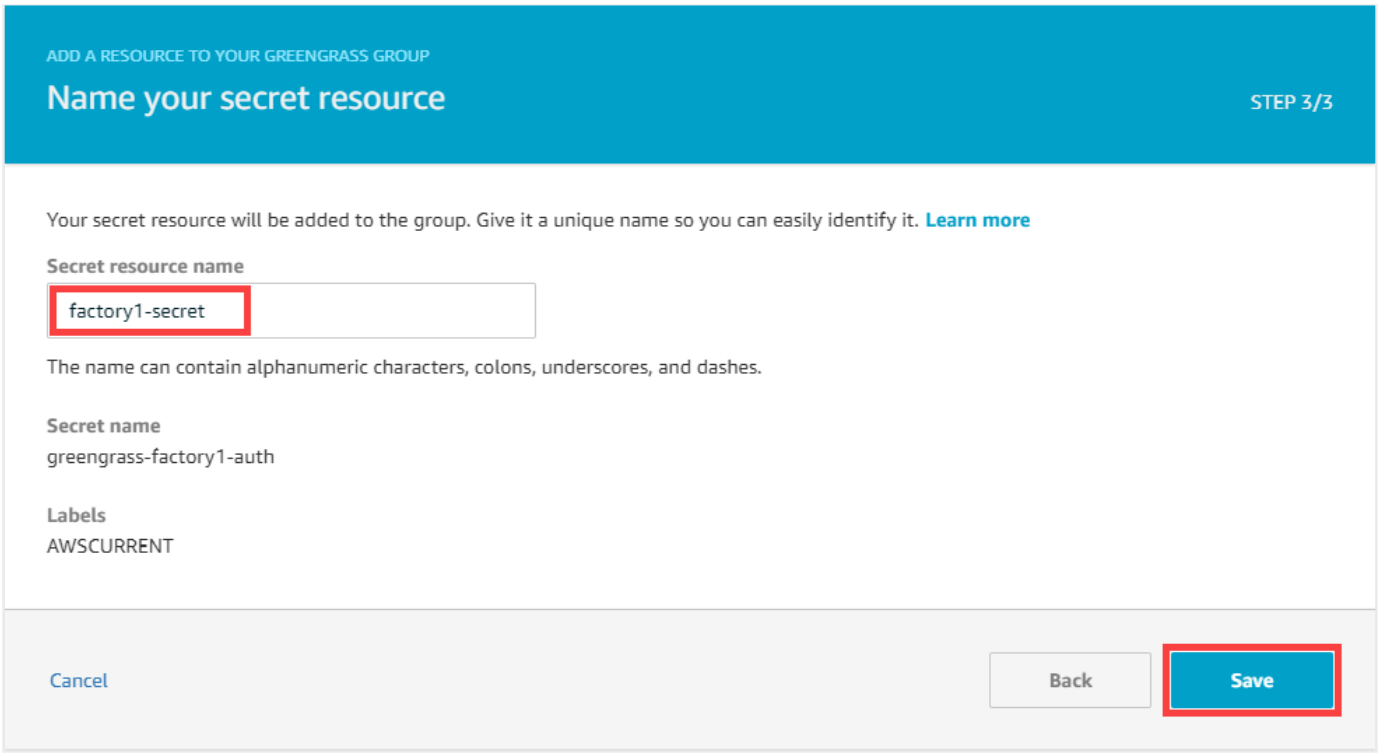
1. AWS IoT Greengrass コンソールに移動します。
2. ナビゲーションペインの、Greengrass で、[Groups] (グループ) を選択し、グループを選択します。



3. ナビゲーションページで、[[Resources] (リソース) を選択します。
4. [リソース] ページで [シークレット] タブを選択してから、[Add a secret resource (シークレットリソースの追加)] を選択します。



5. [Select (シークレット)] を選択し、リストからシークレットを選択します。
6. [Next (次へ)] を選択します。
7. [Secret resource name (シークレットリソース名)] にシークレットリソースの名前を入力し、[Save (保存)] を選択します。

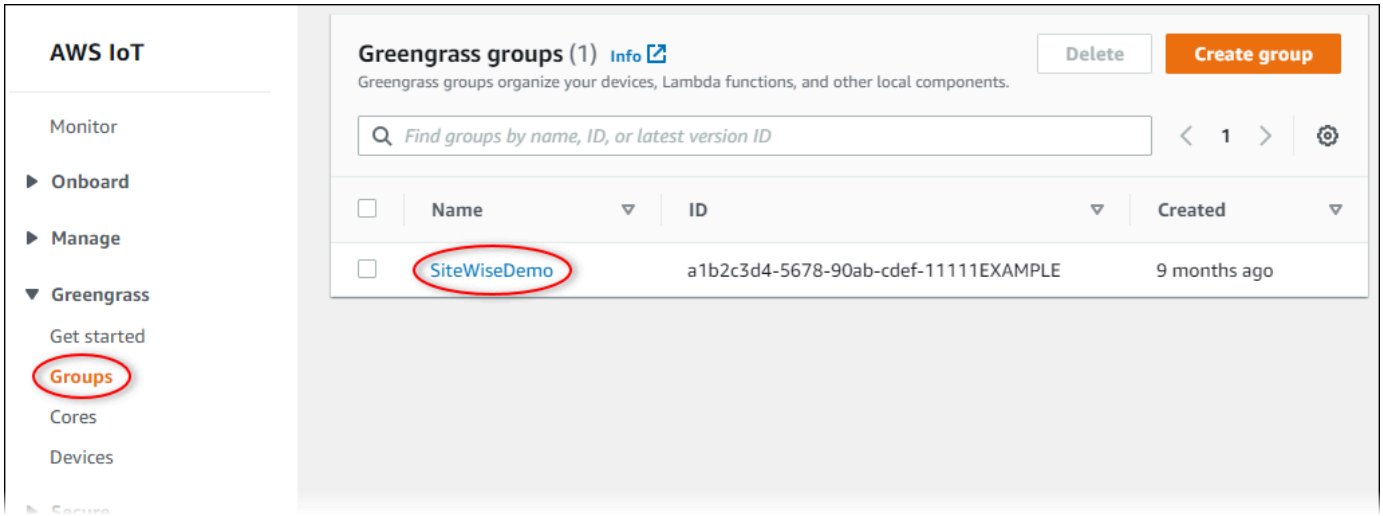


IoT SiteWise コネクタへのシークレットの追加

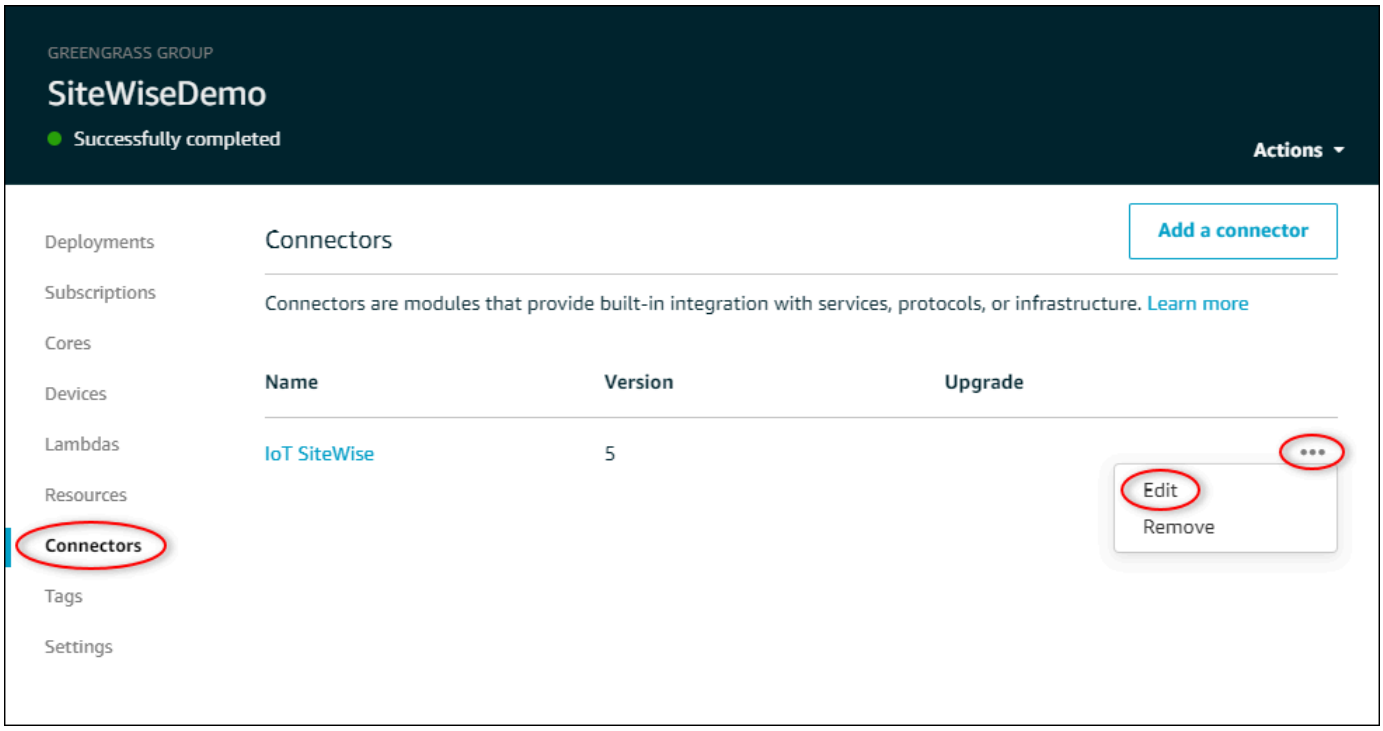
この手順では、ソース認証シークレットを IoT SiteWise コネクタに追加して、 AWS IoT SiteWise および SiteWise Edge ゲートウェイで使用できるようにします。

IoT SiteWise コネクタにシークレットを追加するには

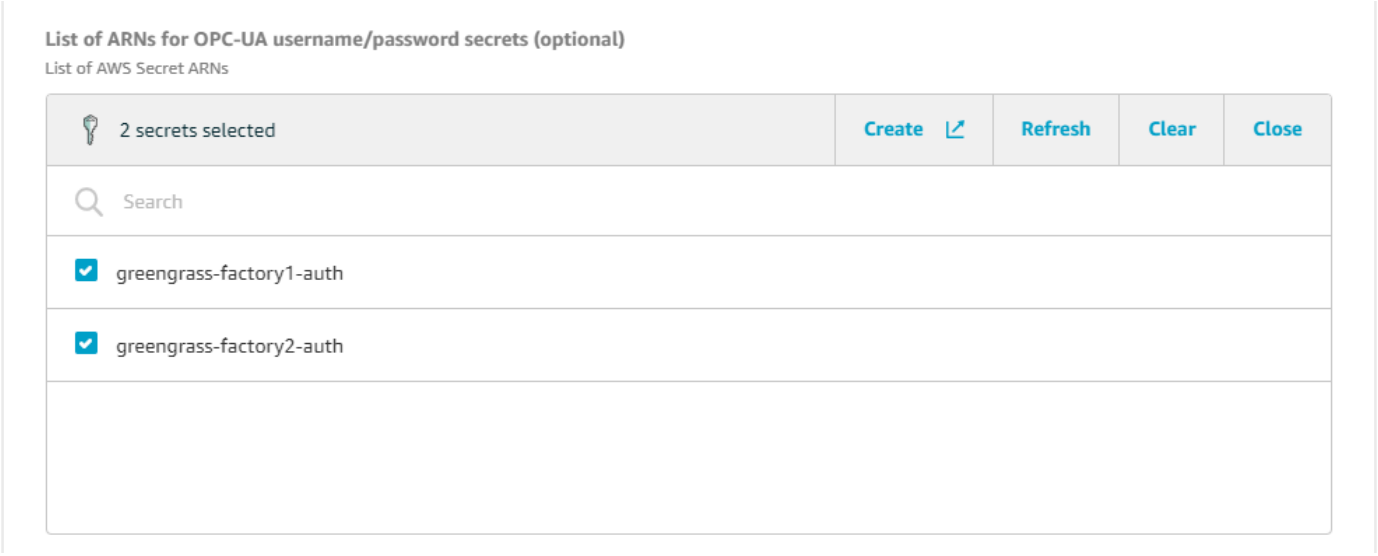
1. AWS IoT Greengrass コンソールに移動します。
2. ナビゲーションペインの、Greengrass で、[Groups] (グループ) を選択し、グループを選択します。



3. ナビゲーションページで、[Connectors]] (コネクタ) を選択します。
4. IoT SiteWise コネクタの省略記号アイコンを選択してオプションメニューを開き、編集 を選択します。



5. ARNs のリスト」で「 の選択」を選択し、この SiteWise Edge ゲートウェイに追加する各シークレットを選択します。シークレットを作成する必要がある場合、「ソース認証シークレットの作成」を参照してください。



シークレットが表示されない場合は、[更新] を選択します。それでもシークレットが表示されない場合は、[added the secret to your Greengrass group] (シークレットを Greengrass グループに追加したこと) を確認してください。

6. [保存] を選択します。
7. ページの右上隅の [Actions (アクション)] メニューで、[Deploy (デプロイ)] を選択します。
8. [Automatic detection (自動検出)] を選択して、デプロイを開始します。

   デプロイが失敗した場合は、再度[デプロイ] を選択します。デプロイが失敗し続ける場合は、「AWS IoT Greengrass デプロイのトラブルシューティング」を参照してください。

   グループがデプロイされたら、新しいシークレットを使用するソースを設定できます。詳細については、「データソースの設定」を参照してください。

コネクタのアップグレード

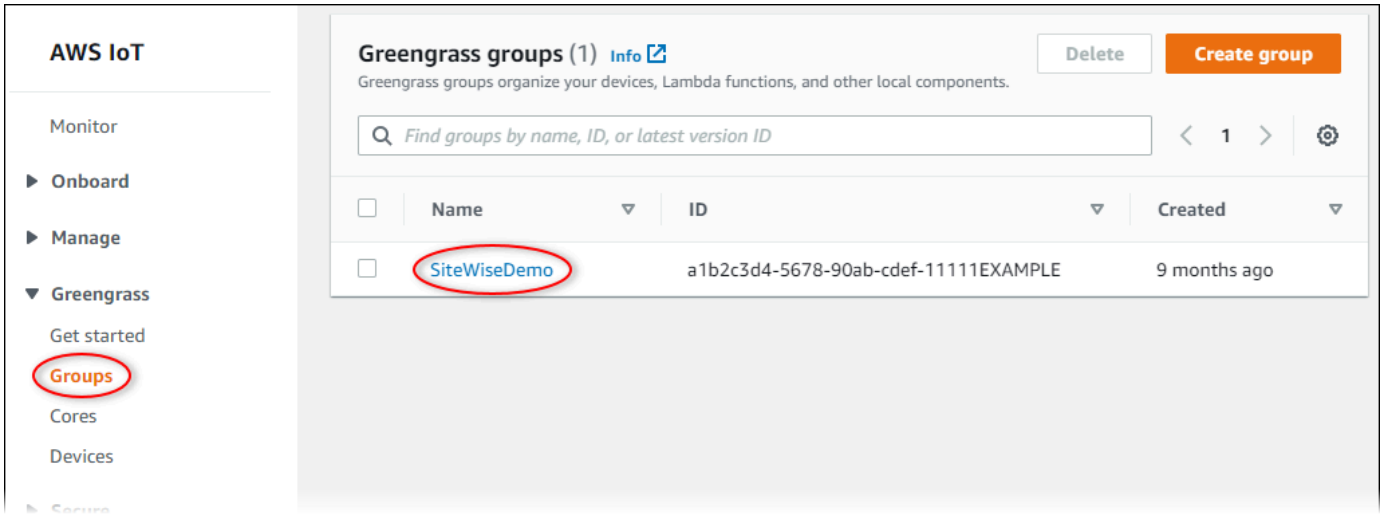
> **Important**
>
> IoT SiteWise コネクタのバージョン 6 では、 Core ソフトウェア v1.10.0 とストリームマネージャー という AWS IoT Greengrass 新しい要件が導入されています。コネクタをアップグレードする前に、 SiteWise エッジゲートウェイがこれらの要件を満たしていることを確認してください。そうしないと、 SiteWise エッジゲートウェイをデプロイできません。

新しい IoT SiteWise コネクタバージョンがリリースされたら、 SiteWise エッジゲートウェイのコネクタを簡単にアップグレードできます。

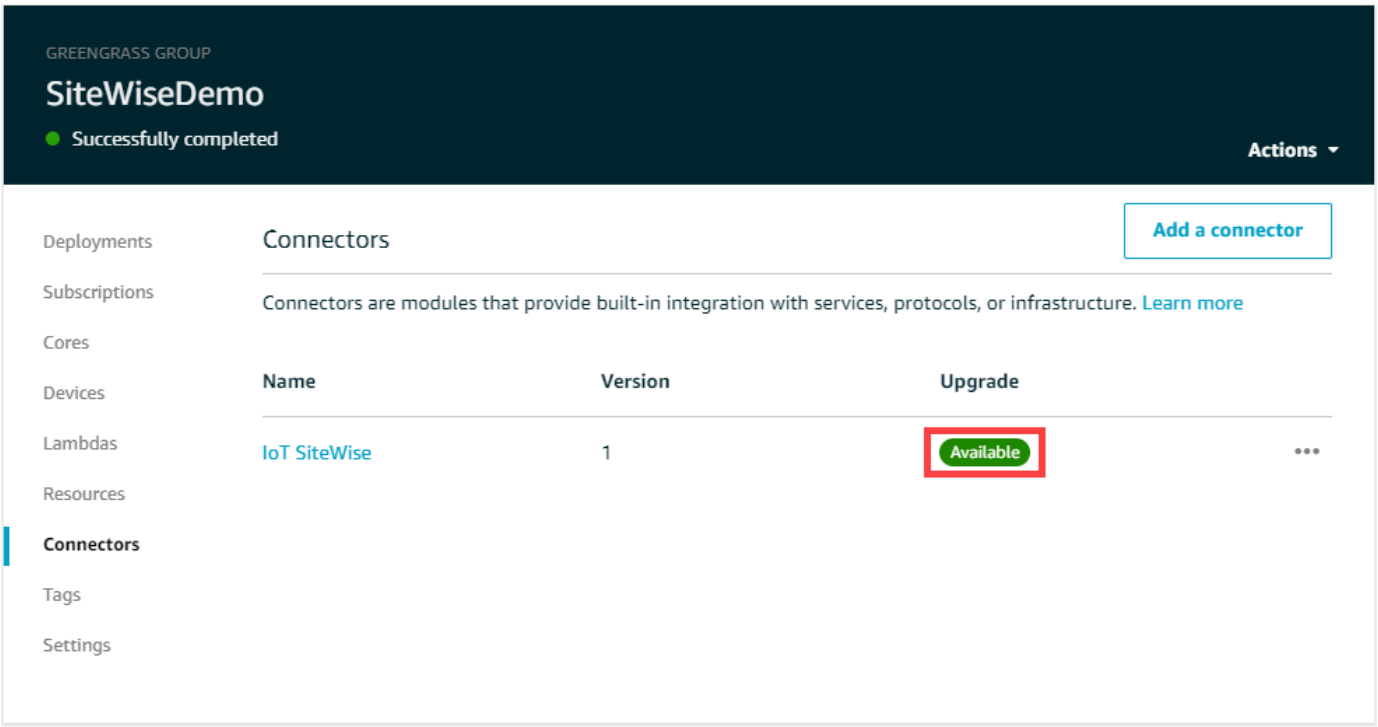
> **Note**
>
> この手順では、Greengrass グループを再デプロイし、 SiteWise エッジゲートウェイを再起動します。 SiteWise Edge ゲートウェイは、再起動中にデータを取り込みません。 SiteWise Edge ゲートウェイを再起動する時間は、 SiteWise Edge ゲートウェイのソースのタグの数によって異なります。再起動時間の範囲は、数秒 (タグ数が少ない SiteWise エッジゲートウェイの場合) から数分 (タグ数が多い SiteWise エッジゲートウェイの場合) です。

IoT SiteWise コネクタをアップグレードするには

1. AWS IoT Greengrass コンソールに移動します。
2. ナビゲーションペインの Greengrass で、グループ を選択し、 SiteWise エッジゲートウェイのセットアップ時に作成したグループを選択します。



3. ナビゲーションペインで、[Connectors] (コネクタ) を選択します。
4. コネクタページで、IoT SiteWise コネクタの横にある「使用可能」を選択します。



[Available (使用可能)] 要素が表示されない場合、コネクタは既に最新バージョンです。

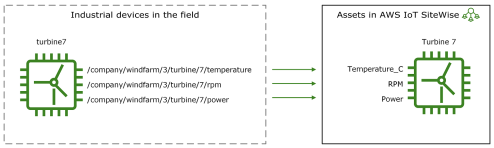
5. [Upgrade connector (アップグレードコネクタ)] ページで、コネクタのパラメータを入力し、[Upgrade (アップグレード)] を選択します。
6. ページの右上隅の [Actions (アクション)] メニューで、[Deploy (デプロイ)] を選択します。
7. [Automatic detection (自動検出)] を選択して、デプロイを開始します。

   デプロイが失敗した場合は、再度[デプロイ] を選択します。デプロイが失敗し続ける場合は、「AWS IoT Greengrass デプロイのトラブルシューティング」を参照してください。

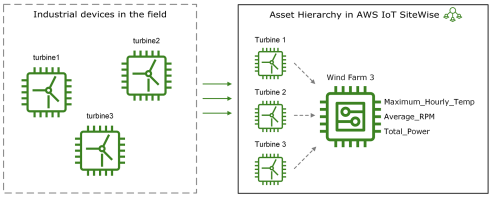
産業用アセットのモデリング

AWS IoT SiteWise アセットを使用して、産業オペレーションの仮想表現を作成できます。アセットは、デバイス、機器、または 1 つ以上のデータストリームを にアップロードするプロセスを表します AWS クラウド。たとえば、アセットデバイスの例として、気温、プロペラ回転速度、および電力出力の時系列測定値を AWS IoT SiteWiseのアセットプロパティに送信する風力タービンがあります。

各データストリームは、一意のプロパティエイリアスに対応します。たとえば、エイリアス `/company/windfarm/3/turbine/7/temperature` は、風力発電所 #3 のタービン #7 からの温度データストリームを一意に識別します。温度データを摂氏から華氏に変換するなど、数式を使用して受信測定データを変換するように AWS IoT SiteWise アセットを設定できます。



アセットは、風力発電所全体など、デバイスの論理的なグループを表すこともできます。アセットを他のアセットに関連付けると、複雑な産業オペレーションを表すアセット階層を作成できます。アセットは、関連付けられた子アセット内のデータにアクセスできます。これにより、 AWS IoT SiteWise 式を使用して風力発電所の正味電力出力などの集計メトリクスを計算できます。



アセットモデル からすべてのアセットを作成する必要があります。アセットモデルはアセットのフォーマットを標準化する宣言的な構造です。アセットモデルは、同じタイプの複数のアセットに一貫した情報を適用し、デバイスのグループを表すアセット内のデータを処理できるようにします。前の図では、3 つのタービンすべてに同じアセットモデルを使用します。これは、すべてのタービンが共通のプロパティセットを共有しているためです。

コンポーネントモデル を作成することもできます。コンポーネントモデルは、アセットモデルやその他のコンポーネントモデルに含めることができる特殊なタイプのアセットモデルです。コンポーネントモデルを使用して、複数のアセットモデル間で共有するセンサー、モーターなどの一般的な再利用可能なサブアセンブリを定義できます。

アセットモデルを定義したら、産業アセットを作成できます。アセットを作成するには、ACTIVE アセットモデルを選択して、そのモデルからアセットを作成します。次に、データストリームのエイリアスや属性など、アセット固有の情報を入力できます。前の図では、1 つのアセットモデルから 3 つのタービンアセットを作成し、各タービンに /company/windfarm/3/turbine/7/temperature のようなデータストリームエイリアスを関連付けます。

既存のアセット、アセットモデル、コンポーネントモデルを更新および削除することもできます。アセットモデルを更新すると、そのアセットモデルに基づくすべてのアセットに、基になるモデルに加えた変更が反映されます。コンポーネントモデルを更新すると、これはコンポーネントモデルを参照するすべてのアセットモデルに基づくすべてのアセットに適用されます。

アセットモデルは非常に複雑な場合があります。例えば、サブコンポーネントが多数ある複雑な機器のモデルを作成する場合などです。このようなアセットモデルを整理して保守しやすくするために、カスタム複合モデルを使用して関連プロパティをグループ化したり、共有コンポーネントを再利用したりできます。詳細については、「カスタム複合モデル (コンポーネント)」を参照してください。

トピック

アセットおよびモデルの状態

アセット、アセットモデル、またはコンポーネントモデルを作成、更新、または削除すると、変更が propagate. AWS IoT SiteWise resolves これらのオペレーションを非同期的に解決し、各リソースの

ステータスを更新します。各アセット、アセットモデル、コンポーネントモデルには、リソースの状態と、該当する場合はエラーメッセージを含むステータスフィールドがあります。ステータスは以下のいずれかの値になります。

- ACTIVE – リソースはアクティブです。これは、アセット、アセットモデル、コンポーネントモデルをクエリして操作できる唯一の状態です。
- CREATING – リソースを作成中です。
- UPDATING – リソースは更新中です。
- DELETING – リソースは削除中です。
- PROPAGATING – (アセットモデルとコンポーネントモデルのみ) 変更は、すべての依存リソース (アセットモデルからアセット、またはコンポーネントモデルからアセットモデル) に反映されます。
- FAILED – リソースは、作成または更新オペレーション中に検証に失敗しました。これは、式内の循環参照が原因である可能性があります。FAILED 状態にあるリソースを削除できます。

アセット、アセットモデル、またはコンポーネントモデルが、オペレーションの解決ACTIVE時以外の状態で AWS IoT SiteWise 設定されている作成、更新、削除オペレーションの一部。これらのオペレーションのいずれかを実行した後にリソースをクエリまたは操作するには、状態が に変わるまで待つ必要がありますACTIVE。そうしないと、リクエストは失敗します。

トピック

- アセットのステータスを確認する
- アセットモデルまたはコンポーネントモデルのステータスの確認

アセットのステータスを確認する

AWS IoT SiteWise コンソールまたは API を使用して、アセットのステータスを確認できます。

トピック

- アセットの状態を確認する (コンソール)
- アセットのステータスの確認 (AWS CLI)

アセットの状態を確認する (コンソール)

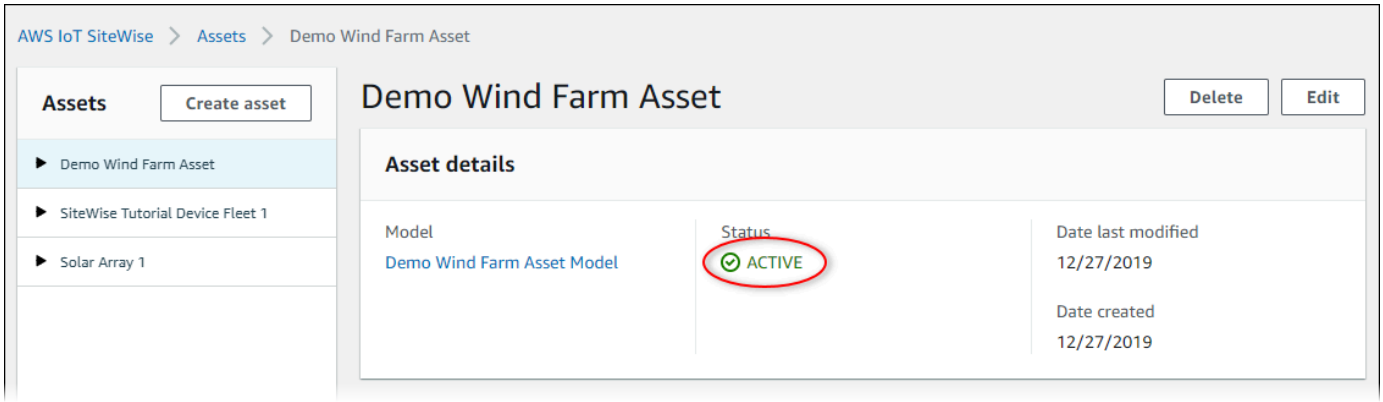
AWS IoT SiteWise コンソールでアセットのステータスを確認するには、次の手順に従います。

アセットの状態を確認するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. チェックするアセットを選択します。

> **Tip**
>
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [Asset details] (アセットの詳細) パネルで[Status] (状態) を検索します。



アセットのステータスの確認 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して、アセットのステータスを確認できます。

アセットのステータスを確認するには、 `assetId` パラメータを指定して DescribeAssetオペレーションを使用します。

アセットのステータスを確認するには (AWS CLI)

- 以下のコマンドを実行して、アセットを記述します。*`asset-id`* をアセットの ID または外部 ID に置き換えます。外部 ID はユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

```
aws iotsitewise describe-asset --asset-id asset-id
```

このオペレーションは、アセットの詳細を含むレスポンスを返します。レスポンスには、次の構造を持つassetStatusオブジェクトが含まれます。

```
{
    ...
    "assetStatus": {
      "state": "String",
      "error": {
         "code": "String",
         "message": "String"
      }
    }
  }
```

アセットの状態は JSON オブジェクトの `assetStatus.state` にあります。

アセットモデルまたはコンポーネントモデルのステータスの確認

AWS IoT SiteWise コンソールまたは API を使用して、アセットモデルまたはコンポーネントモデルのステータスを確認できます。

トピック

- アセットモデルまたはコンポーネントモデルのステータスの確認 (コンソール)
- アセットモデルまたはコンポーネントモデルのステータスの確認 (AWS CLI)

アセットモデルまたはコンポーネントモデルのステータスの確認 (コンソール)

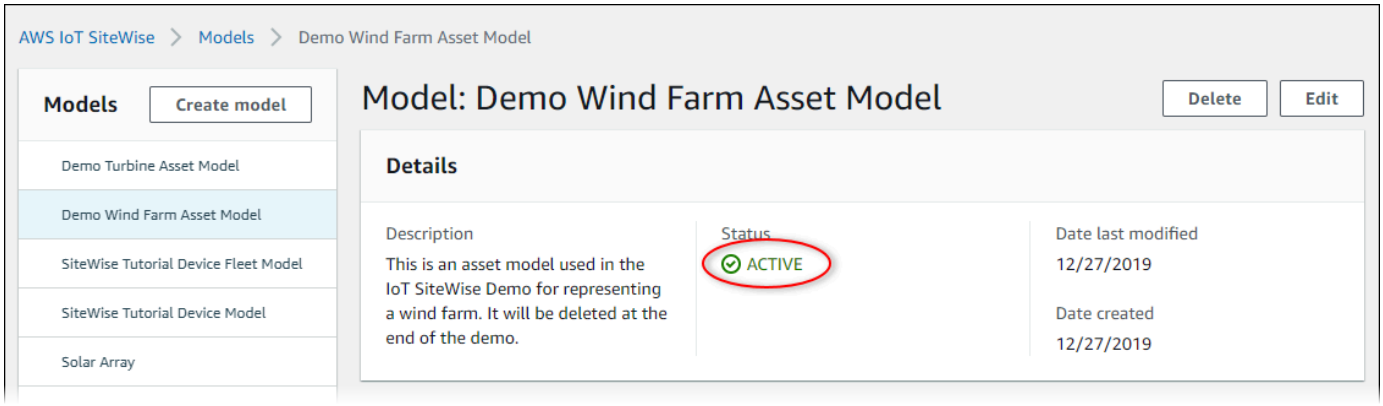
AWS IoT SiteWise コンソールでアセットモデルまたはコンポーネントモデルのステータスを確認するには、次の手順に従います。

> **Tip**
>
> アセットモデルとコンポーネントモデルの両方がナビゲーションペインのモデルの下に一覧表示されます。選択したアセットモデルまたはコンポーネントモデルの詳細パネルには、そのタイプが表示されます。

アセットモデルまたはコンポーネントモデルのステータスを確認するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。

2. ナビゲーションペインで、[モデル] を選択します。
3. チェックするモデルを選択します。
4. [Details] (詳細) パネルで[Status] (状態) を探す。



アセットモデルまたはコンポーネントモデルのステータスの確認 (AWS CLI)

を使用して、アセットモデルまたはコンポーネントモデルのステータス AWS CLI を確認できます。

アセットモデルまたはコンポーネントモデルのステータスを確認するには、 `assetModelId`パラメータを指定して DescribeAssetModel オペレーションを使用します。

> **Tip**
>
> は、コンポーネントモデルをアセットモデルの一種として AWS CLI 定義します。したがって、両方のタイプのDescribeAssetモデルに同じモデルオペレーションを使用します。レスポンスの `assetModelType`フィールドは、それが `ASSET_MODEL`か かを示しますCOMPONENT_MODEL。

アセットモデルまたはコンポーネントモデルのステータスを確認するには (AWS CLI)

- 次のコマンドを実行してモデルを記述します。*`asset-model-id`* をアセットモデルまたはコンポーネントモデルの ID または外部 ID に置き換えます。外部 ID はユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

```
aws iotsitewise describe-asset-model --asset-model-id asset-model-id
```

オペレーションは、モデルの詳細を含むレスポンスを返します。レスポンスには、次の構造を持つ assetModelStatus オブジェクトが含まれています。

```
{
    ...
    "assetModelStatus": {
      "state": "String",
      "error": {
         "code": "String",
         "message": "String"
      }
    }
  }
```

JSON オブジェクトassetModelStatus.state内のモデルの状態は です。

カスタム複合モデル (コンポーネント)

多くの部分がある複雑な機械など、特に複雑な産業アセットをモデル化する場合、アセットモデルを整理して保守しやすくすることが課題になる可能性があります。

このような場合は、カスタム複合モデル、またはコンソールを使用している場合はコンポーネントを既存のアセットモデルとコンポーネントモデルに追加できます。これらは、関連するプロパティをグループ化し、サブコンポーネント定義を再利用することで、整理を維持するのに役立ちます。

カスタム複合モデルには 2 つのタイプがあります。

- インラインカスタム複合モデルは、カスタム複合モデルが属するアセットモデルまたはコンポーネントモデルに適用されるグループ化されたプロパティのセットを定義します。これらを使用して、関連するプロパティをグループ化します。これらは、名前、説明、および一連のアセットモデルプロパティで構成されます。再利用できません。
- コンポーネントモデルベースのカスタム複合モデルは、アセットモデルまたはコンポーネントモデルに含めるコンポーネントモデルを参照します。これらを使用して、モデルに標準サブアセンブリを含めます。これらは、参照するコンポーネントモデルの名前、説明、および ID で構成されます。独自のプロパティはありません。参照されるコンポーネントモデルは、作成されたアセットに関連付けられたプロパティを提供します。

以下のセクションでは、カスタム複合モデルを設計で使用する方法について説明します。

トピック

インラインカスタム複合モデル

インラインカスタム複合モデルは、関連するプロパティをグループ化してアセットモデルを整理する方法を提供します。

例えば、ロボットアセットをモデル化するとします。ロボットには、サーボモーター、電源、バッテリーが含まれています。これらの各構成要素には、モデルに含める独自のプロパティがあります。次のようなプロパティrobot_modelを持つ という名前のアセットモデルを定義できます。

- robot_model
  - servo_status （整数）
  - servo_position （ダブル）
  - powersupply_status （整数）
  - powersupply_temperature （ダブル）
  - battery_status （整数）
  - battery_charge （ダブル）

ただし、サブアセンブリが多数ある場合や、サブアセンブリ自体に多くのプロパティがある場合があります。このような場合、前の例のように、モデルルートの単一のフラットリストで参照して維持するのが面倒になるプロパティが多数ある可能性があります。

このような状況に対処するには、インラインカスタム複合モデルを使用してプロパティをグループ化できます。インラインカスタム複合モデルは、独自のプロパティを定義するカスタム複合モデルです。例えば、次のようにロボットをモデル化できます。

- robot_model
  - servo
    - status （整数）

    - position（ダブル）
  - powersupply
    - status（整数）
    - temperature （ダブル）
  - battery
    - status（整数）
    - charge（ダブル）

前の例では、servo、powersupply、 はrobot_modelアセットモデル内で定義されたインラインカスタム複合モデルの名前batteryです。これらの複合モデルはそれぞれ、独自のプロパティを定義します。

> **Note**
>
> この場合、各カスタム複合モデルは独自のプロパティを定義し、すべてのプロパティがアセットモデル自体の一部になるようにします (robot_modelこの場合は)。これらのプロパティは、他のアセットモデルやコンポーネントモデルと共有されません。例えば、 というインラインカスタム複合モデルも持つ他のアセットモデルを作成してもservo、 servo内でを変更robot_modelしても、他のアセットモデルservoの定義には影響しません。
> このような共有を実装する場合 (例えば、すべてのアセットモデルが共有できるサーボの定義を 1 つだけにする場合)、代わりにコンポーネントモデルを作成し、それを参照するコンポーネントモデルベースの複合モデルを作成します。詳細については、次のセクションを参照してください。

インラインカスタム複合モデルを作成する方法については、「」を参照してくださいカスタム複合モデルの作成 (コンポーネント)。

C omponent-model-based カスタム複合モデル

でコンポーネントモデルを作成して、標準の再利用可能なサブアタッチ AWS IoT SiteWise を定義できます。コンポーネントモデルを作成したら、他のアセットモデルやコンポーネントモデルにそのモデルへの参照を追加できます。これを行うには、コンポーネントを参照する任意のcomponent-model-based モデルにカスタム複合モデルを追加します。コンポーネントへの参照は、多くのモデルから追加することも、同じモデル内で複数回追加することもできます。

このようにして、モデル間で同じ定義を複製することを回避できます。また、コンポーネントモデルに加えた変更は、それを使用するすべてのアセットモデルに反映されるため、モデルのメンテナンスも簡素化されます。

例えば、産業施設に、同じ種類のサーボモーターを使用する多くのタイプの機器があるとします。そのうちのいくつかは、単一の機器に多数のサーボモーターがあります。機器タイプごとにアセットモデルを作成しますが、 の定義をservo毎回複製する必要はありません。一度だけモデル化し、さまざまなアセットモデルで使用します。後で の定義を変更するとservo、すべてのモデルとアセットで更新されます。

この方法で前の例の口ボットをモデル化するには、このように、サーボモーター、電源、バッテリーをコンポーネントモデルとして定義できます。

- servo_component_model
  - status (整数)
  - position (ダブル)

- powersupply_component_model
  - status (整数)
  - temperature (ダブル)

- battery__component_model
  - status (整数)
  - charge (ダブル)

その後、これらのコンポーネントを参照する robot_modelなどのアセットモデルを定義できます。複数のアセットモデルが同じコンポーネントモデルを参照できます。また、口ボットに複数の ServoMotor がある場合など、1 つのアセットモデルで同じコンポーネントモデルを複数回参照することもできます。

- robot_model
  - servo1 (参照： servo_component_model)

- servo2 (リファレンス： servo_component_model)
- servo3 (リファレンス： servo_component_model)
- powersupply (リファレンス： powersupply_component_model)
- battery (リファレンス： battery_component_model)

コンポーネントモデルの作成方法については、「」を参照してくださいコンポーネントモデルの作成。

他のモデルでコンポーネントモデルを参照する方法については、「」を参照してくださいカスタム複合モデルの作成 (コンポーネント)。

パスを使用してカスタム複合モデルプロパティを参照する

アセットモデル、コンポーネントモデル、またはカスタム複合モデルにプロパティを作成すると、変換やメトリクス など、その値を使用する他のプロパティからプロパティを参照できます。

AWS IoT SiteWise には、 プロパティを参照するさまざまな方法が用意されています。最も簡単な方法は、多くの場合、そのプロパティ ID を使用することです。ただし、参照するプロパティがカスタム複合モデルにある場合は、代わりにパスで参照する方が便利な場合があります。

パスは、アセットモデルと複合モデル内のネストされた複合モデル間の位置に関してプロパティを指定する順序付けられたパスセグメントのシーケンスです。

プロパティパスの取得

プロパティのパスは、プロパティ の AssetModelpathフィールドから取得できます。

例えば、 プロパティを持つカスタム複合モデル robot_modelを含むアセットモデルがあるとservoしますposition。で DescribeAssetModelCompositeModel を呼び出すとservo、positionプロパティは次のpathようになります。

```
"path": [
    {
       "id": "asset model ID",
       "name": "robot_model"
    },
    {
       "id": "composite model ID",
       "name": "servo"
    },
```

```
    {
       "id": "property ID",
       "name": "position"
    }
]
```

プロパティパスの使用

プロパティパスは、変換やメトリクスなどの他のプロパティを参照するプロパティを定義するときに使用できます。

プロパティは、変数を使用して別のプロパティを参照します。変数の操作の詳細については、「」を参照してください式での変数の使用。

プロパティを参照する変数を定義する場合、プロパティの ID またはそのパスのいずれかを使用できます。

参照されるプロパティのパスを使用する変数を定義するには、その値の propertyPathフィールドを指定します。

例えば、パスを使用してプロパティを参照するメトリクスを持つアセットモデルを定義するには、次のようなペイロードをCreateAssetモデル に渡します。

```
{
    ...
    "assetModelProperties": [
        {
            ...
            "type": {
                "metric": {
                    ...
                    "variables": [
                        {
                            "name": "variable name",
                            "value": {
                                "propertyPath": [
                                    path segments
                                ]
                            }
                        }
                    ],
                    ...
                }
```

```
            },
            ...
        },
        ...
    ],
    ...
}
```

オブジェクト IDs

AWS IoT SiteWise は、アセット、アセットモデル、プロパティ、階層など、さまざまなタイプの永続オブジェクトを定義します。このようなオブジェクトには、取得、更新、削除に使用できる一意の識別子があります。

AWS IoT SiteWise には、ID 作成のオプションが顧客ごとに異なります。 は、オブジェクト作成時にデフォルトで ID AWS IoT SiteWise を生成します。ユーザーは、オブジェクトに独自の IDsを指定することもできます。

トピック

- オブジェクト UUIDs
- 外部 IDsの使用

オブジェクト UUIDs

のすべての永続オブジェクト AWS IoT SiteWise には、それを識別するための UUID があります。例えば、アセットモデルにはアセットモデル ID があり、アセットにはアセット ID があります。この ID は、オブジェクトの作成時に割り当てられ、オブジェクトの存続期間中は変更されません。

新しいオブジェクトを作成すると、 はデフォルトで一意の ID AWS IoT SiteWise を生成します。作成時に独自の ID を UUID 形式で指定することもできます。

> **Note**
>
> UUIDsは、作成された AWS リージョン内でグローバルに一意で、同じオブジェクトタイプに対して一意である必要があります。が ID を AWS IoT SiteWise 自動生成する場合、常に一意です。独自の ID を選択する場合は、一意であることを確認してください。

例えば、モデル を呼び出して新しいアセットCreateAssetモデルを作成する場合、リクエストのオプションassetModelIdフィールドに独自の UUID を指定できます。

対照的に、リクエストassetModelIdから を省略すると、 は新しいアセットモデルの UUID AWS IoT SiteWise を生成します。

外部 IDsの使用

UUID 以外の形式で独自の ID を定義するには、外部 ID を割り当てることができます。例えば、 ではないシステムで使用している ID を再利用する場合や AWS、人間が読み取れるようにする場合などに、これを行うことができます。外部 IDs形式はより柔軟です。これらを使用して、UUID を使用する API オペレーションで AWS IoT SiteWise オブジェクトを参照できます。

UUIDsはコンテキスト内で一意である必要があります。例えば、同じ外部 ID を持つ 2 つのアセットモデルを持つことはできません。また、UUIDsつの外部 ID のみを持つことができ、変更することはできません。

外部 IDs と UUIDsの違い

外部 IDs、次の点で UUIDsとは異なります。

- すべてのオブジェクトには UUID がありますが、外部 IDsはオプションです。
- AWS IoT SiteWise は外部 IDsを生成しません。これらを自分で指定します。
- オブジェクトにまだ存在しない場合は、いつでも外部 ID を割り当てることができます。

外部 IDsの形式

有効な外部 ID には、次のプロパティがあります。

- 2～128 文字の長さです。
- 最初と最後の文字は英数字 (A～Z、a～z、0～9) である必要があります。
- 最初と最後の文字以外の文字は英数字であるか、次のいずれかである必要があります。 _-.:

例えば、外部 ID は次の正規表現に準拠している必要があります。

```
[a-zA-Z0-9][a-zA-Z0-9_\-.:]*[a-zA-Z0-9]+
```

外部 IDs を持つオブジェクトの参照

UUID を使用してオブジェクトを参照できる多くの場所では、外部 ID がある場合は、代わりにその外部 ID を使用できます。そのためには、外部 ID を文字列 に追加しますexternalId:。

例えば、UUID (アセットモデル ID) が でa1b2c3d4-5678-90ab-cdef-11111EXAMPLE、外部 ID も のアセットモデルがあるとしますmyExternalId。DescribeAssetモデルを呼び出して、詳細を取得します。の値として次のいずれかを使用できますassetModelId。

- アセットモデル ID (UUID) 自体の場合: a1b2c3d4-5678-90ab-cdef-11111EXAMPLE
- 外部 ID の場合: externalId:myExternalId

```
aws iotsitewise describe-asset-model --asset-model-id a1b2c3d4-5678-90ab-
cdef-11111EXAMPLE
aws iotsitewise describe-asset-model --asset-model-id externalId:myExternalId
```

> **Note**
>
> externalId: プレフィックス自体は外部 ID の一部ではありません。プレフィックスを指定する必要があるのは、UUIDs または外部 ID のいずれかを受け入れる API オペレーションに外部 IDs。例えば、既存のオブジェクトをクエリまたは更新するときに プレフィックスを指定します。
> アセットモデルの作成時など、オブジェクトの外部 ID を定義する場合は、 プレフィックスを含めないでください。

この方法で UUIDs の代わりに外部 IDs を の多くの API オペレーションに使用できますが AWS IoT SiteWise、すべてには使用できません。例えば、 GetAssetPropertyValueは UUIDsを使用する必要があります。外部 ID の使用をサポートしていません。

特定の API オペレーションがこの使用をサポートしているかどうかを確認するには、「 API リファレンス」を参照してください。

アセットモデルとコンポーネントモデルの作成

AWS IoT SiteWise アセットモデルとコンポーネントモデルは、産業データの標準化を促進します。アセットモデルまたはコンポーネントモデルには、名前、説明、アセットプロパティ、および (オプ

ションで) プロパティをグループ化するカスタム複合モデル、またはサブアセンブリのコンポーネントモデルを参照するカスタム複合モデルが含まれます。

- アセットモデルを使用してアセットを作成します。上記の機能に加えて、アセットモデルには、アセット間の関係を定義する階層定義を含めることもできます。
- コンポーネントモデルは、アセットモデルまたは別のコンポーネントモデル内のサブアプライドを表します。コンポーネントモデルを作成するときに、アセットモデルや他のコンポーネントモデルでそのモデルへの参照を追加できます。ただし、コンポーネントモデルから直接アセットを作成することはできません。

アセットモデルまたはコンポーネントモデルを作成したら、カスタム複合モデルを作成してプロパティをグループ化したり、既存のコンポーネントモデルを参照したりできます。

アセットモデルとコンポーネントモデルの作成方法の詳細については、以下のセクションを参照してください。

トピック

アセットモデルを作成する

AWS IoT SiteWise アセットモデルは、産業データの標準化を促進します。アセットモデルには、名前、説明、アセットプロパティ、およびアセット階層定義が含まれます。たとえば、気温、1 分あたりの回転数 (RPM) 、および電力特性を使用して風力タービンモデルを定義できます。次に、正味電力出力特性と風力タービンの階層定義を使用して、風力発電所モデルを定義できます。

> **Note**
>
> - 最下位レベルのノードから業務をモデル化することをお勧めします。たとえば、風力発電所モデルを作成する前に、風力タービンモデルを作成します。アセット階層定義には、既存のアセットモデルへの参照が含まれます。この方法により、モデルの作成時にアセット階層を定義できます。

- アセットモデルに他のアセットモデルを含めることはできません。別のモデル内でサブパーティクルとして参照できるモデルを定義する必要がある場合は、代わりにコンポーネント--> モデルを作成する必要があります。詳細については、「コンポーネントモデルの作成」を参照してください。

以下のセクションでは、 AWS IoT SiteWise コンソールまたは API を使用してアセットモデルを作成する方法について説明します。次のセクションでは、モデルの作成に使用できるさまざまなタイプのアセットプロパティとアセット階層についても説明します。

トピック

- アセットモデルを作成する (コンソール)
- アセットモデルの作成 (AWS CLI)
- アセットモデルの例
- アセットモデル階層の定義

アセットモデルを作成する (コンソール)

AWS IoT SiteWise コンソールを使用してアセットモデルを作成できます。 AWS IoT SiteWise コンソールには、有効なアセットモデルの定義に役立つ式の自動補完など、さまざまな機能が用意されています。

アセットモデルを作成するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Models (モデル)] を選択します。
3. [モデルの作成] を選択します。
4. [モデルの作成] ページで、次の操作を行います。

   a. アセットモデルの [名前] を入力します (**`Wind Turbine`** または **`Wind Turbine Model`** など)。この名前は、このリージョンのアカウントのすべてのモデルで一意である必要があります。

   b. (オプション) モデルの外部 ID を追加します。これはユーザー定義の ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

c. (オプション) モデルの [測定の定義] を追加します。測定値は、機器からのデータストリームを表します。詳細については、「機器からのデータストリームの定義 (測定値)」を参照してください。

d. (オプション) モデルの [定義を変換する] を追加します。変換とは、あるフォームから別のフォームにデータをマッピングするための数式です。詳細については、「データの変換 (変換)」を参照してください。

e. (オプション) モデルの [メトリクスの定義] を追加します。メトリクスとは、時間間隔でのデータを集計する数式です。メトリクスは、関連するアセットからデータを入力できるため、オペレーションまたはオペレーションのサブセットを表す値をコンピューティングすることができます。詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください。

f. (オプション) モデルの [階層の定義] を追加します。階層とは、アセット間の関係です。詳細については、「アセットモデル階層の定義」を参照してください。

g. (オプション) アセットモデルのタグを追加します。詳細については、「リソースにタグを付ける AWS IoT SiteWise」を参照してください。

h. [モデルの作成] を選択します。

アセットモデルを作成すると、 AWS IoT SiteWise コンソールは新しいモデルのページに移動します。このページで、モデルの [ステータス] が表示されます。このステータスは、最初は [作成中] です。このページは自動的に更新されるため、モデルのステータスが更新されるまで待つことができます。

> **Note**
>
> 複雑なモデルの場合は、アセットモデルの作成プロセスに数分かかることがあります。アセットモデルの状態が[ACTIVE] (アクティブ) になると、アセットモデルを使用してアセットを作成できるようになります。詳細については、「アセットおよびモデルの状態」を参照してください。

5. (オプション) アセットモデルを作成したら、アセットモデルをエッジに設定することができます。 SiteWise Edge の詳細については、「」を参照してください[エッジデータ処理を有効にする。](#)。

a. モデルページで、[Configure for Edge] (エッジの設定) を選択します。

b. モデル設定ページで、モデルのエッジ設定を選択します。これにより、 がこのアセットモデルに関連付けられたプロパティを計算して保存 AWS IoT SiteWise できる場所が制御され

ます。エッジ用のモデルの設定については、the section called “エッジ機能の設定。” を参照してください。

c. カスタムエッジ設定 で、各アセットモデルプロパティ AWS IoT SiteWise を計算して保存する場所を選択します。

> **Note**
>
> 関連する変換とメトリクスは、同じ場所に設定されている必要があります。エッジ用のモデルの設定については、the section called “エッジ機能の設定。” を参照してください。

d. [保存] を選択します。モデルページで、[Edge configuration] (Edgeの設定) が[Configured] (設定済み) になっていることを確認します。

アセットモデルの作成 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用してアセットモデルを作成できます。

CreateAssetModel オペレーションを使用して、プロパティと階層を持つアセットモデルを作成します。このオペレーションでは、次の構造を持つペイロードが必要です。

```
{
  "assetModelType": "ASSET_MODEL",
  "assetModelName": "String",
  "assetModelDescription": "String",
  "assetModelProperties": Array of AssetModelProperty,
  "assetModelHierarchies": Array of AssetModelHierarchyDefinition
}
```

アセットモデルを作成するには (AWS CLI)

1. asset-model-payload.json という名前のファイルを作成して、次の JSON オブジェクトをファイルにコピーします。

```
{
  "assetModelType": "ASSET_MODEL",
  "assetModelName": "",
  "assetModelDescription": "",
  "assetModelProperties": [
```

```
  ],
  "assetModelHierarchies": [

  ],
  "assetModelCompositeModels": [

  ]
}
```

2. 任意の JSON テキストエディタを使用して、以下の asset-model-payload.json ファイルを編集します。

   a. アセットモデルの名前 (assetModelName) を入力します (**Wind Turbine** または **Wind Turbine Model** など)。この名前は、この のアカウント内のすべてのアセットモデルとコンポーネントモデルで一意である必要があります AWS リージョン。

   b. (オプション) アセットモデルの外部 ID (assetModelExternalId) を入力します。これはユーザー定義の ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

   c. (オプション) アセットモデルの説明 (assetModelDescription) を入力するか、assetModelDescription キーと値のペアを削除します。

   d. (オプション) モデルのアセットプロパティ (assetModelProperties) を定義します。詳細については、「データのプロパティを定義する。」を参照してください。

   e. (オプション) モデルのアセット階層 (assetModelHierarchies) を定義します。詳細については、「アセットモデル階層の定義」を参照してください。

   f. (オプション) モデルのアラームを定義します。アラームは、他のプロパティをモニタリングし、機器やプロセスに注意が必要な時期を特定することができます。各アラーム定義は、アラームが使用する一連のプロパティを標準化した複合モデル (assetModelCompositeModels) である。詳細については、「アラームによるデータのモニタリング。」および「アセットモデルにおけるアラームの定義」を参照してください。

   g. (オプション) アセットモデルのタグ (tags) を追加します。詳細については、「リソースにタグを付ける AWS IoT SiteWise」を参照してください。

3. 次のコマンドを実行して、JSON ファイルの定義からアセットモデルを作成します。

```
aws iotsitewise create-asset-model --cli-input-json file://asset-model-payload.json
```

このオペレーションは、アセットを作成するときに参照する assetModelId を含むレスポンスを返します。レスポンスには、モデルの状態 (assetModelStatus.state) も含まれま

す。これは、最初は CREATING です。アセットモデルのステータスは、変更が反映されるまで CREATING です。

> **Note**
>
> 複雑なモデルの場合は、アセットモデルの作成プロセスに数分かかることがあります。アセットモデルの現在のステータスを確認するには、 を指定して DescribeAssetModel オペレーションを使用しますassetModelId。アセットモデルのステータスが ACTIVE になったら、アセットモデルを使用してアセットを作成できます。詳細については、「アセットおよびモデルの状態」を参照してください。

4. (オプション) アセットモデルのカスタム複合モデルを作成します。カスタム複合モデルでは、モデル内のプロパティをグループ化したり、コンポーネントモデルを参照してサブパーティクルを含めたりできます。詳細については、「カスタム複合モデルの作成 (コンポーネント)」を参照してください。

アセットモデルの例

このセクションでは、 AWS CLI および AWS IoT SiteWise SDKs。これらのアセットモデルは、風力タービンと風力発電所を表しています。風力タービンは、センサーの生データを取り込み、電力や平均風速などの値を算出する。風力発電アセットでは、風力発電所内のすべての風車の総電力量などを算出します。

トピック

- 風力タービンアセットモデル
- 風力発電所アセットモデル

風力タービンアセットモデル

次のアセットモデルは、風力発電所のタービンを表しています。風力タービンはセンサーのデータを取り込み、電力や平均風速などの値を算出する。

> **Note**
>
> このサンプルモデルは、 AWS IoT SiteWise デモの風力タービンモデルに似ています。詳細については、「AWS IoT SiteWise デモを使う」を参照してください。

```
{
  "assetModelType": "ASSET_MODEL",
  "assetModelName": "Wind Turbine Asset Model",
  "assetModelDescription": "Represents a turbine in a wind farm.",
  "assetModelProperties": [
    {
      "name": "Location",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "Renton"
        }
      }
    },
    {
      "name": "Make",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "Amazon"
        }
      }
    },
    {
      "name": "Model",
      "dataType": "INTEGER",
      "type": {
        "attribute": {
          "defaultValue": "500"
        }
      }
    },
    {
      "name": "Torque (KiloNewton Meter)",
      "dataType": "DOUBLE",
      "unit": "kNm",
      "type": {
        "measurement": {}
      }
    },
    {
      "name": "Wind Direction",
      "dataType": "DOUBLE",
```

```
      "unit": "Degrees",
      "type": {
        "measurement": {}
      }
    },
    {
      "name": "RotationsPerMinute",
      "dataType": "DOUBLE",
      "unit": "RPM",
      "type": {
        "measurement": {}
      }
    },
    {
      "name": "Wind Speed",
      "dataType": "DOUBLE",
      "unit": "m/s",
      "type": {
        "measurement": {}
      }
    },
    {
      "name": "RotationsPerSecond",
      "dataType": "DOUBLE",
      "unit": "RPS",
      "type": {
        "transform": {
          "expression": "rpm / 60",
          "variables": [
            {
              "name": "rpm",
              "value": {
                "propertyId": "RotationsPerMinute"
              }
            }
          ]
        }
      }
    },
    {
      "name": "Overdrive State",
      "dataType": "DOUBLE",
      "type": {
        "transform": {
```

```
          "expression": "gte(torque, 3)",
          "variables": [
            {
              "name": "torque",
              "value": {
                "propertyId": "Torque (KiloNewton Meter)"
              }
            }
          ]
        }
      }
    },
    {
      "name": "Average Power",
      "dataType": "DOUBLE",
      "unit": "Watts",
      "type": {
        "metric": {
          "expression": "avg(torque) * avg(rps) * 2 * 3.14",
          "variables": [
            {
              "name": "torque",
              "value": {
                "propertyId": "Torque (Newton Meter)"
              }
            },
            {
              "name": "rps",
              "value": {
                "propertyId": "RotationsPerSecond"
              }
            }
          ],
          "window": {
            "tumbling": {
              "interval": "5m"
            }
          }
        }
      }
    },
    {
      "name": "Average Wind Speed",
      "dataType": "DOUBLE",
```

```
      "unit": "m/s",
      "type": {
        "metric": {
          "expression": "avg(windspeed)",
          "variables": [
            {
              "name": "windspeed",
              "value": {
                "propertyId": "Wind Speed"
              }
            }
          ],
          "window": {
            "tumbling": {
              "interval": "5m"
            }
          }
        }
      }
    },
    {
      "name": "Torque (Newton Meter)",
      "dataType": "DOUBLE",
      "unit": "Nm",
      "type": {
        "transform": {
          "expression": "knm * 1000",
          "variables": [
            {
              "name": "knm",
              "value": {
                "propertyId": "Torque (KiloNewton Meter)"
              }
            }
          ]
        }
      }
    },
    {
      "name": "Overdrive State Time",
      "dataType": "DOUBLE",
      "unit": "Seconds",
      "type": {
        "metric": {
```

```
          "expression": "statetime(overdrive_state)",
          "variables": [
            {
              "name": "overdrive_state",
              "value": {
                "propertyId": "Overdrive State"
              }
            }
          ],
          "window": {
            "tumbling": {
              "interval": "5m"
            }
          }
        }
      }
    }
  ],
  "assetModelHierarchies": []
}
```

風力発電所アセットモデル

次のアセットモデルは、複数の風力タービンで構成される風力発電所を表しています。このアセットモデルは、風力タービンモデルへの[hierarchy] (階層) を定義しています。これにより、風力発電所は、風力発電所内のすべての風力タービンのデータから値 (平均電力など) を計算できます。

> **Note**
>
> このサンプルモデルは、 AWS IoT SiteWise デモの風力発電所モデルに似ています。詳細については、「AWS IoT SiteWise デモを使う」を参照してください。

このアセットモデルは、風力タービンアセットモデル によって異なります。`propertyId` および `childAssetModelId` の値を、既存の風力タービンアセットモデルの値に置き換えます。

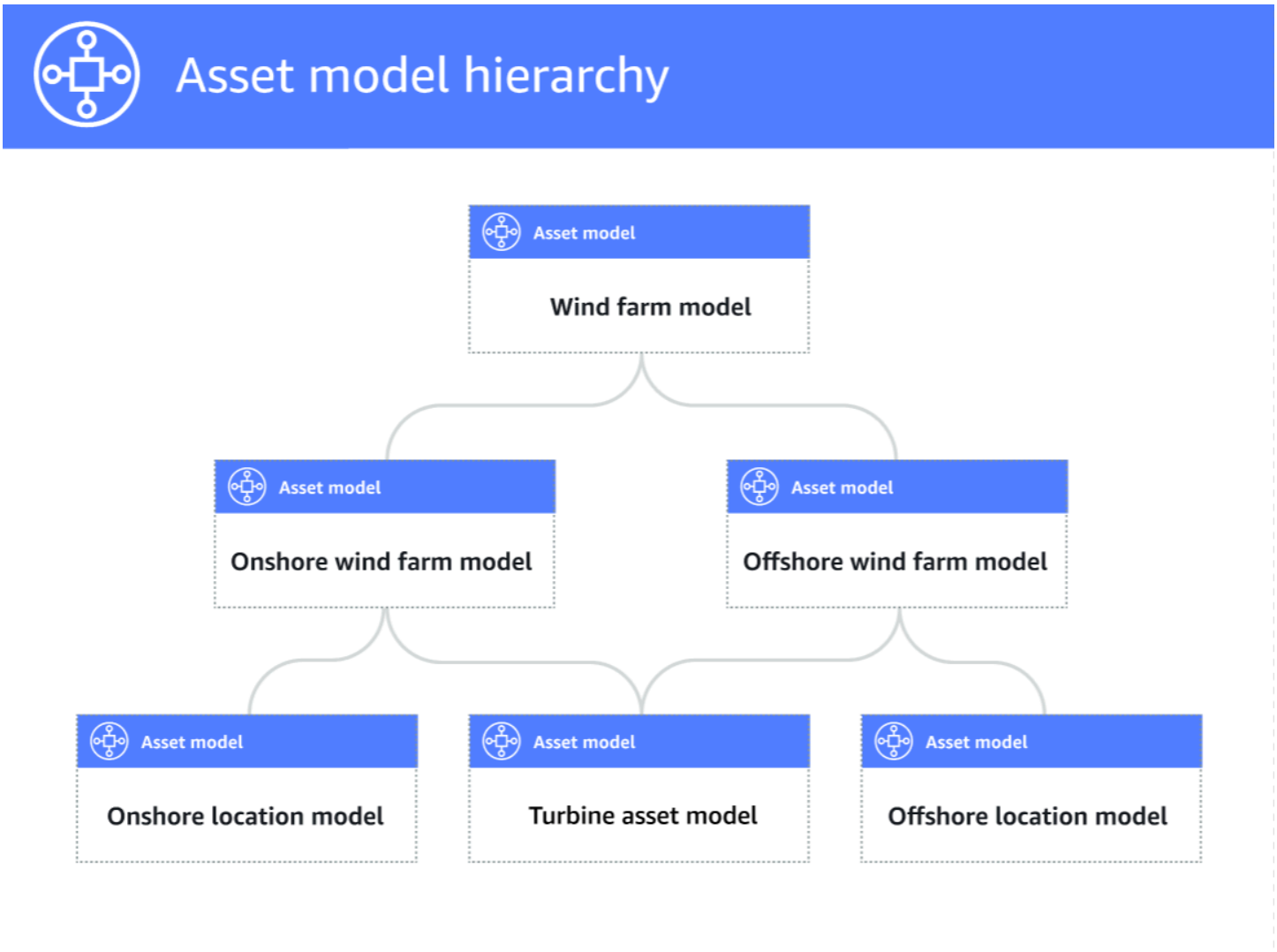
```
{
  "assetModelName": "Wind Farm Asset Model",
  "assetModelDescription": "Represents a wind farm.",
  "assetModelProperties": [
    {
      "name": "Code",
```

```
      "dataType": "INTEGER",
      "type": {
        "attribute": {
          "defaultValue": "300"
        }
      }
    },
    {
      "name": "Location",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "Renton"
        }
      }
    },
    {
      "name": "Reliability Manager",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "Mary Major"
        }
      }
    },
    {
      "name": "Total Overdrive State Time",
      "dataType": "DOUBLE",
      "unit": "seconds",
      "type": {
        "metric": {
          "expression": "sum(overdrive_state_time)",
          "variables": [
            {
              "name": "overdrive_state_time",
              "value": {
                "propertyId": "ID of Overdrive State Time property in Wind Turbine
Asset Model",
                "hierarchyId": "Turbine Asset Model"
              }
            }
          ],
          "window": {
            "tumbling": {
```

```
              "interval": "5m"
            }
          }
        }
      }
    },
    {
      "name": "Total Average Power",
      "dataType": "DOUBLE",
      "unit": "Watts",
      "type": {
        "metric": {
          "expression": "sum(turbine_avg_power)",
          "variables": [
            {
              "name": "turbine_avg_power",
              "value": {
                "propertyId": "ID of Average Power property in Wind Turbine Asset
 Model",
                "hierarchyId": "Turbine Asset Model"
              }
            }
          ],
          "window": {
            "tumbling": {
              "interval": "5m"
            }
          }
        }
      }
    }
  ],
  "assetModelHierarchies": [
    {
      "name": "Turbine Asset Model",
      "childAssetModelId": "ID of Wind Turbine Asset Model"
    }
  ]
}
```

アセットモデル階層の定義

アセットモデル階層を定義することで、産業オペレーションにおけるアセットモデル間の論理的な関連付けを作成することができます。例えば、陸上風力発電所と洋上風力発電所からなる風力発電所を定義することができます。陸上風力発電所には、タービンと陸上の場所が含まれます。洋上風力発電所には、タービンと沖合の場所が含まれます。



子アセットモデルを親アセットモデルに階層的に関連付けると、親アセットモデルのメトリクスは子アセットモデルのメトリクスからデータを入力できるようになります。アセットモデル階層とメトリクスを使用して、オペレーションまたはオペレーションのサブセットに関するインサイトを提供する統計情報をコンピューティングできます。詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください。

各階層は、親アセットモデルと子アセットモデルの関係を定義します。親アセットモデルでは、同じ子アセットモデルに対して複数の階層を定義することができます。例えば、風力発電所に2種類の風

カタービンがあり、すべての風力タービンが同じアセットモデルで表現されている場合、それぞれの種類に階層を定義することができます。そして、風力発電所のモデルで指標を定義し、風力タービンの種類ごとに独立した統計量と複合的な統計量をコンピューティングすることができます。

親アセットモデルには、複数の子アセットモデルを関連付けることができます。例えば、陸上風力発電所と洋上風力発電所が2つの異なるアセットモデルで表現されている場合、これらのアセットモデルを同じ親風力発電所アセットモデルに関連付けることができます。

また、子アセットモデルは、複数の親アセットモデルと関連付けることができます。例えば、2種類の風力発電所があり、すべての風力タービンが同じアセットモデルで表現されている場合、風力タービンアセットモデルと異なる風力発電所アセットモデルを関連付けることができます。

> **Note**
>
> アセットモデル階層を定義する場合、子アセットモデルは ACTIVE であるか、以前の ACTIVE バージョンである必要があります。詳細については、「アセットおよびモデルの状態」を参照してください。

階層アセットモデルを定義してアセットを作成したら、アセットを関連付けて親子関係を完了できます。詳細については、「アセットを作成する」および「アセットの関連付けと関連付け解除」を参照してください。

トピック

- アセットモデル階層の定義 (コンソール)
- アセット階層の定義 (AWS CLI)

アセットモデル階層の定義 (コンソール)

AWS IoT SiteWise コンソールでアセットモデルの階層を定義するときは、次のパラメータを指定します。

- [Hierarchy name] (階層名) - 階層の名前 (**`Wind Turbines`** など)。
- [Hierarchy model] (階層モデル) - 子アセットモデル。
- 階層外部 ID (オプション) – これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

詳細については、「アセットモデルを作成する (コンソール) 」を参照してください。

アセット階層の定義 (AWS CLI)

AWS IoT SiteWise API を使用してアセットモデルの階層を定義する場合は、次のパラメータを指定します。

- `name` 階層の名前 (など)。**`Wind Turbines`**
- `childAssetModelId` – 階層の子アセットモデルの ID または外部 ID。ListAssetModels オペレーションを使用して、既存のアセットモデルの ID を検索できます。

Example 階層定義の例

次の例は、風力発電所の風力タービンとの関係を表すアセットモデル階層を示しています。このオブジェクトは階層 AssetModelの例です。詳細については、「アセットモデルの作成 (AWS CLI)」を参照してください。

```
{
  ...
  "assetModelHierarchies": [
    {
      "name": "Wind Turbines",
      "childAssetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
    },
  ]
}
```

コンポーネントモデルの作成

AWS IoT SiteWise コンポーネントモデルを使用して、アセットモデルやその他のコンポーネントモデルから参照できるサブアセンブリを定義します。このようにして、コンポーネントの定義を他の複数のモデルで再利用することも、同じモデル内で複数回再利用することもできます。

コンポーネントモデルを定義するプロセスは、アセットモデルを定義するのと非常によく似ています。アセットモデルと同様に、コンポーネントモデルには名前、説明、アセットプロパティがあります。ただし、コンポーネントモデル自体を使用してアセットを直接作成することはできないため、コンポーネントモデルにアセット階層定義を含めることはできません。また、コンポーネントモデルではアラームを定義できません。

例えば、モーター温度、エンコーダー温度、およびパーティショニング抵抗プロパティを使用して、サーボモーターのコンポーネントを定義できます。次に、CNC マシンなどのサーボモーターを含む機器のアセットモデルを定義できます。

> **Note**
>
> - 最下位レベルのノードから業務をモデル化することをお勧めします。例えば、CNC マシンのアセットモデルを作成する前に、サーボモーターコンポーネントを作成します。アセットモデルには、既存のコンポーネントモデルへの参照が含まれています。
> - コンポーネントモデルから直接アセットを作成することはできません。コンポーネントを使用するアセットを作成するには、アセットのアセットモデルを作成する必要があります。次に、コンポーネントを参照するカスタム複合モデルを作成します。アセットモデルの作成の詳細については、「」を参照してください。カスタム複合モデルの作成アセットモデルを作成するの詳細については、「」を参照してくださいカスタム複合モデルの作成 (コンポーネント)。

以下のセクションでは、 AWS IoT SiteWise API を使用してコンポーネントモデルを作成する方法について説明します。

トピック

- コンポーネントモデルの作成 (AWS CLI)
- コンポーネントモデルの例

コンポーネントモデルの作成 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用してコンポーネントモデルを作成できます。

Model CreateAssetオペレーションを使用して、プロパティを持つコンポーネントモデルを作成します。このオペレーションでは、次の構造を持つペイロードが必要です。

```
{
  "assetModelType": "COMPONENT_MODEL",
  "assetModelName": "String",
  "assetModelDescription": "String",
  "assetModelProperties": Array of AssetModelProperty,
}
```

コンポーネントモデルを作成するには (AWS CLI)

1. という名前のファイルを作成しcomponent-model-payload.json、次の JSON オブジェクトをファイルにコピーします。

```
{
  "assetModelType": "COMPONENT_MODEL",
  "assetModelName": "",
  "assetModelDescription": "",
  "assetModelProperties": [

  ]
}
```

2. 任意の JSON テキストエディタを使用して、以下の component-model-payload.json ファイルを編集します。

   a. **Servo Motor** や など、コンポーネントモデルの名前 (assetModelName) を入力しますServo Motor Model。この名前は、この のアカウント内のすべてのアセットモデルとコンポーネントモデルで一意である必要があります AWS リージョン。

   b. （ オプション) コンポーネントモデルの外部 ID (assetModelExternalId) を入力します。これはユーザー定義の ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

   c. (オプション) アセットモデルの説明 (assetModelDescription) を入力するか、assetModelDescription キーと値のペアを削除します。

   d. （ オプション) コンポーネントモデルのアセットプロパティ (assetModelProperties) を定義します。詳細については、「データのプロパティを定義する。」を参照してください。

   e. (オプション) アセットモデルのタグ (tags) を追加します。詳細については、「リソースにタグを付ける AWS IoT SiteWise」を参照してください。

3. 次のコマンドを実行して、JSON ファイルの定義からコンポーネントモデルを作成します。

```
aws iotsitewise create-asset-model --cli-input-json file://component-model-
payload.json
```

オペレーションは、アセットモデルまたは別のコンポーネントモデルでコンポーネントモデルへの参照を追加するときに参照assetModelIdする を含むレスポンスを返します。レスポンスには、モデルの状態 (assetModelStatus.state) も含まれます。これは、最初は CREATING です。コンポーネントモデルのステータスは、変更が反映されCREATINGるまで です。

> **Note**
>
> コンポーネントモデルの作成プロセスは、複雑なモデルの場合、数分かかることがあります。コンポーネントモデルの現在のステータスを確認するには、 を指定して DescribeAssetModel オペレーションを使用しますassetModelId。コンポーネントモデルのステータスが になったらACTIVE、アセットモデルまたは他のコンポーネントモデルでコンポーネントモデルへの参照を追加できます。詳細については、「アセットおよびモデルの状態」を参照してください。

4. (オプション) コンポーネントモデルのカスタム複合モデルを作成します。カスタム複合モデルでは、モデル内のプロパティをグループ化したり、別のコンポーネントモデルを参照してサブアプライドを含めたりできます。詳細については、「カスタム複合モデルの作成 (コンポーネント)」を参照してください。

コンポーネントモデルの例

このセクションでは、 AWS CLI および AWS IoT SiteWise SDKsでコンポーネントモデルを作成するために使用できるコンポーネントモデル定義の例を示します。このコンポーネントモデルは、CNC マシンなどの別の機器内で使用できるサーボモーターを表します。

トピック

- Servo モーターコンポーネントモデル

Servo モーターコンポーネントモデル

次のコンポーネントモデルは、CNC マシンなどの機器内で使用できるサーボモーターを表しています。サーボモーターは、温度や耐電力など、さまざまな測定値を提供します。これらの測定値は、サーボモーターコンポーネントモデルを参照するアセットモデルから作成されたアセットのプロパティとして使用できます。

```
{
    "assetModelName": "ServoMotor",
    "assetModelType": "COMPONENT_MODEL",
    "assetModelProperties": [
        {
            "dataType": "DOUBLE",
            "name": "Servo Motor Temperature",
```

```
            "type": {
            "measurement": {}
            },
            "unit": "Celsius"
        },
        {
            "dataType": "DOUBLE",
            "name": "Spindle speed",
            "type": {
            "measurement": {}
            },
            "unit": "rpm"
        }
    ]
}
```

データのプロパティを定義する。

アセットプロパティは、アセットデータを含む各アセット内の構造です。アセットプロパティには、次の型があります。

- [Attributes] (属性) – デバイスの製造元や地理的なリージョンなど、一般的に静的なプロパティを持つアセット。詳細については、「静的データ (属性) の定義」を参照してください。
- [測定値] - タイムスタンプ付きの回転速度値や摂氏のタイムスタンプ付きの温度値など、アセットの raw デバイスのセンサーデータストリーム。測定は、データストリームエイリアスによって定義されます。詳細については、「機器からのデータストリームの定義 (測定値) 」を参照してください。
- [Transforms] (変換) - タイムスタンプ付き華氏温度値など、アセットの変換された時系列値。変換は、式とその式で使用する変数によって定義されます。詳細については、「データの変換 (変換) 」を参照してください。
- [Metrics] (メトリクス) - 時間当たりの平均温度など、指定した時間間隔で集計されたアセットのデータ。メトリクスは、時間間隔、式、およびその式で使用する変数によって定義されます。メトリック表現は、関連するアセットのメトリクスプロパティを入力できるため、オペレーションまたはオペレーションのサブセットを表すメトリックをコンピューティングすることができます。詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください。

詳細については、「アセットモデルを作成する」を参照してください。

測定、変換、およびメトリクスを使用して、総合設備効率 (OEE) をコンピューティングする方法の例については、「でのOEEの計算 AWS IoT SiteWise」を参照してください。

トピック

- 静的データ (属性) の定義
- 機器からのデータストリームの定義 (測定値)
- データの変換 (変換)
- プロパティおよびその他のアセットのデータ (指標) の集計
- 数式の使用

静的データ (属性) の定義

アセット属性は、デバイスの製造元や地理的位置など、一般的に静的な情報を表します。アセットモデルから作成する各アセットには、このモデルの属性が含まれています。

トピック

- 属性の定義 (コンソール)
- 属性の定義 (AWS CLI)

属性の定義 (コンソール)

AWS IoT SiteWise コンソールでアセットモデルの属性を定義するときは、次のパラメータを指定します。

- [Name] (名前) - プロパティの名前。
- [Default value] (デフォルト値) –(オプション) この属性のデフォルト値。モデルから作成されたアセットは、属性に対してこの値を持ちます。モデルから作成されたアセットのデフォルト値を上書きする方法の詳細については、「属性値の更新」を参照してください。
- [Data type] (データ型) - プロパティのデータ型で、次のいずれかになります。
  - [String] (文字列) – 最大 1024 バイトの文字列。
  - [Integer] (整数) - [-2,147,483,648, 2,147,483,647] の範囲を持つ 32 ビットの符合付き整数。
  - [Double] (倍) – [-10^100、10^100] の範囲および IEEE 754 倍精度の浮動小数点数。
  - [Boolean] (ブール値) - `true` または `false`。
- 外部 ID – (オプション) これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

詳細については、「アセットモデルを作成する (コンソール)」を参照してください。

属性の定義 (AWS CLI)

AWS IoT SiteWise API を使用してアセットモデルの属性を定義する場合は、次のパラメータを指定します。

- `name` - プロパティの名前。
- `defaultValue` -(オプション) この属性のデフォルト値。モデルから作成されたアセットは、属性に対してこの値を持ちます。モデルから作成されたアセットのデフォルト値を上書きする方法の詳細については、「属性値の更新」を参照してください。
- `dataType` - プロパティのデータ型。次のいずれかです。
  - `STRING` - 最大 1024 バイトの文字列。
  - `INTEGER` - [-2,147,483,648, 2,147,483,647] の範囲を持つ 32 ビットの符合付き整数。
  - `DOUBLE` - [-10 ^ 100、10 ^ 100] の範囲および IEEE 754 倍精度の浮動小数点数。
  - `BOOLEAN` – `true` または `false`
- `externalId` – (オプション) これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

Example 属性定義の例

次の例では、デフォルト値でアセットのモデル番号を表す属性を示します。このオブジェクトは、属性 を含むAssetModelプロパティの例です。このオブジェクトをCreateAssetモデルリクエストペイロードの一部として指定して、属性プロパティを作成できます。詳細については、「アセットモデルの作成 (AWS CLI)」を参照してください。

```
{
...
"assetModelProperties": [
{
  "name": "Model number",
  "dataType": "STRING",
  "type": {
    "attribute": {
      "defaultValue": "BLT123"
    }
  }
}
],
```

```
...
}
```

機器からのデータストリームの定義 (測定値)

測定値は、タイムスタンプ付き温度値やタイムスタンプ付きの 1 分あたりの回転数 (RPM) 値など、デバイスの raw センサーデータストリームを表します。

トピック

- 測定値の定義 (コンソール)
- 測定値の定義 (AWS CLI)

測定値の定義 (コンソール)

AWS IoT SiteWise コンソールでアセットモデルの測定を定義するときは、次のパラメータを指定します。

- [Name] (名前) - プロパティの名前。
- [Unit] (単位) - プロパティの科学単位 (mm、摂氏など)。
- [Data type] (データ型) - プロパティのデータ型で、次のいずれかになります。
  - [String] (文字列) – 最大 1024 バイトの文字列。
  - [Integer] (整数) - [-2,147,483,648, 2,147,483,647] の範囲を持つ 32 ビットの符合付き整数。
  - [Double] (倍) – [-10^100、10^100] の範囲および IEEE 754 倍精度の浮動小数点数。
  - [Boolean] (ブール値) - `true` または `false`。
- 外部 ID – (オプション) これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

詳細については、「アセットモデルを作成する (コンソール) 」を参照してください。

測定値の定義 (AWS CLI)

AWS IoT SiteWise API を使用してアセットモデルの測定を定義するときは、次のパラメータを指定します。

- `name` - プロパティの名前。
- `dataType` - プロパティのデータ型。次のいずれかです。
  - `STRING` - 最大 1024 バイトの文字列。

- INTEGER - [-2,147,483,648, 2,147,483,647] の範囲を持つ 32 ビットの符合付き整数。
- DOUBLE - [-10 ^ 100、10 ^ 100] の範囲および IEEE 754 倍精度の浮動小数点数。
- BOOLEAN – true または false
- unit –(オプション) プロパティの科学単位 (mm、摂氏など)。
- externalId – (オプション) これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

Example 測定値の定義の例

次の例は、アセットの温度センサーの読み取り値を表す測定値を示しています。このオブジェクトは、測定 を含むAssetModelプロパティの例です。このオブジェクトをCreateAssetモデルリクエストペイロードの一部として指定して、測定プロパティを作成できます。詳細については、「アセットモデルの作成 (AWS CLI)」を参照してください。

アセットモデルを定義するときは、[Measurement] (測定値) は空の構造になります。これは、後で各アセットが一意のデバイスデータストリームを使用するように設定するためです。アセットの測定プロパティをデバイスのセンサーデータストリームに接続する方法の詳細については、「アセットプロパティへの産業データストリームのマッピング」を参照してください。

```
{
      ...
      "assetModelProperties": [
      {
          "name": "Temperature C",
          "dataType": "DOUBLE",
          "type": {
              "measurement": {}
          },
          "unit": "Celsius"
      }
  ],
      ...
}
```

データの変換 (変換)

変換は、アセットプロパティのデータポイントを別のフォームにマッピングする数式です。変換式は、アセットプロパティ変数、リテラル、演算子、関数で構成されます。変換されたデータポイント

は、入力データポイント one-to-one との関係を保持します。 は、入力プロパティが新しいデータポイントを受信するたびに、新しい変換されたデータポイントを AWS IoT SiteWise 計算します。

たとえば、アセットに単位が摂氏の名前 Temperature_C の温度測定ストリームがある場合、数式 Temperature_F = 9/5 * Temperature_C + 32 を使用して各データポイントを華氏に変換できます。がTemperature_C測定ストリームでデータポイント AWS IoT SiteWise を受信するたびに、対応するTemperature_F値は数秒以内に計算され、 Temperature_F プロパティとして使用できます。

変換に複数の変数が含まれる場合、先に到着したデータポイントからすぐにコンピューティングが開始されます。例えば、ある部品メーカーが製品の品質をモニタリングするためにトランスを使用する場合を考えてみましょう。部品の種類によって異なる基準を用いて、メーカーは次のような測定値で工程を表現しています。

- Part_Number - 部品の種類を識別するための文字列。
- Good_Count - 部品が規格に適合している場合、1増加する整数。
- Bad_Count - 部品が規格に適合しない場合、1増加する整数。

また、メーカーは Quality_Monitor に等しい変換 if(eq(Part_Number, "BLT123") and (Bad_Count / (Good_Count + Bad_Count) > 0.1), "Caution", "Normal") を作成します。

この変換は、特定の部品型で生産された不良部品の割合をモニタリングします。部品番号がBLT123で、不良部品の割合が10% (0.1) を超える場合、変換は "Caution" を返します。そうでない場合は、変換は "Normal" を返します。

> **Note**
>
> - Part_Number が他の測定値より先に新しいデータポイントを受信した場合、NQuality_Monitor変換は新しい Part_Number 値と最新の Good_Count および Bad_Count 値を使用します。エラーを回避するため、次の製造の前に Good_Count、Bad_Count をリセットしてください。
> - すべての変数が新しいデータポイントを受け取った後にのみ式を評価したい場合は、[metrics] (メトリクス) を使用します。

トピック

- 変換の定義 (コンソール)
- 変換の定義 (AWS CLI)

変換の定義 (コンソール)

AWS IoT SiteWise コンソールでアセットモデルの変換を定義するときは、次のパラメータを指定します。

- [Name] (名前) - プロパティの名前。
- [Unit] (単位) - プロパティの科学単位 (mm、摂氏など)。
- [Data type] (データ型) - 変換のデータ型で、[Double] (倍) または [String] (文字列) です。
- 外部 ID – (オプション) これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。
- [Formula] (コンピューティング式) – 変換式。変換式では、集計関数や一時関数は使用できません。自動完了機能を開くには、入力を開始するか、下矢印キーを押します。詳細については、「数式の使用」を参照してください。

> **Important**
>
> 変換は、整数、倍、ブール値、文字列型のプロパティを入力することができます。ブール値は、0 (false) 、1 (true) に変換されます。
> 変換には、属性でないプロパティを1つ以上、任意の数だけ入力する必要があります。
> AWS IoT SiteWise は、属性でない入力プロパティが新しいデータポイントを受け取るたびに、新しい変換後のデータポイントをコンピューティングする。新しい属性値は変換更新を起動しません。アセットプロパティデータの API オペレーションのリクエストレートは、変換計算の結果にも適用されます。
> 数式では、倍値または文字列値のみを出力できます。入れ子になった式は、文字列など他のデータ型を出力することができますが、式全体としては、数値または文字列として評価されなければなりません。[jp function] (jp関数) を使うと、文字列を数値に変換することができます。ブール値は 1 (true) または 0 (false) でなければならない。詳細については、「未定義の値、無限の値、およびオーバーフロー値」を参照してください。

詳細については、「アセットモデルを作成する (コンソール) 」を参照してください。

変換の定義 (AWS CLI)

AWS IoT SiteWise API を使用してアセットモデルの変換を定義するときは、次のパラメータを指定します。

- `name` - プロパティの名前。
- `unit` –(オプション) プロパティの科学単位 (mm、摂氏など)。
- `dataType` - 変換のデータ型は、`DOUBLE` または `STRING` である必要があります。
- `externalId` – (オプション) これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。
- `expression` - 変換式。変換式では、集計関数や一時関数は使用できません。詳細については、「数式の使用」を参照してください。
- `variables` - 式で使用するアセットの他のプロパティを定義する変数のリスト。各変数構造には、式で使用する単純な名前と、その変数にリンクされるプロパティを定義する `value` 構造が含まれます。`value` 構造体には、以下の情報が含まれています。
  - `propertyId` - 値を入力するプロパティの ID。ID の代わりにプロパティの名前を使用できます。

> **Important**
>
> 変換は、整数、倍、ブール値、文字列型のプロパティを入力することができます。ブール値は、0 (false) 、1 (true) に変換されます。
> 変換には、属性でないプロパティを1つ以上、任意の数だけ入力する必要があります。
> AWS IoT SiteWise は、属性でない入力プロパティが新しいデータポイントを受け取るたびに、新しい変換後のデータポイントをコンピューティングする。新しい属性値は変換更新を起動しません。アセットプロパティデータの API オペレーションのリクエストレートは、変換計算の結果にも適用されます。
> 数式では、倍値または文字列値のみを出力できます。入れ子になった式は、文字列など他のデータ型を出力することができますが、式全体としては、数値または文字列として評価されなければなりません。[jp function] (jp関数) を使うと、文字列を数値に変換することができます。ブール値は 1 (true) または 0 (false) でなければならない。詳細については、「未定義の値、無限の値、およびオーバーフロー値」を参照してください。

Example [Transform job definition] (変換定義)

次の例は、アセットの温度測定データを摂氏から華氏に変換する変換プロパティを示しています。このオブジェクトは、変換 を含む AssetModelプロパティの例です。このオブジェクトをCreateAssetモデルリクエストペイロードの一部として指定して、変換プロパティを作成できます。詳細については、「アセットモデルの作成 (AWS CLI)」を参照してください。

```
{
...
"assetModelProperties": [
...
{
  "name": "Temperature F",
  "dataType": "DOUBLE",
  "type": {
    "transform": {
      "expression": "9/5 * temp_c + 32",
      "variables": [
        {
          "name": "temp_c",
          "value": {
            "propertyId": "Temperature C"
          }
        }
      ]
    }
  },
  "unit": "Fahrenheit"
}
],
...
}
```

Example 3つの変数を含む変換定義。

次の例では、BLT123の部品の10%以上が基準を満たしていない場合に警告メッセージ ("Caution") を返す変換プロパティを示しています。それ以外の場合は、情報メッセージ ("Normal") を返す。

```
{
...
"assetModelProperties": [
...
```

```
{
"name": "Quality_Monitor",
"dataType": "STRING",
"type": {
    "transform": {
        "expression": "if(eq(Part_Number,"BLT123") and (Bad_Count / (Good_Count +
 Bad_Count) > 0.1), "Caution", "Normal")",
        "variables": [
            {
                "name": "Part_Number",
                "value": {
                    "propertyId": "Part Number"
                }
            },
            {
                "name": "Good_Count",
                "value": {
                    "propertyId": "Good Count"
                }
            },
            {
                "name": "Bad_Count",
                "value": {
                    "propertyId": "Bad Count"
                }
            }
        ]
    }
}
}
...
}
```

プロパティおよびその他のアセットのデータ (指標) の集計

メトリクスは、集計関数を使用してすべての入力データポイントを処理し、指定された時間間隔ごとに 1 つのデータポイントを出力する数式です。たとえば、メトリクスは、温度データストリームから時間当たりの平均温度を計算できます。

メトリクスは、関連付けられたアセットのメトリクスからデータを入力できるため、オペレーションまたはオペレーションのサブセットに関するインサイトを提供する統計情報を計算できます。たとえば、メトリクスは、風力発電所のすべての風力タービンの平均時間温度を計算できます。アセット間の関連付けを定義する方法の詳細については、「アセットモデル階層の定義」を参照してください。

メトリクスは、各時間間隔でのデータを集約することなく、他のプロパティからデータを入力することも可能です。コンピューティング式に[attribute] (属性) を指定した場合、 AWS IoT SiteWise はその属性の[latest] (最新) の値をコンピューティング式に使用します。式でメトリクスを指定すると、 は式を計算する時間間隔の最後の値 AWS IoT SiteWise を使用します。つまり、 `OEE = Availability * Quality * Performance`、 `Availability`、 `Quality` はすべて同じアセットモデル上の他のメトリクスであり、`Performance` のようなメトリクスを定義することができるのです。

AWS IoT SiteWise は、すべてのアセットプロパティの基本的な集計メトリクスのセットも自動的に計算します。計算コストを削減するために、基本的な計算にカスタムメトリクスを定義する代わりに、これらの集計を使用できます。詳細については、「アセットプロパティ集計のクエリ」を参照してください。

トピック

- メトリクスの定義 (コンソール)
- メトリクスの定義 (AWS CLI)

メトリクスの定義 (コンソール)

AWS IoT SiteWise コンソールでアセットモデルのメトリクスを定義するときは、次のパラメータを指定します。

- [Name] (名前) - プロパティの名前。
- [Data type] (データ型) - 変換のデータ型で、[Double] (倍) または [String] (文字列) です。
- 外部 ID – (オプション) これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。
- [Formula] (計算式) - メトリクス式。メトリクス式では、[aggregation functions] (集約関数) を使用して、階層内のすべての関連するアセットに対するプロパティからデータを入力することができます。入力を開始するか、下矢印キーを押すと、オートコンプリート機能を実行できます。詳細については、「数式の使用」を参照してください。

> **Important**
>
> メトリクスは、整数、倍、ブール値、文字列型のプロパティのみ使用可能です。ブール値は、`0` (false) 、1 (true) に変換されます。
> メトリクスの式でメトリクス入力変数を定義する場合、それらの入力には出力メトリクスと同じ時間間隔が必要です。

数式では、倍値または文字列値のみを出力できます。入れ子になった式は、文字列など他のデータ型を出力することができますが、式全体としては、数値または文字列として評価されなければなりません。[jp function] (jp関数) を使うと、文字列を数値に変換することができます。ブール値は 1 (true) または 0 (false) でなければならない。詳細については、「未定義の値、無限の値、およびオーバーフロー値」を参照してください。

- [Time interval] (時間間隔) - メトリクスの時間間隔。AWS IoT SiteWise は次のタンブリングウィンドウ時間間隔をサポートし、各間隔は前の間隔が終了したときに開始される。
  - [1 minute] (1 分) – 1 分。毎分の終わりに処理されます (午前 00:00:00、午前 00:01:00、午前 00:02:00 など)。
  - [5 minutes] (5 分) – 5 分。正時から 5 分ごとに処理されます (午前 00:00:00、午前 00:05:00、午前 00:10:00 など)。
  - [15 minutes] (15 分) – 15 分。正時から 15 分ごとに処理されます (午前 00:00:00、午前 00:15:00、午前 00:30:00 など)。
  - [1 hour] (1 時間) – 1 時間 (60 分)。UTC の毎時の終わりに処理されます (午前 00:00:00、午前 01:00:00、午前 02:00:00 など)。
  - [1 day] (1 日) – 1 日 (24 時間)。UTC の毎日の終わりに処理されます (月曜日の午前 00:00:00、火曜日の午前 00:00:00 など)。
  - [1 week] (1 週間) – 1 週間 (7 日間)。UTC の毎週日曜日の終わりに処理されます (月曜日の午前 00:00:00 ごと)。
  - [Custom interval] (カスタム時間間隔) - 1 分 ~ 1 週間の間で任意の時間間隔を入力できます。
- [Offset date] (オフセット日) -(オプション) データを集計するためのリファレンス日です。
- [Offset time] (オフセット時間) -(オプション) データを集計するリファレンス時間。オフセット時刻は、00:00:00 ~ 23:59:59 の間でなければなりません。
- [Offset time zone] (オフセットタイムゾーン) -(オプション) オフセットのタイムゾーンです。指定しない場合、デフォルトのオフセットタイムゾーンは協定世界時 (UTC) です。

サポートされているタイムゾーン

- (UTC+00:00) 世界協定時間
- (UTC+01:00) 中央ヨーロッパ時間
- (UTC+02:00) 東ヨーロッパ
- (UTC03+:00) 東アフリカ時間
- (UTC+04:00) 近東時間
- (UTC+05:00) パキスタンラホール時間

- (UTC+05:30) インド標準時
- (UTC+06:00) バングラデシュ標準時
- (UTC+07:00) ベトナム標準時
- (UTC+08:00) 中国 [Taiwan] (台湾) (台北市) 時間
- (UTC+09:00) 日本標準時
- (UTC+09:30) オーストラリア中部時間
- (UTC+10:00) オーストラリア東部時間
- (UTC+11:00) ソロモン標準時
- (UTC+12:00) ニュージーランド標準時
- (UTC-11:00) ミッドウェー諸島時間
- (UTC-10:00) ハワイ標準時
- (UTC-09:00) アラスカ標準時
- (UTC-08:00) 太平洋標準時
- (UTC-07:00) フェニックス標準時
- (UTC-06:00) 中央標準時
- (UTC-05:00) 東部標準時
- (UTC-04:00) プエルトリコ米領バージン諸島時間
- (UTC-03:00) アルゼンチン標準時
- (UTC-02:00) サウスジョージア時間
- (UTC-01:00) 中央アフリカ時間

Example オフセット付きのカスタム時間間隔 (コンソール)

次の例では、2021 年 2 月 20 日午後 6 時 30 分 30 秒 (PST) にオフセットされた 12 時間の時間間隔を定義する方法を示します。

オフセット付きのカスタム時間間隔 を定義するには

1. [Time interval] (時間間隔) で [Custom interval] (カスタム間隔) を選択します。
2. [Time interval] (時間間隔) については、次のいずれかを行ってください。
   - **12** を入力し、[hours] (時間) を選択します。
   - **720** を入力し、[minutes] (分) を選択します。

- **43200** を入力し、[seconds] (秒) を選択します。

> **Important**
>
> [Time interval] (時間間隔) は単位に関係なく整数でなければならない。

3. [Offset date] (オフセット日付) は、[2021/02/20]を選択します。
4. [Offset time] (オフセット時間) には、**18:30:30** を入力します。
5. [Offset timezone] (オフセットタイムゾーン) は、[(UTC-08:00) Pacific Standard Time] ((UTC-08:00) 太平洋標準時) を選択してください。

メトリクスを 2021 年 7 月 1 日午後 6 時 30 分 30 秒 (PST) 以前に作成した場合、2021 年 7 月 1 日午後 6 時 30 分 30 秒 (PST) に最初の集計結果が得られます。2 番目の集計結果は 2021 年 7 月 2 日午前 6 時 30 分 30 秒 (PST) に続きます。

メトリクスの定義 (AWS CLI)

AWS IoT SiteWise API を使用してアセットモデルのメトリクスを定義する場合は、次のパラメータを指定します。

- `name` - プロパティの名前。
- `dataType` - メトリクスのデータ型 (`DOUBLE` または `STRING`)。
- `externalId` – (オプション) これはユーザー定義 ID です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。
- `expression` - メトリクス式。メトリクス式では、[aggregation functions] (集約関数) を使用して、階層内のすべての関連するアセットに対するプロパティからデータを入力することができます。詳細については、「数式の使用」を参照してください。
- `window` - メメトリクスタンブリングウィンドウの時間間隔とオフセットで、各間隔は前の間隔が終了したときに開始されます。
  - `interval` - タンブリングウィンドウの時間間隔です。時間間隔は 1 分以上 1 週間以内にする必要があります。
  - `offsets` - タンブリングウィンドウのオフセット。

  詳細については、 API リファレンスTumblingWindowの「」を参照してください。 AWS IoT SiteWise

Example オフセット付きカスタム時間間隔 (AWS CLI)

次の例では、2021 年 2 月 20 日午後 6 時 30 分 30 秒 (PST) にオフセットされた 12 時間の時間間隔を定義する方法を示します。

```
{
    "window": {
        "tumbling": {
            "interval": "12h",
            "offset": " 2021-07-23T18:30:30-08"
        }
    }
}
```

メトリクスを 2021 年 7 月 1 日午後 6 時 30 分 30 秒 (PST) 以前に作成した場合、2021 年 7 月 1 日午後 6 時 30 分 30 秒 (PST) に最初の集計結果が得られます。2 番目の集計結果は 2021 年 7 月 2 日午前 6 時 30 分 30 秒 (PST) に続きます。

- `variables` - 式で使用するアセットまたは子アセットのその他のプロパティを定義する変数のリスト。各変数構造には、式で使用する単純な名前と、その変数にリンクされるプロパティを定義する `value` 構造が含まれます。`value` 構造体には、以下の情報が含まれています。
  - `propertyId` - 値を取得するプロパティの ID。プロパティが (階層からモデルで定義されているのではなく) 現在のモデルで定義されている場合は、ID の代わりにプロパティの名前を使用できます。
  - `hierarchyId` -(オプション) プロパティの子アセットをクエリする階層の ID。ID の代わりに階層定義の名前を使用できます。この値を省略すると、 は現在のモデルで プロパティ AWS IoT SiteWise を検索します。

> **Important**
>
> メトリクスは、整数、倍、ブール値、文字列型のプロパティのみ使用可能です。ブール値は、0 (false) 、1 (true) に変換されます。
> メトリクスの式でメトリクス入力変数を定義する場合、それらの入力には出力メトリクスと同じ時間間隔が必要です。
> 数式では、倍値または文字列値のみを出力できます。入れ子になった式は、文字列など他のデータ型を出力することができますが、式全体としては、数値または文字列として評価されなければなりません。[jp function] (jp関数) を使うと、文字列を数値に変換すること

ができます。ブール値は 1 (true) または 0 (false) でなければならない。詳細については、「未定義の値、無限の値、およびオーバーフロー値」を参照してください。

- `unit` –(オプション) プロパティの科学単位 (mm、摂氏など)。

Example メトリクス定義の例

次の例は、アセットの温度測定データを集計して、時間当たりの最高温度を華氏で計算するメトリクスプロパティを示しています。このオブジェクトは、メトリクス を含むAssetModelプロパティの例です。このオブジェクトをCreateAssetモデルリクエストペイロードの一部として指定して、メトリクスプロパティを作成できます。詳細については、「アセットモデルの作成 (AWS CLI) 」を参照してください。

```
{
      ...
      "assetModelProperties": [
      ...
      {
        "name": "Max temperature",
        "dataType": "DOUBLE",
        "type": {
          "metric": {
            "expression": "max(temp_f)",
            "variables": [
              {
                "name": "temp_f",
                "value": {
                  "propertyId": "Temperature F"
                }
              }
            ],
            "window": {
              "tumbling": {
                "interval": "1h"
              }
            }
          }
        },
        "unit": "Fahrenheit"
      }
    ],
```

```
    ...
}
```

Example 関連アセットからデータを入力するメトリクス定義例。

次の例は、複数の風力タービンの平均電力データを集約し、風力発電所の総平均電力を算出するメトリックプロパティを示しています。このオブジェクトは、メトリクス を含むAssetModelプロパティの例です。このオブジェクトをCreateAssetモデルリクエストペイロードの一部として指定して、メトリクスプロパティを作成できます。

```
{
      ...
      "assetModelProperties": [
      ...
      {
          "name": "Total Average Power",
          "dataType": "DOUBLE",
          "type": {
            "metric": {
              "expression": "avg(power)",
              "variables": [
                {
                  "name": "power",
                  "value": {
                    "propertyId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
                    "hierarchyId": "Turbine Asset Model"
                  }
                }
              ],
              "window": {
                "tumbling": {
                  "interval": "5m"
                }
              }
            }
        },
        "unit": "kWh"
      }
    ],
    ...
}
```

数式の使用

数式を使用すると、raw 産業データを変換および集計する数学関数を定義して、オペレーションに関する洞察を得ることができます。式は、リテラル、演算子、関数、変数を組み合わせてデータを処理します。数式を使用したアセットプロパティの定義方法については、データの変換 (変換) および プロパティおよびその他のアセットのデータ (指標) の集計 を参照してください。変換とメトリックは、式のプロパティです。

トピック

- 式での変数の使用
- 式でのリテラルの使用
- 式での演算子の使用
- 式での定数の使用
- 数式での関数の使用
- 数式表現チュートリアル

式での変数の使用

変数は数式内の AWS IoT SiteWise アセットプロパティを表します。変数を使用して、式中の他のアセットプロパティから値を入力し、定数プロパティ ([attributes]) (属性) 、raw データストリーム ([measurements]) (測定値) 、および他の式プロパティからのデータを処理できるようにします。

変数は、同じアセットモデル、または関連する子アセットモデルのアセットプロパティを表すことができます。子アセットモデルから変数を入力できるのは、メトリクス式のみです。

コンソールと API では、異なる名前で変数を識別します。

- AWS IoT SiteWise コンソール – アセットプロパティ名を式内の変数として使用します。
- AWS IoT SiteWise API (AWS CLI、 AWS SDKs) – ExpressionVariable構造を使用して変数を定義します。これには変数名とアセットプロパティへの参照が必要です。変数名には小文字、数字、アンダースコアを使用することができます。次に、式の中でアセットプロパティをリファレンスするために、変数名を使用します。

変数名名では、大文字と小文字が区別されます。

詳細については、[Defining metrics] (メトリクスの定義) の [Defining transforms] (変換の定義) を参照してください。

変数を使用したプロパティの参照

変数の値は、参照するプロパティを定義します。 AWS IoT SiteWise には、これを行うためのさまざまな方法が用意されています。

- プロパティ ID 別： プロパティの一意の ID (UUID) を指定して識別できます。
- 名前別： プロパティが同じアセットモデルにある場合は、プロパティ ID フィールドで名前を指定できます。
- パス別： 変数値は、パス別にプロパティを参照できます。詳細については、「パスを使用してカスタム複合モデルプロパティを参照する」を参照してください。

> **Note**
>
> 可変は AWS IoT SiteWise コンソールではサポートされていません。これらは、) AWS Command Line Interface AWS CLIや AWS SDKs を含む AWS IoT SiteWise API によって使用されます。

からのレスポンスで受け取る変数 AWS IoT SiteWise には、ID とパスの両方を含む、値に関する完全な情報が含まれます。

ただし、変数を に渡す場合 AWS IoT SiteWise （ 例えば、「create」または「update」呼び出しで ）、これらのいずれかを指定するだけで済みます。例えば、パスを指定した場合、ID を指定する必要はありません。

式でのリテラルの使用

数式には、数値や文字列のリテラルを定義することができます。

- 数字

  数字と科学的記数法を用いて、整数と 2 倍を定義することができる。[E notation] (指数表記) は指数表記で数字を表現することができます。

  例:`1`、 `2.0`、 `.9`、 `-23.1`、 `7.89e3`、 `3.4E-5`

- 文字列

文字列を定義するには、' (引用符) と" (二重引用符) を使用します。開始と終了の引用符の種類は一致させる必要があります。文字列の宣言に使用した引用符と一致する引用符をエスケープするには、その引用符文字を 2 回使用します。これは AWS IoT SiteWise 文字列内の唯一のエスケープ文字です。

例: `'active'`、`"inactive"`、`'{"temp": 52}'`、`"{""temp"": ""high""}"`

式での演算子の使用

式には、次の一般的な算術演算子を使用することができます。

演算子	説明
+	両オペランドが数値の場合、この算術演算子は左右のオペランドを加算します。 どちらかのオペランドが文字列の場合、この算術演算子は左右のオペランドを文字列として連結する。例えば、式 `1 + 2 + " is three"` は `"3 is three"` に評価される。連結された文字列は最大 1024 文字まで可能である。文字列が 1024 文字を超える場合、 AWS IoT SiteWise はそのコンピューティングのためのデータポイントを出力しない。
-	左のオペランドから右のオペランドを減算します。 この算術演算子は、数値オペランドにのみ使用できます。
/	左のオペランドを右のオペランドで除算します。 この算術演算子は、数値オペランドにのみ使用できます。
*	左右のオペランドを乗算します。

演算子	説明
	この算術演算子は、数値オペランドにのみ使用できます。
^	左のオペランドを右のオペランドでべき乗します (べき乗)。 この算術演算子は、数値オペランドにのみ使用できます。
%	左のオペランドを右のオペランドで除算した剰余を返します。結果の符号は、左オペランドと同じです。この動作は、モジュロ演算とは異なります。 この算術演算子は、数値オペランドにのみ使用できます。
`x < y`	1 が x より小さい場合は y を返し、それ以外の場合は 0 を返します。
`x > y`	1 が x より大きい場合は y を返し、それ以外の場合は 0 を返します。
`x <= y`	1 が x より小さいか、または等しい場合は y を返し、それ以外の場合は 0 を返します。
`x >= y`	1 が xより大きいか、または等しい場合は y を返し、それ以外の場合は 0 を返します。
`x == y`	1 が x に等しい場合は y を返し、それ以外の場合は 0 を返します。
`x != y`	1 が x に等しくない場合は y を返し、それ以外の場合は 0 を返します。

演算子	説明
`!x`	1 が x (false) と評価された場合は `0` を返し、それ以外の場合は `0` を返します。 x は、次の場合に (false) と評価される。 • x は数値オペランドであり、`0` に評価される。 • x は空白の文字列に評価される。 • x は空白の配列に評価される。 • x が None に評価される。
`x and y`	`0` が xと評価された場合 (false) 、`0` を返す。それ以外の場合は、y の評価結果を返す。 x または y は、次の場合に (false) と評価される。 • x または y は数値オペランドであり、`0` に評価される。 • x または y は空白の文字列として評価される。 • x または y が空白の配列に評価されます。 • x または y が None に評価される。

演算子	説明
`x or y`	1 が x (true) に評価された場合、1 を返す。それ以外の場合は、y の評価結果を返す。 x または y は、次の場合に (false) と評価される。 • x または y は数値オペランドであり、`0` に評価される。 • x または y は空白の文字列として評価される。 • x または y が空白の配列に評価されます。 • x または y が None に評価される。
`not x`	1 が x (false) と評価された場合は `0` を返し、それ以外の場合は `0` を返します。 x は、次の場合に (false) と評価される。 • x は数値オペランドであり、`0` に評価される。 • x は空白の文字列に評価される。 • x は空白の配列に評価される。 • x が None に評価される。

`[]`
 	
`s[index]`
	文字列 `index` のインデックス `s` にある文字を返す。これは Python のインデックス構文に相当する。
	
	Example 例
	
	• `"Hello!"[1]` は `e` を返します。
	• `"Hello!"[-2]` は `o` を返します。

演算子	説明

`[]`
 	
`s[start:end:step]`
	文字列 `s` のスライスを返す。これは Python のスライス構文に相当する。この演算子は次の引数を持つ。
	
	• `start` -(オプション) スライスの包括的な開始インデックスを指定します。デフォルトは `0` です。
	• `end` -(オプション) スライスの排他的終了インデックスです。デフォルトは文字列の長さです。
	• `step` -(オプション) スライスの各ステップで増分する番号です。例えば、2 を指定すると1文字おきにスライスを返し、`-1` を指定するとスライスを反転させることができる。デフォルトは `1` です。
	
	`step` 引数を省略すると、そのデフォルト値が使用される。たとえば、`s[1:4:1]` と `s[1:4]` は同じです。
	
	引数は整数または[none] (無し) 定数でなければならない。を指定した場合none、 はその引数のデフォルト値 AWS IoT SiteWise を使用します。
	
	Example 例
	
	• `"Hello!"[1:4]` は `"ell"` を返します。
	• `"Hello!"[:2]` は `"He"` を返します。
	• `"Hello!"[3:]` は `"lo!"` を返します。
	• `"Hello!"[:-4]` は `"He"` を返します。
	• `"Hello!"[::2]` は `"Hlo"` を返します。
	• `"Hello!"[::-1]` は `"!olleH"` を返します。

式での定数の使用

式には、次の一般的な数学定数を使用できます。すべての定数は大文字と小文字を区別しません。

> **Note**
> 定数と同じ名前の変数を定義すると、その変数が定数をオーバーライドします。

定数	説明
`pi`	数字のπ (π) です: `3.141592653589793`
`e`	数字のe: `2.718281828459045`
`true`	数字の 1 に相当します。では AWS IoT SiteWise、ブール値は同等の数値に変換されます。
`false`	数字の 0 に相当します。では AWS IoT SiteWise、ブール値は同等の数値に変換されます。
`none`	無価値。この定数を使用すると、[conditional expression] (条件式) の結果として何も出力しないことができます。

数式での関数の使用

次の関数を使用して、数式中のデータを操作することができます。

変換とメトリクスは、それぞれ異なる機能をサポートしています。次の表は、各型の数式プロパティに対応する関数の種類を示したものです。

> **Note**
> 1 つの数式には、最大 10 個の関数を含めることができます。

関数型	変換	メトリクス
数式での一般的な関数の使用	 はい	 はい
数式での比較関数の使用	 はい	 はい
数式での条件関数の使用	 はい	 はい
式での文字列関数の使用	 はい	 はい
数式での集計関数の使用	 いいえ	 はい
数式での時間関数の使用	 はい	 はい
数式での日付および時刻関数の使用	 はい	 はい

関数構文

関数を作成するには、次の構文を使用します。

通常の構文

通常の構文では、関数名の後に括弧が付き、引数は 0 個以上となります。

function_name(*argument1, argument2, argument3, ...*)。例えば、通常の構文を持つ関数は、`log(x)` や `contains(s, substring)` のようになります。

統一関数呼び出し構文 (UFCS)

UFCSでは、オブジェクト指向プログラミングにおけるメソッド呼び出しの構文を用いて、関数を呼び出すことができる。UFCSでは、最初の引数の後にドット (.) が付き、次に関数名、残りの引数がある場合は括弧の中に入る。

argument1.*function_name*(*argument2*, *argument3*, ...)。例えば、UFCSを使った関数は、`x.log()` や `s.contains(substring)` のようになります。

UFCS を使用して後続の関数を連鎖することもできます。 AWS IoT SiteWise は、現在の関数の評価結果を次の関数の最初の引数として使用します。

例えば、`message.jp('$.status').lower().contains('fail')` の代わりに `contains(lower(jp(message, '$.status')),'fail')` を使用することができます。

詳しくは、[D Programming Language] (D プログラミング言語) のウェブサイトをご覧ください。

> **Note**
>
> UFCS はすべての AWS IoT SiteWise 関数に使用できます。
> AWS IoT SiteWise 関数では、大文字と小文字は区別されません。例えば、`lower(s)` と `Lower(s)` を入れ替えて使用することができます。

数式での一般的な関数の使用

[transforms] (変換) と [metrics] (メトリクス) において、一般的な数学的関数をコンピューティングすることができます。

機能	説明
abs(x)	x の絶対値を返します。
acos(x)	x のアークコサイン (逆余弦) を返します。
asin(x)	x のアークサイン (逆正弦) を返します。
atan(x)	x のアークタンジェント (逆正接) を返します。
cbrt(x)	x の立方根を返します。
ceil(x)	x より大きい最も近い整数を返します。
cos(x)	x のコサイン (余弦) を返します。
cosh(x)	x のハイパーボリックコサイン (双曲線余弦) を返します。
cot(x)	x のコタンジェントを返します。
exp(x)	e の x 乗を返します。
expm1(x)	戻り値exp(x) - 1。exp(x) - 1 が小さい値である場合に、より正確に x をコンピューティングするためにこの関数を使用します。
floor(x)	x より小さい最も近い整数を返します。
log(x)	$\log_e$ の e (底 x) を返します。
log10(x)	$\log_{10}$ の 10 (底 x) を返します。
log1p(x)	戻り値log(1 + x)。log(1 + x) が小さい値である場合に、より正確に x をコンピューティングするためにこの関数を使用します。
log2(x)	$\log_2$ の 2 (底 x) を返します。
pow(x, y)	x の y 乗を返します。x ^ y に相当します。

機能	説明
`signum(x)`	x (負の入力の場合は -1、ゼロの入力の場合は 0、正の入力の場合は +1) 符号を返します。
`sin(x)`	x のサイン (正弦) を返します。
`sinh(x)`	x のハイパーボリックサイン (双曲線正弦) を返します。
`sqrt(x)`	x の平方根を返します。
`tan(x)`	x のタンジェント (正接) を返します。
`tanh(x)`	x のハイパーボリックタンジェント (双曲線正接) を返します。

数式での比較関数の使用

変換とメトリクス では、次の比較関数を使用して、2 つの値と出力 1 (true) または 0 (false) を比較できます。 は、辞書順 で文字列 AWS IoT SiteWise を比較します。

機能	説明
`gt(x, y)`	1 が x より大きい場合は y を返し、それ以外の場合は 0 (x > y) を返します。 この関数は、x と y が数値と文字列のように互換性のない型の場合は、値を返さない。
`gte(x, y)`	1 が x より大きいか、または等しい場合は y を返し、それ以外の場合は 0 (x ≥ y) を返します。 AWS IoT SiteWise は、引数が の相対許容値内にある場合、引数を等しいと見なします1E-9。これは Python のisclose 関数と同じような動作をします。

機能	説明
	この関数は、x と y が数値と文字列のように互換性のない型の場合は、値を返さない。
`eq(x, y)`	1 が x に等しい場合は y を返し、それ以外の場合は 0 (x == y) を返します。 AWS IoT SiteWise は、引数が の相対許容値内にある場合、引数を等しいと見なします1E-9。これは Python のisclose 関数と同じような動作をします。 この関数は、x と y が数値と文字列のように互換性のない型の場合は、値を返さない。
`lt(x, y)`	1 が x より小さい場合は y を返し、それ以外の場合は 0 (x < y) を返します。 この関数は、x と y が数値と文字列のように互換性のない型の場合は、値を返さない。
`lte(x, y)`	1 が x より小さいか、または等しい場合は y を返し、それ以外の場合は 0 (x ≤ y) を返します。 AWS IoT SiteWise は、引数が の相対許容値内にある場合、引数を等しいと見なします1E-9。これは Python のisclose 関数と同じような動作をします。 この関数は、x と y が数値と文字列のように互換性のない型の場合は、値を返さない。
`isnan(x)`	1 が x に等しい場合は NaN を返し、それ以外の場合は 0 を返します。 x が文字列の場合、この関数は値を返さない。

数式での条件関数の使用

変換とメトリクス では、次の関数を使用して条件をチェックし、条件が true と false のどちらに評価されるかにかかわらず、異なる結果を返すことができます。

機能	説明
`if(condition, result_if_true, result_if_false)`	`condition` を評価し、条件の評価結果が true の場合に `result_if_true` 、false の場合には `result_if_false` を返します。 `condition` は数字でなければなりません。この関数は、`0` および空の文字列を `false` と見なし、それ以外 (NaN を含む) の場合を `true` と見なします。ブール値は、`0` (false) 、1 (true) に変換されます。 この関数から[none] (無し) 定数を返すことで、特定の条件の出力を破棄することができます。つまり、条件を満たさないデータポイントをフィルタリングすることができるのです。詳細については、「データポイントのフィルタリング」を参照してください。 Example 例 • `if(0, x, y)` は変数 y を返す。 • `if(5, x, y)` は変数 x を返す。 • `if(gt(temp, 300), x, y)` は、変数 x が `temp` よりも大きい場合、変数 `300` を返す。 • `if(gt(temp, 300), temp, none)` が `temp` 以上なら変数 `300` を、none が `temp` より小さいなら (300値なし) を返す。 1つ以上の引数が条件付き関数である入れ子関数には、UFCS を使用することをお勧

機能	説明
	めします。if(condition, result_if_true) で条件を評価し、elif(condition, result_if_true, result_if_false) で追加の条件を評価することができます。 例えば、if(condition1, result1_if_true).elif(condition2, result2_if_true, result2_if_false) の代わりに if(condition1, result1_if_true, if(condition2, result2_if_true, result2_if_false)) を使用することができます。 また、追加の中間条件関数を連結することもできます。たとえば、if(condition1, result1_if_true, if(condition2, result2_if_true, if(condition3, result3_if_true result3_if_false))) のように複数の if ステートメントをネストするのではなく if(condition1, result1_if_true).elif(condition2, result2_if_true).elif(condition3, result3_if_true, result3_if_false) を使用することができます。 **Important** UFCS で elif(condition, result_if_true, result_if_false) を使用する必要があります。

式での文字列関数の使用

[transforms] (変換) と[metrics] (メトリクス) では、次の関数を使用して文字列を操作することができます。詳細については、「数式で文字列を使用する。」を参照してください。

> **Important**
>
> 数式では、倍値または文字列値のみを出力できます。入れ子になった式は、文字列など他のデータ型を出力することができますが、式全体としては、数値または文字列として評価されなければなりません。[jp function] (jp関数) を使うと、文字列を数値に変換することができます。ブール値は 1 (true) または 0 (false) でなければならない。詳細については、「未定義の値、無限の値、およびオーバーフロー値」を参照してください。

関数	説明
`len(s)`	文字列 s の長さをを返します。
`find(s, substring)`	文字列 `substring` が文字列 s に含まれるインデックスを返す。
`contains(s, substring)`	文字列 1 が文字列 s を含んでいれば `substring` 、それ以外の場合は `0` を返します。
`upper(s)`	文字列 s を大文字で返す。
`lower(s)`	文字列 s を小文字に変換して返す。
`jp(s, json_path)`	JsonPath 式sを使用して文字列を評価しjson_path 、結果を返します。 この関数を使うと、次のことができます。 • シリアル化された JSON 構造体から値、配列、またはオブジェクトを取り出す。 • 文字列を数値に変換する。例えば、`jp('111', '$')` という数式は、111 を数値として返す。

関数	説明
	JSON 構造体から文字列値を取り出し、数値として返すには、複数の入れ子 jp 関数を使用する必要があります。外側の jp 関数は JSON 構造体から文字列を抽出し、内側の jp 関数はその文字列を数値に変換する。 文字列 json_path には、文字列リテラルを含める必要があります。つまり、json_path は文字列として評価される式ではありえないということです。 Example 例 • jp('{"status":"active","value":15}', '$.value') は 15 を返します。 • jp('{"measurement":{"reading":25,"confidence":0.95}}', '$.measurement.reading') は 25 を返します。 • jp('[2,8,23]', '$[2]') は 23 を返します。 • jp('{"values":[3,6,7]}', '$.values[1]') は 6 を返します。 • jp('111', '$') は 111 を返します。 • jp(jp('{"measurement":{"reading":25,"confidence":"0.95"}}', '$.measurement.confidence'), '$') は 0.95 を返します。

関数	説明
`join(s0, s1, s2, s3, ...)`	文字列をデリミタ付きで連結して返す。この関数は、最初の入力文字列をデリミタとして使い、残りの入力文字列を結合する。これは、Java の join(CharSequence delimiter, CharSequence... elements) 関数と同様に動作します。 Example 例 • `join("-", "aa", "bb", "cc")` は `aa-bb-cc` を返す
`format(expression: "format")` または `format("format", expression)` **	指定された書式の文字列を返す。この関数は、`expression` を評価し、指定されたフォーマットで値を返す。これは、Java のformat (String format, Object... args)関数と同じような動作をします。サポートされるフォーマットの詳細については、Java Platform, Standard Edition 7 API SpecificationのClass Formatterの Conversions を参照してください。 Example 例 • `format(100+1: "d")` は文字列 `101` を返す。 • `format("The result is %d", 100+1)` は文字列 `The result is 101` を返す。

関数	説明
`f'expression'`	連結された文字列を返す。この書式付き関数を使うと、簡単な式で文字列の連結や書式設定を行うことができます。これらの関数は、入れ子になった式を含むことができる。{} (中括弧) を使って、式を補間することができます。これは、Python の[formatted string literals] (フォーマットされた文字列リテラル) と同じような動作をします。 Example 例 • `f'abc{1+2: "f"}d'` は `abc3.000000d` を返します。この例の式を評価するには、次のようにします。 1. `format(1+2: "f")` は浮動小数点数 `3.000000` を返します。 2. `join('', "abc", 1+2, 'd')` は文字列 `abc3.000000d` を返す。 また、次のように式を書くこともできます `join('', "abc", format(1+2: "f"), 'd')`。

数式での集計関数の使用

[metrics] (メトリクス) のみでは、各タイムインターバルの入力値を集計し、単一の出力値を算出する次の関数を使用することができます。集計関数は、関連付けられたアセットからデータを集計できません。

集計関数の引数には、変数、数値リテラル、時間関数 、入れ子式、または集計関数を使用することができます。式 `max(latest(x), latest(y), latest(z))` は集計関数を引数として使用し、x、y および z プロパティの現在の最大値を返します。

集計関数に入れ子になった式を使用することができます。入れ子になった式を使用する場合、次のルールが適用されます。

- 各引数は1つの変数しか持つことができません。

  Example

  例えば、`avg(x*(x-1))` と `sum(x/2 )/avg(y^2 )` がサポートされています。

  例えば、`min(x/y)` はサポートされていません。
- 各引数は多段階に入れ子になった式を持つことができる。

  Example

  例えば、`sum(avg(x^2 )/2)` がサポートされています。
- 引数によって、異なる変数を持つことができます。

  Example

  例えば、`sum(x/2, y*2)` がサポートされています。

> **Note**
>
> - 式に測定値が含まれている場合、 は測定値の現在の時間間隔の最後の値 AWS IoT SiteWise を使用して集計を計算します。
> - 式に属性が含まれている場合、 は属性の最新の値 AWS IoT SiteWise を使用して集計を計算します。

機能	説明
`avg(`x_0`, ..., `x_n`)`	現在の時間間隔における指定された変数値の平均を返します。 この関数は、与えられた変数が現在の時間間隔に少なくとも 1 つのデータポイントを持つ場合にのみデータポイントを出力します。
`sum(`x_0`, ..., `x_n`)`	現在の時間間隔における指定された変数値の合計を返します。

機能	説明
	この関数は、与えられた変数が現在の時間間隔に少なくとも 1 つのデータポイントを持つ場合にのみデータポイントを出力します。
$min(x_0, ..., x_n)$	現在の時間間隔における指定された変数の最小値を返します。 この関数は、与えられた変数が現在の時間間隔に少なくとも 1 つのデータポイントを持つ場合にのみデータポイントを出力します。
$max(x_0, ..., x_n)$	現在の時間間隔における指定された変数の最大値を返します。 この関数は、与えられた変数が現在の時間間隔に少なくとも 1 つのデータポイントを持つ場合にのみデータポイントを出力します。
$count(x_0, ..., x_n)$	現在の時間間隔における指定された変数のデータポイントの合計数を返します。条件を満たすデータポイントの数を数える方法の詳細については、「条件に一致するデータポイントを数える。」を参照してください。 この関数は、時間間隔ごとにデータポイントを計算します。
$stdev(x_0, ..., x_n)$	現在の時間間隔に対する指定された変の数値の標準偏差を返します。 この関数は、与えられた変数が現在の時間間隔に少なくとも 1 つのデータポイントを持つ場合にのみデータポイントを出力します。

数式での時間関数の使用

時間関数を使用して、データポイントのタイムスタンプに基づいて値を返す。

メトリクスで時間関数を使用する。

[metrics] (メトリクス) のみ、データポイントのタイムスタンプに基づいた値を返す次の関数を使用できます。

時間関数の引数は、ローカルのアセットモデルまたは入れ子式のプロパティである必要があります。これは、子アセットモデルのプロパティを一時関数で使用できないことを意味します。

一時関数で入れ子になった式を使用することができます。入れ子になった式を使用する場合、次のルールが適用されます。

- 各引数は1つの変数しか持つことができません。

  例えば、`latest( t*9/5 + 32 )` がサポートされています。
- 引数には集計関数を指定できません。

  例えば、`first( sum(x) )` はサポートされていません。

機能	説明
`first(x)`	現在の時間間隔における最も早いタイムスタンプを持つ、指定された変数値を返します。
`last(x)`	現在の時間間隔における最新のタイムスタンプを持つ、指定された変数値を返します。
`earliest(x)`	現在の時間間隔の開始前に指定された変数の最後の値を返します。 入力プロパティの履歴に少なくとも 1 つのデータポイントがある場合、この関数は、時間間隔ごとにデータポイントを計算します。詳細については、「time-range-defintion」を参照してください。
`latest(x)`	指定された変数の最後の値を、現在の時間間隔の終了前の最新のタイムスタンプとともに返します。

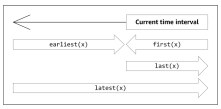
機能	説明
	入力プロパティの履歴に少なくとも 1 つのデータポイントがある場合、この関数は、時間間隔ごとにデータポイントを計算します。詳細については、「time-range-defintion」を参照してください。
`statetime(x)`	指定された時間間隔において、指定された変数が正である時間 (秒単位) を返します。[comparison functions] (比較関数) を使用して、`statetime` 関数が消費する変換プロパティを作成することができます。 たとえば、`Idle` プロパティが `0` または `1` の場合、次の式 `IdleTime = statetime(Idle)` を使用して時間間隔ごとのアイドル時間を計算できます。詳細については、「statetime シナリオの例」を参照してください。 この関数は、入力変数としてメトリクスプロパティをサポートしていません。 入力プロパティの履歴に少なくとも 1 つのデータポイントがある場合、この関数は、時間間隔ごとにデータポイントを計算します。

機能	説明
`TimeWeightedAvg(x, [interpolation])`	ポイント間の時間間隔で重み付けされた入力データの平均を返します。 計算と間隔の詳細については、「時間加重関数のパラメータ」を参照してください。 オプションの引数 `interpolaton` は文字列定数である必要があります。 • `locf` – デフォルト値です。この計算では、データポイント間の間隔に Last Observed Carry Forward (LOCF) 計算アルゴリズムが使用されます。このアプローチでは、データポイントは次の入力データポイントのタイムスタンプまで最後の観測値として使用されます。 良好データポイントの後の値が、次のデータポイントのタイムスタンプまでの値として外挿されます。 • `linear` – この計算では、データポイント間の間隔に線形補間計算アルゴリズムが使用されます。 2 つの良好データポイント間の値が、それらデータポイントの値間の線形補間値として外挿されます。 良好データポイントと不良データポイントの間の値、または最後の良好データポイントの後の値が、良好データポイントとして外挿されます。

機能	説明
`TimeWeightedStDev(x, [algo])`	ポイント間の時間間隔で重み付けされた入力データの標準偏差を返します。 計算と間隔の詳細については、「時間加重関数のパラメータ」を参照してください。 この計算では、データポイント間の間隔に Last Observed Carry Forward (LOCF) 計算アルゴリズムが使用されます。このアプローチでは、データポイントは次の入力データポイントのタイムスタンプまで最後の観測値として使用されます。重みは、データポイント間またはウィンドウ境界間の時間間隔 (秒単位) の形式で求められます。 オプションの引数 `algo` は文字列定数である必要があります。 • f – デフォルト値です。頻度の重みを持つバイアスのない加重サンプル分散を返します。ここで、 TimeWeight は秒単位で計算されます。通常、このアルゴリズムは標準偏差未満であることを想定しており、加重サンプルに対する標準偏差のベッセル補正として知られています。 • p – 偏りあり加重サンプル分散 (母分散とも呼ばれる) を返します。 計算には以下の式が使用されます。 • S_p = 母集団の標準偏差 • S_f = 頻度の標準偏差 • X_i = 入力データ • ω_i = 時間間隔に等しい重み (秒) • μ^* = 入力データの加重平均

機能	説明
	母集団の標準偏差の計算式: $$S_p^2 = \frac{\sum_{i=1}^{N} \omega_i (x_i - \mu^*)^2}{\sum_{i=1}^{N} \omega_i}$$ 頻度の標準偏差の計算式: $$S_f^2 = \frac{\sum_{i=1}^{N} \omega_i (x_i - \mu^*)^2}{\sum_{i=1}^{N} \omega_i - 1}$$

次の図はfirst、現在の時間間隔を基準にして、 が時間関数 last、latest、、earliestおよび AWS IoT SiteWise を計算する方法を示しています。



> **Note**
>
> - の時間範囲はfirst(x)、 (現在のウィンドウ開始、現在のウィンドウ終了] last(x)です。
> - の時間範囲は (開始時間、現在のウィンドウ終了時間] latest(x)です。
> - の時間範囲は (時間の開始、前のウィンドウの終了] earliest(x)です。

時間加重関数のパラメータ

集計ウィンドウ用に計算される時間加重関数では、次のことが考慮されます。

- ウィンドウ内のデータポイント数
- データポイント間の時間間隔数
- ウィンドウの前の最後のデータポイント
- ウィンドウの後の最初のデータポイント (一部のアルゴリズムのみ)

条件:

- 不良データポイント – 品質が良くないか、値が数値以外のデータポイント。不良データポイントは、ウィンドウの結果計算では考慮されません。
- 不良間隔 – 不良データポイントの後の間隔。最初の既知のデータポイントの前の間隔も不良間隔とみなされます。
- 良好データポイント – 品質が良好でかつ値が数値のデータポイント。

> **Note**
>
> - AWS IoT SiteWise は、変換とメトリクスを計算するときにのみGOOD、品質データを消費します。UNCERTAIN および BAD データポイントは無視します。
> - 最初の既知のデータポイントの前の間隔も不良間隔とみなされます。詳細については、「the section called “数式表現チュートリアル”」を参照してください。

最後の既知のデータポイントの後の間隔は無限に続き、以降のすべてのウィンドウに影響します。新しいデータポイントがくると、この関数は間隔を再計算します。

上記のルールに従って、集計ウィンドウの結果が計算され、ウィンドウの境界が制限されます。デフォルトでは、この関数はウィンドウ全体が良好間隔である場合にのみウィンドウの結果を送信します。

ウィンドウの良好間隔がウィンドウの長さよりも短い場合、この関数はウィンドウを送信しません。

ウィンドウの結果に影響するデータポイントに変更があると、データポイントがウィンドウの外側にあっても、ウィンドウは再計算されます。

入力プロパティの履歴に少なくとも 1 つのデータポイントがあり、計算が開始されていた場合は、時間間隔ごとに時間加重集計関数が計算されます。

Example statetime シナリオの例

次のプロパティを持つアセットがある例を考えてみましょう。

- `Idle` - `0` または `1` である測定値をいう。値が `1` の場合、マシンはアイドル状態です。
- `Idle Time` - 1分間隔で、機械がアイドル状態である時間を秒単位でコンピューティングする式 `statetime(Idle)` を使用した指標。

`Idle` プロパティには、以下のデータポイントがあります。

タイムスタンプ	2:00:00 PM	2:00:30 PM	2:01:15 PM	2:02:45 PM	2:04:00 PM
Idle	0	1	1	0	0

AWS IoT SiteWise は、 の値から `Idle Time`プロパティを 1 分ごとに計算しますIdle。この計算が完了すると、`Idle Time` プロパティには次のデータポイントが存在します。

タイムスタンプ	2:00:00 PM	2:01:00 PM	2:02:00 PM	2:03:00 PM	2:04:00 PM
Idle Time	該当なし	30	60	45	0

AWS IoT SiteWise は、毎分の終わりに `Idle Time` に対して次の計算を実行します。

- 2:00 PM 時点 (1:59 PM から 2:00 PM の場合)
  - `Idle` には 2:00 PM より前のデータがないため、データポイントは計算されません。
- 2:01 PM 時点 (2:00 PM から 2:01 PM の場合)
  - 2:00:00 PM の時点で、マシンはアクティブです (`Idle` は `0`)。
  - 2:00:30 PM の時点で、マシンはアイドル状態です (`Idle` は `1`)。
  - 2:01:00 PM にインターバルが終了する前に `Idle` は再度変更されないため、`Idle Time` は 30 秒になります。
- 2:02 PM 時点 (2:01 PM から 2:02 PM の場合)
  - 2:01:00 PM の時点で、マシンはアイドル状態です (2:00:30 PM 時点の直前のデータポイントによる)。
  - 2:01:15 PM の時点で、マシンはまだアイドル状態です。
  - 2:02:00 PM にインターバルが終了する前に `Idle` は再度変更されないため、`Idle Time` は 60 秒になります。
- 2:03 PM 時点 (2:02 PM から 2:03 PM の場合)
  - 2:02:00 PM の時点で、マシンはアイドル状態です (2:01:15 PM 時点の直前のデータポイントによる)。

  - 2:02:45 PM の時点で、マシンはアクティブです。
  - 2:03:00 PM にインターバルが終了する前に `Idle` は再度変更されないため、`Idle Time` は 45 秒になります。
- 2:04 PM 時点 (2:03 PM から 2:04 PM の場合)
  - 2:03:00 PM の時点で、マシンはアクティブです (2:02:45 PM 時点の直前のデータポイントによる)。
  - 2:04:00 PM にインターバルが終了する前に `Idle` は再度変更されないため、`Idle Time` は 0 秒になります。

Example 例 TimeWeightedAvg と TimeWeightedStDev シナリオ

次の表は、1 分間のウィンドウメトリクスの入力例と出力例を示しています: `Avg(x)`, `TimeWeightedAvg(x)`, `TimeWeightedAvg(x, "linear")`, `stDev(x)`, `timeWeightedStDev(x)`, `timeWeightedStDev(x, 'p')`。

1 分間の集計ウィンドウの入力例:

> **Note**
>
> これらのデータポイントにはすべてGOOD品質があります。

03:00:00	4.0
03:01:00	2.0
03:01:10	8.0
03:01:50	20.0
03:02:00	14.0
03:02:05	10.0
03:02:10	3.0
03:02:30	20.0

03:03:30	0.0

集計結果の出力:

> **Note**
>
> なし – このウィンドウでは結果は生成されません。

時間	Avg(x)	TimeWeighted Avg(x)	TimeWeighted Avg(X, "linear")	stDev(X)	timeWeighted StDev(x)	timeWeighted StDev(x, 'p')
3:00:00	4	なし	なし	0	なし	なし
3:01:00	2	4	3	0	0	0
3:02:00	14	9	13	6	5.430610041581775	5.385164807134504
3:03:00	11	13	12.875	8.54400374531753	7.724054437220943	7.659416862050705
3:04:00	0	10	2.5	0	10.084389681792215	10
3:05:00	なし	0	0	なし	0	0

変換で時間関数を使用する

[変換] のみ、`pretrigger()` 関数を使用して、現在の変換の計算を開始させたプロパティ更新前の変数の GOOD 品質値を取得することができます。

メーカーが AWS IoT SiteWise を使用してマシンのステータスをモニタリングする例を考えてみましょう。メーカーは、次のような測定値と変換を使用してプロセスを表現しています。

- 測定値 `current_state` は 0 または 1 となります。

  - 本機がクリーニング状態の場合、current_state は 1 になる。
  - 機械が製造状態にある場合、current_state は 0 になる。
- cleaning_state_durationに相当する変換、if(pretrigger(current_state) == 1, timestamp(current_state) - timestamp(pretrigger(current_state)), none)。この変換は、マシンがクリーニング状態である時間を秒単位で Unix エポックフォーマットで返します。詳しくは、数式での条件関数の使用 およびtimestamp () (タイムスタンプ) 関数を参照してください。

クリーニング状態が予想以上に長く続くようであれば、メーカーが調査する場合もあるります。

また、多変量変換で pretrigger( ) 関数を使うこともできます。例えば、x、y という 2 つの測定値があり、z と等しい変換値 x + y + pretrigger(y) があるとします。次の表は、午前 9 時から午前 9 時 15 分までの x、y、z の値である。

> **Note**
>
> - この例では、測定値の値が時系列に到着することを想定しています。例えば、午前 09 時 00 分のx の値は、午前 09 時 05 分の x の値より先に到着する。
> - 午前 9 時 5 分のデータポイントが午前 9 時のデータポイントより先に到着した場合、午前 9 時 5 分には z はコンピューティングされません。
> - 午前 9 時 5 分 の x の値が午前 9 時の x の値より先に到着し、y の値が時系列に到着する場合、z は午前 9 時 5 分の 22 = 20 + 1 + 1 と等しくなる。

	午前 9 時 0 分	午前 9 時 5 分	午前 9 時 10 分	午前 9 時 15 分
x	10	20		30
y	1	2	3	
z = x + y + pretrigger(y)	y は午前 09:00 以前にデータポイントを受信していません。したがって、z は午前 09:00 には	23 = 20 + 2 + 1 pretrigger(y) は 1 に等しい。	25 = 20 + 3 + 2 x は新しいデータポイントを受信しません。pretrigge	36 = 30 + 3 + 3 y は新しいデータポイントを受信しません。したがって、午前

	午前 9 時 0 分	午前 9 時 5 分	午前 9 時 10 分	午前 9 時 15 分
	コンピューティングされません。		r(y) は 2 に等しい。	09 時 15 分には pretrigger(y) は3に等しい。

数式での日付および時刻関数の使用

[transforms] (変換) と [metrics] (メトリクス) では、日付と時刻の関数を次のように使用できます。

- データポイントの現在のタイムスタンプを UTC またはローカルタイムゾーンで取得する。
- year、month、day_of_month などの引数でタイムスタンプを構成する。
- unix_time 引数で年や月などの期間を抽出する。

機能	説明
now()	現在の日付と時刻を Unix エポックフォーマットで秒単位で返します。
timestamp()	• 変換では、この関数は入力メッセージのタイムスタンプをUnixエポックフォーマットで秒単位で返します。 変換のみでは、次のいずれかを行うことができます。 • 関数の引数として変数を与える。timestamp(*variable-name*) 関数は、指定された変数の最新の GOOD 品質値のタイムスタンプを Unix のエポックフォーマットで秒単位で返す。 例えば、アセットに Temperature_F という変換プロパティがあり、 9/5 * Temperature_C 式を使用して各温度データポイントを摂氏から華氏に変換する場合、timestamp(Temperature_F)

機能	説明
	関数を使用して、GOOD プロパティに対する最新の Temperature_F 品質値のタイムスタンプを取得することができます。 • pretrigger() 関数の引数として使用します。timestamp(pretrigger(*variable-name*)) 関数は、現在の変換の計算を開始させたプロパティ更新前の指定された変数の GOOD 品質値のタイムスタンプを Unix エポック形式で秒単位で返します。詳細については、「変換で時間関数を使用する」を参照してください。 • メトリクスでは、この関数は、現在のウィンドウの終了時に取得したタイムスタンプを Unix エポックフォーマットで秒単位で返します。

機能	説明
`mktime(time_zone, year, month, day_of_month, hour, minute, second)`	入力された時刻を Unix 工ポックフォーマットで秒単位で返します。 この機能を使用するには、次の条件があります。 • タイムゾーンの引数は、引用符で囲まれた文字列 ('UTC') でなければならない。指定されなかった場合、デフォルトのタイムゾーンは UTC です。 タイムゾーン引数は、最初または最後の引数とすることができる。 • 年、月、日、時、分、秒の引数は順番に指定する必要があります。 • 年、月、日付の引数は必須です。 この機能を使用する場合、次の制限があります。 • `year` - 有効な値は 1970 ~ 2250 です。 • `month` - 有効な値は 1 ~ 12 です。 • `day-of-month` - 有効な値は 1 ~ 31 です。 • `hour` - 有効な値は 0 ~ 23 です。 • `minute` - 有効な値は 0 ~ 59 です。 • `second` - 有効な値は 0 ~ 60 です。浮動小数点数であってもよい。 例: • `mktime(2020, 2, 29)` • `mktime('UTC+3', 2021, 12, 31, 22)`

機能	説明
	• `mktime(2022, 10, 13, 2, 55, 13.68, 'PST')`

機能	説明
`localtime(unix_time, time_zone)`	Unix 時間から、指定されたタイムゾーンの年、月、曜日、日、時、分、秒を返します。 この機能を使用するには、次の条件があります。 • タイムゾーンの引数は、引用符で囲まれた文字列 ('UTC') でなければならない。指定されなかった場合、デフォルトのタイムゾーンは UTC です。 • Unix time 引数は、Unix エポックフォーマットによる秒単位の時間である。有効な範囲は 1 ~ 31556889864403199 です。浮動小数点数であってもよい。 レスポンスの例:2007-12-03T10:15:30+01:00[Europe/Paris] `localtime(unix_time, time_zone)` は単体の機能ではありません。year()、mon()、mday、wday()、yday()、hou 関数は、`localtime(unix_time, time_zone)` を引数とする。 例: • `year(localtime('GMT', 1605898608.8113723))` • `now().localtime().year()` • `timestamp().localtime('PST').year()` • `localtime(1605289736, 'Europe/London').year()`

機能	説明
year(localtime(unix_time, time_zone)	localtime(unix_time, time_zone) から年を返します。
mon(localtime(unix_time, time_zone))	localtime(unix_time, time_zone) から月を返します。
mday(localtime(unix_time, time_zone))	localtime(unix_time, time_zone) から月の特定の日を返します。
wday(localtime(unix_time, time_zone))	localtime(unix_time, time_zone) から週の特定の日を返します。
yday(localtime(unix_time, time_zone))	localtime(unix_time, time_zone) から年の特定の日を返します。
hour(localtime(unix_time, time_zone))	localtime(unix_time, time_zone) から時間を返します。
minute(localtime(unix_time, time_zone))	localtime(unix_time, time_zone) からの分数を返します。
sec(localtime(unix_time, time_zone))	localtime(unix_time, time_zone) から 2 番目を返すします

対応するタイムゾーン形式

タイムゾーンの引数は、次の方法で指定することができます。

- タイムゾーンオフセット - UTC の場合は 'Z'、オフセット ('+2' または '-5') を指定します。
- オフセットID - タイムゾーンの略語とオフセットを組み合わせて使用します。例えば、'GMT+2' と 'UTC-01:00' です。タイムゾーンの略語には3文字のみを使用します。
- リージョンベースの ID - 例えば、'Etc/GMT+12' や 'Pacific/Pago_Pago' など。

サポートされているタイムゾーンの略語

日付と時刻の機能は、次の 3 文字のタイムゾーンの略語をサポートしています。

- 東部標準時 -05:00
- 日本時間 - -10:00
- アメリカ標準時 -07:00
- ACT - オーストラリア/ダーウィン
- AET - オーストラリア/シドニー
- AGT - アメリカ/アルゼンチン/ブエノスアイレス
- ART - アフリカ/カイロ
- AST - アメリカ/アンカレッジ
- BET - アメリカ/サンパウロ
- BST - アジア/ダッカ
- CAT - アフリカ/ハラール
- CET - ヨーロッパ/パリ
- CNT - アメリカ/セントジョンズ
- CST - アメリカ/シカゴ
- CTT - アジア/上海
- EAT - アフリカ/アディスアベバ
- IET - アメリカ/インディアナ/インディアナポリス
- IST - アジア/コルカタ
- JST - アジア/東京
- MIT - 太平洋/アピア
- NET - アジア/エレバン
- NST - 太平洋/オークランド
- PLT - アジア/カラチ
- PRT - アメリカ/プエルトリコ
- PST - アメリカ/ロスアンジェルス
- SST - 太平洋/ガダルカナル
- VST - アジア/ホーチミン

サポートされているリージョンベースの IDs

日付および時刻関数は、UTC+00:00 との関係で整理された、次のリージョンベースの IDs をサポートします。

- その他/GMT+12 (UTC-12:00)
- 太平洋/パゴパゴ (UTC-11:00)
- 太平洋/サモア (UTC-11:00)
- 太平洋/二ウエ (UTC-11) :00)
- アメリカ/サモア (UTC-11:00)
- その他/GMT+11 (UTC-11:00)
- 太平洋/ミッドウェイ (UTC-11:00)
- 太平洋/ホノルル (UTC-10:00)
- 太平洋/ラロトンガ島 (UTC-10:00)
- 太平洋/タヒチ (UTC-10:00)
- 太平洋/ジョンストン (UTC-10:00)
- アメリカ/ハワイ (UTC-10:00)
- SystemV/HST10 (UTC-10:00)
- その他/GMT+10 (UTC-10:00)
- 太平洋/マルケサス諸島 (UTC-09:30)
- その他/GMT+9 (UTC-09:00)
- 太平洋/ガンビア (UTC-09:00)
- アメリカ/アトカ (UTC-09:00)
- SystemV/YST9 (UTC-09:00)
- アメリカ/アダック (UTC-09:00)
- アメリカ/アリューシャン (UTC-09:00)
- その他/GMT+8 (UTC-08:00)
- アメリカ/アラスカ (UTC-08:00)
- アメリカ/ジュノー (UTC-08:00)
- アメリカ/メトラカトラ (UTC-08:00)
- アメリカ/ヤクタト (UTC-08:00)

- 太平洋/ピトケアン (UTC-08:00)
- アメリカ/シトカ (UTC-08:00)
- アメリカ/アンカレッジ (UTC-08:00)
- SystemV/PST8 (UTC-08:00)
- アメリカ/ノーム (UTC-08:00)
- SystemV/YST9YDT (UTC-08:00)
- カナダ/ユーコン (UTC-07:00)
- アメリカ/太平洋-新 (UTC-07:00)
- その他/GMT+7 (UTC-07:00)
- アメリカ/アリゾナ州 (UTC-07:00)
- アメリカ/ドーソンクリーク (UTC-07:00)
- カナダ/太平洋 (UTC-07:00)
- PST8PDT (UTC-07:00)
- SystemV/MST7 (UTC-07:00)
- アメリカ/ドーソン (UTC-07:00)
- メキシコ/ BajaNorte (UTC-07:00)
- アメリカ/ティファナ (UTC-07:00)
- アメリカ/クレストン (UTC-07:00)
- アメリカ/エルモシロ (UTC-07:00)
- アメリカ/サンタイザベル (UTC-07:00)
- アメリカ/バンクーバー (UTC-07:00)
- アメリカ/エンセナダ (UTC-07:00)
- アメリカ/フェニックス (UTC-07:00)
- アメリカ/ホワイトホース (UTC-07:00)
- アメリカ/フォートネルソン (UTC-07:00)
- SystemV/PST8PDT (UTC-07:00)
- アメリカ/ロサンゼルス (UTC-07:00)
- アメリカ/太平洋 (UTC-07:00)
- アメリカ/エルサルバドル (UTC-06:00)

- アメリカ/グアテマラ (UTC-06:00)
- アメリカ/ベリーズ (UTC-06:00)
- アメリカ/マナグア (UTC-06:00)
- アメリカ/テグシガルパ (UTC-06:00)
- その他/GMT+6 (UTC-06:00)
- 太平洋/イースター (UTC-06:00)
- メキシコ/ BajaSur (UTC-06:00)
- アメリカ/レジーナ (UTC-06:00)
- アメリカ/デンバー (UTC-06:00)
- 太平洋/ガラパゴス (UTC-06:00)
- アメリカ/イエローナイフ (UTC-06:00)
- アメリカ/スウィフトカレント (UTC-06:00)
- アメリカ/イヌビック (UTC-06:00)
- アメリカ/マサトラン (UTC-06:00)
- アメリカ/ボイズ (UTC-06:00)
- アメリカ/コスタリカ (UTC-06:00)
- MST7MDT (UTC-06:00)
- SystemV/CST6 (UTC-06:00)
- アメリカ/チワワ (UTC-06:00)
- アメリカ/オジナガ (UTC-06:00)
- チリ/ EasterIsland (UTC-06:00)
- アメリカ/山地 (UTC-06:00)
- アメリカ/エドモントン (UTC-06:00)
- カナダ/山地 (UTC-06:00)
- アメリカ/ケンブリッジベイ (UTC-06:00)
- ナバホ (UTC-06:00)
- SystemV/MST7MDT (UTC-06:00)
- カナダ/サスカチュワン (UTC-06:00)
- アメリカ/シップロック (UTC-06:00)

- アメリカ/パナマ (UTC-05:00)
- アメリカ/シカゴ (UTC-05:00)
- アメリカ/シカゴ (UTC-05:00)
- その他/GMT+5 (UTC-05:00)
- メキシコ/一般 (UTC-05:00)
- アメリカ/ポルトアクレ (UTC-05:00)
- アメリカ/グアヤキル (UTC-05:00)
- アメリカ/ランキンインレット (UTC-05:00)
- アメリカ/中部 (UTC-05:00)
- アメリカ/レイニーリバー (UTC-05:00)
- アメリカ/インディアナ/ノックス (UTC-05:00)
- アメリカ/ノースダコタ/ビューラ (UTC-05:00)
- アメリカ/モンテレイ (UTC-05:00)
- アメリカ/ジャマイカ (UTC-05:00)
- アメリカ/アティコカン (UTC-05:00)
- アメリカ/コーラルハーバー (UTC-05:00)
- アメリカ/ノースダコタ/中部 (UTC-05:00)
- アメリカ/ケイマン (UTC-05:00)
- アメリカ/インディアナ/テルシティ (UTC-05:00)
- アメリカ/メキシコシティ (UTC-05:00)
- アメリカ/マタモロス (UTC-05:00)
- CST6CDT (UTC-05:00)
- アメリカ/ノックスインディアナ州 (UTC-05:00)
- アメリカ/ボゴタ (UTC-05:00)
- アメリカ/メノマニー (UTC-05:00)
- アメリカ/レゾリュート (UTC-05:00)
- SystemV/EST5 (UTC-05:00)
- カナダ/中部 (UTC-05:00)
- ブラジル/アクレ (UTC-05:00)

- アメリカ/カンクン (UTC-05:00)
- アメリカ/リマ (UTC-05:00)
- アメリカ/バイアバンデラス (UTC-05:00)
- アメリカ/インディアナ-スターク (UTC-05:00)
- アメリカ/リオブランコ (UTC-05:00)
- SystemV/CST6CDT (UTC-05:00)
- ジャマイカ (UTC-05:00)
- アメリカ/メリダ (UTC-05:00)
- アメリカ/ノースダコタ/ニューセーラム (UTC-05:00)
- アメリカ/ウィニペグ (UTC-05:00)
- アメリカ/クイアバ (UTC-04:00)
- アメリカ/マリゴ (UTC-04:00)
- アメリカ/インディアナ/ピーターズバーグ (UTC-04:00)
- チリ/コンチネンタル (UTC-04:00)
- アメリカ/グランドターク (UTC-04:00)
- キューバ (UTC-04:00)
- その他/GMT+4 (UTC-04:00)
- アメリカ/マナウス (UTC-04:00)
- アメリカ/フォートウェイン (UTC-04:00)
- アメリカ/セントトーマス (UTC-04:00)
- アメリカ/アンギラ (UTC-04:00)
- アメリカ/ハバナ (UTC-04:00)
- アメリカ/ミシガン (UTC-04:00)
- アメリカ/バルバドス (UTC-04:00)
- アメリカ/ルイビル (UTC-04:00)
- アメリカ/キュラソー (UTC-04:00)
- アメリカ/ガイアナ (UTC-04:00)
- アメリカ/マルティニーク (UTC-04:00)
- アメリカ/プエルトリコ (UTC-04:00)

- アメリカ/ポートオブスペイン (UTC-04:00)
- SystemV/AST4 (UTC-04:00)
- アメリカ/インディアナ/ベベイ (UTC-04:00)
- アメリカ/インディアナ/ビンセンズ (UTC-04:00)
- アメリカ/クラレンダイク (UTC-04:00)
- アメリカ/アンティグア (UTC-04:00)
- アメリカ/インディアナポリス (UTC-04:00)
- アメリカ/イカルイト (UTC-04:00)
- アメリカ/セントビンセント (UTC-04:00)
- アメリカ/ケンタッキー/ルイビル (UTC-04:00)
- アメリカ/ドミニカ (UTC-04:00)
- アメリカ/アスンシオン (UTC-04:00)
- EST5EDT (UTC-04:00)
- アメリカ/ナッソー (UTC-04:00)
- アメリカ/ケンタッキー/モンティセロ (UTC-04:00)
- ブラジル/西部 (UTC-04:00)
- アメリカ/アルバ (UTC-04:00)
- アメリカ/インディアナ/インディアナポリス (UTC-04:00)
- アメリカ/サンティアゴ (UTC-04:00)
- アメリカ/ラパス (UTC-04:00)
- アメリカ/サンダーベイ (UTC-04:00)
- アメリカ/インディアナ/マレンゴ (UTC-04:00)
- アメリカ/ブランサブロン (UTC-04:00)
- アメリカ/サントドミンゴ (UTC-04:00)
- アメリカ/東部 (UTC-04:00)
- カナダ/東部 (UTC-04:00)
- アメリカ/ポルトープランス (UTC-04:00)
- アメリカ/サンバルテルミー (UTC-04:00)
- アメリカ/ニピゴン (UTC-04:00)

- アメリカ/東インディアナ (UTC-04:00)
- アメリカ/セントルシア (UTC-04:00)
- アメリカ/モントセラト (UTC-04:00)
- アメリカ/ローワープリンセズ (UTC-04:00)
- アメリカ/デトロイト (UTC-04:00)
- アメリカ/トルトラ (UTC-04:00)
- アメリカ/ポルトヴェーリョ (UTC-04:00)
- アメリカ/カンポグランデ (UTC-04:00)
- アメリカ/バージン (UTC-04:00)
- アメリカ/パングナータング (UTC-04:00)
- アメリカ/モントリオール (UTC-04:00)
- アメリカ/インディアナ/ウィナマック (UTC-04:00)
- アメリカ/ボアビスタ (UTC-04:00)
- アメリカ/グレナダ (UTC-04:00)
- アメリカ/ニューヨーク (UTC-04:00)
- アメリカ/セントキッツ (UTC-04:00)
- アメリカ/カラカス (UTC-04:00)
- アメリカ/グアドループ (UTC-04:00)
- アメリカ/トロント (UTC-04:00)
- SystemV/EST5EDT (UTC-04:00)
- アメリカ/アルゼンチン/カタマルカ (UTC-03:00)
- カナダ/大西洋 (UTC-03:00)
- アメリカ/アルゼンチン/コルドバ (UTC-03:00)
- アメリカ/アラゲイナ (UTC-03:00)
- アメリカ/アルゼンチン/サルタ (UTC-03:00)
- その他/GMT+3 (UTC-03:00)
- アメリカ/モンテビデオ (UTC-03:00)
- ブラジル/東部 (UTC-03:00)
- アメリカ/アルゼンチン/メンドーサ (UTC-03:00)
- アメリカ/アルゼンチン/リオガジェゴス (UTC-03:00)

- アメリカ/カタマルカ (UTC-03:00)
- アメリカ/コルドバ (UTC-03:00)
- アメリカ/サンパウロ (UTC-03:00)
- アメリカ/アルゼンチン/フフイ (UTC-03:00)
- アメリカ/カイエンヌ (UTC-03:00)
- アメリカ/レシフェ (UTC-03:00)
- アメリカ/ブエノスアイレス (UTC-03:00)
- アメリカ/パラマリボ (UTC-03:00)
- アメリカ/モンクトン (UTC-03:00)
- アメリカ/メンドーサ (UTC-03:00)
- アメリカ/サンタレム (UTC-03:00)
- 大西洋/バミューダ (UTC-03:00)
- アメリカ/マセイオ (UTC-03:00)
- 大西洋/スタンレー (UTC-03:00)
- アメリカ/ハリファックス (UTC-03:00)
- 南極大陸/ロセラ (UTC-03:00)
- アメリカ/アルゼンチン/サンルイス (UTC-03:00)
- アメリカ/アルゼンチン/ウシュアイア (UTC-03:00)
- 南極大陸/パーマー (UTC-03:00)
- アメリカ/プンタアレナス (UTC-03:00)
- アメリカ/グレイスベイ (UTC-03:00)
- アメリカ/フォルタレザ (UTC-03:00)
- アメリカ/チュール (UTC-03:00)
- アメリカ/アルゼンチン/ラリオハ州 (UTC-03:00)
- アメリカ/ベレム (UTC-03:00)
- アメリカ/フフイ (UTC-03:00)
- アメリカ/バイア (UTC-03:00)
- アメリカ/グースベイ (UTC-03:00)
- アメリカ/アルゼンチン/サンファン (UTC-03:00)
- アメリカ/アルゼンチン/ ComodRivadavia (UTC-03:00)

- アメリカ/アルゼンチン/トゥクマン (UTC-03:00)
- アメリカ/ロサリオ (UTC-03:00)
- SystemV/AST4ADT (UTC-03:00)
- アメリカ/アルゼンチン/ブエノスアイレス (UTC-03:00)
- アメリカ/セントジョンズ (UTC-02:30)
- カナダ/ニューファンドランド (UTC-02:30)
- アメリカ/ミクロン (UTC-02:00)
- その他/GMT+2 (UTC-02:00)
- アメリカ/ゴッドホープ (UTC-02:00)
- アメリカ/ノローニャ (UTC-02:00)
- ブラジル/ DeNoronha (UTC-02:00)
- 大西洋/サウスジョージア (UTC-02:00)
- その他/GMT+1 (UTC-01:00)
- 大西洋/カーボベルデ (UTC-01:00)
- 太平洋/キリティマティ (UTC+14:00)
- その他/GMT-14 (UTC+ 14:00)
- 太平洋/ファカオフォ (UTC+13:00)
- 太平洋/エンダーベリー (UTC+13:00)
- 太平洋/アピア (UTC+13:00)
- 太平洋/トンガタプ (UTC+13:00)
- その他/GMT-13 (UTC+ 13:00)
- NZ-CHAT (UTC+12:45)
- 太平洋/チャタム (UTC+12:45)
- 太平洋/クェゼリン (UTC+12:00)
- 南極大陸/McMurdo (UTC+12:00)
- 太平洋/ウォリス (UTC+12:00)
- 太平洋/フィジー (UTC+12:00)
- 太平洋/フナフチ (UTC+12:00)
- 太平洋/ナウル (UTC+12:00)
- クェゼリン (UTC+12:00)

- NZ (UTC+ 12:00)
- 太平洋/ウェイク (UTC+12:00)
- 南極大陸/南極 (UTC+12:00)
- 太平洋/タラワ (UTC+12:00)
- 太平洋/オークランド (UTC+12:00)
- アジア/カムチャツカ (UTC+12:00)
- その他/GMT-12 (UTC+ 12:00)
- アジア/アナディリ (UTC+ 12:00)
- 太平洋/マジュロ (UTC+12:00)
- 太平洋/ポナペ (UTC+11:00)
- 太平洋/ブーゲンビル (UTC+11:00)
- 南極大陸/マッカリー (UTC+11:00)
- 太平洋/ポンペイ (UTC+11:00)
- 太平洋/エフェート (UTC+11:00)
- 太平洋/ノーフォーク (UTC+11:00)
- アジア/マガダン (UTC+11:00)
- 太平洋/コスラエ (UTC+11:00)
- アジア/サハリン (UTC+11:00)
- 太平洋/ヌメア (UTC+11:00)
- その他/GMT-11 (UTC+ 11:00)
- アジア/スレドネコリムスク (UTC+11:00)
- 太平洋/ガダルカナル (UTC+11:00)
- オーストラリア/ロードハウ (UTC+10:30)
- オーストラリア/LHI (UTC+10:30)
- オーストラリア/ホバート (UTC+10:00)
- 太平洋/ヤップ (UTC+10:00)
- オーストラリア/タスマニア (UTC+10:00)
- 太平洋/ポートモレスビー (UTC+10:00)
- オーストラリア/ACT (UTC+10:00)
- オーストラリア/ビクトリア (UTC+10:00)

- 太平洋/チューク (UTC+10:00)
- オーストラリア/クイーンズランド (UTC+10:00)
- オーストラリア/キャンベラ (UTC+10:00)
- オーストラリア/カリー (UTC+10:00)
- 太平洋/グアム (UTC+ 10:00)
- 太平洋/トラック (UTC+10:00)
- オーストラリア/NSW (UTC+10:00)
- アジア/ウラジオストク (UTC+10:00)
- 太平洋/サイパン (UTC+10:00)
- 南極大陸/デュモンデュルビル (UTC+10:00)
- オーストラリア/シドニー (UTC+10:00)
- オーストラリア/ブリスベン (UTC+10:00)
- その他/GMT-10 (UTC+ 10:00)
- アジア/ウストネーラ (UTC+10:00)
- オーストラリア/メルボルン (UTC+10:00)
- オーストラリア/リンデマン (UTC+10:00)
- オーストラリア/北部 (UTC+09:30)
- オーストラリア/ヤンコウィンナ (UTC+09:30)
- オーストラリア/アデレード (UTC+09:30)
- オーストラリア/ブロークンヒル (UTC+09:30)
- オーストラリア/南部 (UTC+09:30)
- オーストラリア/ダーウィン (UTC+09:30)
- その他/GMT-9 (UTC+ 09:00)
- 太平洋/パラオ (UTC+09:00)
- アジア/チタ (UTC+09:00)
- アジア/ディリ (UTC+09:00)
- アジア/ジャヤプラ (UTC+09:00)
- アジア/ヤクーツク (UTC+09:00)
- アジア/平壌 (UTC+09:00)
- ROK (UTC+09:00)

- アジア/ソウル (UTC+09:00)
- アジア/ハンディガ (UTC+09:00)
- 日本 (UTC+09:00)
- アジア/東京 (UTC+09:00)
- オーストラリア/ユークラ (UTC+08:45)
- アジア/クチン (UTC+08:00)
- アジア/重慶 (UTC+08:00)
- その他/GMT-8 (UTC+ 08:00)
- オーストラリア/パース (UTC+08:00)
- アジア/マカオ (UTC+08:00)
- アジア/マカオ (UTC+08:00)
- アジア/チョイバルサン (UTC+08:00)
- アジア/上海 (UTC+08:00)
- 南極大陸/ケーシー (UTC+08:00)
- アジア/ウランバートル (UTC+08:00)
- アジア/重慶 (UTC+08:00)
- アジア/ウランバートル (UTC+08:00)
- アジア/台北 (UTC+08:00)
- アジア/マニラ (UTC+08:00)
- PRC (UTC+08:00)
- アジア/ウジュンパンダン (UTC+08:00)
- アジア/ハルビン (UTC+08:00)
- シンガポール (UTC+08:00)
- アジア/ブルネイ (UTC+08:00)
- オーストラリア/西部 (UTC+08:00)
- アジア/香港 (UTC+08:00)
- アジア/マカッサル (UTC+08:00)
- 香港 (UTC+08:00)
- アジア/クアラルンプール (UTC+08:00)
- アジア/イルクーツク (UTC+08:00)

- アジア/シンガポール (UTC+08:00)
- アジア/ポンティアナック (UTC+07:00)
- その他/GMT-7 (UTC+ 07:00)
- アジア/プノンペン (UTC+07:00)
- アジア/ノボシビルスク (UTC+07:00)
- 南極大陸/デイビス (UTC+07:00)
- アジア/トムスク (UTC+07:00)
- アジア/ジャカルタ (UTC+07:00)
- アジア/バルナウル (UTC+07:00)
- インド洋/クリスマス (UTC+07:00)
- アジア/ホーチミン (UTC+07:00)
- アジア/ホバート (UTC+07:00)
- アジア/バンコク (UTC+07:00)
- アジア/ビエンチャン (UTC+07:00)
- アジア/ノボクズネツク (UTC+07:00)
- アジア/クラスノヤルスク (UTC+07:00)
- アジア/サイゴン (UTC+07:00)
- アジア/ヤンゴン (UTC+06:30)
- アジア/ラングーン (UTC+06:30)
- インド洋/ココス (UTC+06:30)
- アジア/カシュガル (UTC+06:00)
- その他/GMT-6 (UTC+ 06:00)
- アジア/アルマティ (UTC+06:00)
- アジア/ダッカ (UTC+06:00)
- アジア/オムスク (UTC+06:00)
- アジア/ダッカ (UTC+06:00)
- インド洋/チャゴス (UTC+06:00)
- アジア/キジローダ (UTC+06:00)
- アジア/ビシュケク (UTC+06:00)
- 南極大陸/ボストーク (UTC+06:00)

- アジア/ウルムチ (UTC+06:00)
- アジア/ティンプー (UTC+06:00)
- アジア/ティンプー (UTC+06:00)
- アジア/カトマンズ (UTC+05:45)
- アジア/カトマンズ (UTC+05:45)
- アジア/コルカタ (UTC+05:30)
- アジア/コロンボ (UTC+05:30)
- アジア/カルカッタ (UTC+05:30)
- アジア/アクタウ (UTC+05:00)
- その他/GMT-5 (UTC+ 05:00)
- アジア/サマルカンド (UTC+05:00)
- アジア/カラチ (UTC+05:00)
- アジア/エカテリンブルク (UTC+05:00)
- アジア/ドゥシャンベ (UTC+ 05:00)
- アジア/ドゥシャンベ (UTC+05:00)
- アジア/オラール (UTC+05:00)
- アジア/タシケント (UTC+05:00)
- 南極大陸/モーソン (UTC+05:00)
- アジア/アクトベ (UTC+05:00)
- アジア/アシュハバード (UTC+05:00)
- アジア/アシガバート (UTC+05:00)
- アジア/アティラウ (UTC+05:00)
- インド洋/ケルゲレン (UTC+05:00)
- イラン (UTC+04:30)
- アジア/テヘラン (UTC+04:30)
- アジア/カブール (UTC+04:30)
- アジア/エレバン (UTC+04:00)
- その他/GMT-4 (UTC+ 04:00)
- その他/GMT-4 (UTC+ 04:00)
- アジア/ドバイ (UTC+04:00)

- インド洋/レユニオン (UTC+04:00)
- ヨーロッパ/サラトフ (UTC+04:00)
- ヨーロッパ/サマラ (UTC+04:00)
- インド/マエ (UTC+04:00)
- アジア/バクー (UTC+04:00)
- アジア/マスカット (UTC+04:00)
- ヨーロッパ/ボルゴグラード (UTC+04:00)
- ヨーロッパ/アストラカン (UTC+04:00)
- アジア/トビリシ (UTC+04:00)
- ヨーロッパ/ウリヤノフスク (UTC+04:00)
- アジア/アデン (UTC+03:00)
- アフリカ/ナイロビ (UTC+03:00)
- ヨーロッパ/イスタンブール (UTC+03:00)
- その他/GMT-3 (UTC+ 03:00)
- ヨーロッパ/ザポロージエ (UTC+03:00)
- イスラエル (UTC+03:00)
- インド洋/コモロ (UTC+03:00)
- 南極大陸/シオワ (UTC+03:00)
- アフリカ/モガディシュ (UTC+03:00)
- ヨーロッパ/ブカレスト (UTC+03:00)
- アフリカ/アスメラ (UTC+03:00)
- ヨーロッパ/マリエハムン (UTC+03:00)
- アジア/イスタンブール (UTC+03:00)
- ヨーロッパ/ティラスポリ (UTC+03:00)
- ヨーロッパ/モスクワ (UTC+03:00)
- ヨーロッパ/キシナウ (UTC+03:00)
- ヨーロッパ/ヘルシンキ (UTC+03:00)
- アジア/ベイルート (UTC+03:00)
- アジア/テルアビブ (UTC+03:00)
- アフリカ/ジブチ (UTC+03:00)

- ヨーロッパ/シンフェロポリ (UTC+03:00)
- ヨーロッパ/ソフィア (UTC+03:00)
- アジア/ガザ (UTC+03:00)
- アフリカ/アスマラ (UTC+03:00)
- ヨーロッパ/リガ (UTC+03:00)
- アジア/バグダッド (UTC+03:00)
- アジア/ダマスカス (UTC+03:00)
- アフリカ/ダルエスサラーム (UTC+03:00)
- アフリカ/アディスアベバ (UTC+03:00)
- ヨーロッパ/ウジゴロド (UTC+03:00)
- アジア/エルサレム (UTC+03:00)
- アジア/リヤド (UTC+03:00)
- アジア/クウェート (UTC+03:00)
- ヨーロッパ/キーロフ (UTC+03:00)
- アフリカ/カンパラ (UTC+03:00)
- ヨーロッパ/ミンスク (UTC+03:00)
- アジア/カタール (UTC+03:00)
- ヨーロッパ/キエフ (UTC+03:00)
- アジア/バーレーン (UTC+03:00)
- ヨーロッパ/ビリニュス (UTC+03:00)
- インド洋/アンタナナリボ (UTC+03:00)
- インド洋/マヨット (UTC+03:00)
- ヨーロッパ/タリン (UTC+03:00)
- トルコ (UTC+03:00)
- アフリカ/ジュバ (UTC+03:00)
- アジア/ニコシア (UTC+03:00)
- アジア/ファマグスタ (UTC+03:00)
- W-SU (UTC+03:00)
- EET (UTC+03:00)
- アジア/ヘブロン (UTC+03:00)

- アジア/アンマン (UTC+03:00)
- ヨーロッパ/ニコシア (UTC+03:00)
- ヨーロッパ/アテネ (UTC+03:00)
- アフリカ/カイロ (UTC+02:00)
- アフリカ/ババーネ (UTC+02:00)
- ヨーロッパ/ブリュッセル (UTC+02:00)
- ヨーロッパ/ワルシャワ (UTC+02:00)
- CET (UTC+02:00)
- ヨーロッパ/ルクセンブルグ (UTC+02:00)
- その他/GMT-2 (UTC+ 02:00)
- リビア (UTC+02:00)
- アフリカ/キガリ (UTC+02:00)
- アフリカ/トリポリ (UTC+02:00)
- ヨーロッパ/カリーニングラード (UTC+02:00)
- アフリカ/ウィンドフック (UTC+02:00)
- ヨーロッパ/マルタ (UTC+02:00)
- ヨーロッパ/ビュッセン (UTC+02:00)
-
- ヨーロッパ/スコピエ (UTC+02:00)
- ヨーロッパ/サラエボ (UTC+02:00)
- ヨーロッパ/ローマ (UTC+02:00)
- ヨーロッパ/チューリッヒ (UTC+02:00)
- ヨーロッパ/ジブラルタル (UTC+02:00)
- アフリカ/ルブンバシ (UTC+02:00)
- ヨーロッパ/ファドゥーツ (UTC+02:00)
- ヨーロッパ/リュブリャナ (UTC+02:00)
- ヨーロッパ/ベルリン (UTC+02:00)
- ヨーロッパ/ストックホルム (UTC+02:00)
- ヨーロッパ/ブダペスト (UTC+02:00)
- ヨーロッパ/ザグレブ (UTC+02:00)

- ヨーロッパ/パリ (UTC+02:00)
- アフリカ/セウタ (UTC+02:00)
- ヨーロッパ/プラハ (UTC+02:00)
- 南極大陸/トロール (UTC+02:00)
- アフリカ/ハボローネ (UTC+02:00)
- ヨーロッパ/コペンハーゲン (UTC+02:00)
- ヨーロッパ/ウィーン (UTC+02:00)
- ヨーロッパ/ティラネ (UTC+02:00)
- MET (UTC+02:00)
- ヨーロッパ/アムステルダム (UTC+02:00)
- アフリカ/マプト (UTC+02:00)
- ヨーロッパ/サンマリノ (UTC+02:00)
- ポーランド (UTC+02:00)
- ヨーロッパ/アンドラ (UTC+02:00)
- ヨーロッパ/オスロ (UTC+02:00)
- ヨーロッパ/ポドゴリツァ (UTC+02:00)
- アフリカ/ブジュンブラ (UTC+02:00)
- 大西洋/ヤンマイエン (UTC+02:00)
- アフリカ/マセル (UTC+02:00)
- ヨーロッパ/マドリッド (UTC+02:00)
- アフリカ/ブランタイヤ (UTC+02:00)
- アフリカ/ルサカ (UTC+02:00)
- アフリカ/ハラレ (UTC+02:00)
- アフリカ/ハルツーム (UTC+02:00)
- アフリカ/ヨハネスブルグ (UTC+02:00)
- ヨーロッパ/ベオグラード (UTC+02:00)
- ヨーロッパ/ブラチスラバ (UTC+02:00)
- 北極/ロンゲルビアン (UTC+02:00)
- エジプト (UTC+02:00)

- ヨーロッパ/バチカン (UTC+02:00)
- ヨーロッパ/モナコ (UTC+02:00)
- ヨーロッパ/ロンドン (UTC+01:00)
- その他/GMT-1 (UTC+ 01:00)
- ヨーロッパ/ジャージー (UTC+01:00)
- ヨーロッパ/ガーンジー (UTC+01:00)
- ヨーロッパ/マン島 (UTC+01:00)
- アフリカ/チュニス (UTC+01:00)
- アフリカ/マラボ (UTC+01:00)
- GB-Eire (UTC+01:00)。
- アフリカ/ラゴス (UTC+01:00)
- アフリカ/アルジェ (UTC+01:00)
- GB (UTC+01:00)
- ポルトガル (UTC+01:00)
- アフリカ/サントメ (UTC+01:00)
- アフリカ/ンジャメナ (UTC+01:00)
- 大西洋/フェロー (UTC+01:00)
- アイルランド (UTC+01:00)
- 大西洋/フェロー (UTC+01:00)
- ヨーロッパ/ダブリン (UTC+01:00)
- アフリカ/リブレビル (UTC+01:00)
- アフリカ/エルアイウン (UTC+01:00)
- アフリカ/エルアイウン (UTC+01:00)
- アフリカ/ドゥアラ (UTC+01:00)
- アフリカ/ブラザビル (UTC+01:00)
- アフリカ/ポルトノボ (UTC+01:00)
- 大西洋/マデイラ (UTC+01:00)
- ヨーロッパ/リスボン (UTC+01:00)
- 大西洋/Canary (UTC+01:00)

- アフリカ/カサブランカ (UTC+01:00)
- ヨーロッパ/ベルファスト (UTC+01:00)
- アフリカ/ルアンダ (UTC+01:00)
- アフリカ/キンシャサ (UTC+01:00)
- アフリカ/バンギ (UTC+01:00)
- WET (UTC+01:00)
- アフリカ/ニアメー (UTC+01:00)
- GMT (UTC+00:00)
- その他/GMT-0 (UTC+ 00:00)
- 大西洋/セントヘレナ (UTC+00:00)
- その他/GMT+0 (UTC+ 00:00)
- アフリカ/バンジュール (UTC+00:00)
- その他etc/GMT (UTC+ 00:00)
- アフリカ/フリータウン (UTC+00:00)
- アフリカ/バマコ (UTC+00:00)
- アフリカ/コナクリ (UTC+00:00)
- ユニバーサル (UTC+ 00:00)
- アフリカ/ヌアクショット (UTC+00:00)
- UTC (UTC+00:00)
- その他/ユニバーサル (UTC+00:00)
- 大西洋/アゾレス (UTC+00:00)
- アフリカ/アビジャン (UTC+00:00)
- アフリカ/アクラ (UTC+00:00)
- その他/UCT (UTC+ 00:00)
- GMT0 (UTC+00:00)
- ズールー (UTC+00:00) ズールー (UTC+00:00)
- アフリカ/ワガドゥグ (UTC+00:00)
- 大西洋/レイキャビク (UTC+00:00)
- その他/ズールー (UTC+ 00:00)
- アイスランド (UTC+00:00)

- アフリカ/ロメ (UTC+00:00)
- グリニッジ (UTC+00:00)
- その他/gmt0 (UTC+ 00:00)
- アメリカ/ダンマルクシャーン (UTC+00:00)
- アフリカ/ダカール (UTC+00:00)
- アフリカ/ビサウ (UTC+00:00)
- その他/グリニッジ (UTC+ 00:00)
- アフリカ/ティンブクトゥ (UTC+00:00)
- UCT (UTC+00:00)
- アフリカ/モンロビア (UTC+00:00)
- その他/UTC (UTC+00:00)

数式表現チュートリアル

AWS IoT SiteWiseで数式表現を使うには、次のチュートリアルを参考にしてください。

トピック

- 数式で文字列を使用する。
- データポイントのフィルタリング
- 条件に一致するデータポイントを数える。
- 数式の遅延データ
- 数式のデータ品質
- 未定義の値、無限の値、およびオーバーフロー値

数式で文字列を使用する。

数式表現の中で文字列を操作することができます。また、属性や測定のプロパティを参照する変数から文字列を入力することができます。

> **Important**
>
> 数式では、倍値または文字列値のみを出力できます。入れ子になった式は、文字列など他のデータ型を出力することができますが、式全体としては、数値または文字列として評価されなければなりません。[jp function] (jp関数) を使うと、文字列を数値に変換することができま

す。ブール値は 1 (true) または 0 (false) でなければならない。詳細については、「未定義の値、無限の値、およびオーバーフロー値」を参照してください。

AWS IoT SiteWise には、文字列の操作に使用できる次の式式機能が用意されています。

- [String literals] (文字列リテラル)
- [index operator] (インデックス演算子) (`s[index]`)
- [slice operator] (スライス演算子) (`s[start:end:step]`)
- [Comparison functions]比較関数: 文字列を[lexicographic order] (辞書式順序) に比較する関数です。
- 文字列関数 。シリアル化された JSON オブジェクトを解析し、文字列を数値に変換できる `jp`関数が含まれます。

データポイントのフィルタリング

[if function] (if関数) を使用すると、条件を満たさないデータポイントをフィルタリングすることができます。`if` 関数は条件を評価し、`true`、`false` の結果に対して異なる値を返す。`if` 関数の 1 ケースの出力として[none constant] (無し定数) を使用すると、そのケースのデータポイントを破棄することができます。

条件に合致するデータポイントをフィルタリングする場合。

- `if` 関数を使用して、ある条件を満たすかどうかをチェックする条件を定義し、none を `result_if_true` または `result_if_false` の値として返す変換を作成します。

Example 例: 水が沸騰していないデータポイントをフィルタリングする。

ある機械の中の水の温度 (摂氏) を測定する測定器 `temp_c` があるとしましょう。次のような変換を定義することで、水が沸騰していないデータポイントをフィルタリングすることができます。

- 変換: `boiling_temps = if(gte(temp_c, 100), temp_c, none)` - 100℃以上であれば温度を返し、そうでなければデータポイントを返さない。

条件に一致するデータポイントを数える。

[comparison functions] (比較関数) と [sum ()] を使用して、条件が真であるデータポイントの数をカウントできます。

条件に一致するデータポイントをカウントする場合。

1. 比較関数を使用して、別のプロパティのフィルター条件を定義する変換を作成します。
2. その条件が満たされるデータポイントを合計するメトリクスを作成します。

Example 例: 水が沸騰しているデータポイントの数を数える

ある機械の中の水の温度 (摂氏) を測定する測定器 `temp_c` があるとしましょう。次の変換プロパティとメトリクスプロパティを定義して、水が沸騰しているデータポイントの数をカウントできます。

- 変換: `is_boiling = gte(temp_c, 100)` - 温度が100℃以上であれば `1`、そうでなければ`0`を返す。
- メトリクス: `boiling_count = sum(is_boiling)` - 水が沸騰しているデータポイントの数を返します。

数式の遅延データ

AWS IoT SiteWise は、最大 7 日経過したデータの遅延データ取り込みをサポートします。が遅延データ AWS IoT SiteWise を受信すると、過去のウィンドウに遅延データを入力するメトリクスの既存の値を再計算します。これらの再計算では、データ処理料金が発生します。

> **Note**
>
> が遅延データを入力するプロパティ AWS IoT SiteWise を計算する場合、各プロパティの現在の式を使用します。

がメトリクスの過去のウィンドウ AWS IoT SiteWise を再計算すると、そのウィンドウの以前の値が置き換えられます。そのメトリクスの通知を有効にした場合、 はプロパティ値通知 AWS IoT SiteWise も発行します。つまり、以前に通知を受信したのと同じプロパティとタイムスタンプについて、新しいプロパティ値の更新通知を受け取ることができます。アプリケーションまたはデータレイクがプロパティ値通知を使用する場合、データが正確になるように、前の値を新しい値で更新する必要があります。

数式のデータ品質

では AWS IoT SiteWise、各データポイントに品質コードがあり、次のいずれかになります。

- GOOD - データはいずれの問題の影響も受けません。
- BAD - データはセンサーの障害などの問題による影響を受けます。
- UNCERTAIN - データはセンサーの不正確さなどの問題による影響を受けます。

AWS IoT SiteWise は、変換とメトリクスを計算するときにGOOD品質データのみを消費します。は、正常な計算のためにGOOD品質データのみを AWS IoT SiteWise 出力します。計算が失敗した場合、 AWS IoT SiteWise はその計算のデータポイントを出力しません。これは、計算の結果、未定義、無限、またはオーバーフロー値になった場合に発生します。

データのクエリとデータ品質によるフィルタ処理の詳細については、「からのデータのクエリ AWS IoT SiteWise」を参照してください。

未定義の値、無限の値、およびオーバーフロー値

一部の数式 (`x / 0`、、 など`log(0)`) は`sqrt(-1)`、実数システムで定義されていない値、無限値、または でサポートされている範囲外の値を計算します AWS IoT SiteWise。アセットプロパティの式が未定義、無限、またはオーバーフローの値を計算する AWS IoT SiteWise 場合、その計算のデータポイントは出力されません。

AWS IoT SiteWise は、数式式の結果として数値以外の値を計算しても、データポイントを出力しません。つまり、文字列、配列、[none constant] (無し定数) をコンピューティングする数式を定義した場合、 AWS IoT SiteWise はそのコンピューティングのデータポイントを出力しないのです。

Example 例

次の式はそれぞれ、数値として表現 AWS IoT SiteWise できない値になります。これらの式を計算するときに、データポイントは出力 AWS IoT SiteWise されません。

- `x / 0` は未定義です。
- `log(0)` は未定義です。
- `sqrt(-1)` は実数系では未定義です。
- `"hello" + " world"` は文字列です。
- `jp('{"values":[3,6,7]}', '$.values')` は配列です。
- `if(gte(temp, 300), temp, none)` が none より小さいとき、`temp` は `300` です。

カスタム複合モデルの作成 (コンポーネント)

カスタム複合モデル、またはコンソールを使用している場合はコンポーネントは、アセットモデルとコンポーネントモデルに別のレベルの組織を提供します。プロパティをグループ化するか、他のモデルを参照することで、モデルを構造化できます。カスタム複合モデルの使用の詳細については、「」を参照してくださいカスタム複合モデル (コンポーネント)。

既存のアセットモデルまたはコンポーネントモデル内にカスタム複合モデルを作成します。カスタム複合モデルには 2 つのタイプがあります。モデル内の関連プロパティをグループ化するには、インラインカスタム複合モデルを作成できます。アセットモデルまたはコンポーネントモデル内のコンポーネントモデルを参照するには、コンポーネントモデルベースのカスタム複合モデルを作成できます。

以下のセクションでは、 AWS IoT SiteWise API を使用してカスタム複合モデルを作成する方法について説明します。

トピック

- インラインコンポーネントの作成 (コンソール)
- インラインカスタム複合モデルの作成 (AWS CLI)
- component-model-based コンポーネントの作成 (コンソール)
- component-model-based カスタム複合モデルの作成 (AWS CLI)

インラインコンポーネントの作成 (コンソール)

AWS IoT SiteWise コンソールを使用して、独自のプロパティを定義するインラインコンポーネントを作成できます。

> **Note**
>
> これはインラインコンポーネントであるため、これらのプロパティは現在のアセットモデルにのみ適用され、他の場所では共有されません。
> 再利用可能なモデルを作成する必要がある場合 (複数のアセットモデル間で共有する場合や、1 つのアセットモデル内に複数のインスタンスを含める場合など)、代わりにコンポーネントモデルに基づいてコンポーネントを作成する必要があります。詳細については、次のセクションを参照してください。

コンポーネントを作成するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[モデル] を選択します。
3. コンポーネントを追加するアセットモデルを選択します。
4. プロパティ タブで、コンポーネント を選択します。
5. [コンポーネントを作成] を選択します。
6. 「コンポーネントの作成」ページで、次の操作を行います。

   a. **ServoMotor** や など、コンポーネントの名前を入力します**ServoMotor Model**。この名前は、このリージョンのアカウント内のすべてのコンポーネントで一意である必要があります。

   b. (オプション) モデルの [属性定義] を追加します。属性は、ほとんど変化しない情報を表します。詳細については、「静的データ (属性) の定義」を参照してください。

   c. (オプション) モデルの [測定の定義] を追加します。測定値は、機器からのデータストリームを表します。詳細については、「機器からのデータストリームの定義 (測定値) 」を参照してください。

   d. (オプション) モデルの [定義を変換する] を追加します。変換とは、あるフォームから別のフォームにデータをマッピングするための数式です。詳細については、「データの変換 (変換) 」を参照してください。

   e. (オプション) モデルの [メトリクスの定義] を追加します。メトリクスとは、時間間隔でのデータを集計する数式です。メトリクスは、関連するアセットからデータを入力できるため、オペレーションまたはオペレーションのサブセットを表す値をコンピューティングすることができます。詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください。

   f. [コンポーネントを作成] を選択します。

インラインカスタム複合モデルの作成 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して、独自のプロパティを定義するインラインカスタム複合モデルを作成できます。

Model CreateAssetModelCompositeオペレーションを使用して、プロパティを含むインラインモデルを作成します。このオペレーションでは、次の構造を持つペイロードが必要です。

> **Note**
>
> これはインライン複合モデルであるため、これらのプロパティは現在のアセットモデルにのみ適用され、他の場所では共有されません。これが「インライン」になるのは、`composedAssetModelId`フィールドに値を提供しないということです。
> 再利用可能なモデルを作成する必要がある場合 (複数のアセットモデル間で共有する場合や、1 つのアセットモデル内に複数のインスタンスを含める場合など) 、代わりにコンポーネントモデルベースの複合モデルを作成する必要があります。詳細については、次のセクションを参照してください。

```
{
    "assetModelCompositeModelName": "CNCLathe_ServoMotorA",
    "assetModelCompositeModelType": "CUSTOM",
    "assetModelCompositeModelProperties": [
        {
            "dataType": "DOUBLE",
            "name": "Servo Motor Temperature",
            "type": {
            "measurement": {}
            },
            "unit": "Celsius"
        },
        {
            "dataType": "DOUBLE",
            "name": "Spindle speed",
            "type": {
            "measurement": {}
            },
            "unit": "rpm"
        }
    ]
}
```

component-model-based コンポーネントの作成 (コンソール)

AWS IoT SiteWise コンソールを使用して、コンポーネントモデルに基づいてコンポーネントを作成できます。

component-model-based コンポーネントを作成するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[モデル] を選択します。
3. コンポーネントを追加するアセットモデルを選択します。
4. プロパティ タブで、コンポーネント を選択します。
5. [コンポーネントを作成] を選択します。
6. 「コンポーネントの作成」ページで、次の操作を行います。

   a. コンポーネントの基礎となるコンポーネントモデルを選択します。

   b. **ServoMotor** や など、コンポーネントの名前を入力します**ServoMotor Model**。この名前は、このリージョンのアカウント内のすべてのコンポーネントで一意である必要があります。

   c. [コンポーネントを作成] を選択します。

component-model-based カスタム複合モデルの作成 (AWS CLI)

を使用して、アセットモデル内に component-model-based カスタム複合モデル AWS CLI を作成できます。 component-model-based カスタム複合モデルは、既に他の場所で定義したコンポーネントモデルへの参照です。

CreateAssetModelCompositeModel オペレーションを使用して、 component-model-based カスタム複合モデルを作成します。このオペレーションでは、次の構造を持つペイロードが必要です。

> **Note**
>
> この例では、 の値は、既存のコンポーネントモデルのアセットモデル ID または外部 ID `composedAssetModelId`です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。コンポーネントモデルの作成方法の例については、「」を参照してくださいコンポーネントモデルの作成 (AWS CLI)。

```
{
    "assetModelCompositeModelName": "CNCLathe_ServoMotorA",
    "assetModelCompositeModelType": "CUSTOM",
    "composedAssetModelId": component model ID
]
```

これは単なるリファレンスであるため、 component-model-based カスタム複合モデルには名前以外の独自のプロパティはありません。

同じコンポーネントの複数のインスタンスをアセットモデル (例えば、複数のサーボモーターを持つ CNC マシン) に追加する場合は、それぞれに独自の名前があるが、すべて同じ を参照する複数の component-model-based カスタム複合モデルを追加できますcomposedAssetModelId。

コンポーネントは他のコンポーネント内にネストできます。そのためには、この例に示すように、component-model-based 複合モデルをコンポーネントモデルの 1 つに追加できます。

アセットを作成する

アセットモデルからアセットを作成できます。アセットを作成する前に、アセットモデルが必要です。アセットモデルを作成していない場合は、「アセットモデルを作成する」を参照してください。

> **Note**
>
> アセットは ACTIVE モデルからのみ作成できます。モデルの状態が ACTIVE でない場合は、そのモデルからアセットを作成できるようになるまで数分待たなければならない場合があります。詳細については、「アセットおよびモデルの状態」を参照してください。

トピック

- アセットを作成する (コンソール)
- アセットの作成 (AWS CLI)
- 新しいアセットの設定

アセットを作成する (コンソール)

AWS IoT SiteWise コンソールを使用してアセットを作成できます。

アセットを作成するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. [Create asset (アセットの作成)] を選択します。
4. [アセットの作成] ページで、次の操作を実行します。

a. [モデル] で、アセットを作成するアセットモデルを選択します。

> **Note**
>
> モデルが [アクティブ] でない場合は、アクティブになるまで待つか、[FAILED (失敗)] の場合は問題を解決する必要があります。

b. アセットの [名前] を入力します。

c. (オプション) アセットにタグを追加します。詳細については、「リソースにタグを付ける AWS IoT SiteWise」を参照してください。

d. [Create asset (アセットの作成)] を選択します。

アセットを作成すると、 AWS IoT SiteWise コンソールは新しいアセットのページに移動します。このページでは、アセットの [ステータス] が表示されます。このステータスは、最初は [作成中] です。このページは自動的に更新されるため、アセットのステータスが更新されるまで待つことができます。

> **Note**
>
> アセットの作成プロセスには、最大で 1 分かかる場合があります。[Status] (状態) が [ACTIVE] (アクティブ) になったら、アセットに対して更新オペレーションを実行できます。詳細については、「アセットおよびモデルの状態」を参照してください。

アセットを作成したら、「新しいアセットの設定」を参照してください。

アセットの作成 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して、アセットモデルからアセットを作成できます。

アセットを作成するには `assetModelId` が必要です。アセットモデルを作成したが、その がわからない場合は`assetModelId`、ListAssetModels API を使用してすべてのアセットモデルを表示します。

アセットモデルからアセットを作成するには、以下のパラメータを指定して CreateAsset API を使用します。

- `assetName` - 新しいアセットの名前。アセットを識別しやすい名前を付けます。
- `assetModelId` – アセットの ID。これは UUID 形式の実際の ID です。持っていexternalId:myExternalIdる場合は です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

アセットを作成するには (AWS CLI)

- 次のコマンドを実行して、アセットを作成します。*asset-name* をアセットの名前に置き換え、 *asset-model-id* をアセットモデルの ID または外部 ID に置き換えます。

```
aws iotsitewise create-asset \
  --asset-name asset-name \
  --asset-model-id asset-model-id
```

このオペレーションは、新しいアセットの詳細およびステータスを含むレスポンスを次の形式で返します。

```
{
  "assetId": "String",
  "assetArn": "String",
  "assetStatus": {
    "state": "String",
    "error": {
      "code": "String",
      "message": "String"
    }
  }
}
```

アセットの `state` は、アセットが作成されるまでは `CREATING` です。

> **Note**
>
> アセットの作成プロセスには、最大で 1 分かかる場合があります。アセットのステータスを確認するには、アセットの ID を `assetId`パラメータとして DescribeAssetオペレーションを使用します。アセットの `state` が `ACTIVE` になったら、アセットに対して更新オペレーションを実行できます。詳細については、「アセットおよびモデルの状態」を参照してください。

アセットを作成したら、「新しいアセットの設定」を参照してください。

新しいアセットの設定

次のいずれかのオプションアクションを使用して、アセットの設定を完了します。

- アセットに測定プロパティがある場合、アセットプロパティへの産業データストリームのマッピング。
- アセットに一意の属性値がある場合、属性値の更新。
- アセットが親アセットの場合、アセットの関連付けと関連付け解除。

アセットの検索

AWS IoT SiteWise コンソール 検索機能を使用して、メタデータとリアルタイムのプロパティ値フィルターに基づいてアセットを検索します。

前提条件

AWS IoT SiteWise では、産業データの整理とモデル化を改善する AWS IoT TwinMaker ために、 と統合するためのアクセス許可が必要です。にアクセス許可を付与している場合は AWS IoT SiteWise、 ExecuteQuery API を使用します。にアクセス許可を付与しておらず AWS IoT SiteWise、開始方法に関するサポートが必要な場合は、「」を参照してくださいAWS IoT SiteWise と AWS IoT TwinMaker の統合。

での高度な検索 AWS IoT SiteWise コンソール

メタデータ検索

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、アセット で詳細検索を選択します。
3. 詳細検索で、メタデータ検索オプションを選択します。
4. パラメータを入力します。効率的な検索のために、できるだけ多くのフィールドに入力します。

    a. アセット名 - 完全なアセット名、または詳細検索用の部分的な名前を入力します。

    b. プロパティ名 - 完全なプロパティ名、または検索範囲の部分的な名前を入力します。

    c. 演算子 — 以下から演算子を選択します。

- =
- <
- >
- <=
- >=

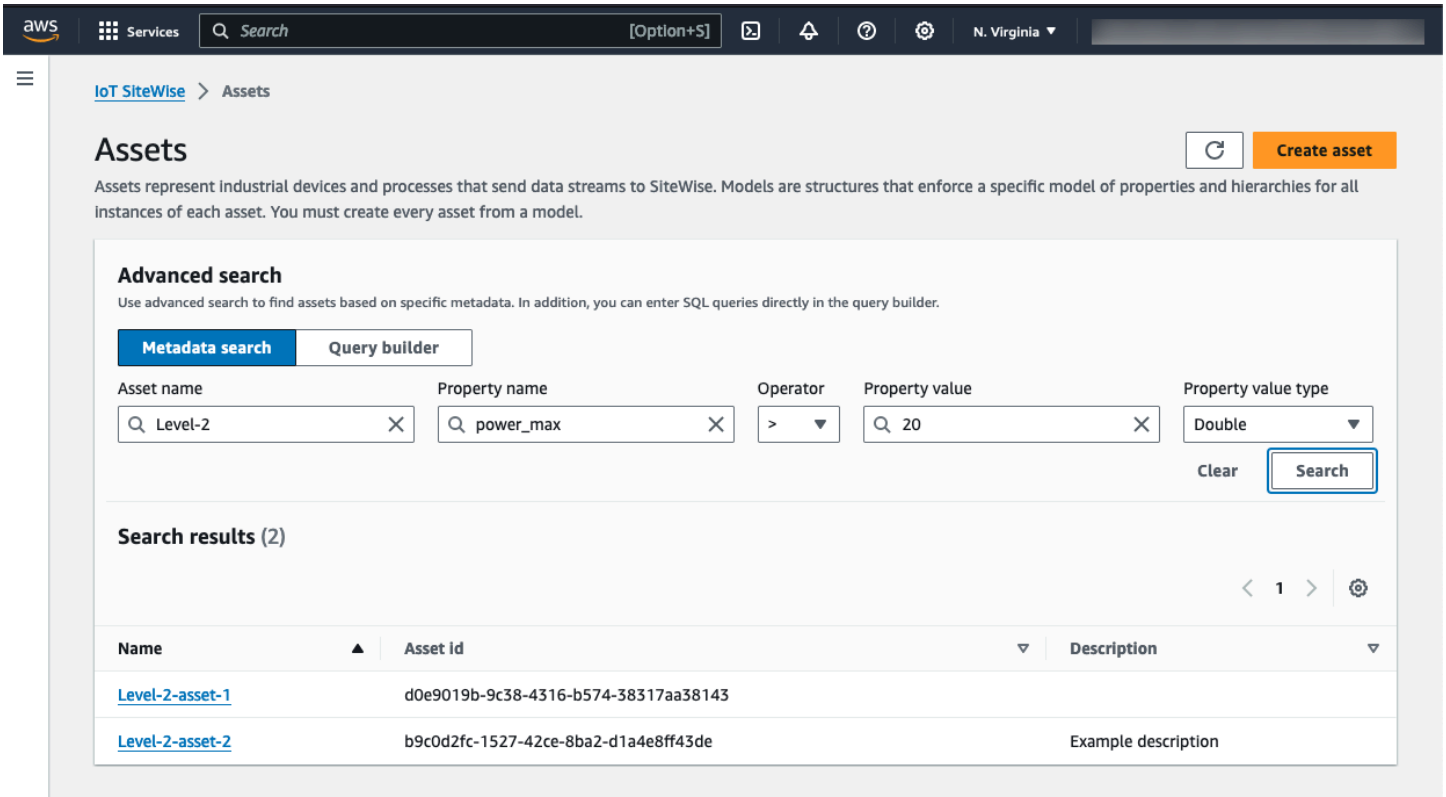
d. プロパティ値 — この値は、プロパティの最新の値と比較されます。

e. プロパティ値タイプ — プロパティのデータ型。次から選択します。

- ダブル
- 整数
- 文字列
- ブール値

5. [検索] を選択します。

6. 検索結果テーブルから、名前列からアセットを選択します。これにより、そのアセットの詳細なアセットページが表示されます。



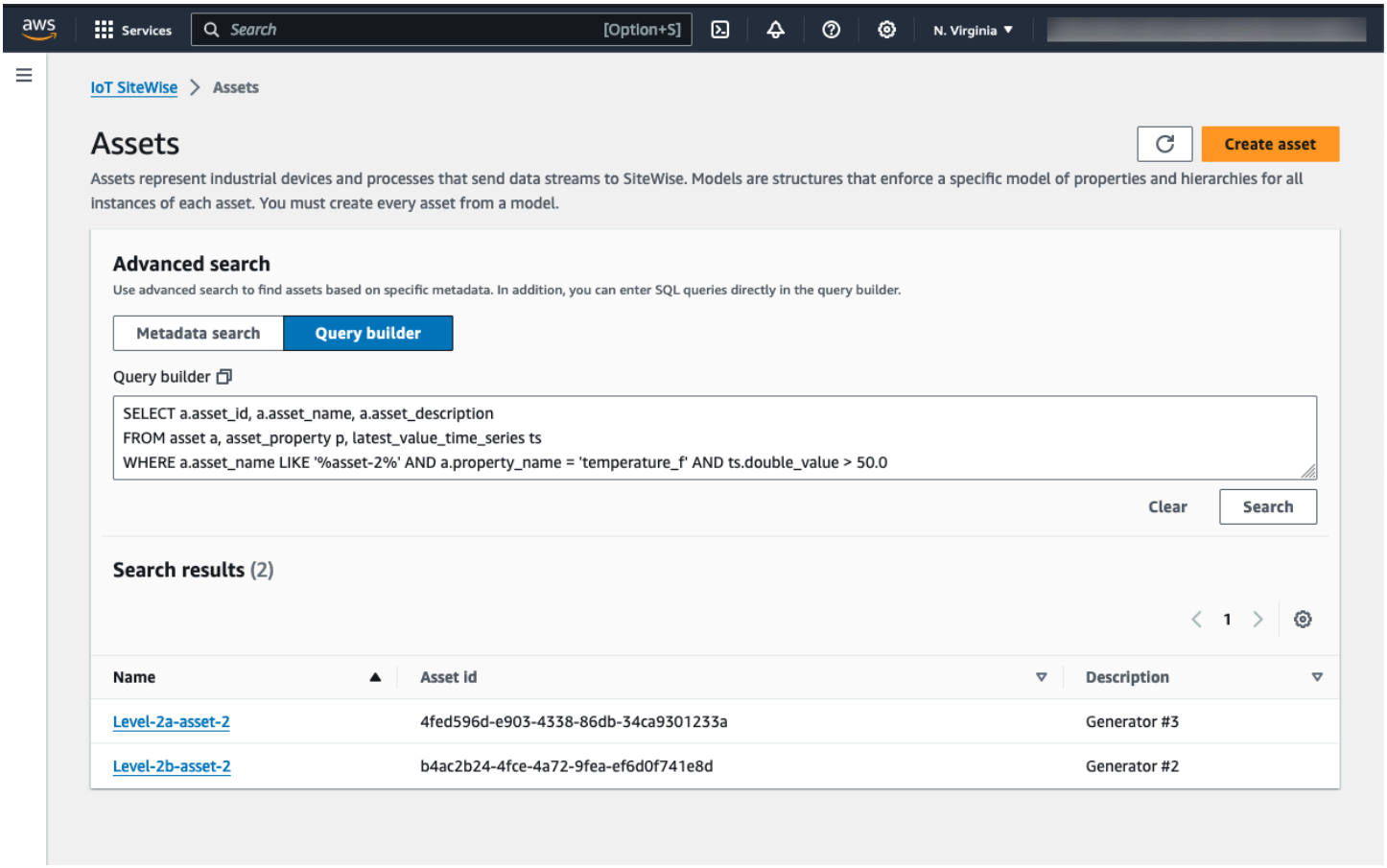
部分検索

アセット検索では、すべてのパラメータを指定する必要はありません。メタデータ検索オプションを使用した部分検索の例を次に示します。

- 名前でアセットを検索します。
  - アセット名フィールドのみに値を入力します。
  - プロパティ名フィールドとプロパティ値フィールドは空です。
- 特定の名前のプロパティを含むアセットを検索します。
  - プロパティ名フィールドのみに値を入力します。
  - アセット名フィールドとプロパティ値フィールドは空です。
- プロパティの最新の値に基づいてアセットを検索します。
  - プロパティ名とプロパティ値フィールドに値を入力します。
  - Operator と Property の値タイプ を選択します。

クエリビルダーの検索

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、アセット の下にある詳細検索を選択します。
3. 詳細検索で、クエリビルダーオプションを選択します。
4. クエリビルダーペインで、SQL クエリを記述して、`asset_name`、`asset_id`、および を取得しますasset_description。
5. [検索] を選択します。
6. 検索結果テーブルから、名前列からアセットを選択します。これにより、そのアセットの詳細なアセットページが表示されます。



> **Note**
>
> - SQL クエリの SELECT句にはasset_name、検索結果テーブルで有効なアセットを確保するために、 および asset_idフィールドを含める必要があります。
> - クエリビルダーは、結果テーブルに名前 、アセット ID 、説明のみを表示します。SELECT句にフィールドを追加しても、結果テーブルに列は追加されません。

アセットプロパティへの産業データストリームのマッピング

アセットプロパティでプロパティエイリアスを定義できます。これにより、アセットデータの取り込みまたは取得時にアセットプロパティを識別できます。アセットに測定プロパティがある場合は、プロパティエイリアスを定義して、データストリームをそのプロパティにマッピングすることができます。

このプロセスでは、プロパティのエイリアスを知っている必要があります。

- SiteWise Edge ゲートウェイ の OPC-UA データソースを使用して OPC-UA サーバーからデータを取り込む場合、プロパティエイリアスは Objects ノードの下の変数へのパスで、 で始まります/。

  Example

  変数へのパスが の場合`company/windfarm/3/turbine/7/temperature`、プロパティエイリアスは です`/company/windfarm/3/turbine/7/temperature`。

  OPC-UA 情報アーキテクチャの詳細については、「OPC UA オンラインリファレンス」の「情報モデルとアドレス空間マッピング」を参照してください。

  > **メモ**
  >
  > - OPC-UA ソースのデータストリームプレフィックスを設定する場合は、そのソースからのすべてのデータストリームのプロパティエイリアスにそのプレフィックスを含める必要があります。
  >
  >   Example
  >
  >   `/RentonWA` がプレフィックスの場合、以前のエイリアスは です`/RentonWA/company/windfarm/3/turbine/7/temperature`。
  > - プロパティのエイリアスは、最大 1,000 バイトまで含むことができます。OPC-UA 変数パスは最大 4,096 バイトまで含むことができます。現在、 AWS IoT SiteWise は、長いパスを持つ OPC-UA 変数からのデータの取り込みをサポートしていません。

- SiteWise Edge ゲートウェイ の Modbus TCP データソースを使用して Modbus サーバーからデータを取り込む場合、プロパティエイリアスは次のとおりです。

  Modbus `register set tag name`

  この値を使用して、このレジスタセットからアセットプロパティにデータを送信します。

- AWS IoT ルールや API など、他のソースからデータを取り込む場合は、プロパティエイリアスを定義する必要があります。デバイス設定に適用可能なプロパティエイリアス命名システムを定義できます。たとえば、 AWS IoT のモノからデータを取り込む場合、プロパティエイリアスにモノの名を含めることで、データストリームを一意に識別できます。この例の詳細については、AWS IoT「モノからのデータの取り込み」チュートリアルを参照してください。

プロパティエイリアスは、リージョンと AWS account AWS IoT SiteWise 内で一意である必要があります。プロパティエイリアスを別のアセットプロパティに既に存在するエイリアスに設定すると、エラーが返されます。

同じデータストリームパスを持つ複数の OPC-UA ソースがある場合は、各ソースのパスにプレフィックスを追加して一意のエイリアスを作成します。詳細については、「データソースの設定」を参照してください。

> **Note**
>
> このセクションでは、測定プロパティのプロパティエイリアスを設定する方法について説明します。外部アラーム状態プロパティのプロパティエイリアスを設定する方法については、外部アラーム状態ストリームのマッピング。 を参照してください。

トピック

- プロパティエイリアスの設定 (コンソール)
- プロパティエイリアスの設定 (AWS CLI)

プロパティエイリアスの設定 (コンソール)

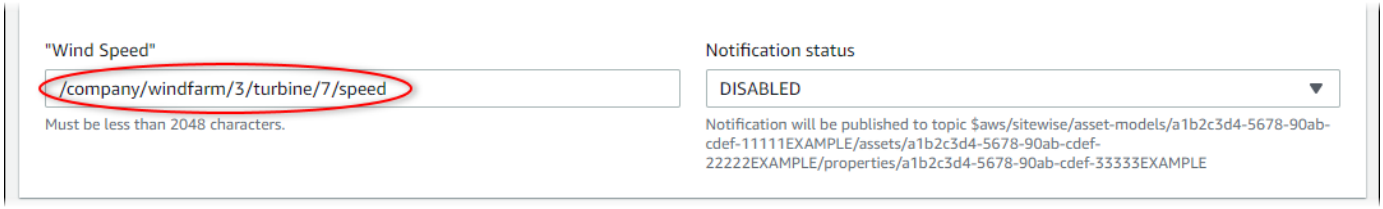
AWS IoT SiteWise コンソールを使用して、アセットプロパティのエイリアスを設定できます。

プロパティエイリアスを設定するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. プロパティエイリアスを設定するアセットを選択します。

   > **Tip**
   >
   > 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [編集] を選択します。
5. エイリアスを設定するプロパティを検索し、プロパティのエイリアスを入力します。



6. [保存] を選択します。

プロパティエイリアスの設定 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して、アセットプロパティのエイリアスを設定します。

この手順を完了するには、アセットの `assetId` とプロパティの `propertyId` を知っている必要があります。外部 ID を使用することもできます。アセットを作成し、その がわからない場合は`assetId`、 ListAssets API を使用して特定のモデルのすべてのアセットを一覧表示します。DescribeAsset オペレーションを使用して、プロパティ IDs を含むアセットのプロパティを表示します。

UpdateAssetプロパティオペレーションを使用して、データストリームをアセットの プロパティにマッピングします。以下のパラメータを指定します。

- `assetId` – アセットの ID または外部 ID。詳細については、AWS IoT SiteWise ユーザーガイドの外部 IDs を持つオブジェクトの参照を参照してください。
- `propertyId` – アセットプロパティの ID または外部 ID。
- `propertyAlias` - プロパティへのエイリアスへのデータストリームのパス。
- `propertyNotificationState` - プロパティ値の通知状態: `ENABLED` または `DISABLED`。プロパティのエイリアスを更新するときに、プロパティの既存の通知状態を指定します。DescribeAsset プロパティオペレーションを使用して、既存の通知状態を取得できます。

  このパラメータを省略すると、新しい通知状態が `DISABLED` になります。プロパティ通知の詳細については、「AWS 他のサービスとのやり取り」を参照してください。

プロパティエイリアスを設定するには (AWS CLI)

1. 次のコマンドを実行して、プロパティの現在の通知状態を取得します。*`asset-id`* と *`property-id`* をアセットプロパティの ID に置き換えます。

```
aws iotsitewise describe-asset-property \
  --asset-id asset-id \
  --property-id property-id
```

このオペレーションは、アセットプロパティの詳細を含むレスポンスを次の形式で返します。プロパティ通知の状態は JSON オブジェクトの assetProperty.notification.state にあります。

```
{
  "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
  "assetName": "Wind Turbine 7",
  "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
  "assetProperty": {
    "id": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
    "name": "Wind Speed",
    "notification": {
      "topic": "$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/
assets/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE/properties/a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE",
      "state": "ENABLED"
    },
    "dataType": "DOUBLE",
    "unit": "m/s",
    "type": {
      "measurement": {}
    }
  }
}
```

2. 次のコマンドを実行して、アセットプロパティのエイリアスを設定します。*property-alias* をプロパティエイリアス、*notification-state* を通知状態に置き換えるか、--property-notification-state を省略して通知を無効にします。オプションで新しい*##*と --property-unit を使用して、アセットの単位を更新することもできます。

```
aws iotsitewise update-asset-property \
  --asset-id asset-id \
  --property-id property-id \
  --property-alias property-alias \
  --property-notification-state notification-state \
  --property-unit unit
```

3. エイリアスが設定されていることを確認するには、以下のコマンドを実行してプロパティの詳細を取得します。*asset-id* と *property-id* をアセットプロパティの ID に置き換えます。

```
aws iotsitewise describe-asset-property \
  --asset-id asset-id \
  --property-id property-id
```

このオペレーションは、アセットプロパティの詳細を含むレスポンスを次の形式で返します。プロパティエイリアスは JSON オブジェクトの `assetProperty.alias` であり、この例では `myAlias` に設定されています。

```
{
  "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
  "assetName": "Wind Turbine 7",
  "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
  "assetProperty": {
    "alias": "myAlias",
    "id": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
    "name": "Wind Speed",
    "notification": {
      "topic": "$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/
assets/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE/properties/a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE",
      "state": "ENABLED"
    },
    "dataType": "DOUBLE",
    "unit": "m/s",
    "type": {
      "measurement": {}
    }
  }
}
```

属性値の更新

アセットは、属性のデフォルト値を含むアセットモデルの属性を継承します。場合によっては、アセットメーカーのプロパティなど、アセットモデルのデフォルト属性をそのままにしておくこともできます。また、アセットの緯度と経度など、継承された属性を更新することもできます。

Updating an attribute value (console)

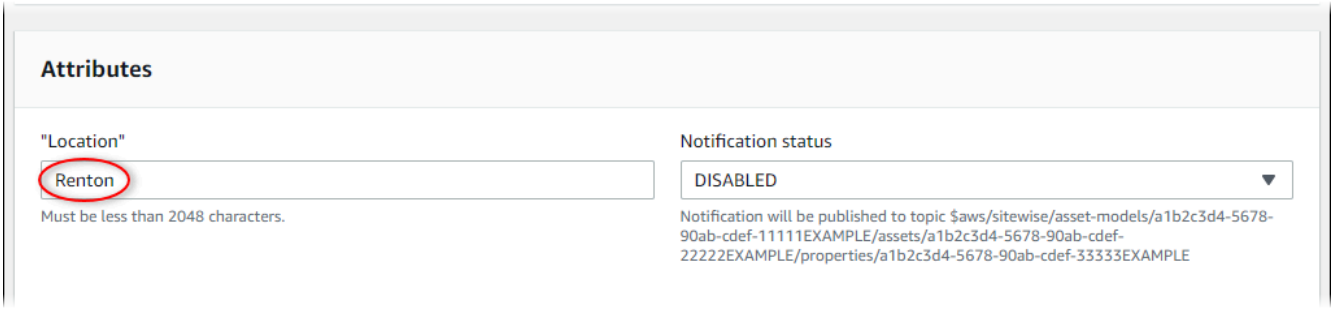
AWS IoT SiteWise コンソールを使用して、属性アセットプロパティの値を更新できます。

属性の値を更新するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. 属性を更新するアセットを選択します。

> **Tip**
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [編集] を選択します。
5. 更新する属性を検索し、新しい値を入力します。



6. [保存] を選択します。

Updating an attribute value (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して属性値を更新できます。

この手順を完了するには、アセットの `assetId` とプロパティの `propertyId` を知っている必要があります。外部 ID を使用することもできます。アセットを作成し、その がわからない場合は`assetId`、 ListAssets API を使用して特定のモデルのすべてのアセットを一覧表示します。DescribeAsset オペレーションを使用して、プロパティ IDs を含むアセットのプロパティを表示します。

BatchPutAssetProperty値 オペレーションを使用して、属性値をアセットに割り当てます。このオペレーションを使用すると、複数の属性を一度に設定できます。このオペレーションのペイ

ロードにはエントリのリストが含まれ、各エントリにはアセット ID、プロパティ ID、属性値が含まれます。

属性の値を更新するには (AWS CLI)

1. batch-put-payload.json という名前のファイルを作成して、次の JSON オブジェクトをファイルにコピーします。このペイロードの例は、風力タービンの緯度と経度を設定する方法を示しています。ID、値、タイムスタンプを更新して、ユースケースのペイロードを変更します。

```
{
  "entries": [
    {
      "entryId": "windfarm3-turbine7-latitude",
      "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "propertyId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
      "propertyValues": [
        {
          "value": {
            "doubleValue": 47.6204
          },
          "timestamp": {
            "timeInSeconds": 1575691200
          }
        }
      ]
    },
    {
      "entryId": "windfarm3-turbine7-longitude",
      "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "propertyId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE",
      "propertyValues": [
        {
          "value": {
            "doubleValue": 122.3491
          },
          "timestamp": {
            "timeInSeconds": 1575691200
          }
        }
      ]
    }
  ]
```

```
}
```

- ペイロードの各エントリには、一意の文字列として定義できる entryId が含まれています。リクエストのエントリが失敗した場合、各エラーには、対応するリクエストの entryId が含まれるため、再試行するリクエストを確認できます。
- 属性値を設定するには、各属性プロパティpropertyValuesの のリストに 1 つの timestamp-quality-value (TQV) 構造を含めることができます。この構造には、新しい value および現在の timestamp を含める必要があります。
  - value - 設定中のプロパティの型に応じて、次のいずれかのフィールドを含む構造。
    - booleanValue
    - doubleValue
    - integerValue
    - stringValue
  - timestamp – 現在の Unix エポック時間を秒単位で含む構造timeInSeconds。. AWS IoT SiteWise は、過去 7 日以上または今後 5 分より新しいタイムスタンプを持つデータポイントを拒否します。

BatchPutAssetProperty値 のペイロードを準備する方法の詳細については、「」を参照してくださいAPI を使用してデータを取り込む AWS IoT SiteWise。

2. 次のコマンドを実行して、属性値を に送信します AWS IoT SiteWise。

```
aws iotsitewise batch-put-asset-property-value -\-cli-input-json file://batch-put-payload.json
```

アセットの関連付けと関連付け解除

アセットのモデルで子アセットモデルの階層が定義されている場合は、子アセットをアセットに関連付けることができます。親アセットは、関連するアセットのデータにアクセスし、集計できます。階層アセットモデルの詳細については、「アセットモデル階層の定義」を参照してください。

トピック

- アセットの関連付けと関連付け解除 (コンソール)
- アセットの関連付けと関連付け解除 (AWS CLI)

アセットの関連付けと関連付け解除 (コンソール)

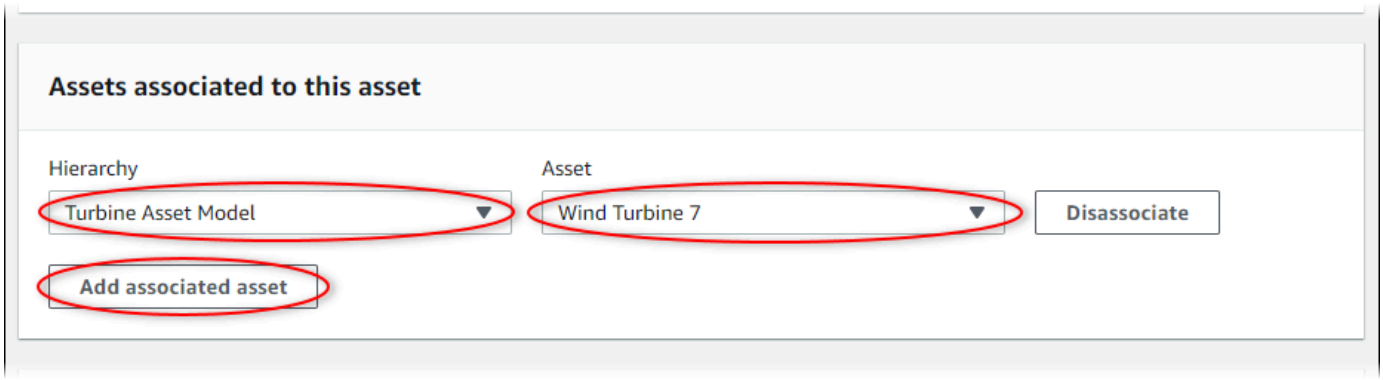
AWS IoT SiteWise コンソールを使用して、アセットの関連付けと関連付け解除を行うことができます。

アセットを関連付けるには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. 子アセットを関連付ける親アセットを選択します。

> **Tip**
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [編集] を選択します。
5. [このアセットに関連付けられているアセット] で、[関連付けられたアセットの追加] を選択します。



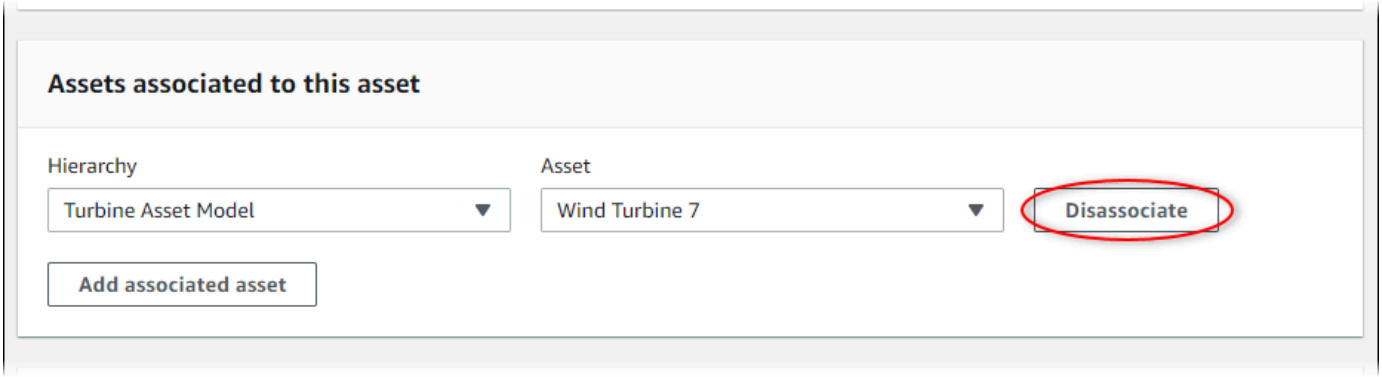
6. [階層] で、親アセットと子アセットの関係を定義する階層を選択します。
7. [アセット] で、関連付ける子アセットを選択します。
8. [保存] を選択します。

アセットの関連付けを解除するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. 子アセットの関連付けを解除する親アセットを選択します。

> **Tip**
>
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [編集] を選択します。
5. [このアセットに関連付けられているアセット] の [関連付けを削除] を選択します。



6. [保存] を選択します。

アセットの関連付けと関連付け解除 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して、アセットの関連付けと関連付け解除を行うことができます。

このステップでは、子アセットモデルとの関係を定義する親アセットモデル内の階層 (`hierarchyId`) の ID を知る必要があります。DescribeAsset オペレーションを使用して、レスポンスで階層 ID を検索します。

階層 ID を検索するには

- 次のコマンドを実行して、親アセットを記述します。*`parent-asset-id`* を親アセットの ID または外部 ID に置き換えます。

```
aws iotsitewise describe-asset --asset-id parent-asset-id
```

このオペレーションは、アセットの詳細を含むレスポンスを返します。レスポンスには、次の構造を持つ `assetHierarchies` リストが含まれます。

```
{
```

```
  ...
  "assetHierarchies": [
    {
      "id": "String",
      "name": "String"
    }
  ],
  ...
}
```

階層 ID は、アセット階層のリスト内の階層の id の値です。

階層 ID を取得したら、アセットをその階層に関連付けるか、関連付けを解除できます。

子アセットを親アセットに関連付けるには、 AssociateAssetsオペレーションを使用します。親アセットから子アセットの関連付けを解除するには、 DisassociateAssetsオペレーションを使用します。次のパラメータを指定します。これらのパラメータは、両方のオペレーションで同じです。

- assetId – 親アセットの ID または外部 ID。
- hierarchyId – 親アセットの階層 ID または外部 ID。
- childAssetId – 子アセットの ID または外部 ID。

アセットを関連付けるには (AWS CLI)

- 子アセットを親アセットに関連付けるには、次のコマンドを実行します。*parent-asset-id* 、 *hierarchy-id* 、および *child-asset-id* をそれぞれの IDs。

```
aws iotsitewise associate-assets \
  --asset-id parent-asset-id \
  --hierarchy-id hierarchy-id \
  --child-asset-id child-asset-id
```

アセットの関連付けを解除するには (AWS CLI)

- 子アセットを親アセットとの関連付けから解除するには、次のコマンドを実行します。*parent-asset-id* 、 *hierarchy-id* 、および *child-asset-id* をそれぞれの IDs。

```
aws iotsitewise disassociate-assets \
```

```
  --asset-id parent-asset-id \
  --hierarchy-id hierarchy-id \
  --child-asset-id child-asset-id
```

アセットとモデルの更新

でアセット、アセットモデル、コンポーネントモデルを更新 AWS IoT SiteWise して、名前と定義を変更できます。これらの更新オペレーションは非同期であり、 を介して伝達されるまでに時間がかかります AWS IoT SiteWise。追加の変更を行う前に、アセットまたはモデルのステータスを確認してください。更新されたアセットまたはモデルを引き続き使用するには、変更が反映されるまで待機する必要があります。

トピック

- アセットを更新する
- アセットモデルとコンポーネントモデルの更新
- カスタム複合モデル (コンポーネント) の更新

アセットを更新する

AWS IoT SiteWise コンソールまたは API を使用して、アセットの名前を更新できます。

アセットを更新すると、変更が反映されるまで、アセットの状態は UPDATING になります。詳細については、「アセットおよびモデルの状態」を参照してください。

トピック

- アセットの更新 (コンソール)
- アセットの更新 (AWS CLI)

アセットの更新 (コンソール)

AWS IoT SiteWise コンソールを使用してアセットの詳細を更新できます。

アセットを更新するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。

3. 更新するアセットを選択します。

> **Tip**
>
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [編集] を選択します。
5. アセットの [名前] を更新します。
6. (オプション) このページで、アセットのその他の情報を更新します。詳細については、次を参照してください。
   - アセットプロパティへの産業データストリームのマッピング
   - 属性値の更新
   - AWS 他のサービスとのやり取り
7. [保存] を選択します。

アセットの更新 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用してアセット名を更新できます。

UpdateAsset オペレーションを使用してアセットを更新します。以下のパラメータを指定します。

- `assetId` – アセットの ID。これは UUID 形式の実際の ID です。持っていexternalId:myExternalIdる場合は `assetId` です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。
- `assetName` - アセットの新しい名前。

アセットの名前を更新するには (AWS CLI)

- アセットの名前を更新するには、次のコマンドを実行します。*asset-id* をアセットの ID または外部 ID に置き換えます。*asset-name* をアセットの新しい名前で更新します。

```
aws iotsitewise update-asset \
  --asset-id asset-id \
  --asset-name asset-name
```

アセットモデルとコンポーネントモデルの更新

AWS IoT SiteWise コンソールまたは API を使用して、アセットモデルまたはコンポーネントモデルを更新できます。

既存のプロパティの型やデータ型、または既存のメトリクスのウィンドウを変更することはできません。また、モデルのタイプをアセットモデルからコンポーネントモデル、またはその逆に変更することもできません。

> **Important**
>
> - アセットモデルまたはコンポーネントモデルからプロパティを削除すると、 はそのプロパティの以前のデータをすべて AWS IoT SiteWise 削除します。コンポーネントモデルの場合、これはそのコンポーネントモデル を使用するすべてのアセットモデルに影響するため、変更がどの程度適用されるかを理解するように特に注意してください。
> - アセットモデルから階層定義を削除すると、 はその階層内のすべてのアセットの関連付け AWS IoT SiteWise を解除します。

アセットモデルを更新すると、基になるモデルに加えた変更がそのモデルに基づくすべてのアセットに反映されます。変更が反映されるまで、各アセットには UPDATING 状態になります。それらのアセットを操作する前に、それらが ACTIVE 状態に戻るまで待機する必要があります。この間、更新されたアセットモデルのステータスは PROPAGATING になります。

コンポーネントモデルを更新すると、そのコンポーネントモデルを組み込むすべてのアセットモデルに変更が反映されます。コンポーネントモデルの変更が伝達されるまで、影響を受ける各アセットモデルUPDATINGの状態は PROPAGATINGになり、その後、前の段落で説明したように、関連するアセットが更新されます。これらのアセットモデルが ACTIVE状態に戻るまで待ってから、操作する必要があります。この間、更新されたコンポーネントモデルのステータスは になりますPROPAGATING。

詳細については、「アセットおよびモデルの状態」を参照してください。

トピック

- アセットまたはコンポーネントモデルの更新 (コンソール)
- アセットまたはコンポーネントモデルの更新 (AWS CLI)

アセットまたはコンポーネントモデルの更新 (コンソール)

AWS IoT SiteWise コンソールを使用して、アセットモデルまたはコンポーネントモデルを更新できます。

アセットモデルまたはコンポーネントモデルを更新するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[モデル] を選択します。
3. 更新するアセットモデルまたはコンポーネントモデルを選択します。
4. [編集] を選択します。
5. [モデルの編集] ページで、次のいずれかの操作を行います。
   - [モデルの詳細] で、モデルの [名前] を変更します。
   - 任意の [属性定義] を変更します。既存の属性の [データ型] は変更できません。詳細については、「静的データ (属性) の定義」を参照してください。
   - [測定の定義] を変更します。既存の測定値の [データ型] は変更できません。詳細については、「機器からのデータストリームの定義 (測定値) 」を参照してください。
   - 任意の [定義を変換する] を変更します。詳細については、「データの変換 (変換) 」を参照してください。
   - [メトリクスの定義] を変更します。既存のメトリクスの [時間間隔] は変更できません。詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください。
   - (アセットモデルのみ) 階層定義 のいずれかを変更します。既存の階層の [階層モデル] を変更することはできません。詳細については、「アセットモデル階層の定義」を参照してください
6. 保存を選択します。

アセットまたはコンポーネントモデルの更新 (AWS CLI)

AWS Command Line Interface (AWS CLI) を使用して、アセットモデルまたはコンポーネントモデルを更新できます。

UpdateAssetModel API を使用して、アセットモデルまたはコンポーネントモデルの名前、説明、プロパティを更新します。アセットモデルのみ、階層を更新できます。以下のパラメータを指定します。

- `assetModelId` – アセットの ID。これは UUID 形式の実際の ID です。持っていexternalId:myExternalIdる場合は です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

ペイロードで更新されたモデルを指定します。アセットモデルまたはコンポーネントモデルの想定される形式については、「」を参照してくださいアセットモデルを作成する。

> **Warning**
>
> UpdateAssetModel API は、ペイロードで指定したモデルで既存のモデルを上書きします。モデルのプロパティや階層を削除しないようにするには、更新されたモデルペイロードにそれらの IDs と定義を含める必要があります。モデルの既存の構造をクエリする方法については、DescribeAsset「モデルオペレーション」を参照してください。

> **Note**
>
> 次の手順では、タイプ の複合モデルのみを更新できますAWS/ALARM。CUSTOM 複合モデルを更新する場合は、代わりに UpdateAssetModelCompositeModel を使用します。詳細については、「カスタム複合モデル (コンポーネント) の更新」を参照してください。

アセットモデルまたはコンポーネントモデルを更新するには (AWS CLI)

1. 既存のモデル定義を取得するには、次のコマンドを実行します。*asset-model-id* を更新するアセットモデルまたはコンポーネントモデルの ID または外部 ID に置き換えます。

   ```
   aws iotsitewise describe-asset-model --asset-model-id asset-model-id
   ```

   オペレーションは、モデルの詳細を含むレスポンスを返します。このレスポンスには以下の構造が含まれます。

   ```
   {
     "assetModelId": "String",
     "assetModelArn": "String",
     "assetModelName": "String",
     "assetModelDescription": "String",
     "assetModelProperties": Array of AssetModelProperty,
   ```

```
  "assetModelHierarchies": Array of AssetModelHierarchyDefinition,
  "assetModelCompositeModels": Array of AssetModelCompositeModel,
  "assetModelCompositeModelSummaries": Array of AssetModelCompositeModelSummary,
  "assetModelCreationDate": "String",
  "assetModelLastUpdateDate": "String",
  "assetModelStatus": {
    "state": "String",
    "error": {
      "code": "String",
      "message": "String"
    },
  "assetModelType": "String"
  }
}
```

詳細については、DescribeAsset「モデルオペレーション」を参照してください。

2. update-asset-model.json という名前のファイルを作成し、前のコマンドのレスポンスをファイルにコピーします。
3. update-asset-model.json の JSON オブジェクトから次のキーと値のペアを削除します。

   - assetModelId
   - assetModelArn
   - assetModelCompositeModelSummaries
   - assetModelCreationDate
   - assetModelLastUpdateDate
   - assetModelStatus
   - assetModelType

   UpdateAssetModel オペレーションでは、次の構造を持つペイロードが必要です。

```
{
  "assetModelName": "String",
  "assetModelDescription": "String",
  "assetModelProperties": Array of AssetModelProperty,
  "assetModelHierarchies": Array of AssetModelHierarchyDefinition,
  "assetModelCompositeModels": Array of AssetModelCompositeModel
}
```

4. update-asset-model.json で、次のいずれかを行ってください。
   - アセットモデルの名前を変更します (assetModelName)。
   - アセットモデルの説明を変更、追加、または削除します (assetModelDescription)。
   - アセットモデルのプロパティを変更、追加、または削除します (assetModelProperties)。既存のプロパティの dataType、または既存のメトリクスの window を変更することはできません。詳細については、「データのプロパティを定義する。」を参照してください。
   - アセットモデルの階層を変更、追加、または削除します (assetModelHierarchies)。既存の階層の childAssetModelId は変更できません。詳細については、「アセットモデル階層の定義」を参照してください。
   - アセットモデルの複合モデルのタイプ AWS/ALARM () を変更、追加、または削除しますassetModelCompositeModels。アラームは、他のプロパティをモニタリングし、機器やプロセスに注意が必要な時期を特定することができます。各アラーム定義は、アラームが使用する一連のプロパティを標準化した複合モデルです。詳細については、「アラームによるデータのモニタリング。」および「アセットモデルにおけるアラームの定義」を参照してください。
5. 次のコマンドを実行して、update-asset-model.json に保存されている定義でアセットモデルを更新します。*asset-model-id* をアセットモデルの ID に置き換えます。

```
aws iotsitewise update-asset-model \
  --asset-model-id asset-model-id \
  --cli-input-json file://model-payload.json
```

カスタム複合モデル (コンポーネント) の更新

AWS IoT SiteWise API を使用してカスタム複合モデルを更新するか、 AWS IoT SiteWise コンソールを使用してコンポーネントを更新できます。

トピック

- コンポーネントの更新 (コンソール)
- カスタム複合モデルの更新 (AWS CLI)

コンポーネントの更新 (コンソール)

AWS IoT SiteWise コンソールを使用してコンポーネントを更新できます。

コンポーネントを更新するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[モデル] を選択します。
3. コンポーネントがあるアセットモデルを選択します。
4. プロパティ タブで、コンポーネント を選択します。
5. 更新するコンポーネントを選択します。
6. [編集] を選択します。
7. 「コンポーネントの編集」ページで、次のいずれかの操作を行います。
   - [モデルの詳細] で、モデルの [名前] を変更します。
   - 任意の [属性定義] を変更します。既存の属性の [データ型] は変更できません。詳細については、「静的データ (属性) の定義」を参照してください。
   - [測定の定義] を変更します。既存の測定値の [データ型] は変更できません。詳細については、「機器からのデータストリームの定義 (測定値) 」を参照してください。
   - 任意の [定義を変換する] を変更します。詳細については、「データの変換 (変換) 」を参照してください。
   - [メトリクスの定義] を変更します。既存のメトリクスの [時間間隔] は変更できません。詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください
8. 保存を選択します。

カスタム複合モデルの更新 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して、カスタム複合モデルを更新できます。

名前または説明を更新するには、UpdateAssetModelCompositeモデルオペレーションを使用します。インラインカスタム複合モデルのみ、プロパティを更新することもできます。 component-model-based カスタム複合モデルのプロパティは、参照されるコンポーネントモデルが関連するプロパティを提供するため、更新できません。

> **Important**
>
> カスタム複合モデルからプロパティを削除すると、 はそのプロパティの以前のデータをすべて AWS IoT SiteWise 削除します。既存のプロパティの型またはデータ型を変更することはできません。

既存の複合モデルプロパティを同じ を持つ新しいモデルプロパティに置き換えるには`name`、次の手順を実行します。

1. 既存のプロパティ全体を削除して`UpdateAssetModelCompositeModel`リクエストを送信します。
2. 新しいプロパティを含む 2 番目の`UpdateAssetModelCompositeModel`リクエストを送信します。新しいアセットプロパティは、前のアセットプロパティ`name`と同じ を持ち AWS IoT SiteWise 、新しい一意の を生成します`id`。

カスタム複合モデルを更新するには (AWS CLI)

1. 既存の複合モデル定義を取得するには、次のコマンドを実行します。*`composite-model-id`* を更新するカスタム複合モデルの ID または外部 ID に置き換え、*`setal-model-id`* をカスタム複合モデルが関連付けられているアセットモデルに置き換えます。詳細については、「 ユーザーガイド」のAWS IoT SiteWise 「」を参照してください。

   ```
   aws iotsitewise describe-asset-model-composite-model \
   --asset-model-composite-model-id composite-model-id \
   --asset-model-id asset-model-id
   ```

   詳細については、DescribeAssetModelComposite「モデルオペレーション」を参照してください。
2. というファイルを作成し`update-custom-composite-model.json`、前のコマンドのレスポンスを ファイルにコピーします。
3. 以下のフィールド`update-custom-composite-model.json`を除き、 の JSON オブジェクトからすべてのキーと値のペアを削除します。

   - `assetModelCompositeModelName`
   - `assetModelCompositeModelDescription` (存在する場合)
   - `assetModelCompositeModelProperties` (存在する場合)
4. `update-custom-composite-model.json` で、次のいずれかを行ってください。

   - の値を変更します`assetModelCompositeModelName`。
   - を追加または削除するか`assetModelCompositeModelDescription`、その値を変更します。

- インラインカスタム複合モデルのみ: のアセットモデルのプロパティを変更、追加、または削除しますassetModelCompositeModelProperties。

このファイルに必要な形式の詳細については、UpdateAssetModelComposite「モデル のリクエスト構文」を参照してください。

5. 次のコマンドを実行して、 に保存されている定義でカスタム複合モデルを更新しますupdate-custom-composite-model.json。*composite-model-id* を複合モデルの ID に、 *asset-model-id* をその中のアセットモデルの ID に置き換えます。

```
aws iotsitewise update-asset-model-composite-model \
--asset-model-composite-model-id composite-model-id \
--asset-model-id asset-model-id \
--cli-input-json file://update-custom-composite-model.json
```

アセットとモデルの削除

アセットとモデルは、使用が終了した AWS IoT SiteWise ら から削除できます。削除オペレーションは非同期であり、 を介して伝達されるまでに時間がかかります AWS IoT SiteWise。

トピック

- アセットの削除
- アセットモデルの削除

アセットの削除

AWS IoT SiteWise コンソールまたは API を使用してアセットを削除できます。

アセットを削除する前に、まずその子アセットの関連付けを解除して、親アセットとの関連付けを解除する必要があります。詳細については、「アセットの関連付けと関連付け解除」を参照してください。 AWS Command Line Interface （ AWS CLI) を使用する場合は、ListAssociatedアセットオペレーションを使用してアセットの子を一覧表示できます。

アセットを削除すると、変更が反映されるまで、アセットのステータスは DELETING になります。詳細については、「アセットおよびモデルの状態」を参照してください。アセットを削除した後は、そのアセットに対してクエリを実行することはできません。使用すると、API は HTTP 404 レスポンスを返します。

> **Important**
>
> AWS IoT SiteWise は、削除されたアセットのすべてのプロパティデータを削除します。

トピック

- アセットの削除 (コンソール)
- アセットの削除 (AWS CLI)

アセットの削除 (コンソール)

AWS IoT SiteWise コンソールを使用してアセットを削除できます。

アセットを削除するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. 削除するアセットを選択します。

   > **Tip**
   >
   > 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. アセットに [関連付けられたアセット] がある場合は、各アセットを削除します。アセットの名前を選択して、そのページに移動して、アセットを削除できます。
5. アセットのページで、[削除] を選択します。
6. 「アセットの削除」ダイアログボックスで、次の操作を行います。
   a. 削除を確定するには、**`Delete`** と入力します。
   b. [削除] を選択します。

アセットの削除 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用してアセットを削除できます。

DeleteAsset オペレーションを使用してアセットを削除します。以下のパラメータを指定します。

- `assetId` – アセットの ID。これは UUID 形式の実際の ID です。持っていて`externalId:myExternalId`る場合は です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

アセットを削除するには (AWS CLI)

1. 次のコマンドを実行して、アセットの階層を一覧表示します。*asset-id* をアセットの ID または外部 ID に置き換えます。

   ```
   aws iotsitewise describe-asset --asset-id asset-id
   ```

   このオペレーションは、アセットの詳細を含むレスポンスを返します。レスポンスには、次の構造を持つ `assetHierarchies` リストが含まれます。

   ```
   {
     ...
     "assetHierarchies": [
       {
         "id": "String",
         "name": "String"
       }
     ],
     ...
   }
   ```

   詳細については、「 DescribeAssetオペレーション」を参照してください。

2. 階層ごとに次のコマンドを実行して、その階層に関連付けられているアセットの子を一覧表示します。*asset-id* をアセットの ID または外部 ID に置き換え、*hierity-id* を階層の ID または外部 ID に置き換えます。

   ```
   aws iotsitewise list-associated-assets \
     --asset-id asset-id \
     --hierarchy-id hierarchy-id
   ```

   詳細については、ListAssociated「アセットオペレーション」を参照してください。

3. 次のコマンドを実行して、関連する各アセットを削除してから、アセットを削除します。*asset-id* をアセットの ID または外部 ID に置き換えます。

```
aws iotsitewise delete-asset --asset-id asset-id
```

アセットモデルの削除

AWS IoT SiteWise コンソールまたは API を使用して、アセットモデルを削除できます。

アセットモデルを削除する前に、アセットモデルから作成されたすべてのアセットを削除する必要があります。

アセットモデルを削除すると、変更が反映されるまで、アセットのステータスは DELETING になります。詳細については、「アセットおよびモデルの状態」を参照してください。アセットモデルを削除した後は、そのアセットモデルをクエリすることはできません。使用すると、API は HTTP 404 レスポンスを返します。

トピック

- アセットモデルの削除 (コンソール)
- アセットモデルの削除 (AWS CLI)

アセットモデルの削除 (コンソール)

AWS IoT SiteWise コンソールを使用してアセットモデルを削除できます。

アセットモデルを削除するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Models (モデル)] を選択します。
3. 削除するアセットモデルを選択します。
4. モデルに [アセット] がある場合は、各アセットを削除します。アセットの名前を選択してそのページに移動し、そこでアセットを削除できます。詳細については、「アセットの削除 (コンソール) 」を参照してください。
5. モデルのページで、[削除] を選択します。
6. 「モデルの削除」ダイアログボックスで、次の操作を行います。
   a. 削除を確定するには、**Delete** と入力します。
   b. [削除] を選択します。

アセットモデルの削除 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用してアセットモデルを削除できます。

DeleteAssetモデルオペレーションを使用して、アセットモデルを削除します。以下のパラメータを指定します。

- `assetModelId` – アセットの ID。これは UUID 形式の実際の ID です。持っていexternalId:myExternalIdる場合は です。詳細については、AWS IoT SiteWise ユーザーガイドの 外部 IDs を持つオブジェクトの参照を参照してください。

アセットモデルを削除するには (AWS CLI)

1. 次のコマンドを実行して、モデルから作成されたすべてのアセットを一覧表示します。*asset-model-id* をアセットモデルの ID または外部 ID に置き換えます。

   ```
   aws iotsitewise list-assets --asset-model-id asset-model-id
   ```

   詳細については、「 ListAssetsオペレーション」を参照してください。
2. 前のコマンドでモデルからアセットが返された場合は、各アセットを削除します。詳細については、「アセットの削除 (AWS CLI) 」を参照してください。
3. 次のコマンドを実行してアセットモデルを削除します。*asset-model-id* をアセットモデルの ID または外部 ID に置き換えます。

   ```
   aws iotsitewise delete-asset-model --asset-model-id asset-model-id
   ```

アセットとモデルによる一括オペレーション

多数のアセットまたはアセットモデルを使用するには、一括操作を使用してリソースを別の場所に一括インポートおよびエクスポートします。例えば、Amazon S3 バケット内のアセットまたはアセットモデルを定義するデータファイルを作成し、一括インポートを使用して で作成または更新できます AWS IoT SiteWise。または、 に多数のアセットまたはアセットモデルがある場合は AWS IoT SiteWise、それらを Amazon S3 にエクスポートできます。

> **Note**
>
> AWS IoT TwinMaker API で オペレーションを呼び出す AWS IoT SiteWise ことで、 で一括オペレーションを実行します。これは、 AWS IoT TwinMaker ワークスペースを設定 AWS IoT TwinMaker または作成しなくても実行できます。必要なのは、 AWS IoT SiteWise コンテンツを配置できる Amazon S3 バケットだけです。

トピック

主要な概念と用語

AWS IoT SiteWise 一括インポートおよびエクスポート機能は、以下の概念と用語に依存しています。

- インポート: Amazon S3 バケット内のファイルから にアセットまたはアセットモデルを移動するアクション AWS IoT SiteWise。
- エクスポート: アセットまたはアセットモデルを から AWS IoT SiteWise Amazon S3 バケットに移動するアクション。
- ソース : コンテンツの移動元の の開始場所。

  例えば、Amazon S3 バケットはインポートソースであり、 はエクスポートソース AWS IoT SiteWise です。
- 送信先 : コンテンツの移動先となる の希望の場所。

例えば、Amazon S3 バケットはエクスポート先であり、 はインポート先 AWS IoT SiteWise です。

- AWS IoT SiteWise スキーマ: このスキーマは、 からメタデータをインポートおよびエクスポートするために使用されます AWS IoT SiteWise。
- 最上位リソース： アセットやアセットモデルなど、個別に作成または更新できる AWS IoT SiteWise リソース。
- サブリソース： 最上位 AWS IoT SiteWise リソース内のネストされたリソース。例としては、プロパティ、階層、複合モデルなどがあります。
- メタデータ: リソースを正常にインポートまたはエクスポートするために必要なキー情報。メタデータの例としては、アセットとアセットモデルの定義があります。
- メタデータ：TransferJob の実行時に作成されたオブジェクト`CreateMetadataTransferJob`。

サポートされている機能

このトピックでは、一括オペレーションを実行するときに実行できる操作について説明します。一括オペレーションは、次の機能をサポートしています。

- 最上位のリソース作成： ID を定義しない、または ID が既存のリソースの ID と一致しないアセットまたはアセットモデルをインポートすると、新しいリソースとして作成されます。
- 最上位のリソース置換： ID が既に存在するものと一致するアセットまたはアセットモデルをインポートすると、既存のリソースが置き換えられます。
- サブリソースの作成、置換、削除： インポートがアセットやアセットモデルなどの最上位のリソースを置き換えると、プロパティ、階層、複合モデルなどのすべてのサブリソースが新しい定義に置き換えられます。

  例えば、一括インポート中にアセットモデルを更新し、更新されたバージョンで元の に存在しなかったプロパティを定義すると、新しいプロパティが作成されます。既に存在するプロパティを定義すると、既存のプロパティが更新されます。更新されたアセットモデルが元の に存在するプロパティを省略すると、 プロパティは削除されます。
- 最上位リソースの削除なし： 一括オペレーションでは、アセットまたはアセットモデルは削除されません。一括オペレーションは、作成または更新のみを行います。

一括オペレーションの前提条件

このセクションでは、 AWS のサービス とローカルマシンの間でリソースを交換するための AWS Identity and Access Management (IAM) アクセス許可など、一括操作の前提条件について説明します。一括オペレーションを開始する前に、次の前提条件を満たしていることを確認してください。

- リソースを保存する Amazon S3 バケットを作成します。Amazon S3 の使用の詳細については、Amazon S3とは」を参照してください。

IAM アクセス許可

一括操作を実行するには、Amazon S3 AWS IoT SiteWiseとローカルマシン間の AWS リソースの交換を許可するアクセス許可を持つ AWS Identity and Access Management (IAM) ポリシーを作成する必要があります。 Amazon S3 IAM ポリシーの作成の詳細については、「IAM ポリシーの作成」を参照してください。

一括操作を実行するには、次のポリシーが必要です。

AWS IoT SiteWise ポリシー

このポリシーは、一括オペレーションに必要な AWS IoT SiteWise API アクションへのアクセスを許可します。

```
{
    "Sid": "SiteWiseApiAccess",
    "Effect": "Allow",
    "Action": [
        "iotsitewise:CreateAsset",
        "iotsitewise:CreateAssetModel",
        "iotsitewise:UpdateAsset",
        "iotsitewise:UpdateAssetModel",
        "iotsitewise:UpdateAssetProperty",
        "iotsitewise:ListAssets",
        "iotsitewise:ListAssetModels",
        "iotsitewise:ListAssetProperties",
        "iotsitewise:ListAssetModelProperties",
        "iotsitewise:ListAssociatedAssets",
        "iotsitewise:DescribeAsset",
        "iotsitewise:DescribeAssetModel",
        "iotsitewise:DescribeAssetProperty",
        "iotsitewise:AssociateAssets",
```

```
        "iotsitewise:DisassociateAssets",
        "iotsitewise:AssociateTimeSeriesToAssetProperty",
        "iotsitewise:DisassociateTimeSeriesFromAssetProperty",
        "iotsitewise:BatchPutAssetPropertyValue",
        "iotsitewise:BatchGetAssetPropertyValue",
        "iotsitewise:TagResource",
        "iotsitewise:UntagResource",
        "iotsitewise:ListTagsForResource",
        "iotsitewise:CreateAssetModelCompositeModel",
        "iotsitewise:UpdateAssetModelCompositeModel",
        "iotsitewise:DescribeAssetModelCompositeModel",
        "iotsitewise:DeleteAssetModelCompositeModel",
        "iotsitewise:ListAssetModelCompositeModels",
        "iotsitewise:ListCompositionRelationships",
        "iotsitewise:DescribeAssetCompositeModel"
    ],
    "Resource": "*"
}
```

AWS IoT TwinMaker ポリシー

このポリシーは、一括オペレーションの操作に使用する AWS IoT TwinMaker API オペレーションへのアクセスを許可します。

```
{
    "Sid": "MetadataTransferJobApiAccess",
    "Effect": "Allow",
    "Action": [
        "iottwinmaker:CreateMetadataTransferJob",
        "iottwinmaker:CancelMetadataTransferJob",
        "iottwinmaker:GetMetadataTransferJob",
        "iottwinmaker:ListMetadataTransferJobs"
    ],
    "Resource": "*"
}
```

Amazon S3 ポリシー

このポリシーは、一括オペレーションのメタデータを転送するための Amazon S3 バケットへのアクセスを提供します。

For a specific Amazon S3 bucket

一括オペレーションメタデータの操作に特定のバケットを使用する場合、このポリシーはそのバケットへのアクセスを提供します。

```
{
    "Effect": "Allow",
    "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:AbortMultipartUpload",
        "s3:ListBucketMultipartUploads",
        "s3:ListMultipartUploadParts"
    ],
    "Resource": [
        "arn:aws:s3:::bucket name",
        "arn:aws:s3:::bucket name/*"
    ]
}
```

To allow any Amazon S3 bucket

バルクオペレーションメタデータを操作するためにさまざまなバケットを使用する場合、このポリシーは任意のバケットへのアクセスを提供します。

```
{
    "Effect": "Allow",
    "Action": [
        "s3:PutObject",
        "s3:GetObject",
        "s3:GetBucketLocation",
        "s3:ListBucket",
        "s3:AbortMultipartUpload",
        "s3:ListBucketMultipartUploads",
        "s3:ListMultipartUploadParts"
    ],
    "Resource": "*"
}
```

インポートおよびエクスポートオペレーションのトラブルシューティングについては、「」を参照してください一括インポートとエクスポートのトラブルシューティング。

一括インポートジョブの実行

一括インポートは、メタデータを AWS IoT SiteWise ワークスペースに移動するアクションです。例えば、一括インポートでは、メタデータをローカルファイルまたは Amazon S3 バケット内のファイルから AWS IoT SiteWise ワークスペースに移動できます。

ステップ 1: インポートするファイルを準備する

AWS IoT SiteWise ネイティブ形式ファイルをダウンロードして、アセットとアセットモデルをインポートします。詳細については、「AWS IoT SiteWise メタデータ転送ジョブスキーマ」を参照してください。

ステップ 2: 準備したファイルを Amazon S3 にアップロードする

このファイルを Amazon S3 にアップロードします。詳細については、Amazon Simple Storage Service Amazon S3 へのファイルのアップロード」を参照してください。

メタデータのインポート (コンソール)

を使用してメタデータ AWS IoT SiteWise コンソール を一括インポートできます。ステップ 1: インポートするファイルを準備する と ステップ 2: 準備したファイルを Amazon S3 にアップロードするに従って、インポートする準備ができたファイルを準備します。

Amazon S3 から にデータをインポートする AWS IoT SiteWise コンソール

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインから一括オペレーション新規を選択します。
3. 新しいインポートを選択してインポートプロセスを開始します。
4. メタデータのインポートページで、次の操作を行います。
   - Amazon S3 を参照 を選択して、Amazon S3 バケットとファイルを表示します。
   - 準備済みのインポートファイルを含む Amazon S3 バケットに移動します。
   - インポートするファイルを選択します。
   - 選択したファイルを確認し、インポート を選択します。
5. の SiteWise メタデータページの一括オペレーションでは、新しく作成されたインポートジョブがジョブの進行状況テーブル AWS IoT SiteWise コンソール に表示されます。

メタデータのインポート (AWS CLI)

インポートアクションを実行するには、次の手順を使用します。

Amazon S3 から にデータをインポートする AWS CLI

1. に続いて、インポートするリソースを指定するメタデータファイルを作成しますAWS IoT SiteWise メタデータ転送ジョブスキーマ。このファイルを Amazon S3 バケットに保存します。

   インポートするメタデータファイルの例については、「」を参照してくださいメタデータのインポートの例。
2. 次に、リクエスト本文を含む JSON ファイルを作成します。リクエストボディは、転送ジョブの送信元と送信先を指定します。このファイルは、前のステップの ファイルとは別のものです。必ず Amazon S3 バケットをソースとして指定し、送信先`iotsitewise`として指定してください。

   次の例は、リクエストボディを示しています。

   ```
   {
         "metadataTransferJobId": "your-transfer-job-Id",
         "sources": [{
             "type": "s3",
             "s3Configuration": {
                 "location": "arn:aws:s3:::your-S3-bucket-name/
   your_import_metadata.json"
             }
         }],
         "destination": {
             "type": "iotsitewise"
         }
     }
   ```
3. 次の AWS CLI コマンド`CreateMetadataTransferJob`を実行して を呼び出します。この例では、前のステップのリクエスト本文ファイルの名前は ですcreateMetadataTransferJobExport.json。

   ```
   aws iottwinmaker create-metadata-transfer-job --region us-east-1 \
     --cli-input-json file://createMetadataTransferJobImport.json
   ```

   これにより、メタデータ転送ジョブが作成され、選択したリソースの転送プロセスが開始されます。

一括エクスポートジョブの実行

一括エクスポートは、 AWS IoT SiteWise ワークスペースから Amazon S3 バケットにメタデータを移動するアクションです。

Amazon S3 への AWS IoT SiteWise コンテンツの一括エクスポートを実行するときに、エクスポートする特定のアセットモデルとアセットを制限するフィルターを指定できます。

フィルターは、JSON リクエストのソース`iotSiteWiseConfiguration`セクション内の セクションで指定する必要があります。

> **Note**
>
> リクエストには複数のフィルターを含めることができます。一括オペレーションでは、いずれかのフィルターに一致するアセットモデルとアセットがエクスポートされます。
> フィルターを指定しない場合、一括オペレーションはすべてのアセットモデルとアセットをエクスポートします。

Example フィルター付きのリクエスト本文

```
{
      "metadataTransferJobId": "your-transfer-job-id",
      "sources": [
       {
        "type": "iotsitewise",
        "iotSiteWiseConfiguration": {
          "filters": [
           {
              "filterByAssetModel": {
                  "assetModelId": "asset model ID"
              }
           },
           {
             "filterByAssetModel": {
                 "assetModelId": "asset model ID",
                 "includeAssets": true
             }
           },
           {
             "filterByAssetModel": {
```

```
                "assetModelId": "asset model ID",
                "includeOffspring": true
              }
            }
          ]
        }
      }
    ],
    "destination": {
      "type": "s3",
      "s3Configuration": {
        "location": "arn:aws:s3:::your-S3-bucket-location"
      }
    }
}
```

メタデータのエクスポート (コンソール)

次の手順では、コンソールのエクスポートアクションについて説明します。

でエクスポートジョブを作成する AWS IoT SiteWise コンソール

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインから一括オペレーション新規を選択します。
3. 新しいエクスポートを選択してエクスポートプロセスを開始します。
4. メタデータのエクスポートページで、次の操作を行います。

   - エクスポートジョブの名前を入力します。これは、Amazon S3 バケット内のエクスポートされたファイルに使用される名前です。
   - エクスポートするリソースを選択すると、ジョブのフィルターが設定されます。
     - すべてのアセットとアセットモデルをエクスポートします。アセットとアセットモデルにフィルターを使用します。
     - アセットをエクスポートします。アセットをフィルタリングします。
       - エクスポートフィルターに使用するアセットを選択します。
       - (オプション) 子または関連するアセットモデルを追加します。
     - アセットモデルをエクスポートします。アセットモデルでフィルタリングします。
       - エクスポートフィルターに使用するアセットモデルを選択します。
       - (オプション) 子孫、関連するアセット、またはその両方を追加します。

    - [次へ] をクリックします。
  - Amazon S3 バケットに移動します。
    - Amazon S3 を参照 を選択して、Amazon S3 バケットとファイルを表示します。
    - ファイルを配置する必要がある Amazon S3 バケットに移動します。
    - [次へ] をクリックします。
  - エクスポートジョブを確認し、エクスポート を選択します。
5. の SiteWise メタデータページの一括オペレーションでは、新しく作成されたインポートジョブがジョブの進行状況テーブル AWS IoT SiteWise コンソール に表示されます。

メタデータのエクスポート時にフィルターを使用するさまざまな方法については、「」を参照してください[メタデータのエクスポート例](#)。

メタデータのエクスポート (AWS CLI)

次の手順では、 AWS CLI エクスポートアクションについて説明します。

から Amazon S3 AWS IoT SiteWise にデータをエクスポートする

1. リクエストボディを使用して JSON ファイルを作成します。リクエストボディは、転送ジョブの送信元と送信先を指定します。次の例は、リクエスト本文の例を示しています。

```
{
    "metadataTransferJobId": "your-transfer-job-Id",
    "sources": [{
        "type": "iotsitewise"
    }],
    "destination": {
        "type": "s3",
        "s3Configuration": {
            "location": "arn:aws:s3:::your-S3-bucket-location"
        }
    }
}
```

   メタデータ転送ジョブの送信先として Amazon S3 バケットを必ず指定してください。

> **Note**
>
> この例では、すべてのアセットモデルとアセットをエクスポートします。エクスポートを特定のアセットモデルまたはアセットに制限するには、リクエスト本文にフィルターを含めることができます。エクスポートフィルターの適用の詳細については、「」を参照してくださいメタデータのエクスポート例。

2. 次のステップで使用するリクエスト本文ファイルを保存します。この例では、ファイル名は `createMetadataTransferJobExport.json` です。
3. 次の AWS CLI コマンド`CreateMetadataTransferJob`を実行して を呼び出します。

```
aws iottwinmaker create-metadata-transfer-job --region us-east-1 \
         --cli-input-json file://createMetadataTransferJobExport.json
```

入力 JSON ファイルを独自の転送ファイル名`createMetadataTransferJobExport.json`に置き換えます。

ジョブの進行状況の追跡とエラー処理

一括処理ジョブの処理には時間がかかります。各ジョブは、リクエスト AWS IoT SiteWise を受け取る順序で処理されます。アカウント one-at-a-time ごとに処理されます。ジョブが完了すると、キュー内の次の は自動的に処理を開始します。 はジョブを非同期的に AWS IoT SiteWise 解決し、進行するたびに各 のステータスを更新します。各ジョブには、リソースの状態と、該当する場合はエラーメッセージを含むステータスフィールドがあります。

ステータスは以下のいずれかの値になります。

- `VALIDATING` - 送信されたファイル形式とその内容を含むジョブを検証します。
- `PENDING` – ジョブはキューにあります。この状態のジョブはコンソール AWS IoT SiteWise からキャンセルできますが、他のすべての状態は最後まで継続されます。
- `RUNNING` – ジョブの処理。インポートファイルで定義されているリソースを作成および更新しているか、選択したエクスポートジョブフィルターに基づいてリソースをエクスポートしています。キャンセルされた場合、このジョブによってインポートされたリソースは削除されません。詳細については、「ジョブの進行状況と詳細を確認する (コンソール)」を参照してください。
- `CANCELLING` – ジョブはアクティブにキャンセルされています。

- ERROR – 1 つ以上のリソースの処理に失敗しました。詳細については、詳細なジョブレポートを確認してください。詳細については、「エラーの詳細を確認する (コンソール)」を参照してください。
- COMPLETED – ジョブはエラーなしで完了しました。
- CANCELLED – ジョブはキャンセルされ、キューに入れられていません。RUNNING ジョブをキャンセルした場合、キャンセル時にこのジョブによって既にインポートされたリソースは から削除されません AWS IoT SiteWise。

トピック

- ジョブの進行状況の追跡
- エラーの検査

ジョブの進行状況の追跡

ジョブの進行状況と詳細を確認する (コンソール)

一括ジョブを開始するには、メタデータのインポート (コンソール)「」またはメタデータのエクスポート (コンソール)「」を参照してください。

AWS IoT SiteWise コンソールでのジョブの進行状況の概要：

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインから一括オペレーション新規を選択します。
3. AWS IoT SiteWise コンソールのジョブの進行状況テーブルに、一括オペレーションジョブのリストが表示されます。
4. ジョブタイプ列は、エクスポートジョブかインポートジョブかを示します。インポートされた日付列には、ジョブが開始された日付が表示されます。
5. ステータス列には、ジョブのステータスが表示されます。ジョブを選択すると、そのジョブの詳細を表示できます。
6. 選択したジョブは成功すると成功、ジョブが失敗した場合は失敗のリストを表示します。エラーの説明は、リソースタイプごとにも表示されます。

AWS IoT SiteWise コンソールでのジョブの詳細の概要：

AWS IoT SiteWise コンソールのジョブの進行状況テーブルに、一括オペレーションジョブのリストが表示されます。

1. ジョブを選択すると、詳細が表示されます。
2. インポートジョブの場合、 はインポートファイルの Amazon S3 の場所Data source ARNを表します。
3. エクスポートジョブの場合、 はエクスポート後のファイルの Amazon S3 の場所Data destination ARNを表します。
4. Status と はStatus reason、現在のジョブに関する追加の詳細を提供します。詳細については、「ジョブの進行状況の追跡とエラー処理」を参照してください。
5. は、プロセスキュー内のジョブの位置Queued positionを表します。ジョブは一度に 1 つずつ処理されます。キューに入れられた位置が 1 の場合、ジョブが次に処理されることを示します。
6. ジョブの詳細ページには、ジョブの進行状況の数も表示されます。
   - ジョブの進行状況カウントタイプは次のとおりです。
     i. Total resources – 転送プロセス内のアセットの合計数を示します。
     ii. Succeeded – プロセス中に正常に転送されたアセットの数を示します。
     iii. Failed – プロセス中に失敗したアセットの数を示します。
     iv. Skipped – プロセス中にスキップされたアセットの数を示します。
7. ジョブのステータスが PENDINGまたは の場合VALIDATING、すべてのジョブの進行状況が としてカウントされます–。これは、ジョブの進行状況カウントが評価されていることを示します。
8. のジョブステータスには、処理のために送信されたジョブTotal resourcesの数RUNNINGが表示されます。詳細なカウント (Succeeded、Failed、および Skipped) は、処理されたリソースに適用されます。詳細なカウントの合計は、ジョブのステータスが COMPLETEDまたは になるまでTotal resourcesカウントよりも小さくなりますERROR。
9. ジョブのステータスが COMPLETEDまたは の場合ERROR、Total resourcesカウントは詳細カウント (Succeeded、、) の合計に等しくなりますFailedSkipped。
10. ジョブのステータスが の場合ERROR、ジョブ失敗テーブルで特定のエラーと失敗の詳細を確認します。詳細については、「エラーの詳細を確認する (コンソール)」を参照してください。

ジョブの進行状況と詳細を確認する (AWS CLI)

一括オペレーションを開始した後、次の API アクションを使用してステータスを確認または更新できます。

- 特定のジョブに関する情報を取得するには、 GetMetadataTransferJob API アクションを使用します。

  GetMetadataTransferJob API を使用して情報を取得します。

  1. 転送ジョブを作成して実行します。GetMetadataTransferJob API を呼び出します。

     Example AWS CLI コマンド：

     ```
     aws iottwinmaker get-metadata-transfer-job \
             --metadata-transfer-job-id your_metadata_transfer_job_id \
             --region your_region
     ```

  2. GetMetadataTransferJob API は、次のパラメータを持つMetadataTransferJobProgressオブジェクトを返します。
     - succeededCount – プロセスで正常に転送されたアセットの数を示します。
     - failedCount – プロセス中に失敗したアセットの数を示します。
     - skippedCount – プロセス中にスキップされたアセットの数を示します。
     - totalCount – 転送プロセスにおけるアセットの合計数を示します。

     これらのパラメータは、ジョブの進行状況を示します。ステータスが の場合RUNNING、まだ処理されていないリソースの数を追跡するのに役立ちます。

     スキーマ検証エラーが発生した場合、または failedCount が 1 以上の場合、ジョブの進行状況は になりますERROR。ジョブの完全なエラーレポートは、Amazon S3 バケットに配置されます。詳細については、「エラーの検査」を参照してください。

- 現在のジョブを一覧表示するには、 ListMetadataTransferJobs API アクションを使用します。

  JSON ファイルを使用して、返されたジョブを現在の状態に基づいてフィルタリングします。次の手順を参照してください。

  1. 使用するフィルターを指定するには、 AWS CLI 入力 JSON ファイルを作成します。 は以下を使用します。

     ```
     {
         "sourceType": "s3",
         "destinationType": "iottwinmaker",
         "filters": [{
             "state": "COMPLETED"
     ```

```
    }]
}
```

有効なstate値のリストについては、AWS IoT TwinMaker 「 API リファレンスガイド」のListMetadataTransferJobs「フィルター」を参照してください。

2. 次のコマンド AWS CLI 例では、JSON ファイルを引数として使用します。

```
aws iottwinmaker list-metadata-transfer-job --region your_region \
        --cli-input-json file://ListMetadataTransferJobsExample.json
```

- ジョブをキャンセルするには、 CancelMetadataTransferJob API アクションを使用します。この API は、エクスポートまたはインポート済みのリソースに影響を与えることなく、特定のメタデータ転送ジョブをキャンセルします。

```
aws iottwinmaker cancel-metadata-transfer-job \
        --region your_region \
        --metadata-transfer-job-id job-to-cancel-id
```

エラーの検査

エラーの詳細を確認する (コンソール)

AWS IoT SiteWise コンソールでのエラーの詳細：

1. AWS IoT SiteWise コンソールに移動します。
2. 一括オペレーションジョブのリストについては、「」の「ジョブの進行状況」表 AWS IoT SiteWise コンソール を参照してください。
3. ジョブを選択して、ジョブの詳細を表示します。
4. ジョブのステータスが COMPLETEDまたは の場合ERROR、Total resourcesカウントは詳細カウント (Succeeded、、および) の合計に等しくなりますFailedSkipped。
5. ジョブのステータスが の場合ERROR、ジョブ失敗テーブルで特定のエラーと失敗の詳細を確認します。
6. ジョブ失敗テーブルには、ジョブレポートのコンテンツが表示されます。Resource type フィールドには、次のようなエラーまたは失敗の場所が表示されます。

- 例えば、 Resource typeフィールドBulk operations templateの の検証エラーは、インポートテンプレートとメタデータスキーマのファイル形式が一致していないことを示します。詳細については、「AWS IoT SiteWise メタデータ転送ジョブスキーマ」を参照してください。
- Resource type フィールドAssetで が失敗した場合、別のアセットとの競合によりアセットが作成されていないことを示します。 AWS IoT SiteWise リソースのエラーと競合については、「一般的なエラー」を参照してください。

エラーの詳細を検査する (AWS CLI)

転送ジョブ中に生成されたエラーを処理および診断するには、 GetMetadataTransferJob API アクションの使用に関する次の手順を参照してください。

1. 転送ジョブを作成して実行したら、 を呼び出しますGetMetadataTransferJob。

```
aws iottwinmaker get-metadata-transfer-job \
        --metadata-transfer-job-id your_metadata_transfer_job_id \
        --region us-east-1
```

2. ジョブの状態が になったらCOMPLETED、ジョブの結果の検証を開始できます。
3. を呼び出すとGetMetadataTransferJob、 というオブジェクトが返されますMetadataTransferJobProgress。

   MetadataTransferJobProgress オブジェクトには、次のパラメータが含まれます。

   - failedCount : 転送プロセス中に失敗したアセットの数を示します。
   - skippedCount : 転送プロセス中にスキップされたアセットの数を示します。
   - succeededCount : 転送プロセス中に成功したアセットの数を示します。
   - totalCount : 転送プロセスに関係するアセットの合計数を示します。
4. さらに、API コールはreportUrl、署名付き URL を含む要素 を返します。転送ジョブでさらに調査する必要がある問題がある場合は、この URL にアクセスしてください。

メタデータのインポートの例

このセクションでは、メタデータファイルを作成して、1 回の一括インポートオペレーションでアセットモデルとアセットをインポートする方法を示します。

一括インポートの例

1 回の一括インポート操作で、多くのアセットモデルとアセットをインポートできます。次の例は、メタデータファイルを作成してこれを行う方法を示しています。

このシナリオ例では、作業セルに産業用ロボットを含むさまざまな作業サイトがあります。

この例では、次の 2 つのアセットモデルを定義します。

- RobotModel1: このアセットモデルは、作業現場にある特定のタイプのロボットを表します。ロボットには測定プロパティ がありますTemperature。
- WorkCell: このアセットモデルは、いずれかの作業サイト内のロボットのコレクションを表します。アセットモデルは、作業セルにロボットが含まれる関係を表すrobotHierarchyOEM1階層を定義します。

この例では、いくつかのアセットも定義しています。

- WorkCell1: ボストンサイト内のワークセル
- RobotArm123456: その作業セル内のロボット
- RobotArm987654: その作業セル内の別のロボット

次の JSON メタデータファイルは、これらのアセットモデルとアセットを定義します。このメタデータを使用して一括インポートを実行すると AWS IoT SiteWise、階層関係を含むアセットモデルとアセットが 内に作成されます。

インポート用のメタデータファイル

```
{
    "assetModels": [
        {
            "assetModelExternalId": "Robot.OEM1.3536",
            "assetModelName": "RobotModel1",
            "assetModelProperties": [
                {
                    "dataType": "DOUBLE",
                    "externalId": "Temperature",
                    "name": "Temperature",
                    "type": {
                        "measurement": {
                            "processingConfig": {
```

```
                            "forwardingConfig": {
                                "state": "ENABLED"
                            }
                        }
                    }
                },
                "unit": "fahrenheit"
            }
        ]
    },
    {
        "assetModelExternalId": "ISA95.WorkCell",
        "assetModelName": "WorkCell",
        "assetModelProperties": [],
        "assetModelHierarchies": [
            {
                "externalId": "workCellHierarchyWithOEM1Robot",
                "name": "robotHierarchyOEM1",
                "childAssetModelExternalId": "Robot.OEM1.3536"
            }
        ]
    }
],
"assets": [
    {
        "assetExternalId": "Robot.OEM1.3536.123456",
        "assetName": "RobotArm123456",
        "assetModelExternalId": "Robot.OEM1.3536"
    },
    {
        "assetExternalId": "Robot.OEM1.3536.987654",
        "assetName": "RobotArm987654",
        "assetModelExternalId": "Robot.OEM1.3536"
    },
    {
        "assetExternalId": "BostonSite.Area1.Line1.WorkCell1",
        "assetName": "WorkCell1",
        "assetModelExternalId": "ISA95.WorkCell",
        "assetHierarchies": [
            {
                "externalId": "workCellHierarchyWithOEM1Robot",
                "childAssetExternalId": "Robot.OEM1.3536.123456"
            },
            {
```

```
                    "externalId": "workCellHierarchyWithOEM1Robot",
                    "childAssetExternalId": "Robot.OEM1.3536.987654"
                }
            ]
        }
    ]
}
```

モデルとアセットの初期オンボーディングの例

このシナリオの例では、会社の産業用ロボットを含むさまざまな作業サイトがあります。

この例では、複数のアセットモデルを定義します。

- Sample_Enterprise – このアセットモデルは、サイトが属する会社を表します。アセットモデルは、サイトとエンタープライズの関係Enterprise to Siteを表す階層 を定義します。
- Sample_Site – このアセットモデルは、社内の製造サイトを表します。アセットモデルでは、階層 を定義してSite to Line、サイトへの行の関係を表します。
- Sample_Welding Line – このアセットモデルは、作業サイト内のアセンブリラインを表します。アセットモデルは、ロボットとラインとの関係Line to Robotを表す階層 を定義します。
- Sample_Welding Robot – このアセットモデルは、作業現場の特定のタイプのロボットを表します。

この例では、アセットモデルに基づいてアセットも定義します。

- Sample_AnyCompany Motor – このアセットはSample_Enterpriseアセットモデルから作成されます。
- Sample_Chicago – このアセットはSample_Siteアセットモデルから作成されます。
- Sample_Welding Line 1 – このアセットはSample_Welding Lineアセットモデルから作成されます。
- Sample_Welding Robot 1 – このアセットはSample_Welding Robotアセットモデルから作成されます。
- Sample_Welding Robot 2 – このアセットはSample_Welding Robotアセットモデルから作成されます。

次の JSON メタデータファイルは、これらのアセットモデルとアセットを定義します。このメタデータを使用して一括インポートを実行すると AWS IoT SiteWise、階層関係を含むアセットモデルとアセットが 内に作成されます。

インポートするアセットとモデルをオンボードする JSON ファイル

```
{
    "assetModels": [
        {
            "assetModelExternalId": "External_Id_Welding_Robot",
            "assetModelName": "Sample_Welding Robot",
            "assetModelProperties": [
                {
                    "dataType": "STRING",
                    "externalId": "External_Id_Welding_Robot_Serial_Number",
                    "name": "Serial Number",
                    "type": {
                        "attribute": {
                            "defaultValue": "-"
                        }
                    },
                    "unit": "-"
                },
                {
                    "dataType": "DOUBLE",
                    "externalId": "External_Id_Welding_Robot_Cycle_Count",
                    "name": "CycleCount",
                    "type": {
                        "measurement": {}
                    },
                    "unit": "EA"
                },
                {
                    "dataType": "DOUBLE",
                    "externalId": "External_Id_Welding_Robot_Joint_1_Current",
                    "name": "Joint 1 Current",
                    "type": {
                        "measurement": {}
                    },
                    "unit": "Amps"
                },
                {
```

```
                    "dataType": "DOUBLE",
                    "externalId": "External_Id_Welding_Robot_Joint_1_Max_Current",
                    "name": "Max Joint 1 Current",
                    "type": {
                        "metric": {
                            "expression": "max(joint1current)",
                            "variables": [
                                {
                                    "name": "joint1current",
                                    "value": {
                                        "propertyExternalId":
"External_Id_Welding_Robot_Joint_1_Current"
                                    }
                                }
                            ],
                            "window": {
                                "tumbling": {
                                    "interval": "5m"
                                }
                            }
                        }
                    },
                    "unit": "Amps"
                }
            ]
        },
        {
            "assetModelExternalId": "External_Id_Welding_Line",
            "assetModelName": "Sample_Welding Line",
            "assetModelProperties": [
                {
                    "dataType": "DOUBLE",
                    "externalId": "External_Id_Welding_Line_Availability",
                    "name": "Availability",
                    "type": {
                        "measurement": {}
                    },
                    "unit": "%"
                }
            ],
            "assetModelHierarchies": [
                {
                    "externalId": "External_Id_Welding_Line_TO_Robot",
                    "name": "Line to Robot",
```

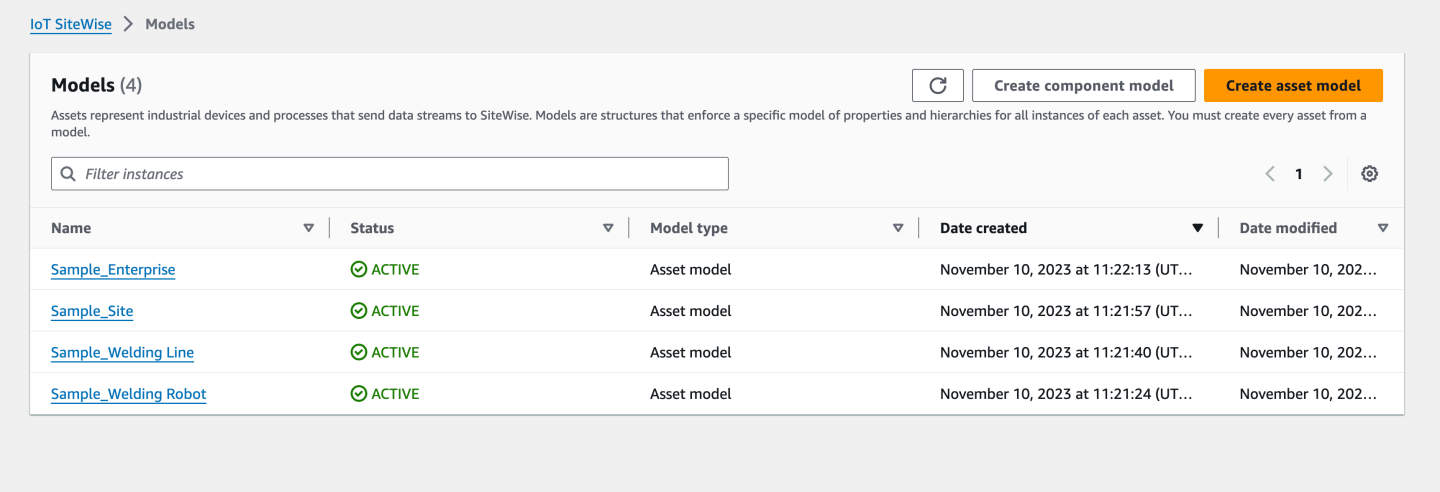
```
                    "childAssetModelExternalId": "External_Id_Welding_Robot"
                }
            ]
        },
        {
            "assetModelExternalId": "External_Id_Site",
            "assetModelName": "Sample_Site",
            "assetModelProperties": [
                {
                    "dataType": "STRING",
                    "externalId": "External_Id_Site_Street_Address",
                    "name": "Street Address",
                    "type": {
                        "attribute": {
                            "defaultValue": "-"
                        }
                    },
                    "unit": "-"
                }
            ],
            "assetModelHierarchies": [
                {
                    "externalId": "External_Id_Site_TO_Line",
                    "name": "Site to Line",
                    "childAssetModelExternalId": "External_Id_Welding_Line"
                }
            ]
        },
        {
            "assetModelExternalId": "External_Id_Enterprise",
            "assetModelName": "Sample_Enterprise",
            "assetModelProperties": [
                {
                    "dataType": "STRING",
                    "name": "Company Name",
                    "externalId": "External_Id_Enterprise_Company_Name",
                    "type": {
                        "attribute": {
                            "defaultValue": "-"
                        }
                    },
                    "unit": "-"
                }
            ],
```

```
            "assetModelHierarchies": [
                {
                    "externalId": "External_Id_Enterprise_TO_Site",
                    "name": "Enterprise to Site",
                    "childAssetModelExternalId": "External_Id_Site"
                }
            ]
        }
    ],
    "assets": [
        {
            "assetExternalId": "External_Id_Welding_Robot_1",
            "assetName": "Sample_Welding Robot 1",
            "assetModelExternalId": "External_Id_Welding_Robot",
            "assetProperties": [
                {
                    "externalId": "External_Id_Welding_Robot_Serial_Number",
                    "attributeValue": "S1000"
                },
                {
                    "externalId": "External_Id_Welding_Robot_Cycle_Count",
                    "alias": "AnyCompany/Chicago/Welding Line/S1000/Count"
                },
                {
                    "externalId": "External_Id_Welding_Robot_Joint_1_Current",
                    "alias": "AnyCompany/Chicago/Welding Line/S1000/1/Current"
                }
            ]
        },
        {
            "assetExternalId": "External_Id_Welding_Robot_2",
            "assetName": "Sample_Welding Robot 2",
            "assetModelExternalId": "External_Id_Welding_Robot",
            "assetProperties": [
                {
                    "externalId": "External_Id_Welding_Robot_Serial_Number",
                    "attributeValue": "S2000"
                },
                {
                    "externalId": "External_Id_Welding_Robot_Cycle_Count",
                    "alias": "AnyCompany/Chicago/Welding Line/S2000/Count"
                },
                {
                    "externalId": "External_Id_Welding_Robot_Joint_1_Current",
```

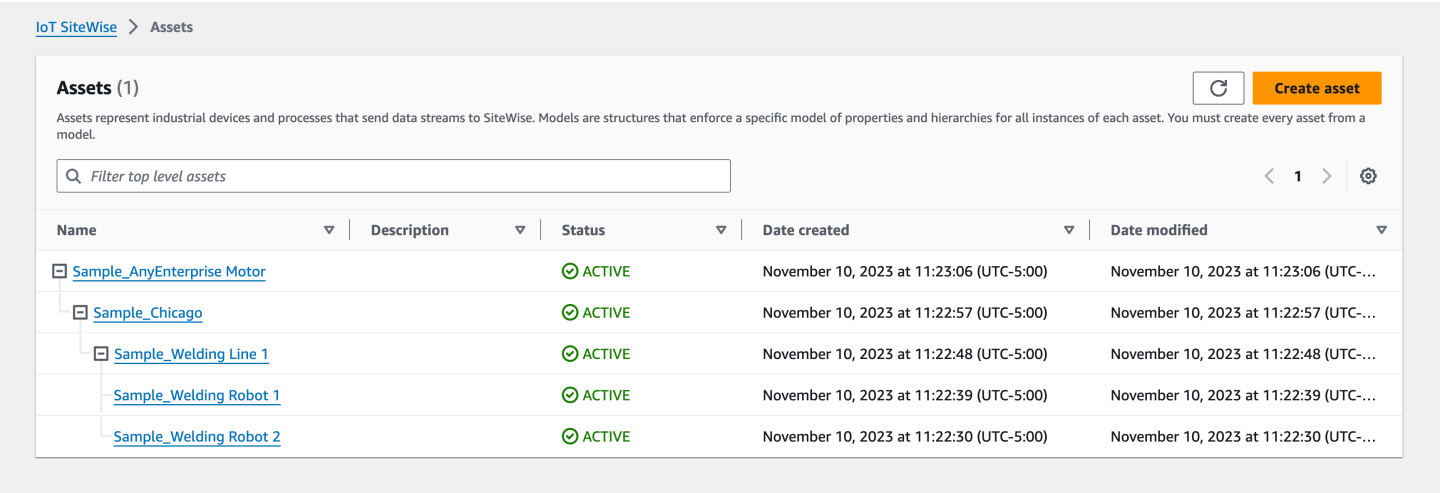
```
                    "alias": "AnyCompany/Chicago/Welding Line/S2000/1/Current"
                }
            ]
        },
        {
            "assetExternalId": "External_Id_Welding_Line_1",
            "assetName": "Sample_Welding Line 1",
            "assetModelExternalId": "External_Id_Welding_Line",
            "assetProperties": [
                {
                    "externalId": "External_Id_Welding_Line_Availability",
                    "alias": "AnyCompany/Chicago/Welding Line/Availability"
                }
            ],
            "assetHierarchies": [
                {
                    "externalId": "External_Id_Welding_Line_TO_Robot",
                    "childAssetExternalId": "External_Id_Welding_Robot_1"
                },
                {
                    "externalId": "External_Id_Welding_Line_TO_Robot",
                    "childAssetExternalId": "External_Id_Welding_Robot_2"
                }
            ]
        },
        {
            "assetExternalId": "External_Id_Site_Chicago",
            "assetName": "Sample_Chicago",
            "assetModelExternalId": "External_Id_Site",
            "assetHierarchies": [
                {
                    "externalId": "External_Id_Site_TO_Line",
                    "childAssetExternalId": "External_Id_Welding_Line_1"
                }
            ]
        },
        {
            "assetExternalId": "External_Id_Enterprise_AnyCompany",
            "assetName": "Sample_AnyEnterprise Motor",
            "assetModelExternalId": "External_Id_Enterprise",
            "assetHierarchies": [
                {
                    "externalId": "External_Id_Enterprise_TO_Site",
                    "childAssetExternalId": "External_Id_Site_Chicago"
```

```
                }
            ]
        }
    ]
}
```

次のスクリーンショットは、前のコード例を実行 AWS IoT SiteWise コンソール した後に に表示されるモデルのスクリーンショットです。



次のスクリーンショットは、前のコード例を実行 AWS IoT SiteWise コンソール した後に に表示されるモデル、アセット、階層を示しています。



追加アセットのオンボーディングの例

この例では、アカウント内の既存のアセットモデルにインポートする追加のアセットを定義します。

- Sample_Welding Line 2 – このアセットはSample_Welding Lineアセットモデルから作成されます。
- Sample_Welding Robot 3– このアセットはSample_Welding Robotアセットモデルから作成されます。
- Sample_Welding Robot 4– このアセットはSample_Welding Robotアセットモデルから作成されます。

この例の初期アセットを作成するには、「」を参照してくださいモデルとアセットの初期オンボーディングの例。

次の JSON メタデータファイルは、これらのアセットモデルとアセットを定義します。このメタデータを使用して一括インポートを実行すると AWS IoT SiteWise、階層関係を含むアセットモデルとアセットが 内に作成されます。

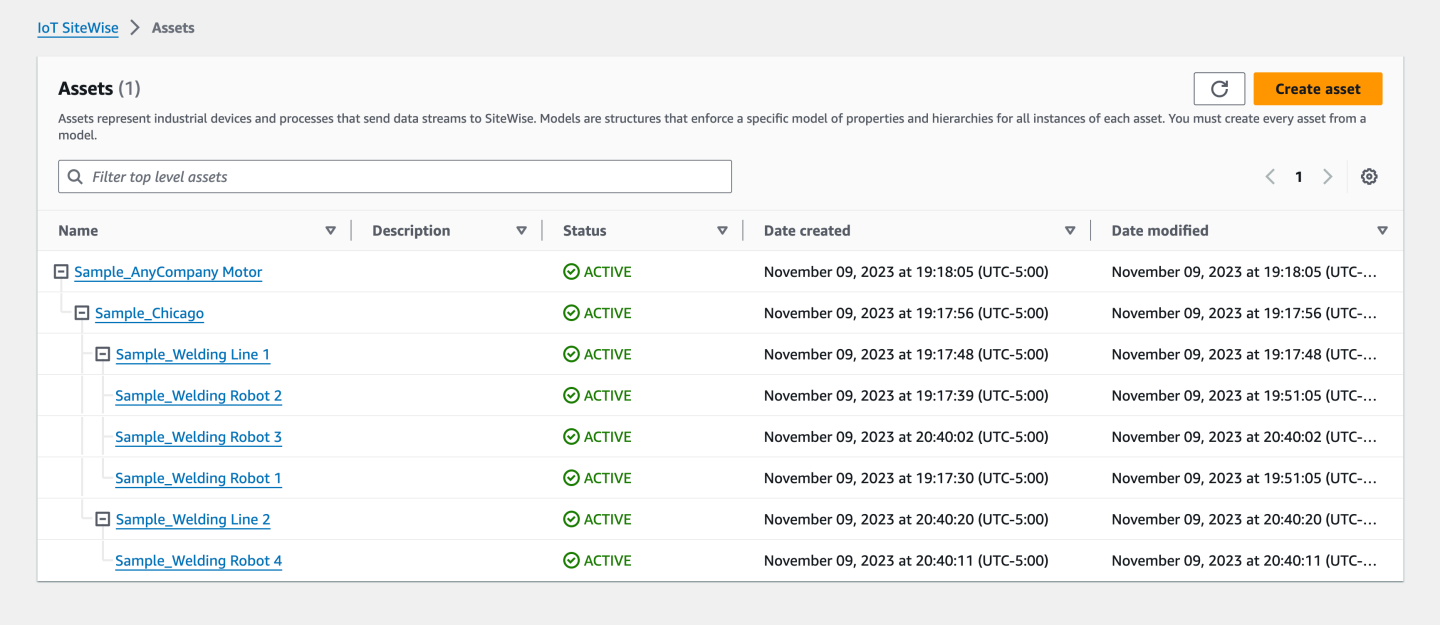
追加のアセットをオンボードするための JSON ファイル

```
{
    "assets": [
        {
            "assetExternalId": "External_Id_Welding_Robot_3",
            "assetName": "Sample_Welding Robot 3",
            "assetModelExternalId": "External_Id_Welding_Robot",
            "assetProperties": [
                {
                    "externalId": "External_Id_Welding_Robot_Serial_Number",
                    "attributeValue": "S3000"
                },
                {
                    "externalId": "External_Id_Welding_Robot_Cycle_Count",
                    "alias": "AnyCompany/Chicago/Welding Line/S3000/Count"
                },
                {
                    "externalId": "External_Id_Welding_Robot_Joint_1_Current",
                    "alias": "AnyCompany/Chicago/Welding Line/S3000/1/Current"
                }
            ]
        },
        {
            "assetExternalId": "External_Id_Welding_Robot_4",
            "assetName": "Sample_Welding Robot 4",
```

```
            "assetModelExternalId": "External_Id_Welding_Robot",
            "assetProperties": [
                {
                    "externalId": "External_Id_Welding_Robot_Serial_Number",
                    "attributeValue": "S4000"
                },
                {
                    "externalId": "External_Id_Welding_Robot_Cycle_Count",
                    "alias": "AnyCompany/Chicago/Welding Line/S4000/Count"
                },
                {
                    "externalId": "External_Id_Welding_Robot_Joint_1_Current",
                    "alias": "AnyCompany/Chicago/Welding Line/S4000/1/Current"
                }
            ]
        },
        {
            "assetExternalId": "External_Id_Welding_Line_1",
            "assetName": "Sample_Welding Line 1",
            "assetModelExternalId": "External_Id_Welding_Line",
            "assetHierarchies": [
                {
                    "externalId": "External_Id_Welding_Line_TO_Robot",
                    "childAssetExternalId": "External_Id_Welding_Robot_1"
                },
                {
                    "externalId": "External_Id_Welding_Line_TO_Robot",
                    "childAssetExternalId": "External_Id_Welding_Robot_2"
                },
                {
                    "externalId": "External_Id_Welding_Line_TO_Robot",
                    "childAssetExternalId": "External_Id_Welding_Robot_3"
                }
            ]
        },
        {
            "assetExternalId": "External_Id_Welding_Line_2",
            "assetName": "Sample_Welding Line 2",
            "assetModelExternalId": "External_Id_Welding_Line",
            "assetHierarchies": [
                {
                    "externalId": "External_Id_Welding_Line_TO_Robot",
                    "childAssetExternalId": "External_Id_Welding_Robot_4"
                }
```

```
            ]
        },
        {
            "assetExternalId": "External_Id_Site_Chicago",
            "assetName": "Sample_Chicago",
            "assetModelExternalId": "External_Id_Site",
            "assetHierarchies": [
                {
                    "externalId": "External_Id_Site_TO_Line",
                    "childAssetExternalId": "External_Id_Welding_Line_1"
                },
                {
                    "externalId": "External_Id_Site_TO_Line",
                    "childAssetExternalId": "External_Id_Welding_Line_2"
                }
            ]
        }
    ]
}
```

次のスクリーンショットは、前のコード例を実行 AWS IoT SiteWise コンソール した後に に表示されるモデル、アセット、階層を示しています。



新しいプロパティのオンボーディングの例

この例では、既存のアセットモデルに新しいプロパティを定義します。追加のアセットとモデルをオンボード追加アセットのオンボーディングの例するには、「」を参照してください。

- Joint 1 Temperature – このプロパティはSample_Welding Robotアセットモデルに追加されます。この新しいプロパティは、アセットモデルから作成された各Sample_Welding Robotアセットにも伝播されます。

既存のアセットモデルに新しいプロパティを追加するには、次の JSON メタデータファイルの例を参照してください。JSON に示すように、既存のSample_Welding Robotアセットモデル定義全体を新しいプロパティとともに提供する必要があります。既存の定義のプロパティリスト全体が指定されていない場合、 は省略されたプロパティ AWS IoT SiteWise を削除します。

新しいプロパティをオンボードするための JSON ファイル

この例では、アセットモデルJoint 1 Temperatureに新しいプロパティを追加します。

```
{
    "assetModels": [
        {
            "assetModelExternalId": "External_Id_Welding_Robot",
            "assetModelName": "Sample_Welding Robot",
            "assetModelProperties": [
                {
                    "dataType": "STRING",
                    "externalId": "External_Id_Welding_Robot_Serial_Number",
                    "name": "Serial Number",
                    "type": {
                        "attribute": {
                            "defaultValue": "-"
                        }
                    },
                    "unit": "-"
                },
                {
                    "dataType": "DOUBLE",
                    "externalId": "External_Id_Welding_Robot_Cycle_Count",
                    "name": "CycleCount",
                    "type": {
                        "measurement": {}
                    },
                    "unit": "EA"
                },
                {
                    "dataType": "DOUBLE",
```

```
                    "externalId": "External_Id_Welding_Robot_Joint_1_Current",
                    "name": "Joint 1 Current",
                    "type": {
                        "measurement": {}
                    },
                    "unit": "Amps"
                },
                {
                    "dataType": "DOUBLE",
                    "externalId": "External_Id_Welding_Robot_Joint_1_Max_Current",
                    "name": "Max Joint 1 Current",
                    "type": {
                        "metric": {
                            "expression": "max(joint1current)",
                            "variables": [
                                {
                                    "name": "joint1current",
                                    "value": {
                                        "propertyExternalId":
"External_Id_Welding_Robot_Joint_1_Current"
                                    }
                                }
                            ],
                            "window": {
                                "tumbling": {
                                    "interval": "5m"
                                }
                            }
                        }
                    },
                    "unit": "Amps"
                },
                {
                    "dataType": "DOUBLE",
                    "externalId": "External_Id_Welding_Robot_Joint_1_Temperature",
                    "name": "Joint 1 Temperature",
                    "type": {
                        "measurement": {}
                    },
                    "unit": "degC"
                }
            ]
        }
    ]
```

```
}
```

メタデータのエクスポート例

Amazon S3 への AWS IoT SiteWise コンテンツの一括エクスポートを実行するときに、エクスポートする特定のアセットモデルとアセットを制限するフィルターを指定できます。

フィルターは、リクエスト本文の `iotSiteWiseConfiguration`セクション内の `sources`セクションで指定します。

> **Note**
>
> 複数のフィルターを含めることができます。一括オペレーションでは、いずれかのフィルターに一致するアセットモデルまたはアセットがエクスポートされます。
> フィルターを指定しない場合、オペレーションはすべてのアセットモデルとアセットをエクスポートします。

```
{
    "metadataTransferJobId": "your-transfer-job-id",
    "sources": [{
        "type": "iotsitewise",
        "iotSiteWiseConfiguration": {
            "filters": [{
                list of filters
            }]
        }
    }],
    "destination": {
        "type": "s3",
        "s3Configuration": {
            "location": "arn:aws:s3:::your-S3-bucket-location"
        }
    }
}
```

アセットモデルによるフィルタリング

特定のアセットモデルをフィルタリングできます。そのモデルを使用するすべてのアセット、またはその階層内のすべてのアセットモデルを含めることもできます。アセットと階層の両方を含めることはできません。

階層の詳細については、「アセットモデル階層の定義」を参照してください。

Asset model

このフィルターには、指定されたアセットモデルが含まれます。

```
"filterByAssetModel": {
    "assetModelId": "asset model ID"
}
```

Asset model and its assets

このフィルターには、指定されたアセットモデルとそのアセットモデルを使用するすべてのアセットが含まれます。

```
"filterByAssetModel": {
    "assetModelId": "asset model ID",
    "includeAssets": true
}
```

Asset model and its hierarchy

このフィルターには、指定されたアセットモデルとその階層内のすべての関連するアセットモデルが含まれます。

```
"filterByAssetModel": {
    "assetModelId": "asset model ID",
    "includeOffspring": true
}
```

アセットによるフィルタリング

特定のアセットをフィルタリングできます。そのアセットモデル、または関連するすべてのアセットを階層に含めることもできます。アセットモデルと階層の両方を含めることはできません。

階層の詳細については、「アセットモデル階層の定義」を参照してください。

Asset

このフィルターには、指定されたアセットが含まれます。

```
"filterByAsset": {
    "assetId": "asset ID"
}
```

Asset and its asset model

このフィルターには、指定したアセットと、使用するアセットモデルが含まれます。

```
"filterByAsset": {
    "assetId": "asset ID",
    "includeAssetModel": true
}
```

Asset and its hierarchy

このフィルターには、指定されたアセットと、その階層内のすべての関連アセットが含まれます。

```
"filterByAsset": {
    "assetId": "asset ID",
    "includeOffspring": true
}
```

AWS IoT SiteWise メタデータ転送ジョブスキーマ

独自の一括インポートおよびエクスポートオペレーションを実行するときは、 AWS IoT SiteWise メタデータ転送ジョブスキーマを参照として使用します。

```
{
  "$schema": "https://json-schema.org/draft/2020-12/schema",
  "title": "IoTSiteWise",
  "description": "Metadata transfer job resource schema for IoTSiteWise",
  "definitions": {
    "Name": {
      "type": "string",
```

```
      "minLength": 1,
      "maxLength": 256,
      "pattern": "[^\\u0000-\\u001F\\u007F]+"
    },
    "Description": {
      "type": "string",
      "minLength": 1,
      "maxLength": 2048,
      "pattern": "[^\\u0000-\\u001F\\u007F]+"
    },
    "ID": {
      "type": "string",
      "minLength": 36,
      "maxLength": 36,
      "pattern": "^[0-9a-f]{8}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{4}-[0-9a-f]{12}$"
    },
    "ExternalId": {
      "type": "string",
      "minLength": 2,
      "maxLength": 128,
      "pattern": "[a-zA-Z0-9_][a-zA-Z_\\-0-9.:]*[a-zA-Z0-9_]+"
    },
    "AttributeValue": {
      "description": "The value of the property attribute.",
      "type": "string",
      "minLength": 1,
      "maxLength": 1024,
      "pattern": "[^\\u0000-\\u001F\\u007F]+"
    },
    "PropertyUnit": {
      "description": "The unit of measure (such as Newtons or RPM) of the asset
property.",
      "type": "string",
      "minLength": 1,
      "maxLength": 256,
      "pattern": "[^\\u0000-\\u001F\\u007F]+"
    },
    "PropertyAlias": {
      "description": "The property alias that identifies the property.",
      "type": "string",
      "minLength": 1,
      "maxLength": 1000,
      "pattern": "[^\\u0000-\\u001F\\u007F]+"
    },
```

```
    "AssetProperty": {
      "description": "The asset property's definition, alias, unit, and notification
 state.",
      "type": "object",
      "additionalProperties": false,
      "anyOf": [
        {
          "required": [
            "id"
          ]
        },
        {
          "required": [
            "externalId"
          ]
        }
      ],
      "properties": {
        "id": {
          "description": "The ID of the asset property.",
          "$ref": "#/definitions/ID"
        },
        "externalId": {
          "description": "The ExternalID of the asset property.",
          "$ref": "#/definitions/ExternalId"
        },
        "alias": {
          "$ref": "#/definitions/PropertyAlias"
        },
        "unit": {
          "$ref": "#/definitions/PropertyUnit"
        },
        "attributeValue": {
          "$ref": "#/definitions/AttributeValue"
        },
        "retainDataOnAliasChange": {
          "type": "string",
          "default": "TRUE",
          "enum": [
            "TRUE",
            "FALSE"
          ]
        },
        "propertyNotificationState": {
```

```
          "description": "The MQTT notification state (ENABLED or DISABLED) for this
 asset property.",
          "type": "string",
          "enum": [
            "ENABLED",
            "DISABLED"
          ]
        }
      }
    },
    "AssetHierarchy": {
      "description": "A hierarchy specifies allowed parent/child asset relationships.",
      "type": "object",
      "additionalProperties": false,
      "anyOf": [
        {
          "required": [
            "id",
            "childAssetId"
          ]
        },
        {
          "required": [
            "externalId",
            "childAssetId"
          ]
        },
        {
          "required": [
            "id",
            "childAssetExternalId"
          ]
        },
        {
          "required": [
            "externalId",
            "childAssetExternalId"
          ]
        }
      ],
      "properties": {
        "id": {
          "description": "The ID of a hierarchy in the parent asset's model.",
          "$ref": "#/definitions/ID"
```

```
        },
        "externalId": {
          "description": "The ExternalID of a hierarchy in the parent asset's model.",
          "$ref": "#/definitions/ExternalId"
        },
        "childAssetId": {
          "description": "The ID of the child asset to be associated.",
          "$ref": "#/definitions/ID"
        },
        "childAssetExternalId": {
          "description": "The ExternalID of the child asset to be associated.",
          "$ref": "#/definitions/ExternalId"
        }
      }
    },
    "Tag": {
      "type": "object",
      "additionalProperties": false,
      "required": [
        "key",
        "value"
      ],
      "properties": {
        "key": {
          "type": "string"
        },
        "value": {
          "type": "string"
        }
      }
    },
    "AssetModelType": {
      "type": "string",
      "default": null,
      "enum": [
        "ASSET_MODEL",
        "COMPONENT_MODEL"
      ]
    },
    "AssetModelCompositeModel": {
      "description": "Contains a composite model definition in an asset model. This
composite model definition is applied to all assets created from the asset model.",
      "type": "object",
      "additionalProperties": false,
```

```
      "anyOf": [
        {
          "required": [
            "id"
          ]
        },
        {
          "required": [
            "externalId"
          ]
        }
      ],
      "required": [
        "name",
        "type"
      ],
      "properties": {
        "id": {
          "description": "The ID of the asset model composite model.",
          "$ref": "#/definitions/ID"
        },
        "externalId": {
          "description": "The ExternalID of the asset model composite model.",
          "$ref": "#/definitions/ExternalId"
        },
        "parentId": {
          "description": "The ID of the parent asset model composite model.",
          "$ref": "#/definitions/ID"
        },
        "parentExternalId": {
          "description": "The ExternalID of the parent asset model composite model.",
          "$ref": "#/definitions/ExternalId"
        },
        "composedAssetModelId": {
          "description": "The ID of the composed asset model.",
          "$ref": "#/definitions/ID"
        },
        "composedAssetModelExternalId": {
          "description": "The ExternalID of the composed asset model.",
          "$ref": "#/definitions/ExternalId"
        },
        "description": {
          "description": "A description for the asset composite model.",
          "$ref": "#/definitions/Description"
```

```
        },
        "name": {
          "description": "A unique, friendly name for the asset composite model.",
          "$ref": "#/definitions/Name"
        },
        "type": {
          "description": "The type of the composite model. For alarm composite models,
 this type is AWS/ALARM.",
          "$ref": "#/definitions/Name"
        },
        "properties": {
          "description": "The property definitions of the asset model.",
          "type": "array",
          "items": {
            "$ref": "#/definitions/AssetModelProperty"
          }
        }
      }
    },
    "AssetModelProperty": {
      "description": "Contains information about an asset model property.",
      "type": "object",
      "additionalProperties": false,
      "anyOf": [
        {
          "required": [
            "id"
          ]
        },
        {
          "required": [
            "externalId"
          ]
        }
      ],
      "required": [
        "name",
        "dataType",
        "type"
      ],
      "properties": {
        "id": {
          "description": "The ID of the asset model property.",
          "$ref": "#/definitions/ID"
```

```
        },
        "externalId": {
          "description": "The ExternalID of the asset model property.",
          "$ref": "#/definitions/ExternalId"
        },
        "name": {
          "description": "The name of the asset model property.",
          "$ref": "#/definitions/Name"
        },
        "dataType": {
          "description": "The data type of the asset model property.",
          "$ref": "#/definitions/DataType"
        },
        "dataTypeSpec": {
          "description": "The data type of the structure for this property.",
          "$ref": "#/definitions/Name"
        },
        "unit": {
          "description": "The unit of the asset model property, such as Newtons or
RPM.",
          "type": "string",
          "minLength": 1,
          "maxLength": 256,
          "pattern": "[^\\u0000-\\u001F\\u007F]+"
        },
        "type": {
          "description": "The property type",
          "$ref": "#/definitions/PropertyType"
        }
      }
    },
    "DataType": {
      "type": "string",
      "enum": [
        "STRING",
        "INTEGER",
        "DOUBLE",
        "BOOLEAN",
        "STRUCT"
      ]
    },
    "PropertyType": {
      "description": "Contains a property type, which can be one of attribute,
measurement, metric, or transform.",
```

```
      "type": "object",
      "additionalProperties": false,
      "properties": {
        "attribute": {
          "$ref": "#/definitions/Attribute"
        },
        "transform": {
          "$ref": "#/definitions/Transform"
        },
        "metric": {
          "$ref": "#/definitions/Metric"
        },
        "measurement": {
          "$ref": "#/definitions/Measurement"
        }
      }
    },
    "Attribute": {
      "type": "object",
      "additionalProperties": false,
      "properties": {
        "defaultValue": {
          "type": "string",
          "minLength": 1,
          "maxLength": 1024,
          "pattern": "[^\\u0000-\\u001F\\u007F]+"
        }
      }
    },
    "Transform": {
      "type": "object",
      "additionalProperties": false,
      "required": [
        "expression",
        "variables"
      ],
      "properties": {
        "expression": {
          "description": "The mathematical expression that defines the transformation
function.",
          "type": "string",
          "minLength": 1,
          "maxLength": 1024
        },
```

```
        "variables": {
          "description": "The list of variables used in the expression.",
          "type": "array",
          "items": {
            "$ref": "#/definitions/ExpressionVariable"
          }
        },
        "processingConfig": {
          "$ref": "#/definitions/TransformProcessingConfig"
        }
      }
    },
    "TransformProcessingConfig": {
      "description": "The processing configuration for the given transform property.",
      "type": "object",
      "additionalProperties": false,
      "required": [
        "computeLocation"
      ],
      "properties": {
        "computeLocation": {
          "description": "The compute location for the given transform property.",
          "$ref": "#/definitions/ComputeLocation"
        },
        "forwardingConfig": {
          "description": "The forwarding configuration for a given property.",
          "$ref": "#/definitions/ForwardingConfig"
        }
      }
    },
    "Metric": {
      "type": "object",
      "additionalProperties": false,
      "required": [
        "expression",
        "variables",
        "window"
      ],
      "properties": {
        "expression": {
          "description": "The mathematical expression that defines the metric
aggregation function.",
          "type": "string",
          "minLength": 1,
```

```
          "maxLength": 1024
        },
        "variables": {
          "description": "The list of variables used in the expression.",
          "type": "array",
          "items": {
            "$ref": "#/definitions/ExpressionVariable"
          }
        },
        "window": {
          "description": "The window (time interval) over which AWS IoT SiteWise
computes the metric's aggregation expression",
          "$ref": "#/definitions/MetricWindow"
        },
        "processingConfig": {
          "$ref": "#/definitions/MetricProcessingConfig"
        }
      }
    },
    "MetricProcessingConfig": {
      "description": "The processing configuration for the metric.",
      "type": "object",
      "additionalProperties": false,
      "required": [
        "computeLocation"
      ],
      "properties": {
        "computeLocation": {
          "description": "The compute location for the given metric property.",
          "$ref": "#/definitions/ComputeLocation"
        }
      }
    },
    "ComputeLocation": {
      "type": "string",
      "enum": [
        "EDGE",
        "CLOUD"
      ]
    },
    "ForwardingConfig": {
      "type": "object",
      "additionalProperties": false,
      "required": [
```

```
        "state"
      ],
      "properties": {
        "state": {
          "type": "string",
          "enum": [
            "ENABLED",
            "DISABLED"
          ]
        }
      }
    },
    "MetricWindow": {
      "description": "Contains a time interval window used for data aggregate
computations (for example, average, sum, count, and so on).",
      "type": "object",
      "additionalProperties": false,
      "properties": {
        "tumbling": {
          "description": "The tumbling time interval window.",
          "type": "object",
          "additionalProperties": false,
          "required": [
            "interval"
          ],
          "properties": {
            "interval": {
              "description": "The time interval for the tumbling window.",
              "type": "string",
              "minLength": 2,
              "maxLength": 23
            },
            "offset": {
              "description": "The offset for the tumbling window.",
              "type": "string",
              "minLength": 2,
              "maxLength": 25
            }
          }
        }
      }
    },
    "ExpressionVariable": {
      "type": "object",
```

```
      "additionalProperties": false,
      "required": [
        "name",
        "value"
      ],
      "properties": {
        "name": {
          "description": "The friendly name of the variable to be used in the
expression.",
          "type": "string",
          "minLength": 1,
          "maxLength": 64,
          "pattern": "^[a-z][a-z0-9_]*$"
        },
        "value": {
          "description": "The variable that identifies an asset property from which to
use values.",
          "$ref": "#/definitions/VariableValue"
        }
      }
    },
    "VariableValue": {
      "type": "object",
      "additionalProperties": false,
      "anyOf": [
        {
          "required": [
            "propertyId"
          ]
        },
        {
          "required": [
            "propertyExternalId"
          ]
        }
      ],
      "properties": {
        "propertyId": {
          "$ref": "#/definitions/ID"
        },
        "propertyExternalId": {
          "$ref": "#/definitions/ExternalId"
        },
        "hierarchyId": {
```

```
          "$ref": "#/definitions/ID"
        },
        "hierarchyExternalId": {
          "$ref": "#/definitions/ExternalId"
        }
      }
    },
    "Measurement": {
      "type": "object",
      "additionalProperties": false,
      "properties": {
        "processingConfig": {
          "$ref": "#/definitions/MeasurementProcessingConfig"
        }
      }
    },
    "MeasurementProcessingConfig": {
      "type": "object",
      "additionalProperties": false,
      "required": [
        "forwardingConfig"
      ],
      "properties": {
        "forwardingConfig": {
          "description": "The forwarding configuration for the given measurement
property.",
          "$ref": "#/definitions/ForwardingConfig"
        }
      }
    },
    "AssetModelHierarchy": {
      "description": "Contains information about an asset model hierarchy.",
      "type": "object",
      "additionalProperties": false,
      "anyOf": [
        {
          "required": [
            "id",
            "childAssetModelId"
          ]
        },
        {
          "required": [
            "id",
```

```
            "childAssetModelExternalId"
          ]
        },
        {
          "required": [
            "externalId",
            "childAssetModelId"
          ]
        },
        {
          "required": [
            "externalId",
            "childAssetModelExternalId"
          ]
        }
      ],
      "required": [
        "name"
      ],
      "properties": {
        "id": {
          "description": "The ID of the asset model hierarchy.",
          "$ref": "#/definitions/ID"
        },
        "externalId": {
          "description": "The ExternalID of the asset model hierarchy.",
          "$ref": "#/definitions/ExternalId"
        },
        "name": {
          "description": "The name of the asset model hierarchy.",
          "$ref": "#/definitions/Name"
        },
        "childAssetModelId": {
          "description": "The ID of the asset model. All assets in this hierarchy must
be instances of the child AssetModelId asset model.",
          "$ref": "#/definitions/ID"
        },
        "childAssetModelExternalId": {
          "description": "The ExternalID of the asset model. All assets in this
hierarchy must be instances of the child AssetModelId asset model.",
          "$ref": "#/definitions/ExternalId"
        }
      }
    },
```

```
    "AssetModel": {
      "type": "object",
      "additionalProperties": false,
      "anyOf": [
        {
          "required": [
            "assetModelId"
          ]
        },
        {
          "required": [
            "assetModelExternalId"
          ]
        }
      ],
      "required": [
        "assetModelName"
      ],
      "properties": {
        "assetModelId": {
          "description": "The ID of the asset model.",
          "$ref": "#/definitions/ID"
        },
        "assetModelExternalId": {
          "description": "The ID of the asset model.",
          "$ref": "#/definitions/ExternalId"
        },
        "assetModelName": {
          "description": "A unique, friendly name for the asset model.",
          "$ref": "#/definitions/Name"
        },
        "assetModelDescription": {
          "description": "A description for the asset model.",
          "$ref": "#/definitions/Description"
        },
        "assetModelType": {
          "description": "The type of the asset model.",
          "$ref": "#/definitions/AssetModelType"
        },
        "assetModelProperties": {
          "description": "The property definitions of the asset model.",
          "type": "array",
          "items": {
            "$ref": "#/definitions/AssetModelProperty"
```

```
          }
        },
        "assetModelCompositeModels": {
          "description": "The composite asset models that are part of this asset model.
Composite asset models are asset models that contain specific properties.",
          "type": "array",
          "items": {
            "$ref": "#/definitions/AssetModelCompositeModel"
          }
        },
        "assetModelHierarchies": {
          "description": "The hierarchy definitions of the asset model. Each hierarchy
specifies an asset model whose assets can be children of any other assets created from
this asset model.",
          "type": "array",
          "items": {
            "$ref": "#/definitions/AssetModelHierarchy"
          }
        },
        "tags": {
          "description": "A list of key-value pairs that contain metadata for the asset
model.",
          "type": "array",
          "items": {
            "$ref": "#/definitions/Tag"
          }
        }
      }
    },
    "Asset": {
      "type": "object",
      "additionalProperties": false,
      "anyOf": [
        {
          "required": [
            "assetId",
            "assetModelId"
          ]
        },
        {
          "required": [
            "assetExternalId",
            "assetModelId"
          ]
```

```
        },
        {
          "required": [
            "assetId",
            "assetModelExternalId"
          ]
        },
        {
          "required": [
            "assetExternalId",
            "assetModelExternalId"
          ]
        }
      ],
      "required": [
        "assetName"
      ],
      "properties": {
        "assetId": {
          "description": "The ID of the asset",
          "$ref": "#/definitions/ID"
        },
        "assetExternalId": {
          "description": "The external ID of the asset",
          "$ref": "#/definitions/ExternalId"
        },
        "assetModelId": {
          "description": "The ID of the asset model from which to create the asset.",
          "$ref": "#/definitions/ID"
        },
        "assetModelExternalId": {
          "description": "The ExternalID of the asset model from which to create the
asset.",
          "$ref": "#/definitions/ExternalId"
        },
        "assetName": {
          "description": "A unique, friendly name for the asset.",
          "$ref": "#/definitions/Name"
        },
        "assetDescription": {
          "description": "A description for the asset",
          "$ref": "#/definitions/Description"
        },
        "assetProperties": {
```

```
          "type": "array",
          "items": {
            "$ref": "#/definitions/AssetProperty"
          }
        },
        "assetHierarchies": {
          "type": "array",
          "items": {
            "$ref": "#/definitions/AssetHierarchy"
          }
        },
        "tags": {
          "description": "A list of key-value pairs that contain metadata for the
 asset.",
          "type": "array",
          "uniqueItems": false,
          "items": {
            "$ref": "#/definitions/Tag"
          }
        }
      }
    }
  },
  "additionalProperties": false,
  "properties": {
    "assetModels": {
      "type": "array",
      "uniqueItems": false,
      "items": {
        "$ref": "#/definitions/AssetModel"
      }
    },
    "assets": {
      "type": "array",
      "uniqueItems": false,
      "items": {
        "$ref": "#/definitions/Asset"
      }
    }
  }
}
```

アラームによるデータのモニタリング。

データに対してアラームを設定し、機器の動作やプロセスが最適な状態でない場合に、チームにアラート通知を送信することができます。機器やプロセスの最適なパフォーマンスとは、特定のメトリクスの値が上限と下限の範囲内にあることを指します。これらのメトリクスが運用の範囲外の場合、機器のオペレーターに通知して問題を解決する必要があります。アラームを使用して、問題を迅速に特定し、オペレーターに通知して、機器やプロセスのパフォーマンスを最大限に高めることができます。

トピック

アラーム型

AWS クラウドで検出するアラームと、外部プロセスで検出するアラームを定義できます。 では、次のタイプのアラーム AWS IoT SiteWise がサポートされています。

- AWS IoT Events アラーム

  AWS IoT Events アラームは、 で を検出するアラーム AWS IoT SiteWise です AWS IoT Events。は、アセットプロパティ値を のアラームモデルに送信します AWS IoT Events。次に、 はアラーム状態を AWS IoT Events に送信します AWS IoT SiteWise。アラームが検出されるタイミングや、アラームの状態が変化したときの通知先などのオプションを設定することができます。また、アラームの状態が変化したときに発生する [AWS IoT Events actions] (アクション) を定義することができます。

  のアラーム AWS IoT Events は、アラームモデルのインスタンスです。アラームモデルは、アラームのしきい値や重要度、アラームの状態が変化したときの処理などを指定します。アラームモデルの各属性を設定するとき、アラームがモニタリングするアセットモデルから属性プロパティを指定

します。アセットモデルに基づくすべてのアセットは、 がアラームの特性 AWS IoT Events を評価するときに 属性の値を使用します。詳細については、[AWS IoT Events Developer Guide] (デベロッパーガイド) の[Using alarms] (アラームの使用) を参照してください。

AWS IoT Events アラームの状態が変わったときに応答できます。例えば、アラームがアクティブになったときに、それを確認したり、スヌーズしたりすることができます。また、アラームの有効化、無効化、リセットも可能です。

SiteWise Monitor ユーザーは、 SiteWise Monitor ポータルで AWS IoT Events アラームを視覚化、設定、対応できます。詳細については、[AWS IoT SiteWise Monitor Developer Guide] (デベロッパーガイド) の[Monitoring with alarms] (アラームによるモニタリング) を参照してください。

> **Note**
>
> AWS IoT Events 料金は、これらのアラームを評価し、 AWS IoT SiteWise と の間でデータを転送するために適用されます AWS IoT Events。詳細については、AWS IoT Events 料金を参照してください。

- 外部アラーム

  外部アラームは、 の外部で評価するアラームです AWS IoT SiteWise。アラームの状態を報告するデータソースがある場合は、外部アラームを使用します。外部アラームは、アラーム状態データを取り込む測定プロパティを含んでいます。

  外部アラームの状態が変わっても、確認やスヌーズはできません。

  SiteWise Monitor ユーザーは Monitor SiteWise ポータルで外部アラームの状態を確認できますが、これらのアラームを設定したり応答したりすることはできません。

  AWS IoT SiteWise は外部アラームの状態を評価しません。

アラームの状態

産業用アラームには、モニタリングする機器やプロセスの状態に関する情報と、アラーム状態に対するオペレータの対応に関する情報(オプション) が含まれています。

AWS IoT Events アラームを定義するときは、確認応答フロー を有効にするかどうかを指定します。フロー承認はデフォルトで有効になっています。このオプションを有効にすると、オペレーターはアラームを承認し、アラームに関する詳細や対処方法をメモに残すことができます。アクティブなア

ラームが非アクティブになる前にオペレーターが承認しないと、アラームはラッチされた状態になります。ラッチされた状態は、アラームがアクティブになり、承認されなかったことを示します。したがって、オペレータは装置またはプロセスをチェックし、ラッチされたアラームを承認する必要があります。

アラームには次の状態があります。

- [正常] (Normal) – アラームは有効ですが、非アクティブです。工業プロセスや機器が期待通りに動作している。
- [アクティブ] (Active) – アラームはアクティブです。産業プロセスまたは装置が動作範囲外にあり、注意が必要である。
- [承認済み] (Acknowledged) – オペレータがアラームの状態を承認しました。

  この状態は、承認フローを有効にしたアラームにのみ適用されます。
- [ラッチされた] (Latched) – アラームは正常に戻ったが、アクティブであり、オペレーターはそれを承認しなかった。産業プロセスや機器では、アラームを正常に戻すためにオペレーターの注意が必要です。

  この状態は、承認フローを有効にしたアラームにのみ適用されます。
- [スヌーズ] (SnoozeDisabled) – オペレータがアラームをスヌーズしたため、アラームは無効になります。アラームをスヌーズさせる時間をオペレータが定義します。この時間が経過すると、アラームは通常状態に戻ります。
- [無効] (Disabled) — アラームは無効になり、検出されません。

アラーム状態のプロパティ

AWS IoT SiteWise は、アラーム状態データを文字列にシリアル化された JSON オブジェクトとして保存します。このオブジェクトは、アラームの状態、およびオペレータの応答アクションやアラームが評価するルールなどの追加情報を含みます。

アラーム状態のプロパティは、その名前と構造体型である AWS/ALARM_STATE によって識別されます。詳細については、「アセットモデルにおけるアラームの定義」を参照してください。

アラーム状態データオブジェクトは、次の情報が含まれています。

`stateName`

アラームの状態。詳細については、「アラームの状態」を参照してください。

データ型: STRING

customerAction

(オプション) アラームに対するオペレータの応答に関する情報を含むオブジェクト。オペレータは、アラームの有効化、無効化、確認、スヌーズを行うことができます。その際、アラームの状態データには、彼らの応答と、応答時に残すことのできるメモが含まれます。このオブジェクトには、次の情報が含まれます。

actionName

オペレータがアラームに対応するために取るアクションの名前。この値には、次の文字列のいずれかが含まれます。

- ENABLE
- DISABLE
- SNOOZE
- ACKNOWLEDGE
- RESET

データ型: STRING

enable

(オプション) オペレータがアラームを有効にしたときに、customerAction に存在するオブジェクトです。オペレータがアラームを有効にすると、アラーム状態は Normal になります。このオブジェクトには、次の情報が含まれます。

note

(オプション) 顧客がアラームを有効にする際に残すメモ。

データ型: STRING

最大文字数: 128 文字

disable

(オプション) オペレータがアラームを無効にしたときに customerAction に存在するオブジェクト。オペレータがアラームを有効にすると、アラーム状態は Disabled になります。このオブジェクトには、次の情報が含まれます。

note

(オプション) 顧客がアラームを無効にする際に残すメモ。

データ型: STRING

最大文字数: 128 文字

acknowledge

(オプション) オペレータがアラームを承認するときに customerAction に存在するオブジェクト。オペレータがアラームを有効にすると、アラーム状態は Acknowledged になります。このオブジェクトには、次の情報が含まれます。

note

(オプション) 顧客がアラームを確認する際に残すメモ。

データ型: STRING

最大文字数: 128 文字

snooze

(オプション) オペレータがアラームをスヌーズしたときに customerAction に存在するオブジェクト。オペレータがアラームを有効にすると、アラーム状態は SnoozeDisabled になります。このオブジェクトには、次の情報が含まれます。

snoozeDuration

オペレータがアラームをスヌーズする時間 (秒)。この時間経過後、アラームは Normal 状態になります。

データ型: INTEGER

note

(オプション) お客様がアラームをスヌーズした際に残すメモです。

データ型: STRING

最大文字数: 128 文字

ruleEvaluation

(オプション) アラームを評価するルールに関する情報を含むオブジェクトです。このオブジェクトには、次の情報が含まれます。

simpleRule

プロパティ値としきい値を比較演算子で比較する、単純なルールに関する情報を持つオブジェクト。このオブジェクトには、次の情報が含まれます。

`inputProperty`

このアラームが評価するプロパティの値。

データ型: DOUBLE

`operator`

このアラームがプロパティをしきい値を比較するために使用する比較演算子。この値には、次の文字列のいずれかが含まれます。

- < - 未満
- <= - 次
- == - 等しい。
- != - 等しくない。
- >= - 以上
- > - 次より大きい

データ型: STRING

`threshold`

プロパティ値に対して、本アラームが比較するしきい値を設定する。

データ型: DOUBLE

アセットモデルにおけるアラームの定義

アセットモデルは、産業用データとアラームの標準化を推進します。アセットモデルにアラーム定義を行うことで、アセットモデルに基づくすべてのアセットのアラームを標準化することができます。

アセットモデルにアラームを定義するには、[composite asset model] (複合アセットモデル) を使用します。複合アセットモデルとは、他のアセットモデルに特定のプロパティのセットを標準化したアセットモデルです。複合アセットモデルは、アセットモデルに特定のプロパティが存在することを保証します。アラームは型、ステート、および (オプションの) 出典プロパティを持つため、アラーム複合モデルはこれらのプロパティの存在を強制する。

各複合アセットモデルには、その複合モデルのプロパティを定義する型があります。アラーム複合モデルは、アラーム型、アラーム状態、および (オプションの) アラーム出典のプロパティを定義しま

す。複合モデルを使ってアセットモデルからアセットを作成すると、アセットモデルで指定したプロパティに加えて、複合モデルのプロパティもアセットに含まれます。

複合モデル内の各プロパティは、その複合モデルの型に応じて識別できる名前を持つ必要があります。複合モデルプロパティは、複雑なデータ型を持つプロパティに対応しています。これらのプロパティは、STRUCT データ型と、プロパティの複合データ型を指定する dataTypeSpec 特性を持つ。複合データ型プロパティは、文字列としてシリアライズされた JSON データを含む。

アラーム複合モデルは次のような特性を持ちます。各プロパティは、この型の複合モデル用に識別するための名前を持つ必要があります。

アラーム型

アラームの型。次のいずれかを指定します。

- IOT_EVENTS – AWS IoT Events alarm. AWS IoT SiteWise sends data to AWS IoT Events は、このアラームの状態を評価します。このアラーム定義の AWS IoT Events アラームモデルを定義するには、アラームソースプロパティを指定する必要があります。
- EXTERNAL - 外部アラーム アラームの状態を測定値として取り込みます。

プロパティ名: AWS/ALARM_TYPE

プロパティ型: [attribute] (属性)

データ型: STRING

[Alarm state] (アラームの状態)

アラームの状態を表す時系列データです。これは文字列としてシリアライズされたオブジェクトであり、アラームの状態やその他の情報を含む。詳細については、「アラーム状態のプロパティ」を参照してください。

プロパティ名: AWS/ALARM_STATE

プロパティの種類: [measurement] (測定)

データ型: STRUCT

データ構造型:AWS/ALARM_STATE

[Alarm source] (アラーム出典)

(オプション) アラームの状態を評価するリソースの Amazon リソースネーム (ARN) です。 AWS IoT Events アラームの場合、これはアラームモデルの ARN です。

プロパティ名: AWS/ALARM_SOURCE

プロパティ型: [attribute] (属性)

データ型: STRING

Example アラーム複合モデルの例

次のアセットモデルは、温度をモニタリングするアラームがあるボイラーを表します。 は温度データを AWS IoT SiteWise に送信 AWS IoT Events してアラームを検出します。

```
{
  "assetModelName": "Boiler",
  "assetModelDescription": "A boiler that alarms when its temperature exceeds its
 limit.",
  "assetModelProperties": [
    {
      "name": "Temperature",
      "dataType": "DOUBLE",
      "unit": "Celsius",
      "type": {
        "measurement": {}
      }
    },
    {
      "name": "High Temperature",
      "dataType": "DOUBLE",
      "unit": "Celsius",
      "type": {
        "attribute": {
          "defaultValue": "105.0"
        }
      }
    }
  ],
  "assetModelCompositeModels": [
    {
      "name": "BoilerTemperatureHighAlarm",
      "type": "AWS/ALARM",
      "properties": [
        {
          "name": "AWS/ALARM_TYPE",
          "dataType": "STRING",
```

```
          "type": {
            "attribute": {
              "defaultValue": "IOT_EVENTS"
            }
          }
        },
        {
          "name": "AWS/ALARM_STATE",
          "dataType": "STRUCT",
          "dataTypeSpec": "AWS/ALARM_STATE",
          "type": {
            "measurement": {}
          }
        },
        {
          "name": "AWS/ALARM_SOURCE",
          "dataType": "STRING",
          "type": {
            "attribute": {}
          }
        }
      ]
    }
  ]
}
```

トピック

- AWS IoT Events アラームの定義
- 外部アラームの定義

AWS IoT Events アラームの定義

AWS IoT Events アラームを作成すると、 AWS IoT SiteWise はアセットプロパティ値を に送信 AWS IoT Events して alarm. AWS IoT Events alarm 定義の状態を評価します。 は、 で定義したアラームモデルによって異なります AWS IoT Events。アセットモデルに AWS IoT Events アラームを定義するには、アラームモデルを AWS IoT Events アラームソースプロパティとして指定するアラーム複合モデルを定義します。

AWS IoT Events アラームは、アラームのしきい値やアラーム通知設定などの入力によって異なります。これらの入力は、アセットモデル上の属性として定義します。そして、これらの入力をモデルに

基づいて各アセットでカスタマイズすることができます。 AWS IoT SiteWise コンソールでこれらの属性を作成できます。 AWS CLI または API でアラームを定義する場合は、アセットモデルでこれらの属性を手動で定義する必要があります。

また、カスタムアラーム通知アクションなど、アラームが検出されたときに発生するその他のアクションを定義することも可能です。例えば、Amazon SNSのトピックにプッシュ通知を送信するアクションを設定することができます。定義できるアクションの詳細については、「 AWS IoT Events デベロッパーガイド」の「他の AWS のサービスの使用」を参照してください。

アセットモデルを更新または削除すると、 は、 の AWS IoT Events アラームモデルがこのアセットモデルに関連付けられたアセットプロパティをモニタリングしているかどうか AWS IoT SiteWise を確認できます。これにより、 AWS IoT Events アラームが現在使用しているアセットプロパティを削除できなくなります。でこの機能を有効にするには AWS IoT SiteWise、 アクセス`iotevents:ListInputRoutings`許可が必要です。このアクセス許可により AWS IoT SiteWise 、 は でサポートされている ListInputRoutings API オペレーションを呼び出すことができます AWS IoT Events。詳細については、「(オプション) ListInputRoutings アクセス許可」を参照してください。

> **Note**
>
> アラーム通知機能は、中国 (北京) リージョンでは利用できません。

トピック

- アラーム通知の要件
- AWS IoT Events アラームの定義AWS IoT SiteWise （ コンソール ）
- AWS IoT Events アラームの定義AWS IoT Events （ コンソール ）
- AWS IoT Events アラームの定義 (AWS CLI)

アラーム通知の要件

AWS IoT Events は、 AWS アカウントの AWS Lambda 関数を使用してアラーム通知を送信します。アラーム通知を有効にするには、アラームと同じ AWS リージョンにこの Lambda 関数を作成する必要があります。このLambda関数は、テキスト通知の送信に[Amazon Simple Notification Service (Amazon SNS)]、メール通知の送信に[Amazon Simple Email Service (Amazon SES)]を使用しています。 AWS IoT Events アラームを作成するときは、アラームが通知を送信するために使用するプロトコルと設定を構成します。

AWS IoT Events は、 アカウントでこの Lambda 関数を作成するために使用できる AWS CloudFormation スタックテンプレートを提供します。詳細については、[AWS IoT Events Developer Guide] (デベロッパーガイド) の[Alarm notification Lambda function] (アラーム通知 Lambda 関数) を参照してください。

AWS IoT Events アラームの定義AWS IoT SiteWise （ コンソール ）

AWS IoT SiteWise コンソールを使用して、既存のアセットモデルに AWS IoT Events アラームを定義できます。新しいアセットモデルに AWS IoT Events アラームを定義するには、アセットモデルを作成し、次のステップを実行します。詳細については、「アセットモデルを作成する」を参照してください。

> **Important**
>
> 各アラームには、そのアラームの比較対象となるしきい値を指定する属性が必要です。アラームを定義する前に、アセットモデルでしきい値属性を定義する必要があります。
> 例えば、風力タービンが定格最大風速 50mph を超えた場合にアラームを発生させたい場合を考えてみましょう。アラームを定義する前に、デフォルト値が `50` の属性 ([最大風速]) を定義する必要があります。

アセットモデルに AWS IoT Events アラームを定義するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[モデル] を選択します。
3. アラームを定義するアセットモデルを選択します。
4. [アラーム] タブを選択します。
5. [アラームの追加] を選択します。
6. [アラーム型のオプション] セクションで、[AWS IoT Events アラーム] を選択します。
7. [アラームの詳細] セクションで、次の手順を実行します。
   a. アラームの名前を入力します。
   b. (オプション) アラームの説明を入力します。
8. [しきい値定義] セクションでは、アラーム検出のタイミングとアラームの重要度を定義します。以下の操作を実行します。

a. アラームを検出した[プロパティ] を選択します。このプロパティが新しい値を受信するたびに、 は値を AWS IoT SiteWise に送信 AWS IoT Events してアラームの状態を評価します。

b. プロパティとしきい値の比較に使用する [演算子] を選択します。次のオプションから選択します。

- [< 未満]
- [<= より小さい、または等しい]
- == 等しい
- != 等しくない
- [>= より大きい、または等しい]
- [> より大きい]

c. 値 で、しきい値として使用する属性プロパティを選択します。 AWS IoT Events は、 プロパティの値をこの属性の値と比較します。

d. アラームの[Severity] (重要度) を入力します。このアラームの重要度を反映させるために、チームが理解できる数字を使用してください。

9. (オプション) [通知設定 - オプション] セクションで、次の操作を行います。

a. [アクティブ] を選択します。

> **Note**
>
> [非アクティブ] を選択した場合、アラーム通知は送信されません。

b. [受信者] では、受信者を選択します。

> **Important**
>
> AWS IAM Identity Center ユーザーにアラーム通知を送信できます。この機能を使用するには、IAM Identity Center を有効にする必要があります。IAM Identity Center は、一度に 1 つの AWS リージョンでのみ有効にできます。このことは、IAM Identity Center を有効にしたリージョンでのみ、アラーム通知を定義できることを意味します。詳細については、[AWS IAM Identity Center User Guide] (ユーザーガイド) の[Getting started] (使用開始) を参照してください。

c. [Protocol] (プロトコル) については、次のオプションから選択してください。

- E メールとテキスト – SMS メッセージと E メールで IAM Identity Center ユーザーに通知します。
- E メール – E メールで IAM Identity Center ユーザーに通知します。
- テキスト – SMS メッセージで IAM Identity Center ユーザーに通知します。

d. [送信者] は、送信者を選択します。

> **Important**
>
> Amazon Simple Email Service (Amazon SES) で送信者メールアドレスの確認が必要です。詳細については、[Amazon Simple Email Service Developer Guide] (Amazon Simple Email Service デベロッパーガイド) の[Verifying email addresses in Amazon SES] (Amazon SES でのメールアドレスの確認) を参照してください。

10. [デフォルトのアセット状態] セクションでは、このアセットモデルから作成されたアラームに対するデフォルトの状態を設定できます。

> **Note**
>
> このアセットモデルから作成するすべてのアセットに対して、後のステップでこのアラームを有効または無効にすることができます。

11. 詳細設定セクションでは、アクセス許可、追加の通知設定、アラーム状態アクション、SiteWise Monitor のアラームモデル、確認フローを設定できます。

> **Note**
>
> AWS IoT Events アラームには、次のサービスロールが必要です。
>
> - がアラーム状態値を に送信することを AWS IoT Events 引き受けるロール AWS IoT SiteWise。
> - Lambda にデータを送信するために が AWS IoT Events 引き受けるロール。このロールは、アラームが通知を送信する場合にのみ必要です。

[許可] セクションで、以下の操作を行います。

a. [AWS IoT Events ロール] の場合、既存のロールを使用するか、必要なアクセス許可を持つロールを作成します。このロールには、`iotsitewise:BatchPutAssetPropertyValue` アクセス許可と、 iotevents.amazonaws.com がロールを引き受けることを許可する信頼関係が必要です。

b. [AWS IoT Events Lambda ロール] については、既存のロールを使用するか、必要なアクセス許可を持つロールを作成します。このロールには、`lambda:InvokeFunction`、 `sso-directory:DescribeUser` のアクセス許可と、`iotevents.amazonaws.com` がそのロールを担うことができる信頼関係が必要です。

12. (オプション) [追加の通知設定] セクションで、次を実行します。

a. [受信者属性] には、通知の受信者を指定する値を持つ属性を定義します。IAM Identity Center ユーザーを受信者として選択することができます。

属性を作成するか、アセットモデル上の既存の属性を使用することができます。

- [新規受信者属性を作成] を選択した場合は、受信者属性名とその属性に対する受信者のデフォルト値 - (オプション) を指定します。
- 「既存の属性を使用する」を選択した場合は、[受信者の属性名] で属性を選択します。アラームは、選択した属性のデフォルト値を使用します。

このアセットモデルから作成する各アセットで、デフォルト値を上書きすることができます。

b. [カスタムメッセージ属性] には、デフォルトの状態変化メッセージに加えて、送信するカスタムメッセージを指定する属性を定義します。例えば、このアラームの対処方法をチームが理解するのに役立つメッセージを指定することができます。

属性を作成するか、アセットモデル上の既存の属性を使用するかを選択することができます。

- 「新しいカスタムメッセージ属性を作成する」を選択した場合は、その属性に対するカスタムメッセージ属性名とカスタムメッセージのデフォルト値 - オプションを指定します。
- 「既存の属性を使用する」を選択した場合は、[カスタムメッセージの属性名] でその属性を選択します。アラームは、選択した属性のデフォルト値を使用します。

このアセットモデルから作成する各アセットで、デフォルト値を上書きすることができます。

c. [Lambda 関数の管理] では、次のいずれかを実行します。

- で新しい Lambda 関数 AWS IoT SiteWise を作成するには、AWS マネージドテンプレート から新しい Lambda を作成する を選択します。
- 既存の Lambda 関数を使用するには、[既存の Lambda を使用する] を選択して、関数の名前を選択します。

詳細については、AWS IoT Events デベロッパーガイドの「アラーム通知の管理」を参照してください。

13. (オプション) [状態アクションの設定] セクションでは、次を実行します。

a. [アクションを編集] を選択します。

b. [アラーム状態アクションの追加] でアクションを追加して、[保存] を選択します。

アクションは最大10個まで追加可能です。

AWS IoT Events は、アラームがアクティブなときにアクションを実行できます。組み込みアクションを定義して、タイマーを使用したり、変数を設定したり、他の AWS リソースにデータを送信したりできます。詳細については、[AWS IoT Events Developer Guide] (デベロッパーガイド) の[Supported actions] (対応アクション) を参照してください。

14. (オプション) SiteWise Monitor のアラームモデルを管理する - オプション で、アクティブまたは非アクティブ を選択します。

このオプションを使用すると、 SiteWise Monitors でアラームモデルを更新できます。デフォルトでは、このオプションは有効になっています。

15. [承認フロー] で、アクティブまたは非アクティブを選択します。フロー承認の詳細については、アラームの状態 を参照してください。

16. [アラームの追加] を選択します。

> **Note**
>
> AWS IoT SiteWise コンソールは、複数の API リクエストを実行して、アセットモデルにアラームを追加します。[アラームの追加] を選択すると、コンソールがダイアログボックスを開き、これらの API リクエストの進行状況が表示されます。各 API リクエストが成功するまで、または API リクエストが失敗するまで、このページに滞在してく

ださい。リクエストに失敗した場合は、ダイアログボックスを閉じて問題を修正し、[アラームの追加] を選択して再試行してください。

AWS IoT Events アラームの定義AWS IoT Events （ コンソール ）

AWS IoT Events コンソールを使用して、既存のアセットモデルに AWS IoT Events アラームを定義できます。新しいアセットモデルに AWS IoT Events アラームを定義するには、アセットモデルを作成し、次のステップを実行します。詳細については、「アセットモデルを作成する」を参照してください。

> **Important**
>
> 各アラームには、そのアラームの比較対象となるしきい値を指定する属性が必要です。アラームを定義する前に、アセットモデルでしきい値属性を定義する必要があります。
> 例えば、風力タービンが定格最大風速 50mph を超えた場合にアラームを発生させたい場合を考えてみましょう。アラームを定義する前に、デフォルト値が 50 の属性 ([最大風速]) を定義する必要があります。

アセットモデルに AWS IoT Events アラームを定義するには

1. AWS IoT Events コンソールに移動します。
2. ナビゲーションペインで、アラームモデルを選択します。
3. [アラームモデルの作成] を選択します。
4. アラームの名前を入力します。
5. (オプション) アラームの説明を入力します。
6. アラームターゲットセクションで、次の手順を実行します。
   a. ターゲットオプションでは、AWS IoT SiteWise アセットプロパティを選択します。
   b. アラームを追加するアセットモデルを選択します。
7. [しきい値定義] セクションでは、アラーム検出のタイミングとアラームの重要度を定義します。以下の操作を実行します。
   a. アラームを検出した[プロパティ] を選択します。このプロパティが新しい値を受信するたびに、 は値を AWS IoT SiteWise に送信 AWS IoT Events してアラームの状態を評価します。

b. プロパティとしきい値の比較に使用する [演算子] を選択します。次のオプションから選択します。

- [< 未満]
- [<= より小さい、または等しい]
- == 等しい
- != 等しくない
- [>= より大きい、または等しい]
- [> より大きい]

c. 値 で、しきい値として使用する属性プロパティを選択します。 AWS IoT Events は、 プロパティの値をこの属性の値と比較します。

d. アラームの[Severity] (重要度) を入力します。このアラームの重要度を反映させるために、チームが理解できる数字を使用してください。

8. (オプション) [通知設定 - オプション] セクションで、次の操作を行います。

a. [プロトコル] については、次のオプションから選択してください。

- E メールとテキスト – SMS メッセージと E メールで IAM Identity Center ユーザーに通知します。
- E メール – E メールで IAM Identity Center ユーザーに通知します。
- テキスト – SMS メッセージで IAM Identity Center ユーザーに通知します。

b. [送信者] は、送信者を選択します。

> **Important**
>
> Amazon Simple Email Service (Amazon SES) で送信者メールアドレスの確認が必要です。詳細については、[Amazon Simple Email Service Developer Guide] (Amazon Simple Email Service デベロッパーガイド) の[Verifying email addresses in Amazon SES] (Amazon SES でのメールアドレスの確認) を参照してください。

c. [受信者属性 - オプション] で属性を選択します。アラームは、選択した属性のデフォルト値を使用します。

d. [カスタムメッセージ属性 - オプション] で属性を選択します。アラームは、選択した属性のデフォルト値を使用します。

9. [インスタンス] セクションで、このアラームのデフォルト状態を指定します。後のステップで、このアセットモデルから作成するすべてのアセットに対して、このアラームを有効または無効にすることができます。
10. 詳細設定では、アクセス許可、追加の通知設定、アラーム状態アクション、 SiteWise Monitor のアラームモデル、確認フローを設定できます。

> **Note**
>
> AWS IoT Events アラームには、次のサービスロールが必要です。
>
> - がアラーム状態値を に送信することを AWS IoT Events 引き受けるロール AWS IoT SiteWise。
> - Lambda にデータを送信するために が AWS IoT Events 引き受けるロール。このロールは、アラームが通知を送信する場合にのみ必要です。

a. [承認フロー] セクションで有効または無効を選択します。フロー承認の詳細については、アラームの状態 を参照してください。
b. [許可] セクションで、以下の操作を行います。

   i. [AWS IoT Events ロール] の場合、既存のロールを使用するか、必要なアクセス許可を持つロールを作成します。このロールには、`iotsitewise:BatchPutAssetPropertyValue` アクセス許可と、iotevents.amazonaws.com がロールを引き受けることを許可する信頼関係が必要です。

   ii. [Lambda ロール] については、既存のロールを使用するか、必要なアクセス許可を持つロールを作成します。このロールには、`lambda:InvokeFunction`、 `sso-directory:DescribeUser` のアクセス許可と、`iotevents.amazonaws.com` がそのロールを担うことができる信頼関係が必要です。

c. (オプション) [Additional notification settings] (追加の通知設定) ペインで、次を実行します。

   - [Lambda 関数の管理] では、次のいずれかを実行します。
     - で新しい Lambda 関数 AWS IoT Events を作成するには、新しい Lambda 関数の作成 を選択します。
     - 既存の Lambda 関数を使用するには、[既存の Lambda 関数を使用する] を選択して、関数の名前を選択します。

詳細については、AWS IoT Events デベロッパーガイドの「アラーム通知の管理」を参照してください。

d. (オプション) [状態アクションの設定 - オプション] では、次を実行します。

- [アラーム状態アクション] でアクションを追加して、[保存] を選択します。

  アクションは最大10個まで追加可能です。

AWS IoT Events は、アラームがアクティブなときにアクションを実行できます。組み込みアクションを定義して、タイマーを使用したり、変数を設定したり、他の AWS リソースにデータを送信したりできます。詳細については、[AWS IoT Events Developer Guide] (デベロッパーガイド) の[Supported actions] (対応アクション) を参照してください。

11. [作成] を選択します。

> **Note**
>
> AWS IoT Events コンソールは、複数の API リクエストを実行して、アセットモデルにアラームを追加します。[アラームの追加] を選択すると、コンソールがダイアログボックスを開き、これらの API リクエストの進行状況が表示されます。各 API リクエストが成功するまで、または API リクエストが失敗するまで、このページに滞在してください。リクエストに失敗した場合は、ダイアログボックスを閉じて問題を修正し、[アラームの追加] を選択して再試行してください。

AWS IoT Events アラームの定義 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して、アセットプロパティをモニタリングする AWS IoT Events アラームを定義できます。新規または既存のアセットモデルにアラームを定義することができます。アセットモデルにアラームを定義したら、 でアラームを作成し AWS IoT Events 、アセットモデルに接続します。このプロセスでは、次のことを行います。

ステップ

- ステップ1: アセットモデルにアラームを定義する。
- ステップ 2: AWS IoT Events アラームモデルを定義する
- ステップ 3: AWS IoT SiteWise と の間のデータフローを有効にする AWS IoT Events

ステップ1: アセットモデルにアラームを定義する。

新規または既存のアセットモデルに、アラーム定義と関連プロパティを追加します。

アセットモデルにアラームを定義するには (CLI)。

1. `asset-model-payload.json` という名前のファイルを作成します。これらの他のセクションのステップに従って、アセットモデルの詳細をファイルに追加しますが、アセットモデルの作成または更新のリクエストは送信しないでください。このセクションでは、`asset-model-payload.json` ファイルのアセットモデル詳細にアラーム定義を追加します。
   - アセットモデルの作成方法については、アセットモデルの作成 (AWS CLI) を参照してください。
   - 既存のアセットモデルを更新する方法については、アセットまたはコンポーネントモデルの更新 (AWS CLI) を参照してください。

   > **Note**
   >
   > アセットモデルには、アラームでモニタリングするアセットプロパティを含め、少なくとも1つのアセットプロパティが定義されている必要があります。

2. アセットモデルにアラーム複合モデル (`assetModelCompositeModels`) を追加する。 AWS IoT Events アラーム複合モデルでは、`IOT_EVENTS`タイプを指定し、アラームソースプロパティを指定します。でアラームモデルを作成した後、アラームソースプロパティを追加します AWS IoT Events。

   > **Important**
   >
   > アラーム複合モデルの名前は、後で作成する AWS IoT Events アラームモデルと同じである必要があります。アラームモデル名には、英数字のみ使用できます。アラームモデルに同じ名前を使用できるように、一意な英数字の名前を指定します。

```
{
  ...
  "assetModelCompositeModels": [
    {
      "name": "BoilerTemperatureHighAlarm",
      "type": "AWS/ALARM",
      "properties": [
```

```
        {
          "name": "AWS/ALARM_TYPE",
          "dataType": "STRING",
          "type": {
            "attribute": {
              "defaultValue": "IOT_EVENTS"
            }
          }
        },
        {
          "name": "AWS/ALARM_STATE",
          "dataType": "STRUCT",
          "dataTypeSpec": "AWS/ALARM_STATE",
          "type": {
            "measurement": {}
          }
        }
      ]
    }
  ]
}
```

3. アセットモデルにアラームしきい値属性を追加する。このしきい値に使用するデフォルト値を指定します。このモデルに基づいて、各アセットでこのデフォルト値を上書きすることができます。

> **Note**
>
> アラームしきい値属性は、INTEGER または DOUBLE でなければならない。

```
{
  ...
  "assetModelProperties": [
    ...
    {
      "name": "Temperature Max Threshold",
      "dataType": "DOUBLE",
      "type": {
        "attribute": {
          "defaultValue": "105.0"
        }
```

```
      }
    }
  ]
}
```

4. (オプション) アセットモデルにアラーム通知属性を追加します。これらの属性は、アラームの状態が変更されたときに AWS IoT Events が通知を送信するために使用する IAM Identity Center 受信者およびその他の入力を指定します。このモデルに基づいて、各アセットでこれらのデフォルトをオーバーライドすることができます。

   > **Important**
   >
   > AWS IAM Identity Center ユーザーにアラーム通知を送信できます。この機能を使用するには、IAM Identity Center を有効にする必要があります。IAM Identity Center は、一度に 1 つの AWS リージョンでのみ有効にできます。このことは、IAM Identity Center を有効にしたリージョンでのみ、アラーム通知を定義できることを意味します。詳細については、[AWS IAM Identity Center User Guide] (ユーザーガイド) の[Getting started] (使用開始) を参照してください。

   以下の操作を実行します。

   a. IAM Identity Center ID ストアの ID を指定する属性を追加します。IAM Identity Center ListInstances API オペレーションを使用して、ID ストアを一覧表示できます。このオペレーションは、IAM Identity Center を有効にしたリージョンでのみ機能します。

      ```
      aws sso-admin list-instances
      ```

      次に、ID ストア ID (例: d-123EXAMPLE) を属性のデフォルト値として指定します。

      ```
      {
        ...
        "assetModelProperties": [
          ...
          {
            "name": "identityStoreId",
            "dataType": "STRING",
            "type": {
              "attribute": {
                "defaultValue": "d-123EXAMPLE"
      ```

```
        }
      }
    }
  ]
}
```

b. 通知の送信先の IAM Identity Center ユーザーの ID を指定する属性を追加します。デフォルトの通知受信者を定義するには、デフォルト値として IAM Identity Center ユーザー ID を追加します。次のいずれかを行って、IAM Identity Center ユーザー ID を取得します。

i. IAM Identity Center ListUsers API を使用して、ユーザー名がわかっているユーザーの ID を取得できます。*d-123EXAMPLE* は ID ストアの ID に、*[Name]* (名前) はユーザーのユーザー名に置き換えてください。

```
aws identitystore list-users \
  --identity-store-id d-123EXAMPLE \
  --filters AttributePath=UserName,AttributeValue=Name
```

ii. IAM Identity Center コンソール を使用してユーザーをブラウズして、ユーザー ID を探すことができます。

そして、ユーザー ID (例えば、123EXAMPLE-a1b2c3d4-5678-90ab-cdef-33333EXAMPLE) を属性のデフォルト値として指定するか、デフォルト値なしで属性を定義します。

```
{
  ...
  "assetModelProperties": [
    ...
    {
      "name": "userId",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "123EXAMPLE-a1b2c3d4-5678-90ab-cdef-33333EXAMPLE"
        }
      }
    }
  ]
}
```

c. (オプション) SMS (テキスト) メッセージ通知のデフォルトの送信者 ID を指定する属性を追加します。Amazon Simple Notification Service (Amazon SNS) が送信するメッセージの送信者として、送信者 ID が表示されます。詳細については、[Amazon Simple Notification Service Developer Guide] (Amazon Simple Notification Service デベロッパーガイド) の[Requesting sender IDs for SMS messaging with Amazon SNS] (Amazon SNS を利用した SMS メッセージの送信者 ID を要求する) を参照してください。

```
{
  ...
  "assetModelProperties": [
    ...
    {
      "name": "senderId",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "MyFactory"
        }
      }
    }
  ]
}
```

d. (オプション) E メール通知で[from] (送信元) アドレスとして使用するデフォルトのE メールアドレスを指定する属性を追加します。

```
{
  ...
  "assetModelProperties": [
    ...
    {
      "name": "fromAddress",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "my.factory@example.com"
        }
      }
    }
  ]
}
```

e. (オプション) E メール通知で使用するデフォルトの件名を指定する属性を追加します。

```
{
  ...
  "assetModelProperties": [
    ...
    {
      "name": "emailSubject",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "[ALERT] High boiler temperature"
        }
      }
    }
  ]
}
```

f. (オプション) 通知に含める追加メッセージを指定する属性を追加します。デフォルトでは、通知メッセージはアラームに関する情報を含んでいます。また、ユーザーにより多くの情報を与える追加メッセージを含めることもできます。

```
{
  ...
  "assetModelProperties": [
    ...
    {
      "name": "additionalMessage",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "Turn off the power before you check the alarm."
        }
      }
    }
  ]
}
```

5. アセットモデルを作成、または既存のアセットモデルを更新します。次のいずれかを行います。

   - アセットモデルを作成するには、次のコマンドを実行します。

```
aws iotsitewise create-asset-model --cli-input-json file://asset-model-
payload.json
```

- 既存のアセットモデルを更新するには、次のコマンドを実行します。*asset-model-id* をアセットモデルの ID に置き換えます。

```
aws iotsitewise update-asset-model \
  --asset-model-id asset-model-id \
  --cli-input-json file://asset-model-payload.json
```

コマンドを実行したら、レスポンスに assetModelId に注意してください。

例:ボイラーアセットモデル

次のアセットモデルは、温度データを報告するボイラーを表しています。このアセットモデルでは、ボイラーのオーバーヒートを検出するアラームを定義しています。

```
{
  "assetModelName": "Boiler Model",
  "assetModelDescription": "Represents a boiler.",
  "assetModelProperties": [
    {
      "name": "Temperature",
      "dataType": "DOUBLE",
      "unit": "C",
      "type": {
        "measurement": {}
      }
    },
    {
      "name": "Temperature Max Threshold",
      "dataType": "DOUBLE",
      "type": {
        "attribute": {
          "defaultValue": "105.0"
        }
      }
    },
    {
      "name": "identityStoreId",
      "dataType": "STRING",
```

```
      "type": {
        "attribute": {
          "defaultValue": "d-123EXAMPLE"
        }
      }
    },
    {
      "name": "userId",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "123EXAMPLE-a1b2c3d4-5678-90ab-cdef-33333EXAMPLE"
        }
      }
    },
    {
      "name": "senderId",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "MyFactory"
        }
      }
    },
    {
      "name": "fromAddress",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "my.factory@example.com"
        }
      }
    },
    {
      "name": "emailSubject",
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "[ALERT] High boiler temperature"
        }
      }
    },
    {
      "name": "additionalMessage",
```

```
      "dataType": "STRING",
      "type": {
        "attribute": {
          "defaultValue": "Turn off the power before you check the alarm."
        }
      }
    }
  ],
  "assetModelHierarchies": [

  ],
  "assetModelCompositeModels": [
    {
      "name": "BoilerTemperatureHighAlarm",
      "type": "AWS/ALARM",
      "properties": [
        {
          "name": "AWS/ALARM_TYPE",
          "dataType": "STRING",
          "type": {
            "attribute": {
              "defaultValue": "IOT_EVENTS"
            }
          }
        },
        {
          "name": "AWS/ALARM_STATE",
          "dataType": "STRUCT",
          "dataTypeSpec": "AWS/ALARM_STATE",
          "type": {
            "measurement": {}
          }
        }
      ]
    }
  ]
}
```

ステップ 2: AWS IoT Events アラームモデルを定義する

でアラームモデルを作成します AWS IoT Events。では AWS IoT Events、式を使用してアラームモデルの値を指定します。式を使用して の値を指定 AWS IoT SiteWise し、アラームへの入力として評価および使用できます。 AWS IoT SiteWise がアセットプロパティ値をアラームモデルに送信する

と、 AWS IoT Events は式を評価してプロパティの値またはアセットの ID を取得します。アラームモデルでは、次のような表現が可能です。

- アセットプロパティ値

  アセットプロパティの値を取得するには、次の表現を使用します。*####ModelId*をアセットモデルの ID に置き換え、*propertyId* をプロパティの ID に置き換えます。

  ```
  $sitewise.assetModel.`assetModelId`.`propertyId`.propertyValue.value
  ```

- [Asset IDs] (アセット ID)

  アセットのIDを取得するには、次の表現を使用します。*####ModelId*をアセットモデルの ID に置き換え、*propertyId* をプロパティの ID に置き換えます。

  ```
  $sitewise.assetModel.`assetModelId`.`propertyId`.assetId
  ```

> **Note**
>
> アラームモデルを作成するときは、 AWS IoT SiteWise 値に評価される式の代わりにリテラルを定義できます。これにより、アセットモデルに定義する属性の数を減らすことができます。ただし、値をリテラルで定義した場合、アセットモデルに基づいてアセットでその値をカスタマイズすることはできません。また、 AWS IoT SiteWise Monitor ユーザーはアセットに対してのみアラーム設定を構成できるため、アラームをカスタマイズすることはできません。

AWS IoT Events アラームモデルを作成するには (CLI)

1. でアラームモデルを作成するときは AWS IoT Events、アラームが使用する各プロパティの ID を指定する必要があります。これには、以下が含まれます。

   - 複合アセットモデルにおけるアラーム状態プロパティ。
   - アラームがモニタリングしているプロパティ。
   - しきい値属性
   - (オプション) IAM Identity Center ID ストア ID 属性
   - (オプション) IAM Identity Center ユーザー ID 属性

- (オプション) SMS 送信者 ID 属性
- (オプション) E メールの[from] (送信元) アドレス属性
- (オプション) E メールの件名属性
- (オプション) 追加のメッセージ属性

次のコマンドを実行し、アセットモデル上のこれらのプロパティの ID を取得します。*asset-model-id* を前のステップのアセットモデルの ID に置き換えてください。

```
aws iotsitewise describe-asset-model --asset-model-id asset-model-id
```

このオペレーションは、アセットモデルの詳細を含むレスポンスを返します。アラームが使用する各プロパティの ID を記録しておきます。これらの ID は、次のステップで AWS IoT Events アラームモデルを作成するときに使用します。

2. でアラームモデルを作成します AWS IoT Events。以下の操作を実行します。

   a. `alarm-model-payload.json` という名前のファイルを作成します。

   b. 次の JSON オブジェクトをファイルにコピーしてください。

   c. アラームの名前 (`alarmModelName`) 、説明 (`alarmModelDescription`) 、重要度 (`severity`) を入力します。重要度には、お客様の会社の重要度レベルを反映した整数を指定します。

   > **Important**
   >
   > アラームモデルは、先にアセットモデルで定義したアラーム複合モデルと同じ名前である必要があります。
   > アラームモデル名には、英数字のみ使用できます。

```
{
  "alarmModelName": "BoilerTemperatureHighAlarm",
  "alarmModelDescription": "Detects when the boiler temperature is high.",
  "severity": 3
}
```

d. アラームに比較ルール (alarmRule) を追加します。このルールでは、モニタリングするプロパティ (inputProperty) 、比較するしきい値 (threshold) 、使用する比較演算子 (comparisonOperator) を定義する。

- *####ModelId*をアセットモデルの ID に置き換えます。
- alarm を、*####PropertyId*がモニタリングする プロパティの ID に置き換えます。
- threshold を threshold 属性プロパティの ID *AttributeId*に置き換えます。
- *GREATER* は、プロパティ値としきい値の比較に使用する演算子で置き換えてください。次のオプションから選択します。
  - LESS
  - LESS_OR_EQUAL
  - EQUAL
  - NOT_EQUAL
  - GREATER_OR_EQUAL
  - GREATER

```
{
  "alarmModelName": "BoilerTemperatureHighAlarm",
  "alarmModelDescription": "Detects when the boiler temperature is high.",
  "severity": 3,
  "alarmRule": {
    "simpleRule": {
      "inputProperty":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.propertyValue.value",
      "comparisonOperator": "GREATER",
      "threshold":
 "$sitewise.assetModel.`assetModelId`.`thresholdAttributeId`.propertyValue.value"
    }
  }
}
```

e. アラームの状態が変化した時に、 AWS IoT SiteWise にアラームの状態を送信するアクション (alarmEventActions) を追加する。

> **Note**
>
> アドバンストな設定として、アラームの状態が変化したときに実行する追加アクションを定義することができます。例えば、 AWS Lambda 関数を呼び出したり、MQTT トピックに発行したりします。詳細については、「 AWS IoT Events デベロッパーガイド」の「他の AWS のサービスの使用」を参照してください。

- *####ModelId*をアセットモデルの ID に置き換えます。
- alarm を、*####PropertyId*がモニタリングする プロパティの ID に置き換えます。
- alarm *StatePropertyId* を、アラーム複合モデルのアラーム状態プロパティの ID に置き換えます。

```
{
  "alarmModelName": "BoilerTemperatureHighAlarm",
  "alarmModelDescription": "Detects when the boiler temperature is high.",
  "severity": 3,
  "alarmRule": {
    "simpleRule": {
      "inputProperty":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.propertyValue.value",
      "comparisonOperator": "GREATER",
      "threshold":
 "$sitewise.assetModel.`assetModelId`.`thresholdAttributeId`.propertyValue.value"
    }
  },
  "alarmEventActions": {
    "alarmActions": [
      {
        "iotSiteWise": {
          "assetId":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.assetId",
          "propertyId": "'alarmStatePropertyId'"
        }
      }
    ]
  }
}
```

f. (オプション) アラーム通知設定を行います。アラーム通知アクションは、アカウント内のLambda関数を使用して、アラーム通知を送信します。詳細については、「アラーム通知の要件」を参照してください。アラーム通知の設定では、IAM Identity Center ユーザーに送信する SMS と E メールの通知を設定できます。以下の操作を実行します。

i. `alarm-model-payload.json` のペイロードにアラーム通知設定 (`alarmNotification`) を追加する。

- alarm *`NotificationFunctionArn`* を、アラーム通知を処理する Lambda 関数の ARN に置き換えます。

```
{
  "alarmModelName": "BoilerTemperatureHighAlarm",
  "alarmModelDescription": "Detects when the boiler temperature is high.",
  "severity": 3,
  "alarmRule": {
    "simpleRule": {
      "inputProperty":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.propertyValue.value",
      "comparisonOperator": "GREATER",
      "threshold":
 "$sitewise.assetModel.`assetModelId`.`thresholdAttributeId`.propertyValue.value"
    }
  },
  "alarmEventActions": {
    "alarmActions": [
      {
        "iotSiteWise": {
          "assetId":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.assetId",
          "propertyId": "'alarmStatePropertyId'"
        }
      }
    ]
  },
  "alarmNotification": {
    "notificationActions": [
      {
        "action": {
          "lambdaAction": {
            "functionArn": "alarmNotificationFunctionArn"
```

```
          }
        }
      }
    ]
  }
}
```

ii. (オプション) アラームの状態が変化したときに IAM Identity Center ユーザーに送信する SMS 通知 (smsConfigurations) を設定し ます。

- *IDStoreIdAttributeId* を、IAM Identity Center アイデンティティストアの ID を含む属性の ID に置き換えます。
- *#### IdAttributeID* を、IAM Identity Center ユーザーの ID を含む属性の ID に置き換えます。
- *### IdAttributeID* を Amazon SNS 送信者 ID を含む属性の ID に置き換えるか、ペイロードsenderIdから削除します。
- *### MessageAttributeID* を、追加のメッセージを含む属性の ID に置き換えるか、ペイロードadditionalMessageから削除します。

```
{
  "alarmModelName": "BoilerTemperatureHighAlarm",
  "alarmModelDescription": "Detects when the boiler temperature is high.",
  "severity": 3,
  "alarmRule": {
    "simpleRule": {
      "inputProperty":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.propertyValue.value",
      "comparisonOperator": "GREATER",
      "threshold":
 "$sitewise.assetModel.`assetModelId`.`thresholdAttributeId`.propertyValue.value"
    }
  },
  "alarmEventActions": {
    "alarmActions": [
      {
        "iotSiteWise": {
          "assetId":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.assetId",
          "propertyId": "'alarmStatePropertyId'"
        }
```

```
      }
    ]
  },
  "alarmNotification": {
    "notificationActions": [
      {
        "action": {
          "lambdaAction": {
            "functionArn": "alarmNotificationFunctionArn"
          }
        },
        "smsConfigurations": [
          {
            "recipients": [
              {
                "ssoIdentity": {
                  "identityStoreId":
"$sitewise.assetModel.`assetModelId`.`identityStoreIdAttributeId`.propertyValue.va
                  "userId":
"$sitewise.assetModel.`assetModelId`.`userIdAttributeId`.propertyValue.value"
                }
              }
            ],
            "senderId":
"$sitewise.assetModel.`assetModelId`.`senderIdAttributeId`.propertyValue.value",
            "additionalMessage":
"$sitewise.assetModel.`assetModelId`.`additionalMessageAttributeId`.propertyValue.
          }
        ]
      }
    ]
  }
}
```

iii. (オプション) アラームの状態が変化したときに IAM Identity Center ユーザーに送信する E メール通知 (`emailConfigurations`) を設定します。

- *`IDStoreIdAttributeId`* を IAM Identity Center アイデンティティストア ID 属性プロパティの ID に置き換えます。
- *`user IdAttributeId`* を IAM Identity Center ユーザー ID 属性プロパティの ID に置き換えます。

- *from AddressAttributeId* を「from」アドレス属性プロパティの ID に置き換えるか、ペイロードfromから削除します。
- *email SubjectAttributeId* を email subject 属性プロパティの ID に置き換えるか、ペイロードsubjectから削除します。
- *### MessageAttributeID* を追加のメッセージ属性プロパティの ID に置き換えるか、ペイロードadditionalMessageから削除します。

```
{
  "alarmModelName": "BoilerTemperatureHighAlarm",
  "alarmModelDescription": "Detects when the boiler temperature is high.",
  "severity": 3,
  "alarmRule": {
    "simpleRule": {
      "inputProperty":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.propertyValue.value",
      "comparisonOperator": "GREATER",
      "threshold":
 "$sitewise.assetModel.`assetModelId`.`thresholdAttributeId`.propertyValue.value"
    }
  },
  "alarmEventActions": {
    "alarmActions": [
      {
        "iotSiteWise": {
          "assetId":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.assetId",
          "propertyId": "'alarmStatePropertyId'"
        }
      }
    ]
  },
  "alarmNotification": {
    "notificationActions": [
      {
        "action": {
          "lambdaAction": {
            "functionArn": "alarmNotificationFunctionArn"
          }
        },
        "smsConfigurations": [
          {
```

```
            "recipients": [
              {
                "ssoIdentity": {
                  "identityStoreId":
"$sitewise.assetModel.`assetModelId`.`identityStoreIdAttributeId`.propertyValue.va
                  "userId":
"$sitewise.assetModel.`assetModelId`.`userIdAttributeId`.propertyValue.value"
                }
              }
            ],
            "senderId":
"$sitewise.assetModel.`assetModelId`.`senderIdAttributeId`.propertyValue.value",
            "additionalMessage":
"$sitewise.assetModel.`assetModelId`.`additionalMessageAttributeId`.propertyValue.
          }
        ],
        "emailConfigurations": [
          {
            "from":
"$sitewise.assetModel.`assetModelId`.`fromAddressAttributeId`.propertyValue.value"
            "recipients": {
              "to": [
                {
                  "ssoIdentity": {
                    "identityStoreId":
"$sitewise.assetModel.`assetModelId`.`identityStoreIdAttributeId`.propertyValue.va
                    "userId":
"$sitewise.assetModel.`assetModelId`.`userIdAttributeId`.propertyValue.value"
                  }
                }
              ]
            },
            "content": {
              "subject":
"$sitewise.assetModel.`assetModelId`.`emailSubjectAttributeId`.propertyValue.value
              "additionalMessage":
"$sitewise.assetModel.`assetModelId`.`additionalMessageAttributeId`.propertyValue.
            }
          }
        ]
      }
    ]
  }
```

```
}
```

g. (オプション) alarm-model-payload.jsonのペイロードにアラーム機能 (alarmCapabilities) を追加する。このオブジェクトでは、アセットモデルに基づいて、フロー承認が有効かどうか、およびアセットに対するデフォルトの有効状態を指定することができます。フロー承認の詳細については、アラームの状態 を参照してください。

```
{
  "alarmModelName": "BoilerTemperatureHighAlarm",
  "alarmModelDescription": "Detects when the boiler temperature is high.",
  "severity": 3,
  "alarmRule": {
    "simpleRule": {
      "inputProperty":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.propertyValue.value",
      "comparisonOperator": "GREATER",
      "threshold":
 "$sitewise.assetModel.`assetModelId`.`thresholdAttributeId`.propertyValue.value"
    }
  },
  "alarmEventActions": {
    "alarmActions": [
      {
        "iotSiteWise": {
          "assetId":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.assetId",
          "propertyId": "'alarmStatePropertyId'"
        }
      }
    ]
  },
  "alarmNotification": {
    "notificationActions": [
      {
        "action": {
          "lambdaAction": {
            "functionArn": "alarmNotificationFunctionArn"
          }
        },
        "smsConfigurations": [
          {
            "recipients": [
              {
```

```
              "ssoIdentity": {
                "identityStoreId":
"$sitewise.assetModel.`assetModelId`.`identityStoreIdAttributeId`.propertyValue.value"
                "userId":
"$sitewise.assetModel.`assetModelId`.`userIdAttributeId`.propertyValue.value"
              }
            }
          ],
          "senderId":
"$sitewise.assetModel.`assetModelId`.`senderIdAttributeId`.propertyValue.value",
          "additionalMessage":
"$sitewise.assetModel.`assetModelId`.`additionalMessageAttributeId`.propertyValue.valu
        }
      ],
      "emailConfigurations": [
        {
          "from":
"$sitewise.assetModel.`assetModelId`.`fromAddressAttributeId`.propertyValue.value",
          "recipients": {
            "to": [
              {
                "ssoIdentity": {
                  "identityStoreId":
"$sitewise.assetModel.`assetModelId`.`identityStoreIdAttributeId`.propertyValue.value"
                  "userId":
"$sitewise.assetModel.`assetModelId`.`userIdAttributeId`.propertyValue.value"
                }
              }
            ]
          },
          "content": {
            "subject":
"$sitewise.assetModel.`assetModelId`.`emailSubjectAttributeId`.propertyValue.value",
            "additionalMessage":
"$sitewise.assetModel.`assetModelId`.`additionalMessageAttributeId`.propertyValue.valu
          }
        }
      ]
    }
  ]
 },
 "alarmCapabilities": {
   "initializationConfiguration": {
     "disabledOnInitialization": false
```

```
    },
    "acknowledgeFlow": {
      "enabled": true
    }
  }
}
```

h. にデータを送信するために が引き受け AWS IoT Events ることができる IAM サービスロール (roleArn) を追加します AWS IoT SiteWise。このロールを担うには、iotsitewise:BatchPutAssetPropertyValue の許可と、iotevents.amazonaws.com がそのロールを担うことができる信頼関係が必要です。通知を送信するには、このロールには lambda:InvokeFunction および sso-directory:DescribeUser のアクセス許可も必要です。詳細については、[AWS IoT Events Developer Guide] (デベロッパーガイド) の[Alarm service roles] (アラームサービスロール) を参照してください。

- を、これらのアクションを実行するために が引き受け AWS IoT Events ることができるロールの ARN roleArnに置き換えます。

```
{
  "alarmModelName": "BoilerTemperatureHighAlarm",
  "alarmModelDescription": "Detects when the boiler temperature is high.",
  "severity": 3,
  "alarmRule": {
    "simpleRule": {
      "inputProperty":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.propertyValue.value",
      "comparisonOperator": "GREATER",
      "threshold":
 "$sitewise.assetModel.`assetModelId`.`thresholdAttributeId`.propertyValue.value"
    }
  },
  "alarmEventActions": {
    "alarmActions": [
      {
        "iotSiteWise": {
          "assetId":
 "$sitewise.assetModel.`assetModelId`.`alarmPropertyId`.assetId",
          "propertyId": "'alarmStatePropertyId'"
        }
      }
```

```
    ]
  },
  "alarmNotification": {
    "notificationActions": [
      {
        "action": {
          "lambdaAction": {
            "functionArn": "alarmNotificationFunctionArn"
          }
        },
        "smsConfigurations": [
          {
            "recipients": [
              {
                "ssoIdentity": {
                  "identityStoreId":
"$sitewise.assetModel.`assetModelId`.`identityStoreIdAttributeId`.propertyValue.value"
                  "userId":
"$sitewise.assetModel.`assetModelId`.`userIdAttributeId`.propertyValue.value"
                }
              }
            ],
            "senderId":
"$sitewise.assetModel.`assetModelId`.`senderIdAttributeId`.propertyValue.value",
            "additionalMessage":
"$sitewise.assetModel.`assetModelId`.`additionalMessageAttributeId`.propertyValue.valu
          }
        ],
        "emailConfigurations": [
          {
            "from":
"$sitewise.assetModel.`assetModelId`.`fromAddressAttributeId`.propertyValue.value",
            "recipients": {
              "to": [
                {
                  "ssoIdentity": {
                    "identityStoreId":
"$sitewise.assetModel.`assetModelId`.`identityStoreIdAttributeId`.propertyValue.value"
                    "userId":
"$sitewise.assetModel.`assetModelId`.`userIdAttributeId`.propertyValue.value"
                  }
                }
              ]
            },
```

```
            "content": {
              "subject":
 "$sitewise.assetModel.`assetModelId`.`emailSubjectAttributeId`.propertyValue.value",
              "additionalMessage":
 "$sitewise.assetModel.`assetModelId`.`additionalMessageAttributeId`.propertyValue.valu
            }
          }
        ]
      }
    ]
  },
  "alarmCapabilities": {
    "initializationConfiguration": {
      "disabledOnInitialization": false
    },
    "acknowledgeFlow": {
      "enabled": false
    }
  },
  "roleArn": "arn:aws:iam::123456789012:role/MyIoTEventsAlarmRole"
}
```

i. 次のコマンドを実行して、 のペイロードから AWS IoT Events アラームモデルを作成しますalarm-model-payload.json。

```
aws iotevents create-alarm-model --cli-input-json file://alarm-model-
payload.json
```

j. この操作では、アラームモデルの ARN、alarmModelArn を含む応答が返されます。この ARN をコピーして、次のステップでアセットモデルのアラーム定義に設定します。

ステップ 3: AWS IoT SiteWise と の間のデータフローを有効にする AWS IoT Events

AWS IoT SiteWise および で必要なリソースを作成したら AWS IoT Events、リソース間のデータフローを有効にしてアラームを有効にできます。このセクションでは、前のステップで作成したアラームモデルを使用するために、アセットモデルのアラーム定義を更新します。

AWS IoT SiteWise と AWS IoT Events (CLI) 間のデータフローを有効にするには

- アセットモデルにアラームの出典としてアラームモデルを設定します。以下の操作を実行します。

a. 次のコマンドを実行して、既存のアセットモデル定義を取得します。*asset-model-id* をアセットモデルの ID に置き換えます。

```
aws iotsitewise describe-asset-model --asset-model-id asset-model-id
```

このオペレーションは、アセットモデルの詳細を含むレスポンスを返します。

b. update-asset-model-payload.json という名前のファイルを作成し、前のコマンドのレスポンスをファイルにコピーします。

c. update-asset-model-payload.json ファイルから次の key-value ペアを削除します。

- assetModelId
- assetModelArn
- assetModelCreationDate
- assetModelLastUpdateDate
- assetModelStatus

d. 先に定義したアラーム複合モデルに、アラーム出典プロパティ (AWS/ALARM_SOURCE) を追加します。alarm をアラームモデルの ARN *ModelArn*に置き換え、アラームソースプロパティの値を設定します。

```
{
  ...
  "assetModelCompositeModels": [
    ...
    {
      "name": "BoilerTemperatureHighAlarm",
      "type": "AWS/ALARM",
      "properties": [
        {
          "id": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
          "name": "AWS/ALARM_TYPE",
          "dataType": "STRING",
          "type": {
            "attribute": {
              "defaultValue": "IOT_EVENTS"
            }
          }
        },
        {
```

```
          "id": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
          "name": "AWS/ALARM_STATE",
          "dataType": "STRUCT",
          "dataTypeSpec": "AWS/ALARM_STATE",
          "type": {
            "measurement": {}
          }
        },
        {
          "name": "AWS/ALARM_SOURCE",
          "dataType": "STRING",
          "type": {
            "attribute": {
              "defaultValue": "alarmModelArn"
            }
          }
        }
      ]
    }
  ]
}
```

e. 次のコマンドを実行し、update-asset-model-payload.json ファイルに保存されている定義でアセットモデルを更新します。*asset-model-id* をアセットモデルの ID に置き換えます。

```
aws iotsitewise update-asset-model \
  --asset-model-id asset-model-id \
  --cli-input-json file://update-asset-model-payload.json
```

これで、アセットモデルには AWS IoT Eventsで検出するアラームが定義されました。このアセットモデルに基づくすべてのアセットにおいて、対象プロパティをモニタリングするアラームです。各アセットにアラームを設定することで、各アセットのしきい値や IAM Identity Center 受信者などのプロパティをカスタマイズすることができます。詳細については、「アセットにアラームを設定する。」を参照してください。

外部アラームの定義

外部アラームは、 AWS IoT SiteWise以外で検知したアラームの状態を含みます。

外部アラーム (コンソール) を定義する。

AWS IoT SiteWise コンソールを使用して、既存のアセットモデルに外部アラームを定義できます。新しいアセットモデルに外部アラームを定義するには、アセットモデルを作成し、次のステップを実行します。詳細については、「アセットモデルを作成する」を参照してください。

アセットモデルでアラームを定義するには。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[モデル] を選択します。
3. アラームを定義するアセットモデルを選択します。
4. [Alarm definitions] (アラーム定義) タブを選択します。
5. [Add alarm] (アラームの追加) を選択します。
6. [Alarm type options] (アラーム型オプション) で、[External alarm] (外部アラーム) を選択します。
7. アラームの名前を入力します。
8. (オプション) アラームの説明を入力します。
9. [Add alarm] (アラームの追加) を選択します。

外部アラームの定義 (CLI)

を使用して、新規または既存のアセットモデルに外部アラーム AWS CLI を定義できます。

アセットモデルに外部アラームを追加するには、アセットモデルにアラーム複合モデルを追加します。外部アラーム複合モデルは、EXTERNAL 型を指定し、アラーム出典プロパティを指定しない。次の複合アラームの例では、外部温度アラームを定義しています。

```
{
  ...
  "assetModelCompositeModels": [
    {
      "name": "BoilerTemperatureHighAlarm",
      "type": "AWS/ALARM",
      "properties": [
        {
          "name": "AWS/ALARM_TYPE",
          "dataType": "STRING",
```

```
          "type": {
            "attribute": {
              "defaultValue": "EXTERNAL"
            }
          }
        },
        {
          "name": "AWS/ALARM_STATE",
          "dataType": "STRUCT",
          "dataTypeSpec": "AWS/ALARM_STATE",
          "type": {
            "measurement": {}
          }
        }
      ]
    }
  ]
}
```

新規または既存のアセットモデルに複合モデルを追加する方法の詳細については、次を参照してください。

- アセットモデルの作成 (AWS CLI)
- アセットまたはコンポーネントモデルの更新 (AWS CLI)

外部アラームを定義した後、アセットモデルに基づいてアラームの状態をアセットに取り込むことができます。詳細については、「外部アラームの状態を取り込む。」を参照してください。

アセットにアラームを設定する。

アセットモデルに AWS IoT Events アラームを定義したら、アセットモデルに基づいて各アセットにアラームを設定できます。アラームのしきい値や通知設定を編集することができます。これらの値はそれぞれアセット上の属性であるため、属性のデフォルト値を更新することでこれらの値を設定することができます。

> **Note**
>
> これらの値は AWS IoT Events 、アラームには設定できますが、外部アラームには設定できません。

トピック

しきい値を設定する (コンソール)。

AWS IoT SiteWise コンソールを使用して、アラームのしきい値を指定する属性の値を更新できます。

アラームのしきい値を更新するには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. アラームしきい値を更新したいアセットを選択します。

> **Tip**
>
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [編集] を選択します。
5. アラームがしきい値に使用している属性を検索し、新しい値を入力します。
6. [保存] を選択します。

しきい値の設定 (AWS CLI)

AWS Command Line Interface （ AWS CLI) を使用して、アラームのしきい値を指定する属性の値を更新できます。

この手順を完了するには、アセットの `assetId` とプロパティの `propertyId` を知っている必要があります。外部 ID を使用することもできます。アセットを作成し、その がわからない場合は`assetId`、 ListAssets API を使用して特定のモデルのすべてのアセットを一覧表示します。DescribeAsset オペレーションを使用して、プロパティ IDs を含むアセットのプロパティを表示します。

BatchPutAssetProperty値 オペレーションを使用して、属性値をアセットに割り当てます。このオペレーションを使用すると、複数の属性を一度に設定できます。このオペレーションのペイロードにはエントリのリストが含まれ、各エントリにはアセット ID、プロパティ ID、属性値が含まれます。

属性の値を更新するには (AWS CLI)

1. batch-put-payload.json という名前のファイルを作成して、次の JSON オブジェクトをファイルにコピーします。このペイロードの例は、風力タービンの緯度と経度を設定する方法を示しています。ID、値、タイムスタンプを更新して、ユースケースのペイロードを変更します。

```
{
  "entries": [
    {
      "entryId": "windfarm3-turbine7-latitude",
      "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "propertyId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
      "propertyValues": [
        {
          "value": {
            "doubleValue": 47.6204
          },
          "timestamp": {
            "timeInSeconds": 1575691200
          }
        }
      ]
    },
    {
      "entryId": "windfarm3-turbine7-longitude",
      "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "propertyId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE",
      "propertyValues": [
        {
          "value": {
            "doubleValue": 122.3491
          },
          "timestamp": {
            "timeInSeconds": 1575691200
          }
        }
      ]
    }
  ]
```

```
}
```

- ペイロードの各エントリには、一意の文字列として定義できる entryId が含まれています。リクエストのエントリが失敗した場合、各エラーには、対応するリクエストの entryId が含まれるため、再試行するリクエストを確認できます。
- 属性値を設定するには、各属性プロパティpropertyValuesの のリストに 1 つの timestamp-quality-value (TQV) 構造を含めることができます。この構造には、新しい value および現在の timestamp を含める必要があります。
  - value - 設定中のプロパティの型に応じて、次のいずれかのフィールドを含む構造。
    - booleanValue
    - doubleValue
    - integerValue
    - stringValue
  - timestamp – 現在の Unix エポック時間を秒単位で含む構造timeInSeconds。. AWS IoT SiteWise は、過去 7 日以上または今後 5 分より新しいタイムスタンプを持つデータポイントを拒否します。

BatchPutAssetProperty値 のペイロードを準備する方法の詳細については、「」を参照してくださいAPI を使用してデータを取り込む AWS IoT SiteWise。

2. 次のコマンドを実行して、属性値を に送信します AWS IoT SiteWise。

```
aws iotsitewise batch-put-asset-property-value -\-cli-input-json file://batch-put-payload.json
```

通知設定を行う (コンソール)。

AWS IoT SiteWise コンソールを使用して、アラームの通知設定を指定する属性の値を更新できます。

アラームの通知設定を更新するには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [アセット] を選択します。
3. アラーム設定を更新するアセットを選択します。

4. [編集] を選択します。
5. 変更したい通知設定にアラームが使用している属性を探し、その新しい値を入力します。
6. [保存] を選択します。

通知設定を行う (CLI)。

AWS Command Line Interface （ AWS CLI) を使用して、アラームの通知設定を指定する属性の値を更新できます。

この手順を完了するには、アセットの `assetId` とプロパティの `propertyId` を知っている必要があります。外部 ID を使用することもできます。アセットを作成し、その がわからない場合は`assetId`、 ListAssets API を使用して特定のモデルのすべてのアセットを一覧表示します。DescribeAsset オペレーションを使用して、プロパティ IDs を含むアセットのプロパティを表示します。

BatchPutAssetProperty値 オペレーションを使用して、属性値をアセットに割り当てます。このオペレーションを使用すると、複数の属性を一度に設定できます。このオペレーションのペイロードにはエントリのリストが含まれ、各エントリにはアセット ID、プロパティ ID、属性値が含まれます。

属性の値を更新するには (AWS CLI)

1. `batch-put-payload.json` という名前のファイルを作成して、次の JSON オブジェクトをファイルにコピーします。このペイロードの例は、風力タービンの緯度と経度を設定する方法を示しています。ID、値、タイムスタンプを更新して、ユースケースのペイロードを変更します。

```
{
  "entries": [
    {
      "entryId": "windfarm3-turbine7-latitude",
      "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "propertyId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
      "propertyValues": [
        {
          "value": {
            "doubleValue": 47.6204
          },
          "timestamp": {
            "timeInSeconds": 1575691200
          }
        }
```

```
      ]
    },
    {
      "entryId": "windfarm3-turbine7-longitude",
      "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
      "propertyId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE",
      "propertyValues": [
        {
          "value": {
            "doubleValue": 122.3491
          },
          "timestamp": {
            "timeInSeconds": 1575691200
          }
        }
      ]
    }
  ]
}
```

- ペイロードの各エントリには、一意の文字列として定義できる `entryId` が含まれています。リクエストのエントリが失敗した場合、各エラーには、対応するリクエストの `entryId` が含まれるため、再試行するリクエストを確認できます。
- 属性値を設定するには、各属性プロパティ`propertyValues`の のリストに 1 つの timestamp-quality-value (TQV) 構造を含めることができます。この構造には、新しい `value` および現在の `timestamp` を含める必要があります。
  - `value` - 設定中のプロパティの型に応じて、次のいずれかのフィールドを含む構造。
    - `booleanValue`
    - `doubleValue`
    - `integerValue`
    - `stringValue`
  - `timestamp` – 現在の Unix エポック時間を秒単位で含む構造`timeInSeconds`。. AWS IoT SiteWise は、過去 7 日以上または今後 5 分より新しいタイムスタンプを持つデータポイントを拒否します。

BatchPutAssetProperty値 のペイロードを準備する方法の詳細については、「」を参照してくださいAPI を使用してデータを取り込む AWS IoT SiteWise。

2. 次のコマンドを実行して、属性値を に送信します AWS IoT SiteWise。

```
aws iotsitewise batch-put-asset-property-value -\-cli-input-json file://batch-put-payload.json
```

アラームへの対応。

AWS IoT Events アラームの状態が変わったら、次の操作を実行してアラームに応答できます。

- アラームを確認することで、問題に対処していることを示すことができます。
- アラームをスヌーズして、一時的に無効にします。
- アラームを無効にすると、再度有効にするまで永久にアラームを無効にすることができます。
- 無効になっているアラームを有効にして、アラーム状態を検出する。
- アラームをリセットすると、アラームの状態や最新の値がクリアされます。

AWS IoT SiteWise コンソールまたは AWS IoT Events API を使用してアラームに応答できます。

> **Note**
>
> AWS IoT Events アラームには応答できますが、外部アラームには応答できません。

トピック

- アラームに対応する (コンソール)。
- アラームに対応する (API)。

アラームに対応する (コンソール)。

AWS IoT SiteWise コンソールを使用して、アラームの確認、スヌーズ、無効化、または有効化を行うことができます。

トピック

- アラームを承認する (コンソール)。
- アラームをスヌーズする (コンソール)。

- アラームを無効にする (コンソール)。
- アラームを有効にする (コンソール)。
- アラームをリセットする (コンソール)。

アラームを承認する (コンソール)。

問題を処理していることを示すために、アラームを承認することができます。

> **Note**
>
> アラームを確認できるように、アラームのフロー承認を有効にする必要があります。このオプションは、 AWS IoT SiteWise コンソールからアラームを定義した場合、デフォルトで有効になります。

アラームを確認するには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. アラームを確認したいアセットを選択します。

   > **Tip**
   >
   > 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [Alarms] (アラーム) を選択します。
5. 確認するアラームを選択し、[Actions] (アクション) を選択して応答アクションメニューを表示します。
6. [Acknowledge] (承認) を選択します。アラームの状態が [Acknowledged] (承認済み) に変化します。

アラームをスヌーズする (コンソール)。

アラームをスヌーズして、一時的にアラームを無効にすることができます。アラームをスヌーズする時間を指定します。

アラームをスヌーズするには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. アラームをスヌーズさせたいアセットを選択します。

> **Tip**
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [Alarms] (アラーム) を選択します。
5. スヌーズするアラームを選択し、[Actions] (アクション) を選択すると、対応アクションメニューが表示されます。
6. [Snooze] (スヌーズ) を選択します。スヌーズする時間を指定するモデルが開きます。
7. [Snooze length] (スヌーズの長さ) を選択するか、 [Custom snooze length] (スヌーズの長さをカスタマイズする) を入力します。
8. [保存] を選択します。アラームの状態が [Snoozed] (スヌーズ済み) に切り替わります。

アラームを無効にする (コンソール)。

アラームを無効にして、検出しないようにすることができます。アラームを無効にした後、アラームを検出させたい場合は、再度有効にする必要があります。

アラームを無効にするには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. アラームを無効にするアセットを選択します。

> **Tip**
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [Alarms] (アラーム) を選択します。
5. 無効にするアラームを選択し、[Actions] (アクション) を選択してレスポンスアクションメニューを表示します。

6. [Disable (無効化)] を選択します。アラームの状態が[Disabled] (無効) に変化します。

アラームを有効にする (コンソール)。

アラームを無効にしたり、スヌーズした後、再び検出するように有効化することができます。

アラームを有効にするには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. アラームを有効にするアセットを選択します。

> **Tip**
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [Alarms] (アラーム) を選択します。
5. 有効にするアラームを選択し、[Actions] (アクション) を選択してレスポンスアクションメニューを表示します。
6. [Enable (有効化)] を選択します。アラームの状態が[Normal] (正常) に変化します。

アラームをリセットする (コンソール)。

アラームをリセットして、その状態や最新の値をクリアすることができます。

アラームをリセットするには (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. アラームをリセットするアセットを選択します。

> **Tip**
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [Alarms] (アラーム) を選択します。

5. 有効にするアラームを選択し、[Actions] (アクション) を選択してレスポンスアクションメニューを表示します。
6. [リセット] を選択します。アラームの状態が[Normal] (正常) に変化します。

アラームに対応する (API)。

AWS IoT Events API を使用して、アラームの確認、スヌーズ、無効化、有効化、リセットを行うことができます。詳細は、[AWS IoT Events API Reference] (API リファレンス) の次の操作を参照してください。

- BatchAcknowledgeアラーム
- BatchSnoozeアラーム
- BatchDisableアラーム
- BatchEnableアラーム
- BatchResetアラーム

詳細については、[AWS IoT Events Developer Guide] (デベロッパーガイド) の[Responding to alarms] (アラームへの対応) を参照してください。

外部アラームの状態を取り込む。

外部アラームは、 の外部で評価するアラームです AWS IoT SiteWise。アラームの状態を報告するデータソースがあり、それを AWS IoT SiteWiseに取り込みたい場合に外部アラームを使用することができます。

アラーム状態プロパティは、アラーム状態データ値に特定のフォーマットを要求します。各データ値は、文字列にシリアライズされた JSON オブジェクトである必要があります。そして、シリアライズされた文字列を文字列の値として取り込みます。詳細については、「アラーム状態のプロパティ」を参照してください。

Example アラーム状態データ値例 (シリアル化されていない)

```
{
  "stateName": "Active"
}
```

Example アラーム状態データ値の例 (シリアル化)

```
{\"stateName\":\"Active\"}
```

> **Note**
>
> データソースがこのフォーマットでデータをレポートできない場合、またはデータを取り込む前にこのフォーマットに変換できない場合、アラームプロパティを使用しないことを選択することができます。その代わり、例えば文字列データ型を持つ計測プロパティとしてデータを取り込むことができます。詳細については、「機器からのデータストリームの定義 (測定値)」および「へのデータの取り込み AWS IoT SiteWise」を参照してください。

外部アラーム状態ストリームのマッピング。

データストリームをアラーム状態のプロパティにマッピングするために、プロパティのエイリアスを定義することができます。これにより、データの取り込みや取得の際に、アラーム状態のプロパティを簡単に特定することができます。プロパティのエイリアスについての詳細は、アセットプロパティへの産業データストリームのマッピング を参照してください。

トピック

- 外部アラーム状態ストリームをマッピングする (コンソール)。
- 外部アラーム状態ストリームのマッピング (AWS CLI)

外部アラーム状態ストリームをマッピングする (コンソール)。

データストリームをアラーム状態のプロパティにマッピングするために、プロパティのエイリアスを定義することができます。これにより、データの取り込みや取得の際に、アラーム状態のプロパティを簡単に特定することができます。プロパティのエイリアスについての詳細は、アセットプロパティへの産業データストリームのマッピング を参照してください。

AWS IoT SiteWise コンソールを使用して、アラーム状態プロパティのエイリアスを設定できます。

アラーム状態プロパティにプロパティエイリアスを設定する方法 (コンソール)。

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。

3. プロパティエイリアスを設定するアセットを選択します。

> **Tip**
>
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [編集] を選択します。
5. アラームまでスクロールし、 セクションを展開します。
6. 外部アラーム で、プロパティエイリアス - オプション にエイリアスを入力します。
7. [保存] を選択します。

外部アラーム状態ストリームのマッピング (AWS CLI)

データストリームをアラーム状態のプロパティにマッピングするために、プロパティのエイリアスを定義することができます。これにより、データの取り込みや取得の際に、アラーム状態のプロパティを簡単に特定することができます。プロパティのエイリアスについての詳細は、アセットプロパティへの産業データストリームのマッピング を参照してください。

AWS Command Line Interface （ AWS CLI) を使用して、アラーム状態プロパティのエイリアスを設定できます。

この手順を完了するには、アセットの `assetId` とプロパティの `propertyId` を知っている必要があります。外部 ID を使用することもできます。アセットを作成し、その がわからない場合は`assetId`、 ListAssets API を使用して特定のモデルのすべてのアセットを一覧表示します。DescribeAsset オペレーションを使用して、プロパティ IDs を含むアセットのプロパティを表示します。

> **Note**
>
> DescribeAsset レスポンスには、アセットの複合アセットモデルのリストが含まれます。各アラームは複合モデルです。`propertyId` を求めるには、アラームの合成モデルを見つけ、その合成モデルの中で `AWS/ALARM_STATE` のプロパティを見つけます。

プロパティのエイリアスの設定方法については、プロパティエイリアスの設定 (AWS CLI) を参照してください。

アラーム状態データの取り込み。

アラーム状態プロパティは、アラーム状態をシリアル化された JSON 文字列として受け取る。でアラーム状態を外部アラームに取り込むには AWS IoT SiteWise、このシリアル化された文字列をタイムスタンプ付き文字列値として取り込みます。次の例は、アクティブなアラームの状態データの値を示しています。

```
{\"stateName\":\"Active\"}
```

アラーム状態プロパティを特定するために、次のいずれかを指定することができます。

- データデスティネーションのアラームプロパティの `assetId`、 `propertyId` です。
- データストリームのエイリアスである `propertyAlias` (例えば、`/company/windfarm/3/turbine/7/temperature/high`)。このオプションを使用するには、まず、アラームプロパティのエイリアスを設定する必要があります。アラーム状態のプロパティにプロパティエイリアスを設定する方法については、外部アラーム状態ストリームのマッピング。 を参照してください。

次の BatchPutAssetPropertyValue API ペイロードの例は、外部アラームの状態をフォーマットする方法を示しています。風力タービンの1分間あたりの回転数 (RPM) が高すぎる場合に報告される外部アラームです。

Example アラーム状態データの BatchPutAssetPropertyValue ペイロードの例

```
{
    "entries": [
      {
        "entryId": "unique entry ID",
        "propertyAlias": "/company/windfarm/3/turbine/7/temperature/high",
        "propertyValues": [
          {
            "value": {
              "stringValue": "{\"stateName\":\"Active\"}"
            },
            "timestamp": {
              "timeInSeconds": 1607550262
            }
          }
        ]
      }
    ]
```

```
  }
```

`BatchPutAssetPropertyValue` API を使ったデータ取り込みの方法については、API を使用してデータを取り込む AWS IoT SiteWise を参照してください。

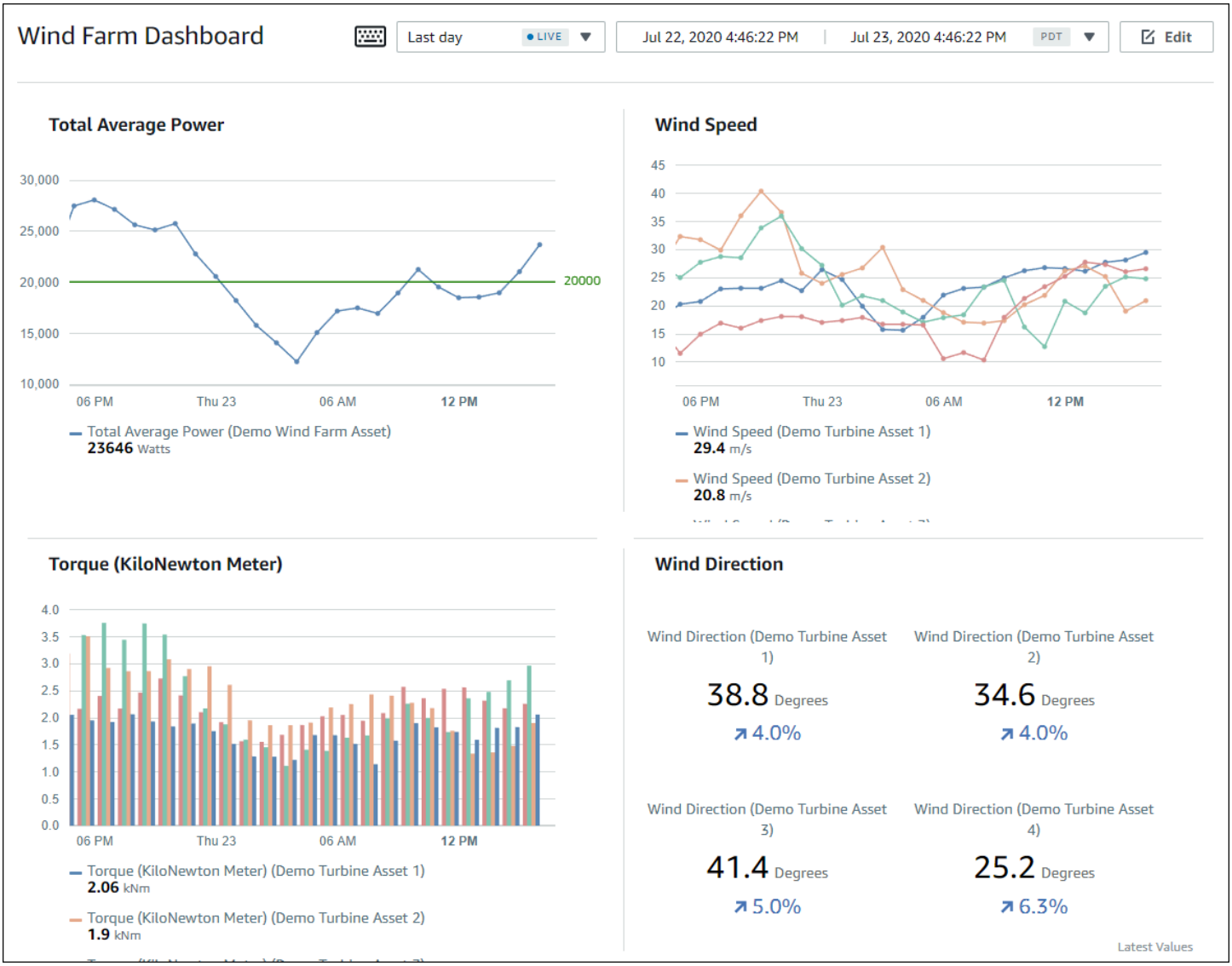
その他のデータ取り込み方法については、へのデータの取り込み AWS IoT SiteWise を参照してください。

によるデータの監視 AWS IoT SiteWise Monitor

Monitor Web ポータルを作成することで、プロセス、デバイス、 SiteWise 機器からのデータを監視できます。 AWS IoT SiteWise SiteWise Monitor は、マネージド Web アプリケーションの形式でポータルを作成するために使用できる機能です。 AWS IoT SiteWise その後、これらのポータルを使用して、運用データを表示および共有できます。ダッシュボードを使用してプロジェクトを作成し、AWS IoTに接続されているプロセス、デバイス、機器からのデータを可視化できます。

プロセスエンジニアのようなドメインエキスパートは、ポータルを使用して運用データに関するインサイトを迅速に取得し、デバイスや機器の動作を把握できます。

以下は、風力発電所のデータを表示するダッシュボードの例です。



SiteWise Monitor AWS IoT SiteWise ではデータを時系列的にキャプチャするので、時間の経過に伴う運用データや、特定の時点で最後に報告された値を表示できます。これにより、通常であれば見つけることが難しいインサイトを発見できます。

SiteWise モニターの役割

SiteWise モニターと相互作用する役割は次の 4 つです。

AWS 管理者

AWS AWS IoT SiteWise 管理者はコンソールを使用してポータルを作成します。 AWS 管理者は、ポータル管理者を割り当て、ポータルユーザーを追加することもできます。ポータル管理者は、後でポータルユーザーを所有者または閲覧者としてプロジェクトに割り当てます。 AWS AWS 管理者はコンソールでのみ作業します。

ポータル管理者

各 SiteWise Monitor ポータルには 1 人以上のポータル管理者がいます。ポータル管理者は、ポータルを使用して、アセットとダッシュボードのコレクションを含むプロジェクトを作成します。その後で、ポータル管理者はアセットと所有者を各プロジェクトに割り当てます。プロジェクトへのアクセスを制御することにより、ポータル管理者は、プロジェクトの所有者とビューワーが参照できるアセットを指定します。

プロジェクトの所有者

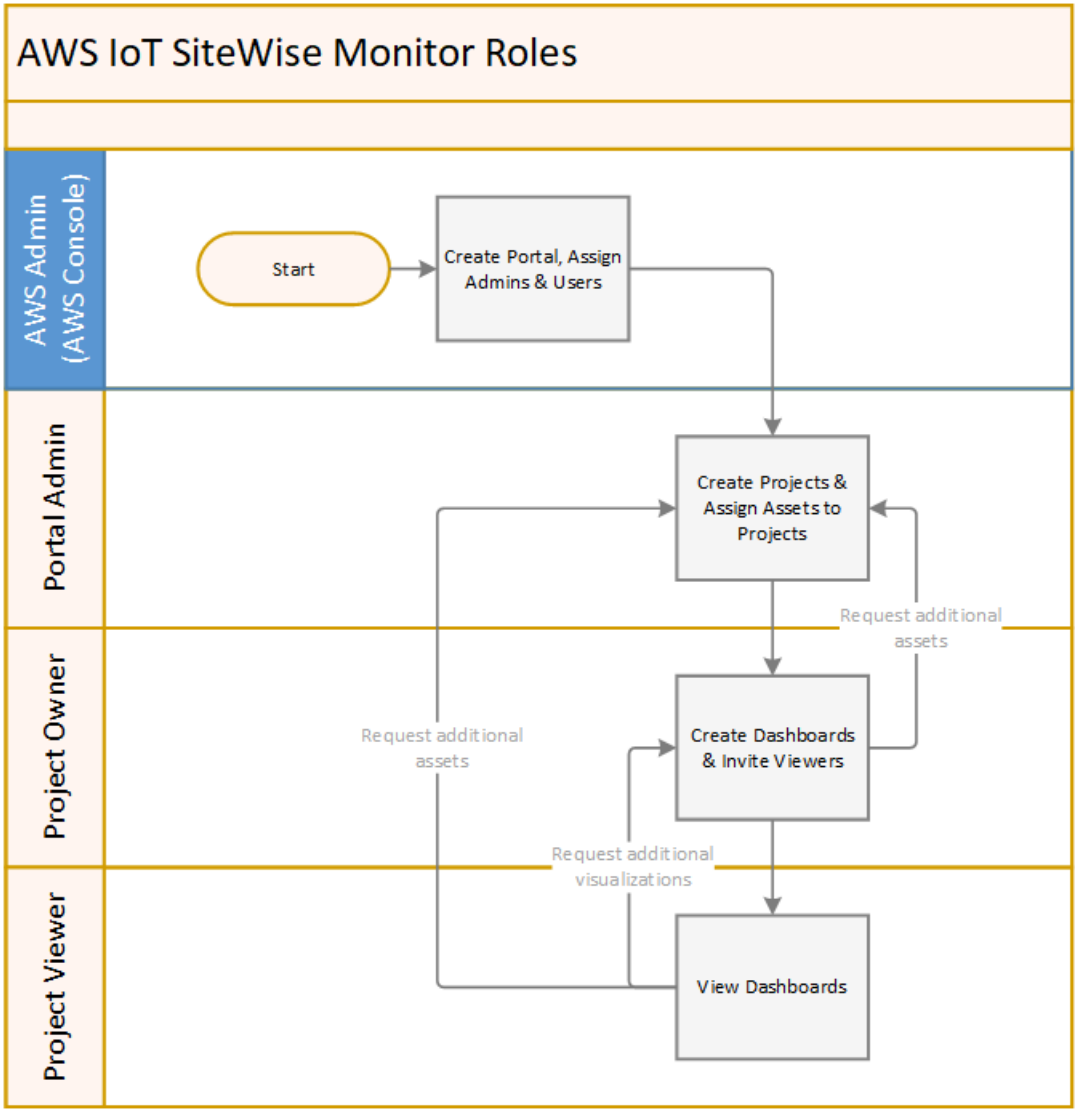
各 SiteWise Monitor プロジェクトには所有者がいます。プロジェクトの所有者は、一貫した方法で運用データを表すために、ダッシュボードの形式で可視化を作成します。ダッシュボードを共有する準備ができたら、プロジェクト所有者はビューワーをプロジェクトに招待できます。プロジェクトの所有者は、他の所有者もプロジェクトに割り当てることができます。プロジェクト所有者は、アラームのしきい値や通知設定を設定することができます。

プロジェクトビューワー

各 SiteWise Monitor プロジェクトには視聴者がいます。プロジェクト閲覧者は、ポータルに接続して、プロジェクト所有者が作成したダッシュボードを表示できます。各ダッシュボードでは、プロジェクトビューワーが時間範囲を調整することで、運用データをより深く理解することができます。プロジェクト閲覧者は、アクセス権のあるプロジェクトのダッシュボードのみを表示できます。プロジェクトビューワーは、アラームを確認したり、スヌーズさせたりすることができます。

組織によっては、同じユーザーが複数のロールを実行することがあります。

以下の画像は、これら 4 SiteWise つのロールがモニターポータルでどのように相互作用するかを示しています。



[AWS IAM Identity Center or] (または) IAM を使用して、データにアクセスできるユーザーを管理できます。データユーザーは、IAM Identity Center または IAM 認証情報を使用して、デスクトップまたはモバイルブラウザーから SiteWise Monitor にサインインできます。

SAML フェデレーション

IAM Identity Center および IAM は、SAML (Security Assertion Markup Language) 2.0 によるID フェデレーションをサポートしています。SAML 2.0 は、多くの外部 ID プロバイダー (IdPs) がユーザーを認証し、ID とセキュリティ情報をサービスプロバイダー (SP) に渡すために使用しているオープンスタンダードです。SP は通常、アプリケーションまたはサービスです。SAML フェデレーションに

より、 SiteWise Monitor ポータルの管理者とユーザーは、会社のユーザー名やパスワードなどの外部認証情報を使用して、割り当てられたポータルにサインインできます。

モニターポータルへのアクセスに SAML ベースのフェデレーションを使用するように IAM Identity Center と IAM を設定できます。 SiteWise

IAM アイデンティティセンター

ポータル管理者とユーザーは、 AWS 会社のユーザー名とパスワードを使用してアクセスポータルにサインインできます。その後、 SiteWise 割り当てられた監視ポータルに移動できます。IAM Identity Center は、証明書を使用して ID プロバイダーとの SAML 信頼関係を設定します。 AWS 詳細については、AWS IAM Identity Center ユーザーガイドの「SCIM プロファイルと SAML 2.0 の実装」を参照してください。

IAM

ポータルの管理者とユーザーは、 SiteWise 割り当てられたモニターポータルにアクセスするための一時的なセキュリティ認証情報をリクエストできます。IAM で SAML ID プロバイダー ID を作成して、ID プロバイダーとの間に信頼関係を設定します。 AWS詳細については、『IAM ユーザーガイド』の「API アクセスに SAML ベースのフェデレーションを使用する」を参照してください。 AWS

ポータルの管理者とユーザーは、会社のポータルにサインインして、管理コンソールに移動するオプションを選択できます。 AWS その後、 SiteWise 割り当てられた監視ポータルに移動できます。ID AWSプロバイダーとの間の信頼の交換は、会社のポータルが行います。詳細については、IAM ユーザーガイドの「SAML 2.0 AWS フェデレーティッドユーザーが管理コンソールにアクセスできるようにする」を参照してください。

> **Note**
>
> ポータルへのユーザーまたは管理者の追加では、IP の制限などのユーザー許可を制限する IAM ポリシーを作成しないようにしてください。権限が制限された添付ポリシーは、ポータルに接続できません。 AWS IoT SiteWise

SiteWise モニターのコンセプト

SiteWise Monitor を使用するには、以下の概念を理解している必要があります。

Portal

AWS IoT SiteWise Monitor ポータルは、 AWS IoT SiteWise データを視覚化して共有できる Web アプリケーションです。ポータルには 1 人以上の管理者があり、0 個以上のプロジェクトが含まれています。

プロジェクト

SiteWise 各モニターポータルには一連のプロジェクトが含まれています。各プロジェクトには、AWS IoT SiteWise アセットと関連付けられたのサブセットがあります。プロジェクト所有者は、1 つ以上のダッシュボードを作成して、それらのアセットに関連付けられたデータの整合性ある表示方法を提供します。プロジェクトの所有者は、プロジェクトに閲覧者を招待して、プロジェクト内のアセットとダッシュボードを表示できるようにすることができます。 SiteWise プロジェクトはモニター内での共有の基本単位です。プロジェクトオーナーは、 AWS 管理者からポータルへのアクセス権を与えられたユーザーを招待できます。ユーザーは、そのポータル内のプロジェクトをそのユーザーと共有する前に、ポータルへのアクセス権を持っている必要があります。

アセット

AWS IoT SiteWise 産業機器からデータを取り込むと、デバイス、機器、プロセスはそれぞれ資産として扱われます。各アセットには、関連付けられたプロパティとアラームがあります。ポータル管理者は、各プロジェクトにアセットのセットを割り当てます。

プロパティ

プロパティは、アセットに関連する時系列データです。たとえば、機器には、シリアル番号、ロケーション、製造元とモデル、およびインストール日があります。また、可用性、性能、品質、温度、圧力などの時系列値を持つ場合もあります。

アラーム

アラームは、機器の動作範囲外を識別するために、プロパティをモニタリングします。各アラームは、しきい値とモニタリングするプロパティを定義します。プロパティがしきい値を超えると、アラームがアクティブになり、お客様やお客様のチームの誰かが問題に対処する必要があることを示します。プロジェクト所有者は、アラームのしきい値や通知設定をカスタマイズすることができます。プロジェクトビューワーはアラームを確認したり、スヌーズしたりすることができ、アラームの詳細や対処方法をメッセージとして残すことができます。

ダッシュボード

各プロジェクトには、ダッシュボードのセットが含まれています。ダッシュボードは、一連のアセットの値に対する一連の可視化を提供します。プロジェクトの所有者は、ダッシュボードと

それに含まれる可視化を作成します。プロジェクト所有者がダッシュボードのセットを共有する準備ができたら、所有者はプロジェクトに閲覧者を招待し、プロジェクト内のすべてのダッシュボードにアクセスできるようにします。ダッシュボードごとに異なる閲覧者のセットが必要な場合は、プロジェクト間でダッシュボードを分割する必要があります。閲覧者はダッシュボードを見るときに、特定のデータを見るように時間範囲をカスタマイズできます。

視覚化

各ダッシュボードでは、プロジェクトに関連するアセットのプロパティとアラームをどのように表示するかをプロジェクト所有者が決定します。可用性は折れ線グラフで、その他の数値は棒グラフや重要業績評価指標 (KPI、 Key Performance Indicator) で表示されるかもしれません。アラームは、状態グリッドと状態タイムラインでの表示が最適です。プロジェクトの所有者は、それぞれの可視化をカスタマイズして、そのアセットのデータを最もよく理解できるようにしています。

はじめに AWS IoT SiteWise Monitor

AWS 組織の管理者の場合は、 AWS IoT SiteWise コンソールからポータルを作成します。 AWS IoT SiteWise 組織のメンバーがデータを閲覧できるようにポータルを作成するには、次の手順を実行します。

1. ポータルを設定して作成する
2. ポータル管理者の追加と招待メールの送信
3. ポータルユーザーの追加

ポータルを作成すると、 AWS IoT SiteWise ポータル管理者はアセットを表示し、ポータル内のプロジェクトに割り当てることができます。プロジェクト所有者は、ダッシュボードを作成してアセットのプロパティを可視化できます。これは、プロジェクトビューワーがデバイス、プロセス、および機器のパフォーマンスを理解するのに役立ちます。

> **Note**
>
> ユーザーまたは管理者をポータルに追加するときは、制限付き IP などのユーザー権限を制限する AWS Identity and Access Management (IAM) ポリシーを作成しないでください。権限が制限された添付ポリシーは、 AWS IoT SiteWise ポータルに接続できません。

チュートリアルでは、風力発電施設データを使用して、特定のシナリオのプロジェクト、ダッシュボード、複数のユーザーを含むポータルをセットアップするために必要な手順を説明します。詳細については、「Monitor SiteWise での風力発電所データの視覚化と共有」を参照してください。

トピック

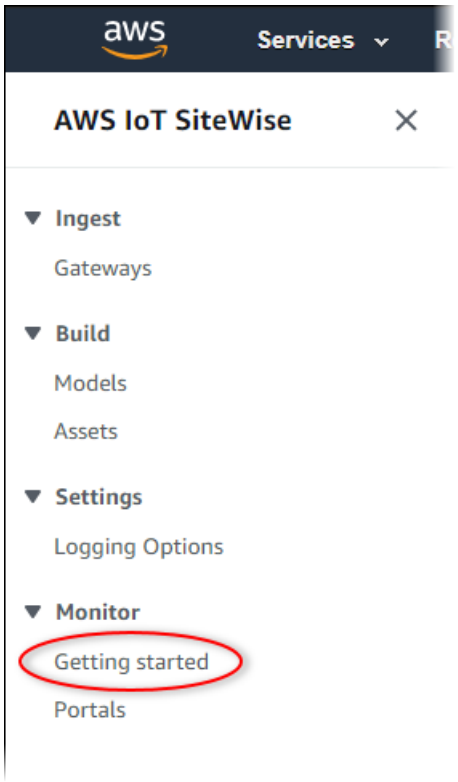
- ポータルの作成
- ポータルの設定
- 管理者の招待
- ポータルユーザーを追加する

ポータルの作成

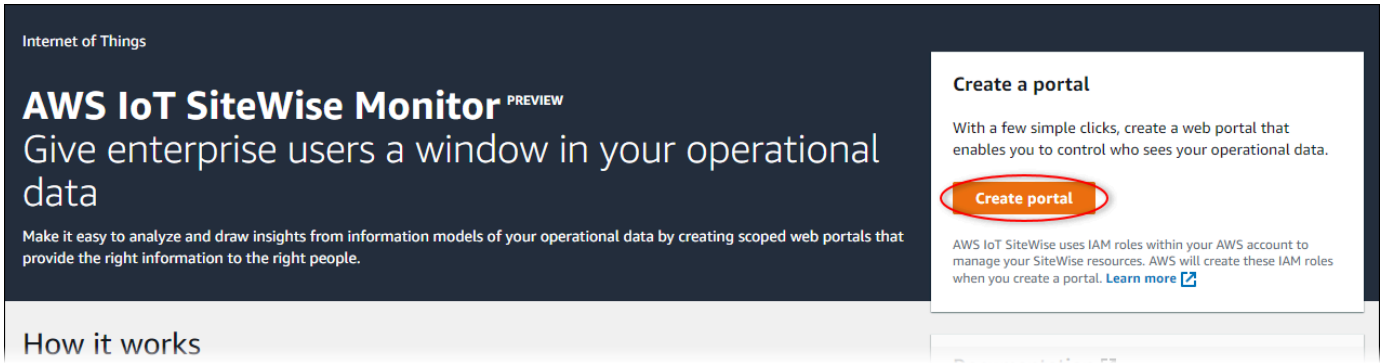
SiteWise AWS IoT SiteWise モニターポータルはコンソールで作成します。

ポータルを作成するには

1. AWS IoT SiteWise コンソールにサインインします。
2. ナビゲーションペインで、[モニタリング]、[開始方法] の順に選択します。



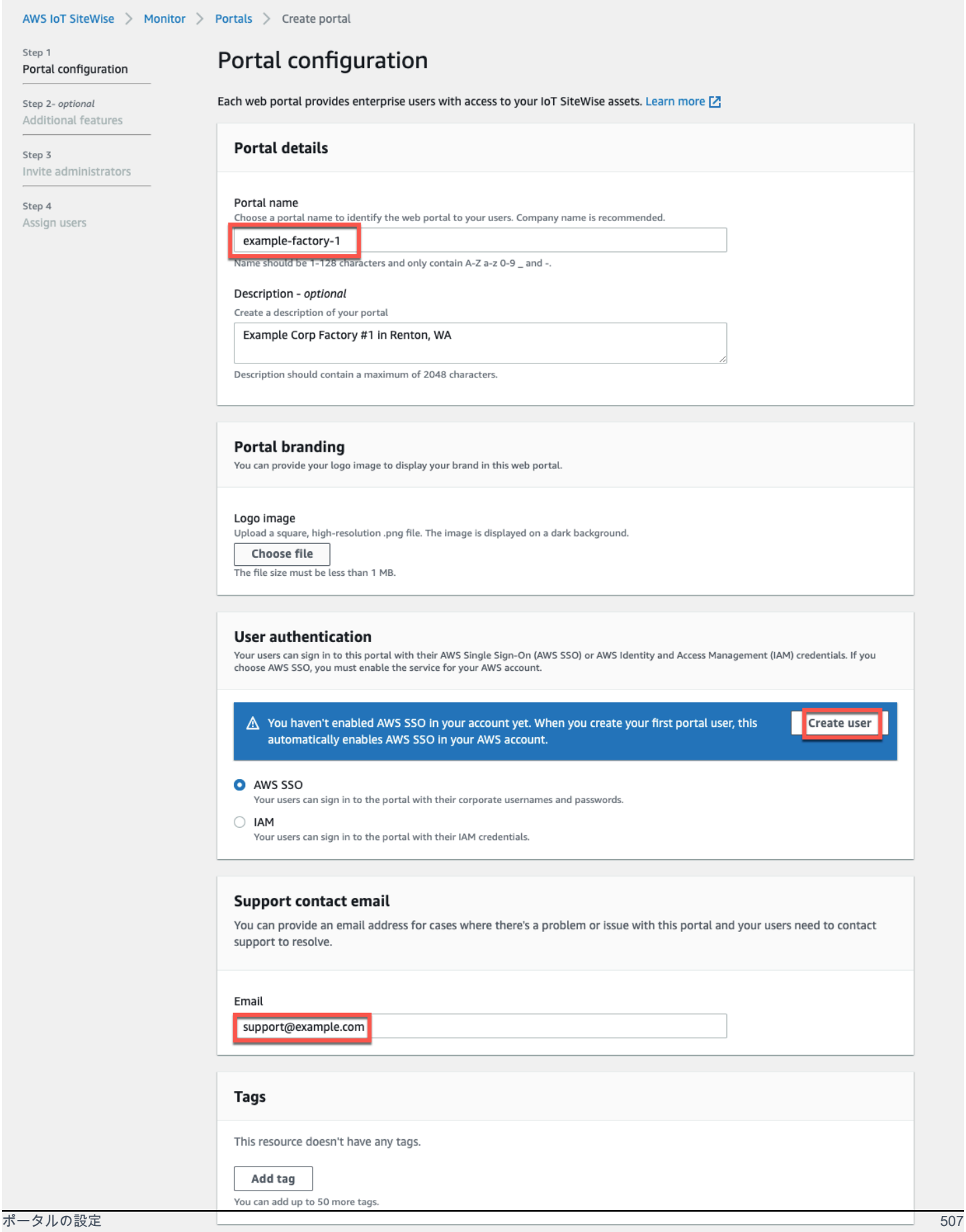
3. [ポータルの作成] を選択します。



次に、ポータルを設定するために基本情報を指定する必要があります。

ポータルの設定

ユーザーはポータルを使用してデータを表示します。ポータルの名前、説明、ブランディング、ユーザー認証、サポートお問い合わせメール、アクセス許可などをカスタマイズできます。


AWS IoT SiteWise > Monitor > Portals > Create portal
Step 1
Portal configuration
Step 2- optional
Additional features
Step 3
Invite administrators
Step 4
Assign users
Portal configuration
Each web portal provides enterprise users with access to your IoT SiteWise assets. Learn more
Portal details
Portal name
Choose a portal name to identify the web portal to your users. Company name is recommended.
example-factory-1
Name should be 1-128 characters and only contain A-Z a-z 0-9 _ and -.
Description - optional
Create a description of your portal
Example Corp Factory #1 in Renton, WA
Description should contain a maximum of 2048 characters.
Portal branding
You can provide your logo image to display your brand in this web portal.
Logo image
Upload a square, high-resolution .png file. The image is displayed on a dark background.
Choose file
The file size must be less than 1 MB.
User authentication
Your users can sign in to this portal with their AWS Single Sign-On (AWS SSO) or AWS Identity and Access Management (IAM) credentials. If you choose AWS SSO, you must enable the service for your AWS account.
You haven't enabled AWS SSO in your account yet. When you create your first portal user, this automatically enables AWS SSO in your AWS account.
Create user
AWS SSO
Your users can sign in to the portal with their corporate usernames and passwords.
IAM
Your users can sign in to the portal with their IAM credentials.
Support contact email
You can provide an email address for cases where there's a problem or issue with this portal and your users need to contact support to resolve.
Email
support@example.com
Tags
This resource doesn't have any tags.
Add tag
You can add up to 50 more tags.
Permissions
SiteWise Monitor assumes this role to give permissions to your federated users to access AWS IoT SiteWise resources. Learn

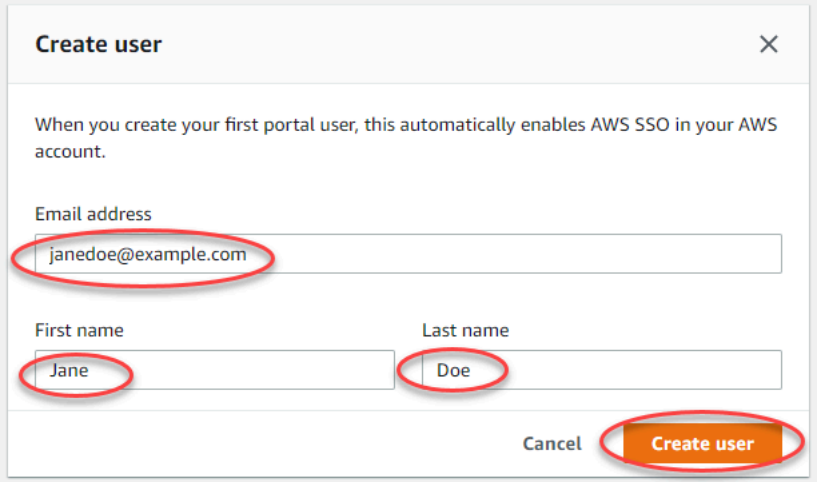
ポータルを設定するには

1. ポータルの名前を入力します。
2. (オプション) ポータルの説明を入力します。複数のポータルがある場合は、わかりやすい説明を使用すると、各ポータルに含まれる内容を追跡できます。
3. (オプション) 画像をアップロードして、ブランドをポータルに表示します。正方形の PNG 画像を選択します。正方形でない画像をアップロードすると、画像は正方形に縮小されます。
4. 以下のオプションのいずれかを選択します。
   - ポータルのユーザーが会社のユーザー名とパスワードでこのポータルにサインインする場合は、[IAM Identity Center] を選択します。

     アカウントで IAM Identity Center を有効にしていない場合は、次のことを行います。

     a. [ユーザーの作成] を選択します。

     b. [ユーザーの作成] ページで、最初のポータルを作成するために、ユーザーの E メールアドレス、名前、姓を入力し、[ユーザーの作成] を選択します。



> **Note**
>
> - AWS 最初のポータルユーザーを作成すると、アカウントの IAM Identity Center が自動的に有効になります。
> - IAM Identity Center は、一度に 1 つのリージョンでしか設定できません。SiteWise モニターは IAM ID センター用に設定したリージョンに接続します。つまり、IAM Identity Center へのアクセスには 1 つのリージョンを使用しますが、どのリージョンにもポータルを作成することができます。

- ポータルのユーザーが IAM の認証情報を使ってこのポータルにサインインする場合は、[IAM] を選択します。

> **Important**
>
> ユーザーまたはロールがこのポータルにサインインするには `iotsitewise:DescribePortal` アクセス許可が必要です。

5. ポータルに問題があり、解決するためにサポートが必要な場合にポータルユーザーが連絡できる E メールアドレスを入力します。
6. (オプション) ポータルにタグを追加します。詳細については、「リソースにタグを付ける AWS IoT SiteWise」を参照してください。
7. 以下のオプションのいずれかを選択します。
   - [新しいサービスロールの作成と使用] を選択します。デフォルトでは、 SiteWise Monitor はポータルごとにサービスロールを自動的に作成します。このロールにより、 AWS IoT SiteWise ポータルユーザーはリソースにアクセスできます。詳細については、「のサービスロールの使用 AWS IoT SiteWise Monitor」を参照してください。
   - 既存の[サービスロールの使用] を選択し、対象のロールを選択します。
8. [次へ] を選択します。
9. (オプション) ポータルのアラームを有効にします。詳細については、「ポータルのアラームの有効化」を参照してください。
10. [作成] を選択します。 AWS IoT SiteWise ポータルを作成します。

> **Note**
>
> コンソールを閉じる場合、管理者とユーザーを追加することでセットアッププロセスを完了できます。詳細については、「ポータル管理者の追加または削除」を参照してください。このポータルを維持しない場合は、リソースを使用しないようにするため、ポータルを削除します。詳細については、「ポータルの削除」を参照してください。

[状態] 欄には、次のいずれかの値を指定することができる。

- AWS IoT SiteWise CREATING-は、ポータルを作成するリクエストを処理しています。このプロセスは完了までに数分かかることがあります。
- AWS IoT SiteWise UPDATING-ポータルを更新するリクエストを処理しています。このプロセスは完了までに数分かかることがあります。
- 保留中-DNS AWS IoT SiteWise レコードの伝達が完了するのを待っています。このプロセスは完了までに数分かかることがあります。ステータスが「保留中」の場合は、ポータルを削除できます。
- 削除中- AWS IoT SiteWise ポータルを削除するユーザーのリクエストを処理しています。このプロセスは完了までに数分かかることがあります。
- アクティブ - ポータルがアクティブになると、ポータルユーザーはポータルにアクセスできます。
- 失敗-ポータルの作成、更新、 AWS IoT SiteWise 削除のリクエストを処理できませんでした。Amazon CloudWatch Logs AWS IoT SiteWise への ログ送信を有効にした場合は、これらのログを使用して問題のトラブルシューティングを行うことができます。詳細については、「AWS IoT SiteWise CloudWatch ログを使ったモニタリング」を参照してください。

ポータルを作成すると、メッセージが表示されます。

次に、1 人以上のポータル管理者をポータルに招待する必要があります。これまでのところ、ポータルを作成しましたが、誰もアクセスできません。

管理者の招待

新しいポータルの使用を開始するには、ポータル管理者を割り当てる必要があります。ポータル管理者は、プロジェクトを作成し、プロジェクト所有者を選択して、アセットをプロジェクトに割り当てます。 AWS IoT SiteWise ポータル管理者はすべてのアセットを閲覧できます。

ユーザー認証サービスに基づいて、次のいずれかを選択します。

IAM Identity Center

SiteWise モニターを初めて使用する場合は、前に作成したユーザーをポータル管理者として選択できます。ポータル管理者として別のユーザーを追加する場合は、このページから IAM Identity Center ユーザーを作成することができます。もう一つの方法として、外部 ID プロバイダを IAM Identity Center に接続することもできます。詳細については、『AWS IAM Identity Center ユーザーガイド』を参照してください。

管理者を招待するには

1. ポータル管理者になってもらいたいユーザーのチェックボックスをオンにします。これにより、ユーザーが [ポータル管理者] リストに追加されます。

   > **Note**
   >
   > IAM Identity Center をアイデンティティストアとして使用しており、 AWS Organizations 管理アカウントにサインインしている場合は、[ユーザーの作成] を選択することで IAM Identity Center ユーザーを作成することができます。IAM Identity Center から新しいユーザーに、パスワードを設定するための E メールが送信されます。そして、そのユーザーを管理者としてポータルに割り当てることができます。詳細については、「IAM Identity Center での ID の管理」を参照してください。

2. (オプション) [選択したユーザーに招待を送信] を選択します。E メールクライアントが開き、招待状がメッセージ本文に入力されます。

   E メールは、ポータル管理者に送信する前にカスタマイズできます。後でポータル管理者に E メールを送信することもできます。 SiteWise Monitor を初めて試し、新しい IAM Identity Center、IAM ユーザー、またはロールをポータル管理者として追加する場合は、自分宛にメールを送信する必要はありません。

3. 管理者として不要なユーザーを追加する場合は、そのユーザーのチェックボックスをオフにします。
4. ポータル管理者の招待が終了したら、[Next (次へ)] を選択します。

IAM

ポータル管理者にするユーザーまたはロールを選択することができます。ポータル管理者として別のユーザーまたはロールを追加する場合は、IAM コンソールでユーザーまたはロールを作成することができます。詳細については、IAM ユーザーガイドの「AWS アカウントでの IAM ユーザーの作成」および「IAM ロールの作成」を参照してください。

管理者を招待するには

1. 以下の操作を実行します。

   - [IAM ユーザー] を選択して、IAM ユーザーをポータル管理者として追加します。
   - [IAM ロール] を選択して、ポータル管理者として IAM ロールを追加します。

2. ポータルサイトの管理者にしたいユーザーやロールのチェックボックスを選択します。これにより、ユーザーまたはロールが [Portal administrators] (ポータル管理者) リストに追加されます。
3. 管理者として不要なユーザーまたはロールを追加した場合は、そのユーザーまたはロールのチェックボックスを外します。
4. ポータル管理者の招待が終了したら、[次へ] を選択します。

> **Important**
>
> ユーザーまたはロールがこのポータルにサインインするには `iotsitewise:DescribePortal` アクセス許可が必要です。

> **Note**
>
> IAM Identity Center をアイデンティティストアとして使用しており、 AWS Organizations 管理アカウントにサインインしている場合は、[ユーザーの作成] を選択することで IAM Identity Center ユーザーを作成することができます。IAM Identity Center から新しいユーザーに、パスワードを設定するための E メールが送信されます。そして、そのユーザーを管理者としてポータルに割り当てることができます。詳細については、「IAM Identity Center での ID の管理」を参照してください。

ポータル管理者のリストは後で変更できます。詳細については、「ポータル管理者の追加または削除」を参照してください。

> **Note**
>
> ポータル管理者のみがプロジェクトを作成し、アセットを割り当てることができるため、少なくとも1 人のポータル管理者を指定する必要があります。

最後の手順として、新しいポータルにアクセスできるユーザーを追加します。

ポータルユーザーを追加する

ポータルにアクセスできるユーザーを制御します。各ポータルでは、ポータル管理者は 1 つ以上のプロジェクトを作成し、ポータルユーザーを各プロジェクトの所有者または閲覧者として割り当てます。各プロジェクト所有者は、プロジェクトを所有または閲覧するように追加のポータルユーザーを招待することができます。

ユーザー認証サービスに基づいて、次のいずれかを選択します。

IAM Identity Center

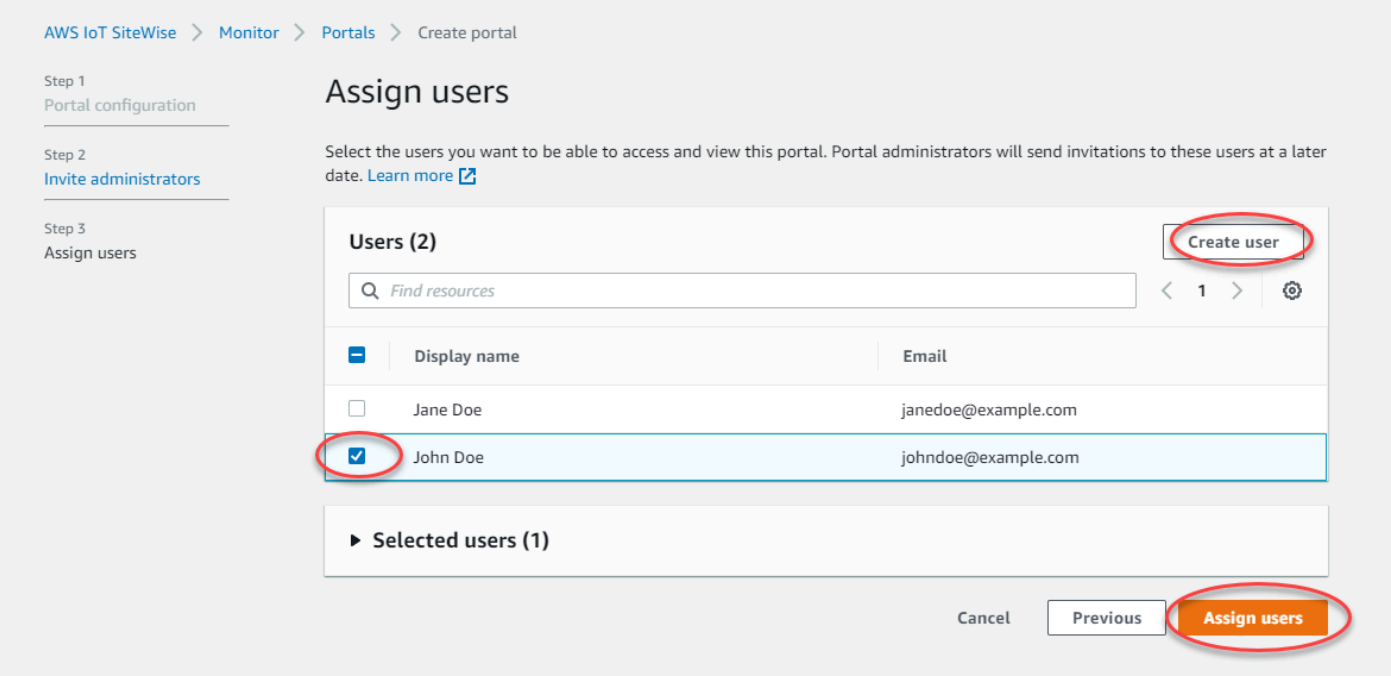
ユーザーを [ユーザー] リストに追加する場合は、次のステップを実行します。

ポータルユーザーを追加するには

1. ポータルに追加するユーザーを [ユーザー] リストから選択します。これにより、ユーザーが [ポータルユーザー] リストに追加されます。 SiteWise モニターを初めて使用する場合は、ポータル管理者をポータルユーザーとして追加する必要はありません。

   > **Note**
   >
   > IAM Identity Center をアイデンティティストアとして使用しており、 AWS Organizations 管理アカウントにサインインしている場合は、[ユーザーの作成] を選択することで IAM Identity Center ユーザーを作成することができます。IAM Identity Center から新しいユーザーに、パスワードを設定するための E メールが送信されます。そして、そのユーザーをユーザーとしてポータルに割り当てることができます。詳細については、「IAM Identity Center での ID の管理」を参照してください。

2. ポータルにアクセスさせたくないユーザーを追加する場合は、そのユーザーのチェックボックスをオフにします。
3. ユーザーの選択が終了したら、[ユーザーの割り当て] を選択します。



IAM

[IAM ユーザー] または [IAM ロール] のリストに追加したいユーザーまたはロールが表示されたら、次のステップを実行します。

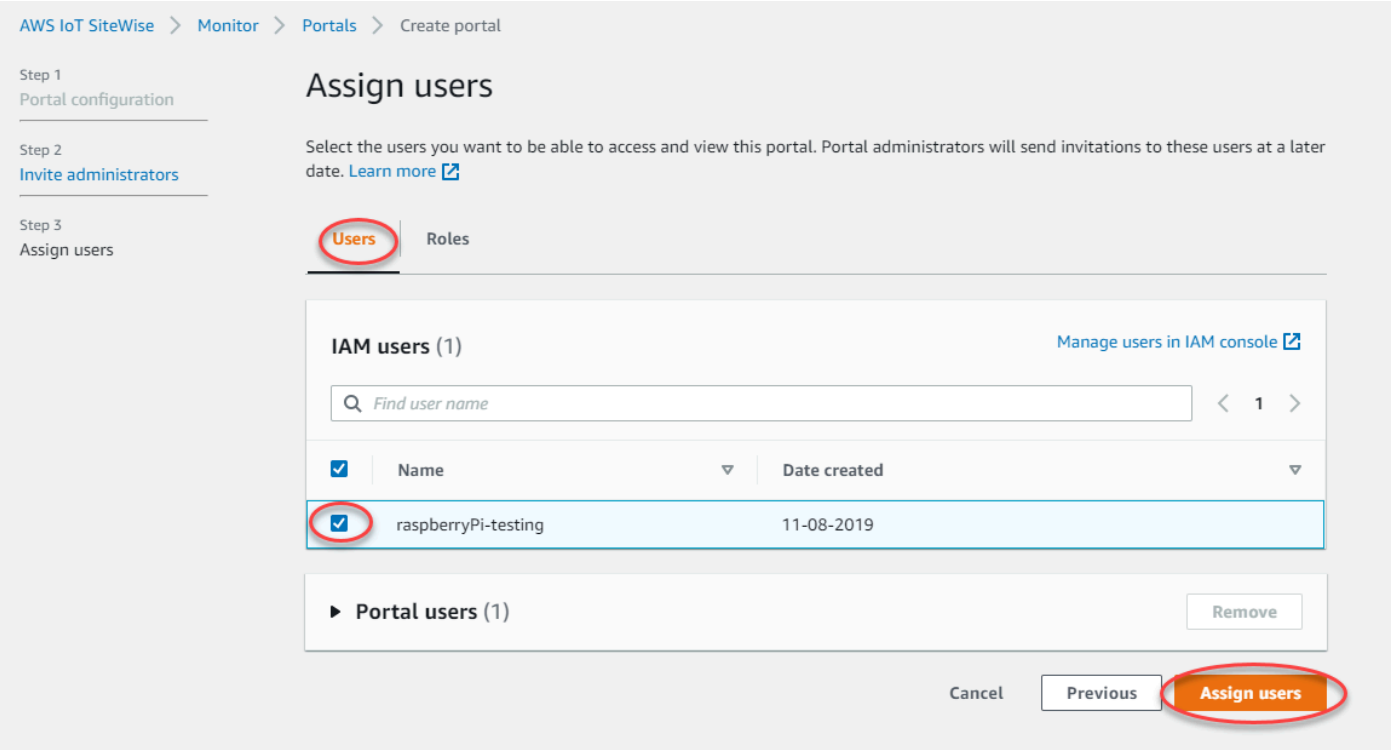
ポータルユーザーを追加するには

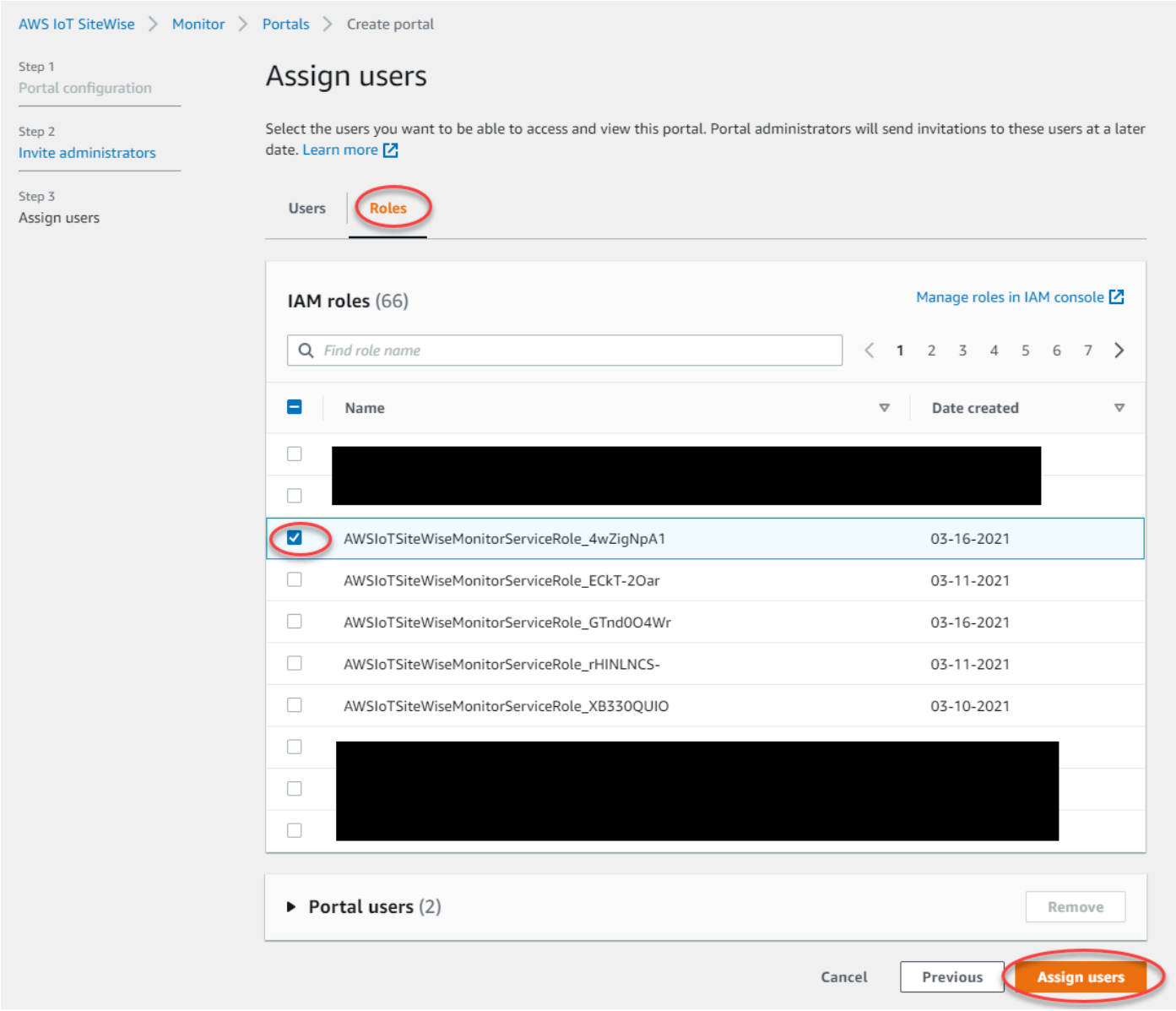
1. 次のオプションを実行します。

   - [IAM ユーザー] () を選択して、IAM ユーザーをポータルユーザーとして追加します。
   - [IAM ロール] を選択して、IAM ロールをポータルユーザーとして追加します。

   SiteWise Monitor を初めて使用する場合は、ポータル管理者をポータルユーザーとして追加する必要はありません。
2. ポータルユーザーにするユーザーまたはロールのチェックボックスを選択します。これにより、ユーザーまたはロールが [Portal users] (ポータルユーザー) リストに追加されます。
3. ポータルにアクセスさせたくないユーザーを追加する場合は、そのユーザーのチェックボックスをオフにします。
4. ユーザーの選択が終了したら、[ユーザーの割り当て] を選択します。

> **Important**
>
> ユーザーまたはロールがこのポータルにサインインするには `iotsitewise:DescribePortal` アクセス許可が必要です。





お疲れ様でした。ポータルの作成、ポータル管理者の割り当て、ポータルを利用するユーザーの割り当てが完了しました。ポータル管理者は、プロジェクトを作成し、それらのプロジェクトにアセットを追加できるようになりました。次に、プロジェクトの所有者はダッシュボードを作成して、各プロジェクトのアセットのデータを可視化することができます。

ポータルユーザーのリストは、後で変更することができます。詳細については、「ポータルユーザーの追加または削除」を参照してください。

ポータルに変更を加える必要がある場合は、「SiteWise 監視ポータルの管理」を参照してください。

ポータルを始めるには、『SiteWise Monitor アプリケーションガイド』の「はじめに」を参照してください。

ダッシュボードの作成 (AWS Command Line Interface)

AWS CLI を使用してダッシュボードで視覚化 (またはウィジェット) を定義する場合、`dashboardDefinition` JSON ドキュメントで次の情報を指定する必要があります。この定義は、 CreateDashboardおよび UpdateDashboardオペレーションのパラメータです。

`widgets`

ウィジェット定義構造のリスト。それぞれに次の情報が含まれています。

`type`

ウィジェットのタイプ。AWS IoT SiteWise は、次のウィジェットタイプを提供します。

- `sc-line-chart` - 折れ線グラフ 詳しくは、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[Line charts] (折れ線グラフ) をご覧ください。
- `sc-scatter-chart` — 散布図。詳細については、AWS IoT SiteWise Monitor アプリケーションガイドの「散布図」を参照してください。
- `sc-bar-chart` - 棒グラフ 詳細については、AWS IoT SiteWise Monitor アプリケーションガイドの「棒グラフ」を参照してください。
- `sc-status-grid` - アセットプロパティの最新値をグリッドで表示する状態ウィジェットです。詳しくは、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[Status widgets] (状態ウィジェット) をご覧ください。
- `sc-status-timeline` - アセットプロパティの履歴値を時系列で表示する状態ウィジェットです。詳しくは、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[Status widgets] (状態ウィジェット) をご覧ください。
- `sc-kpi` - 重要業績評価指標 (KPI、 key performance indicator) の視覚化。詳しくは、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[KPI widgets] (KPI ウィジェット) をご覧ください。
- `sc-table` — テーブルウィジェット 詳しくは、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[Table widgets] (テーブルウィジェット) をご覧ください。

`title`

ウィジェットのタイトル。

`x`

ウィジェットの水平位置。グリッドの左から開始します。この値は、ダッシュボードのグリッド内のウィジェットの位置を参照します。

`y`

グリッドの上部から始まる、ウィジェットの垂直位置。この値は、ダッシュボードのグリッド内のウィジェットの位置を参照します。

`width`

ダッシュボードのグリッド上のスペース数で表されるウィジェットの幅。

`height`

ダッシュボードのグリッド上のスペースの数で表されるウィジェットの高さ。

`metrics`

それぞれがこのウィジェットのデータストリームを定義するメトリクス構造のリスト。リスト内の各構造には、次の情報が含まれている必要があります。

`label`

このメトリクスに表示するラベル。

`type`

このメトリクスのデータソースのタイプ。AWS IoT SiteWise は、次のメトリクスタイプを提供します。

- `iotsitewise` – ダッシュボードには、AWS IoT SiteWise にあるアセットプロパティのデータがフェッチされます。このオプションを選択した場合は、このメトリクスに対して `assetId` および `propertyId` を定義する必要があります。

`assetId`

(オプション) AWS IoT SiteWise のアセットの ID。

このフィールドは、このメトリクスで `type` に `iotsitewise` を選択した場合、必須です。

`propertyId`

(オプション) AWS IoT SiteWise のアセットプロパティの ID。

このフィールドは、このメトリクスで `type` に `iotsitewise` を選択した場合、必須です。

`analysis`

(オプション) ウィジェットに表示するトレンドラインなどの分析を定義する構造です。詳しくは、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[Configuring trend lines] (トレンドラインの設定) をご覧ください。ウィジェットのプロパティごとに、各型のトレンドラインを 1 つずつ追加することができます。解析構造には、次の情報が含まれています。

`trends`

(オプション) 各ウィジェットのトレンド分析を定義するトレンド構造のリスト。リスト内の各構造には、次の情報が含まれています。

`type`

トレンドラインの種類。次のオプションを選択します。

- `linear-regression` - 線形回帰ラインを表示します。 SiteWise モニタリングは最小二乗法を使用して線形回帰を計算します。

`annotations`

(オプション) ウィジェットのしきい値を定義するアノテーション構造。詳しくは、[AWS IoT SiteWise Monitor Application Guide] (アプリケーションガイド) の[Configuring thresholds] (しきい値の設定) をご覧ください。1 つのウィジェットにつき最大 6 つのアノテーションを追加することができます。アノテーション構造には、次の情報が含まれます。

`y`

(オプション) このウィジェットの水平方向のしきい値をそれぞれ定義するアノテーション構造のリスト。リスト内の各構造には、次の情報が含まれています。

`comparisonOperator`

しきい値の比較演算子。以下のうちのひとつを選択します。

- `LT` - `value` より小さいデータポイントを少なくとも 1 つ持つプロパティがハイライトされます。

- GT - value より大きいデータポイントを少なくとも 1 つ持つプロパティがハイライトされます。
- LTE - value 以下のデータポイントを少なくとも 1 つ持つプロパティがハイライトされます。
- GTE - value 以上のデータポイントを少なくとも 1 つ持つプロパティがハイライトされます。
- EQ - value と同じデータポイントを少なくとも 1 つ持つプロパティがハイライトされます。

value

データポイントを comparisonOperator と比較するためのしきい値。

color

(オプション) しきい値の色を表す 6 桁の 16 進数コード。視覚化では、しきい値のルールを満たすデータポイントが少なくとも 1 つあるプロパティについて、この色でプロパティの凡例が表示されます。デフォルトは黒 (#000000) です。

showValue

(オプション) ウィジェットの余白にしきい値の値を表示するかどうかを設定します。デフォルトは true です。

properties

(オプション) ウィジェットのプロパティのフラットディクショナリです。この構造のメンバーはコンテキストに依存します。AWS IoT SiteWise は、properties を使用する次のウィジェットを提供します。

- 折れ線グラフ、散布図、棒グラフには次のプロパティがあります。

  colorDataAcrossThresholds

  (オプション) このウィジェットでしきい値を超えたデータの色を変更するかどうかを設定します。このオプションを有効にすると、しきい値を超えたデータが選択した色で表示されます。デフォルトは true です。

- [Status grids] (状態グリッド) は、次の特性を持ちます。

  labels

  (オプション) 状態グリッドに表示するラベルを定義する構造。ラベル構造には、次の情報が含まれる。

`showValue`

(オプション) このウィジェットの各アセットプロパティの単位と値を表示するかどうかを指定します。デフォルトは `true` です。

Example ダッシュボード定義の例

次の例では、JSON ファイルに保存されているペイロードからダッシュボードを定義します。

```
aws iotsitewise create-dashboard \
  --project-id a1b2c3d4-5678-90ab-cdef-eeeeeEXAMPLE \
  --dashboard-name "Wind Farm Dashboard" \
  --dashboard-definition file://dashboard-definition.json
```

次の `dashboard-definition.json` の JSON 例では、次の視覚化ウィジェットを使用してダッシュボードを定義します。

- ダッシュボードの左上にある風力発電所の総電力を視覚化する折れ線グラフ。この折れ線グラフには、風力発電所の出力が期待される最小出力よりも小さくなるタイミングを示すしきい値が含まれています。この折れ線グラフには、線形回帰のトレンドラインも含まれています。
- ダッシュボードの右上にある 4 つのタービンの風速を表示する棒グラフ。

> **Note**
>
> 次の例は、ダッシュボードの折れ線グラフと棒グラフの視覚化を表します。このダッシュボードは、風力発電所のダッシュボードの例に似ています。

```
{
  "widgets": [
    {
      "type": "sc-line-chart",
      "title": "Total Average Power",
      "x": 0,
      "y": 0,
      "height": 3,
      "width": 3,
      "metrics": [
        {
```

```
          "label": "Power",
          "type": "iotsitewise",
          "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
          "propertyId": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
          "analysis": {
            "trends": [
              {
                "type": "linear-regression"
              }
            ]
          }
        }
      ],
      "annotations": {
        "y": [
          {
            "comparisonOperator": "LT",
            "value": 20000,
            "color": "#D13212",
            "showValue": true
          }
        ]
      }
    },
    {
      "type": "sc-bar-chart",
      "title": "Wind Speed",
      "x": 3,
      "y": 3,
      "height": 3,
      "width": 3,
      "metrics": [
        {
          "label": "Turbine 1",
          "type": "iotsitewise",
          "assetId": "a1b2c3d4-5678-90ab-cdef-2a2a2EXAMPLE",
          "propertyId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE"
        },
        {
          "label": "Turbine 2",
          "type": "iotsitewise",
          "assetId": "a1b2c3d4-5678-90ab-cdef-2b2b2EXAMPLE",
          "propertyId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE"
        },
```

```
        {
          "label": "Turbine 3",
          "type": "iotsitewise",
          "assetId": "a1b2c3d4-5678-90ab-cdef-2c2c2EXAMPLE",
          "propertyId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE"
        },
        {
          "label": "Turbine 4",
          "type": "iotsitewise",
          "assetId": "a1b2c3d4-5678-90ab-cdef-2d2d2EXAMPLE",
          "propertyId": "a1b2c3d4-5678-90ab-cdef-55555EXAMPLE"
        }
      ]
    }
  ]
}
```

ポータルのアラームの有効化

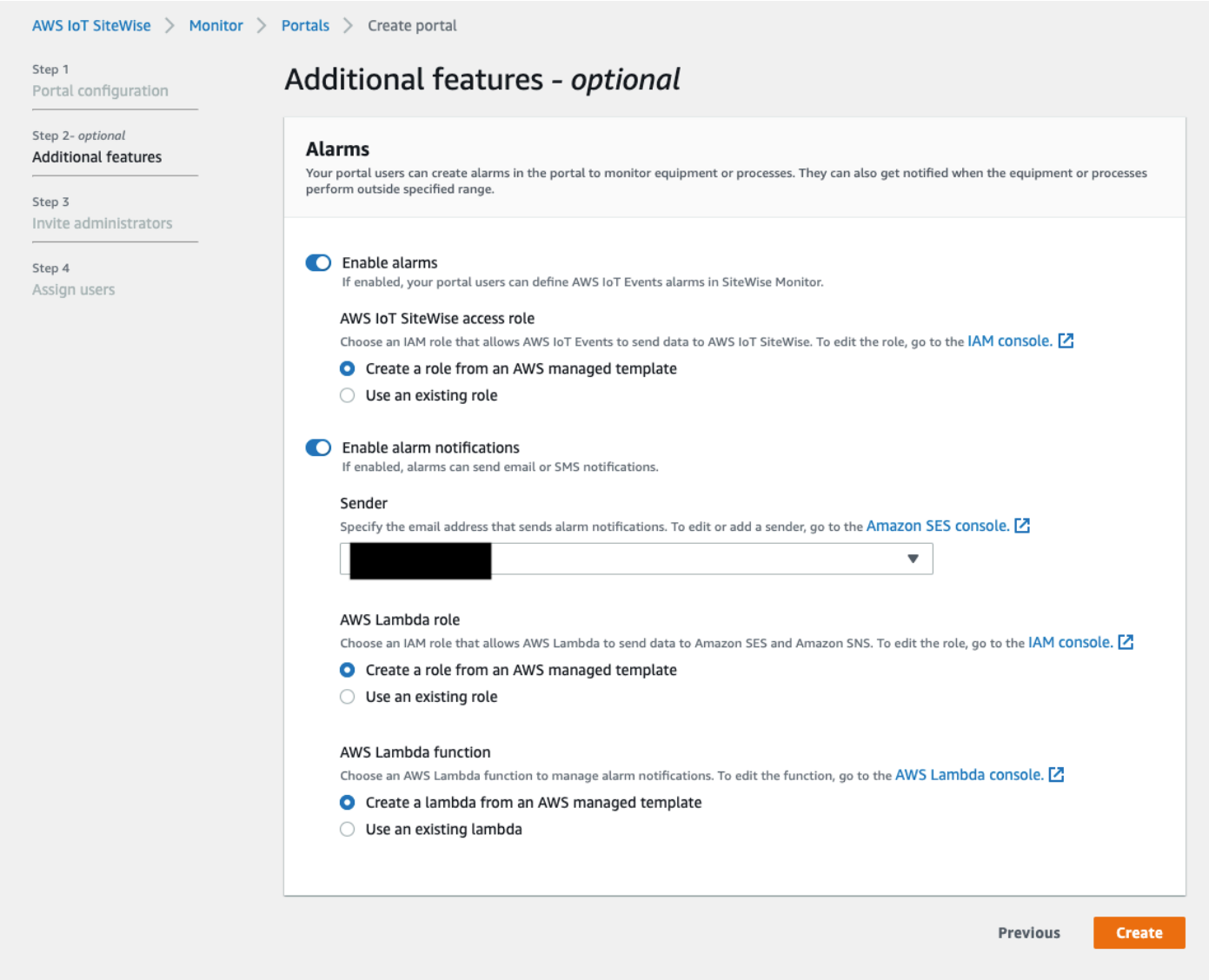
AWS IoT Events ポータルでサポートされているアラーム機能を有効にして、 AWS IoT Events ポータル管理者が監視ポータルでアラームモデルを作成、編集、削除できるようにすることができます。SiteWise プロジェクト所有者は、アラームを設定することができます。プロジェクトビューワーは、アラームの詳細を確認することができます。このセクションでは、 AWS IoT SiteWise コンソールを使用してポータルのアラーム機能を有効にする方法について説明します。

Important

- ポータルに外部アラームを作成することはできません。
- アラーム通知を送信する場合は、ユーザー認証サービスに IAM Identity Center を選択する必要があります。
- アラーム通知機能は中国 (北京) では使用できません。 AWS リージョン

ポータルを設定し作成すると、[ステップ 2 の追加機能] でアラームとアラーム通知を有効にすることができます。ユーザー認証サービスに基づいて、次のいずれかを選択します。

IAM Identity Center



ポータルのアラームを有効にするには

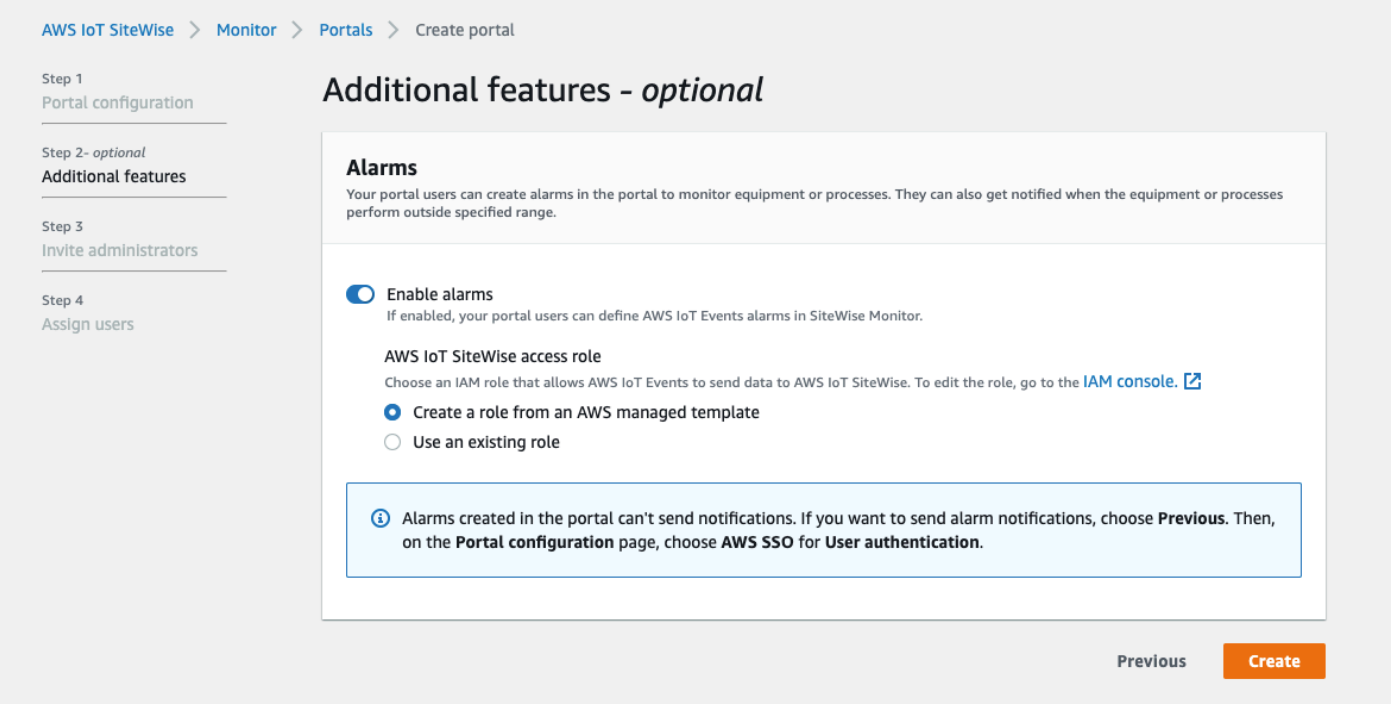
1. (オプション) [アラームを有効にする] を選択します。
   - [AWS IoT SiteWise アクセスロール] では、既存のロールを使用するか、必要なアクセス許可を持つロールを作成します。このロールには、`iotevents:BatchPutMessage` のアクセス許可と、`iot.amazonaws.com` と `iotevents.amazonaws.com` がロールを担うことができる信頼関係が必要です。
2. (オプション) [アラーム通知を有効にする] を選択します。
   a. [Sender] (送信者) は、送信者を選択します。

> **Important**
>
> Amazon SES で送信者メールアドレスの確認が必要です。詳細については、[Amazon Simple Email Service Developer Guide] (Amazon Simple Email Service デベロッパーガイド) の[Verifying email addresses in Amazon SES] (Amazon SES でのメールアドレスの確認) を参照してください。

b. [AWS Lambda ロール] の場合、既存のロールを使用するか、必要なアクセス許可を持つロールを作成します。このロールには、`lambda:InvokeFunction`、`sso-directory:DescribeUser` のアクセス許可と、`iotevents.amazonaws.com` と `lambda.amazonaws.com` がロールを担うことができる信頼関係が必要です。

c. [AWS Lambda 関数] では、既存の Lambda 関数を選択するか、アラーム通知を管理する関数を作成します。詳細については、AWS IoT Events デベロッパーガイドの「アラーム通知の管理」を参照してください。

IAM



ポータルのアラームを有効にするには

- (オプション) [アラームを有効にする] を選択します。

- [AWS IoT SiteWise アクセスロール] では、既存のロールを使用するか、必要なアクセス許可を持つロールを作成します。このロールには、`iotevents:BatchPutMessage` のアクセス許可と、`iot.amazonaws.com` と `iotevents.amazonaws.com` がロールを担うことができる信頼関係が必要です。

SiteWise Monitor のアラームについて詳しくは、『AWS IoT SiteWise アプリケーションガイド』の「アラームによる監視」を参照してください。

エッジでのポータルの有効化

Edge でポータルを有効にすると、アカウントでデータ処理パックが有効になっているすべての SiteWise Edge ゲートウェイでこのポータルを利用できるようになります。

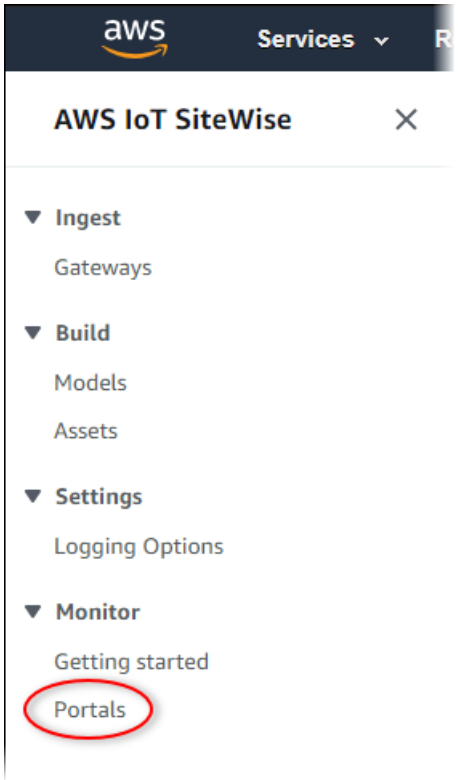
エッジでポータルを有効にするには

1. [エッジ設定] セクションで、[エッジでこのポータルを有効にする] をオンにします。
2. [作成] を選択します。

SiteWise 監視ポータルの管理

ポータル詳細の更新、管理者の変更、またはポータルへのユーザーの追加が必要になる場合があります。このセクションでは、 SiteWise モニターポータルの基本的な管理タスクを実行する方法について説明します。

1. AWS IoT SiteWise コンソールにサインインします。
2. ナビゲーションペインで、[モニター]、[ポータル] の順に選択します。



3. ポータルを選択し、[詳細の表示] を選択します (または [名前] を選択します)。
4. 次のいずれかの管理タスクを実行できます。

   - ポータルの名前、説明、ブランド、サポート E メール、アクセス許可の変更
   - ポータル管理者の追加または削除
   - ポータル管理者への招待メールの送信
   - ポータルユーザーの追加または削除
   - ポータルの削除

ポータルを作成する方法については、「はじめに AWS IoT SiteWise Monitor」を参照してください。

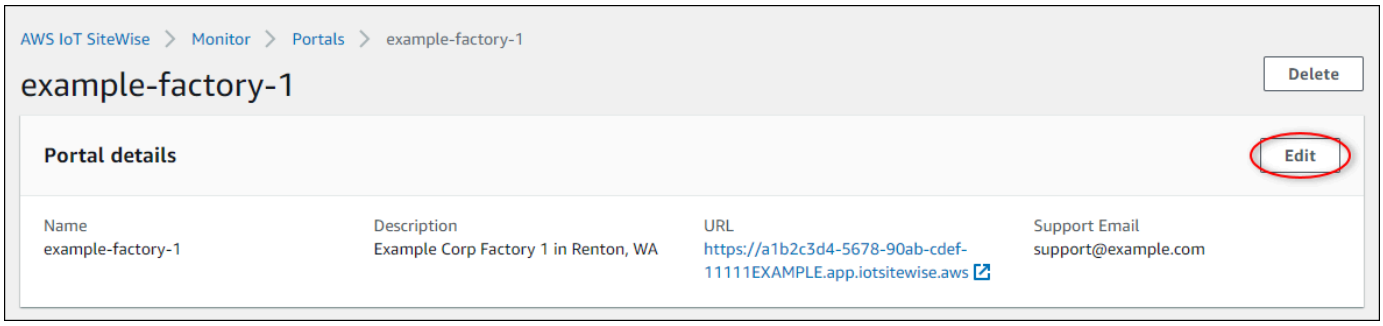
トピック

- ポータルの名前、説明、ブランド、サポート E メール、アクセス許可の変更
- ポータル管理者の追加または削除
- ポータル管理者への招待メールの送信
- ポータルユーザーの追加または削除
- ポータルの削除

ポータルの名前、説明、ブランド、サポート E メール、アクセス許可の変更

ポータルの名前、説明、サポート E メール、ブランド、アクセス許可を変更できます。

1. ポータルの詳細ページの [ポータルの詳細] セクションで、[編集] を選択します。

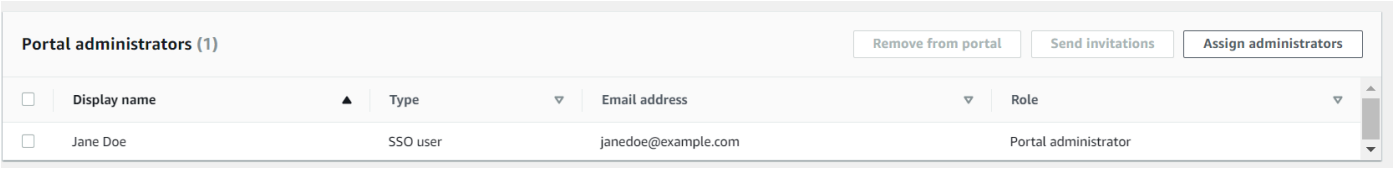


2. [名前]、[説明]、[ポータルのブランド]、[サポートお問い合わせ E メール]、または [アクセス許可] を更新します。
3. 完了したら、[保存] を選択します。

ポータル管理者の追加または削除

いくつかのステップで、ポータルの管理者としてユーザーを追加または削除できます。ユーザー認証サービスに基づいて、次のいずれかを選択します。

IAM Identity Center

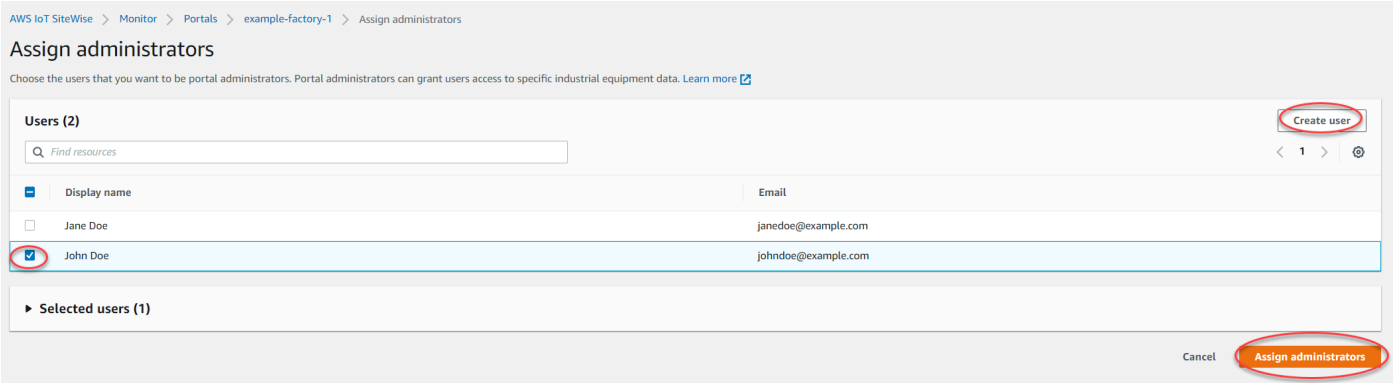


ポータル管理者を追加するには

1. ポータルの詳細ページで、[Portal administrators] (ポータル管理者) セクションの [Assign administrators] (管理者の割り当て) を選択します。
2. [管理者の割り当て] () ページで、ポータルに管理者として追加するユーザーのチェックボックスを選択します。

> **Note**
>
> IAM Identity Center をアイデンティティストアとして使用しており、 AWS Organizations 管理アカウントにサインインしている場合は、[ユーザーの作成] を選択することで IAM Identity Center ユーザーを作成することができます。IAM Identity Center から新しいユーザーに、パスワードを設定するための E メールが送信されます。そして、そのユーザーを管理者としてポータルに割り当てることができます。詳細については、「IAM Identity Center での ID の管理」を参照してください。

3. [Assign administrators] (管理者の割り当て) を選択します。

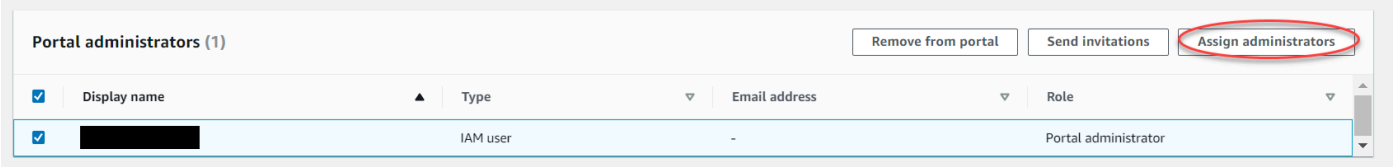


ポータル管理者を削除するには

- ポータルの詳細ページの [Portal administrators (ポータル管理者)] セクションで、削除する各ユーザーのチェックボックスをオンにし、[Remove from portal (ポータルから削除)] を選択します。

> **Note**
>
> ポータル管理者を最低 1 人選択することをお勧めします。

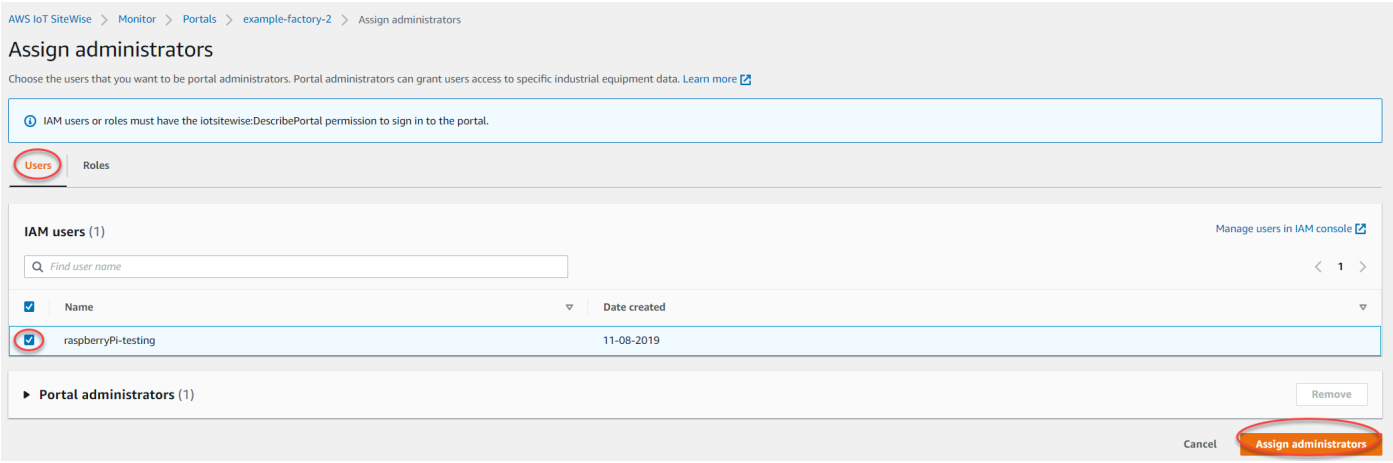
IAM

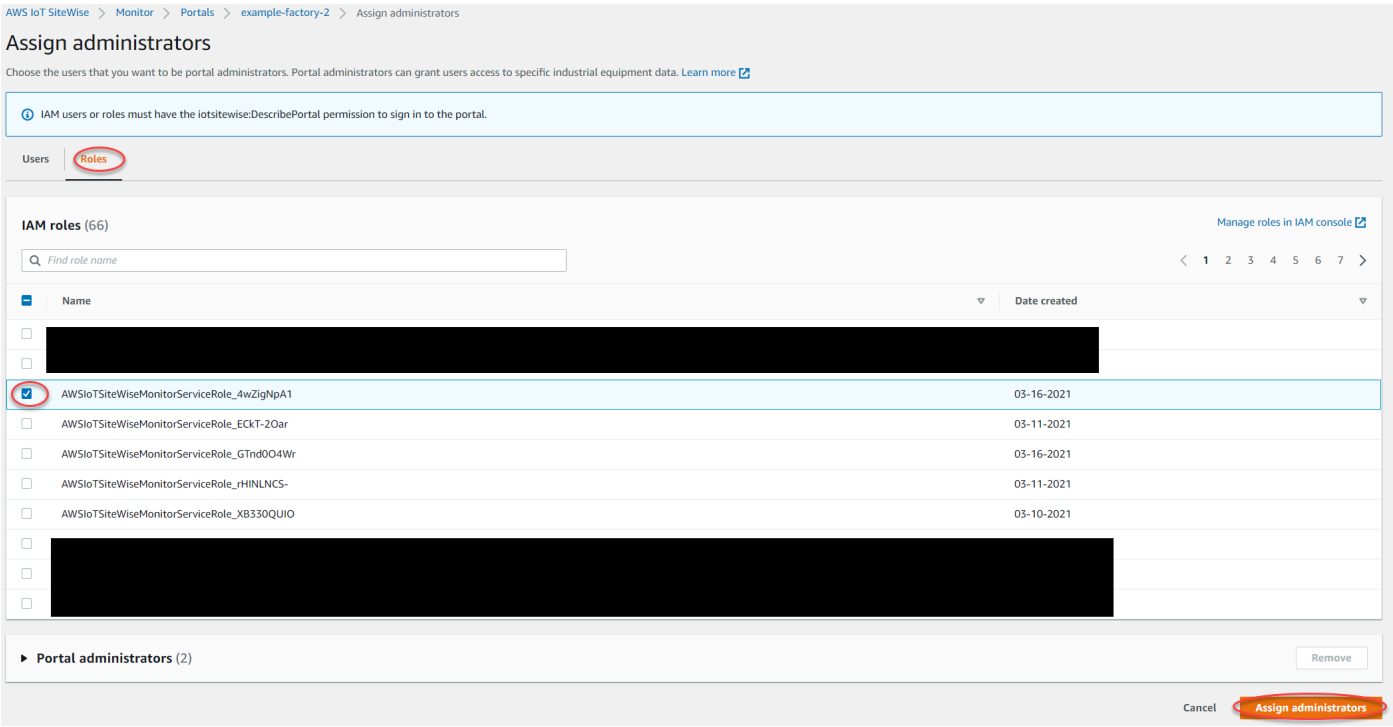


ポータル管理者を追加するには

1. ポータルの詳細ページで、[ポータル管理者] () セクションの [管理者の割り当て] を選択します。
2. [管理者の割り当て] ページで、次の操作を行います。
   - IAM ユーザーをポータル管理者として追加する場合は、[IAM ユーザー] を選択します。
   - ポータル管理者として IAM ロールを追加する場合は、[IAM ロール] () を選択します。
3. ポータルサイトの管理者にしたいユーザーやロールのチェックボックスを選択します。これにより、ユーザーまたはロールが [ポータル管理者] リストに追加されます。
4. [管理者の割り当て] を選択します。

> **Important**
>
> ユーザーまたはロールがこのポータルにサインインするには `iotsitewise:DescribePortal` アクセス許可が必要です。





ポータル管理者を削除するには

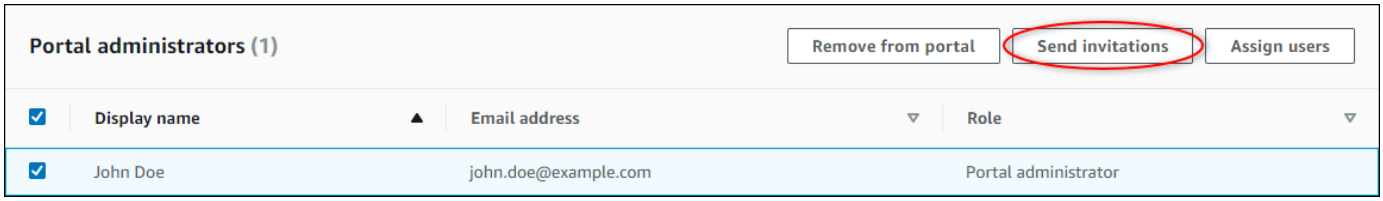
- ポータルの詳細ページの [ポータル管理者] セクションで、削除する各ユーザーのチェックボックスをオンにし、[ポータルから削除] を選択します。

> **Note**
>
> ポータル管理者なしでポータルを離れることはお勧めしません。

ポータル管理者への招待メールの送信

ポータルサイトの管理者に招待メールを送信することができます。

1. ポータルの詳細ページの [ポータル管理者] セクションで、ポータル管理者のチェックボックスをオンにします。



2. [招待状を送信] を選択します。E メールクライアントが開き、招待状がメッセージ本文に入力されます。

   E メールは、ポータル管理者に送信する前にカスタマイズできます。

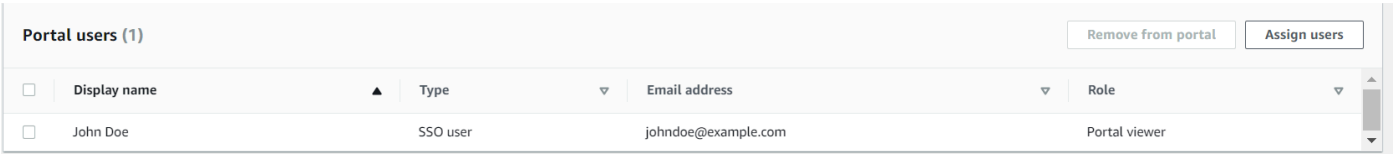
ポータルユーザーの追加または削除

ポータルにアクセスできるユーザーを選択します。ポータルユーザーは、 SiteWise モニターポータル内のユーザーリストに表示されます。このリストから、ポータル管理者はプロジェクト所有者を追加でき、プロジェクト所有者はプロジェクトビューワーを追加できます。

> **Note**
>
> ポータルの管理者およびポータルユーザーが、ユーザーの追加または削除を必要とする場合、ポータルのサポートメールを通じて連絡してくる場合があります。

ユーザー認証サービスに基づいて、次のいずれかを選択します。

IAM Identity Center



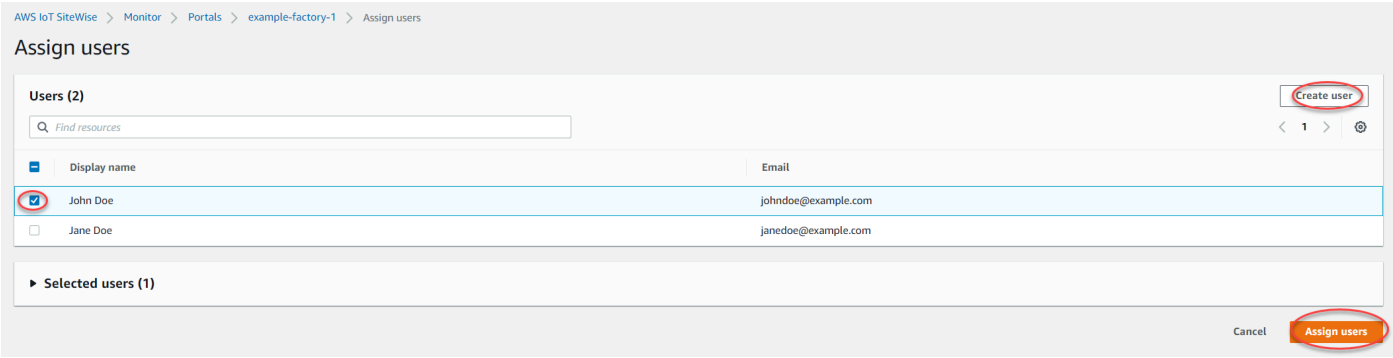
ポータルユーザーを追加するには

1. ポータルの詳細ページの [Portal users (ポータルユーザー)] セクションで [ユーザーの割り当て] を選択します。
2. [ユーザーの割り当て] ページで、ポータルに追加するユーザーのチェックボックスを選択します。

   > **Note**
   >
   > IAM Identity Center をアイデンティティストアとして使用しており、 AWS Organizations 管理アカウントにサインインしている場合は、[ユーザーの作成] を選択することで IAM Identity Center ユーザーを作成することができます。IAM Identity Center から新しいユーザーに、パスワードを設定するための E メールが送信されま

す。そして、そのユーザーをユーザーとしてポータルに割り当てることができます。詳細については、「IAM Identity Center での ID の管理」を参照してください。

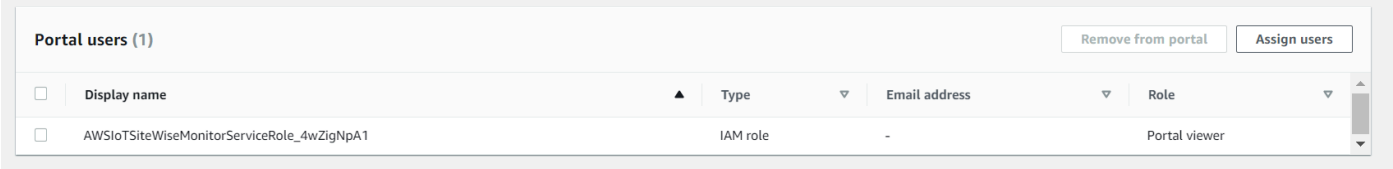
3. [ユーザーの割り当て] を選択します。



ポータルユーザーを削除するには

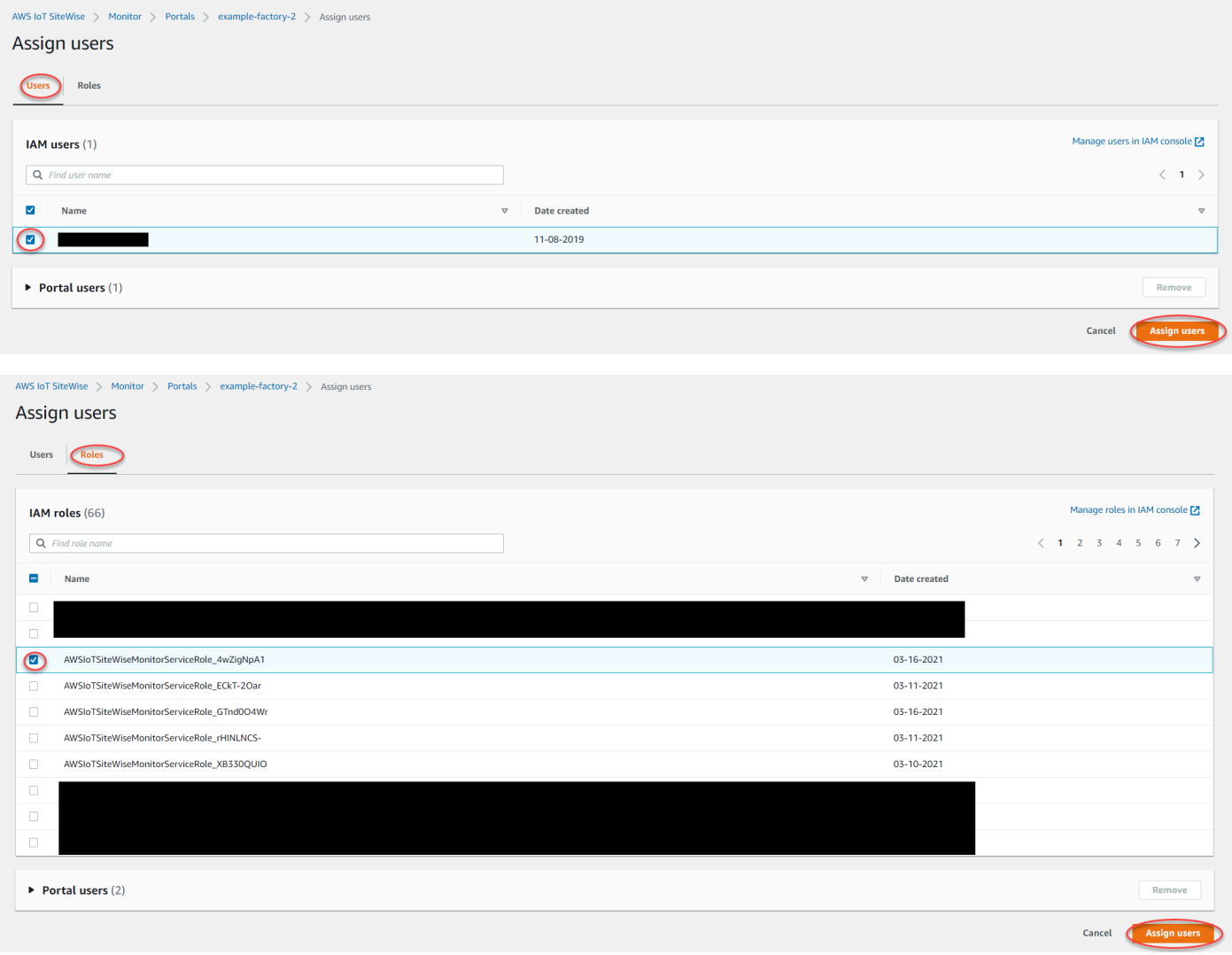
- ポータルの詳細ページで、[Portal users] (ポータルのユーザー) セクションで、ポータルから削除するユーザーのチェックボックスを選択し、[Remove from portal] (ポータルから削除) を選択します。

IAM



ポータルユーザーを追加するには

1. ポータルの詳細ページの [ポータルユーザー] セクションで [ユーザーの割り当て] を選択します。
2. [ユーザーの割り当て] ページで、次の操作を行います。
   - [IAM ユーザー] を選択して、IAM ユーザーをポータルユーザーとして追加します。
   - [IAMロール] を選択して、IAM ロールをポータルユーザーとして追加します。
3. ポータルユーザーとして追加したいユーザーまたはロールのチェックボックスを選択します。これにより、ユーザーまたはロールが [ポータルユーザー] リストに追加されます。
4. [ユーザーの割り当て] を選択します。



ポータルユーザーを削除するには

- ポータルの詳細ページで、[ポータルのユーザー] セクションで、ポータルから削除するユーザーのチェックボックスを選択し、[ポータルから削除] を選択します。

> **Important**
>
> ユーザーまたはロールがこのポータルにサインインするには `iotsitewise:DescribePortal` アクセス許可が必要です。

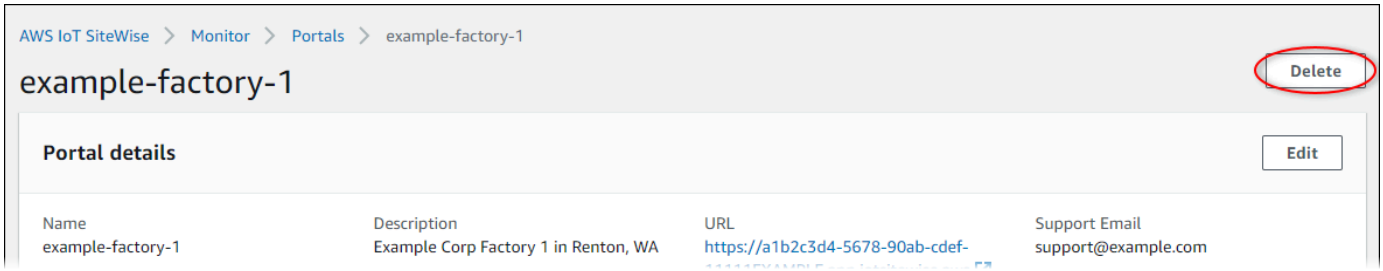
ポータルの削除

テスト目的でポータルを作成した場合、またはすでに存在するポータルの複製を作成した場合は、ポータルを削除することがあります。

> **Note**
>
> ポータルを削除するには、まずポータル内のすべてのダッシュボードとプロジェクトを手動で削除する必要があります。詳細については、『SiteWise Monitor アプリケーションガイド』の「プロジェクトの削除」と「ダッシュボードの削除」を参照してください。

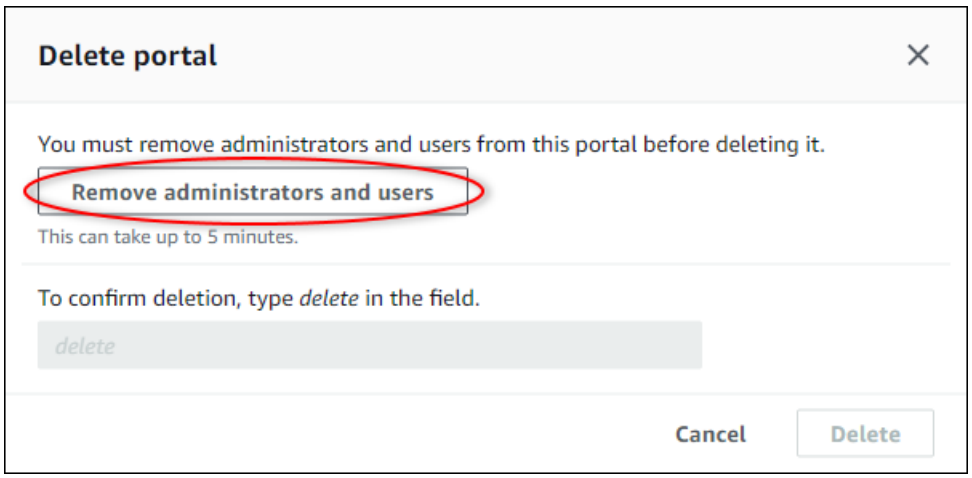
1. ポータルの詳細ページで、[削除] を選択します。

   > **Important**
   >
   > ポータルを削除すると、ポータルに含まれるすべてのプロジェクト、および各プロジェクトのすべてのダッシュボードが失われます。この操作は元に戻すことができません。アセットデータは影響を受けません。

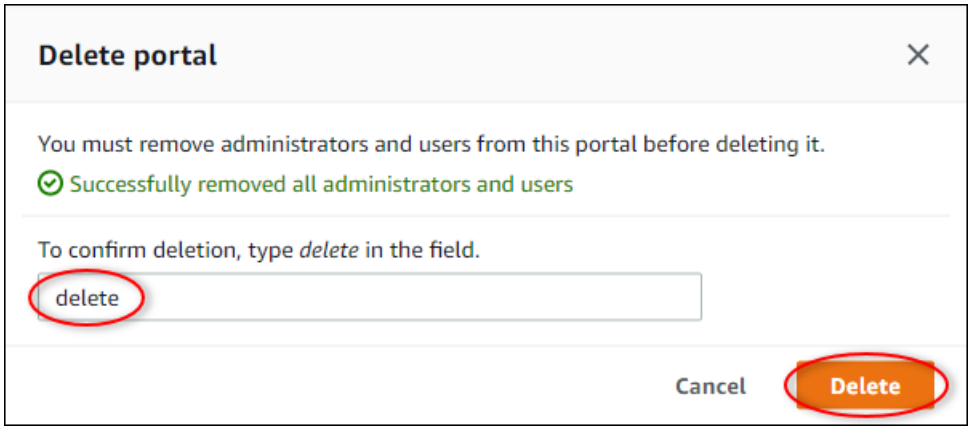


2. [ポータルの削除] ダイアログボックスで、[管理者とユーザーの削除] を選択します。

   ポータルを削除する前に、ポータルから管理者とユーザーを削除する必要があります。ポータルに管理者またはユーザーがいない場合、ボタンは表示されず、次の手順に進むことができます。



3. ポータル全体を削除する場合は、フィールドに **`delete`** と入力して削除を確認します。



4. [削除] をクリックします。

IoT ダッシュボードアプリケーションでのデータのモニタリング

IoT ダッシュボードアプリケーションは、運用データを視覚化して操作できるオープンソースのダッシュボードアプリケーションです。を使用して AWS Cloud Development Kit (AWS CDK) IoT IoT ダッシュボードアプリケーションをデプロイできます。

IoT ダッシュボードアプリケーションのカスタマイズ可能なデータ視覚化機能の例を次に示します。

- 1 つの折れ線グラフでの複数のプロパティのサポート。
- アセットとプロパティの検索を強化しました。

製造、物流、エネルギー、その他の業界のお客様は、IoT ダッシュボードアプリケーションを使用して、機器のパフォーマンスの追跡、運用効率の最適化、データ主導の決定など、特定の課題に対処できます。詳細については、GitHub 「 IoT ダッシュボードアプリケーション用のリポジトリ」を参照してください。

からのデータのクエリ AWS IoT SiteWise

AWS IoT SiteWise API オペレーションを使用して、特定の期間におけるアセットプロパティの現在の値、過去の値、および集計をクエリできます。

これらの機能を使用して、データに関する洞察を得てください。たとえば、特定のプロパティ値を持つすべての資産を検索したり、データをカスタム表示したりできます。API オペレーションを使用して、 AWS IoT SiteWise 資産に保存されている産業データと統合するソフトウェアソリューションを開発することもできます。また、 AWS IoT SiteWise Monitorではリアルタイムのアセットデータを調べることもできます。 SiteWise Monitor の設定方法については、を参照してくださいによるデータの監視 AWS IoT SiteWise Monitor。

このセクションで説明する操作は、タイムスタンプ、品質、値 (TQV) 構造を含むプロパティ値オブジェクトを返します。

- `timestamp` には、ナノ秒のオフセットがある現在の UNIX エポック時刻 (秒単位) が含まれます。
- `quality` には、データポイントの品質を示す、以下のいずれかの文字列が含まれます。
  - `GOOD` - データはいずれの問題の影響も受けません。
  - `BAD` - データはセンサーの障害などの問題による影響を受けます。
  - `UNCERTAIN` - データはセンサーの不正確さなどの問題による影響を受けます。
- `value` には、プロパティのタイプに応じて、以下のいずれかのフィールドが含まれます。
  - `booleanValue`
  - `doubleValue`
  - `integerValue`
  - `stringValue`

トピック

- 現在のアセットのプロパティ値のクエリ
- 履歴アセットプロパティ値をクエリする
- アセットプロパティ集計のクエリ
- AWS IoT SiteWise クエリ言語

現在のアセットのプロパティ値のクエリ

このチュートリアルでは、アセットプロパティの現在の値を取得する 2 つの方法を紹介します。AWS IoT SiteWise コンソールを使用することも、 AWS Command Line Interface (AWS CLI) 内の API を使用することもできます。

トピック

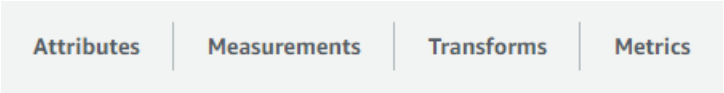
- アセットプロパティの現在の値を問い合わせる (コンソール)
- アセットプロパティの現在の値 (AWS CLI) をクエリします。

アセットプロパティの現在の値を問い合わせる (コンソール)

AWS IoT SiteWise コンソールを使用してアセットプロパティの現在の値を表示できます。

アセットプロパティの現在の値を取得するには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [Assets (アセット)] を選択します。
3. クエリするプロパティを持つアセットを選択します。
4. 矢印アイコンを選択して資産階層を展開し、目的の資産を見つけてください。
5. プロパティの種類のタブを選択します。たとえば、測定プロパティの現在の値を表示するには、[測定] を選択します。



6. 表示するプロパティを検索します。現在の値が [最新の値] 列に表示されます。

アセットプロパティの現在の値 (AWS CLI) をクエリします。

AWS Command Line Interface (AWS CLI) を使用して、アセットプロパティの現在の値をクエリできます。

GetAssetPropertyValueオペレーションを使用して、アセットプロパティの現在の値をクエリします。

アセットプロパティを識別するには、次のいずれかを指定します。

- データの送信先となるアセットプロパティの AND。assetId propertyId
- データストリームのエイリアスである propertyAlias (例えば、/company/windfarm/3/turbine/7/temperature)。このオプションを使用するには、最初にアセットプロパティのエイリアスを設定する必要があります。プロパティエイリアスを設定するには、を参照してくださいアセットプロパティへの産業データストリームのマッピング。

アセットプロパティの現在の値を取得するには ()AWS CLI

- 次のコマンドを実行して、アセットプロパティの現在の値を取得します。*asset-id* をアセットの ID に置き換え、*property-id* をプロパティの ID に置き換えます。

```
aws iotsitewise get-asset-property-value \
  --asset-id asset-id \
  --property-id property-id
```

オペレーションは、プロパティの現在の TQV を含むレスポンスを次の形式で返します。

```
{
  "propertyValue": {
    "value": {
      "booleanValue": Boolean,
      "doubleValue": Number,
      "integerValue": Number,
      "stringValue": "String"
    },
    "timestamp": {
      "timeInSeconds": Number,
      "offsetInNanos": Number
    },
    "quality": "String"
  }
}
```

履歴アセットプロパティ値をクエリする

AWS IoT SiteWise API GetAssetPropertyValueHistoryオペレーションを使用して、アセットプロパティの履歴値をクエリできます。

アセットプロパティを識別するには、次のいずれかを指定します。

- データの送信先となるアセットプロパティの AND。assetId propertyId
- データストリームのエイリアスである propertyAlias (例えば、/company/windfarm/3/turbine/7/temperature)。このオプションを使用するには、最初にアセットプロパティのエイリアスを設定する必要があります。プロパティエイリアスを設定するには、を参照してくださいアセットプロパティへの産業データストリームのマッピング。

次のパラメータを渡して結果を絞り込みます。

- startDate - 履歴データをクエリる範囲の開始点を (Unix エポックタイムで秒単位)。
- endDate - 履歴データをクエリする範囲の終端を (Unix エポックタイムで秒単位)。
- maxResults - 1 回のリクエストで返す結果の最大数。デフォルトは 20 結果です。
- nextToken - このオペレーションの前の呼び出しから返されたページ割りトークン。
- timeOrdering - 返された値に適用する順序。ASCENDING または DESCENDING。
- qualities – 結果をフィルタリングする品質: GOOD、BAD または UNCERTAIN。

トピック

- アセットプロパティ ()AWS CLIの値履歴をクエリします。

アセットプロパティ ()AWS CLIの値履歴をクエリします。

アセットプロパティの値履歴をクエリするには (AWS CLI)

1. 次のコマンドを実行して、アセットプロパティの値の履歴を取得します。このコマンドは、特定の 10 分間隔でプロパティの履歴をクエリします。*asset-id* をアセットの ID に置き換え、*property-id* をプロパティの ID に置き換えます。日付パラメータをクエリする間隔に置き換えます。

   ```
   aws iotsitewise get-asset-property-value-history \
     --asset-id asset-id \
     --property-id property-id \
     --start-date 1575216000 \
     --end-date 1575216600
   ```

   このオペレーションは、プロパティの履歴 TQV を含むレスポンスを以下の形式で返します。

```
{
  "assetPropertyValueHistory": [
    {
      "value": {
        "booleanValue": Boolean,
        "doubleValue": Number,
        "integerValue": Number,
        "stringValue": "String"
      },
      "timestamp": {
        "timeInSeconds": Number,
        "offsetInNanos": Number
      },
      "quality": "String"
    }
  ],
  "nextToken": "String"
}
```

2. 値エントリが他にも存在する場合は、nextTokenそのフィールドから操作への以降の呼び出しにページネーショントークンを渡すことができます。GetAssetPropertyValueHistory

アセットプロパティ集計のクエリ

AWS IoT SiteWise 複数の時間間隔で計算される基本指標の集約された資産プロパティ値を自動的に計算します。 AWS IoT SiteWise アセットプロパティについて、分、時間、日ごとに以下の集計を計算します。

- [average] (平均) - 時間間隔におけるプロパティ値の平均 (中間)。
- [count] (カウント) - 時間間隔におけるプロパティのデータポイント数
- [maximum] (最大) - 時間間隔におけるプロパティの最大値。
- [minimum] (最小) - 時間間隔におけるプロパティの最小値。
- [standard deviation] (標準偏差) - 時間間隔におけるプロパティ値の標準偏差。
- [sum] (合計) - 時間間隔におけるプロパティ値の合計。

文字列やブール値などの数値以外のプロパティでは、 AWS IoT SiteWise カウント集計のみを計算します。

アセットデータのカスタムメトリクスを計算することもできます。メトリクスプロパティでは、操作に固有の集計を定義します。メトリクスプロパティには、API 用に事前に計算されていない追加の集計関数と時間間隔があります。 AWS IoT SiteWise 詳細については、「プロパティおよびその他のアセットのデータ (指標) の集計」を参照してください。

トピック

- アセットプロパティ (API) の集計
- アセットプロパティ () の集計AWS CLI

アセットプロパティ (API) の集計

AWS IoT SiteWise API を使用してアセットプロパティの集計を取得できます。

GetAssetPropertyAggregatesオペレーションを使用して、アセットプロパティの集計をクエリします。

資産プロパティを識別するには、次のいずれかを指定します。

- データの送信先となるアセットプロパティの AND。`assetId` `propertyId`
- データストリームのエイリアスである `propertyAlias` (例えば、`/company/windfarm/3/turbine/7/temperature`)。このオプションを使用するには、最初にアセットプロパティのエイリアスを設定する必要があります。プロパティエイリアスを設定するには、を参照してくださいアセットプロパティへの産業データストリームのマッピング。

また、次の必須パラメータも渡す必要があります。

- `aggregateTypes` - 取得する集計のリスト。`AVERAGE`、`COUNT`、`MAXIMUM`、`MINIMUM`、`STANDARD_DEVIATION`、`SUM` のいずれかを指定できます。
- `resolution` - メトリクスを取得する時間間隔を指定します。`1m` (1 分) 、`1h` (1 時間) 、`1d` (1 日) のいずれかを指定します。
- `startDate` - 履歴データをクエリる範囲の開始点を (Unix エポックタイムで秒単位)。
- `endDate` - 履歴データをクエリする範囲の終端を (Unix エポックタイムで秒単位)。

次のパラメータのいずれかを渡して、結果を絞り込むこともできます。

- `maxResults` - 1 回のリクエストで返す結果の最大数。デフォルトは `20` 結果です。

- nextToken - このオペレーションの前の呼び出しから返されたページ割りトークン。
- timeOrdering - 返された値に適用する順序。ASCENDING または DESCENDING。
- qualities – 結果をフィルタリングする品質: GOOD、BAD または UNCERTAIN。

> **Note**
>
> GetAssetPropertyAggregatesこのオペレーションは、このセクションで説明する他のオペレーションとは異なる形式の TQV を返します。value 構造には、リクエスト内の各 aggregateTypes のフィールドが含まれます。timestamp には、集計が発生した時間 (秒単位の Unix エポック時間) が含まれます。

アセットプロパティ () の集計AWS CLI

アセットプロパティのアグリゲートをクエリするには ()AWS CLI

1. アセットプロパティの集計を取得するには、次のコマンドを実行します。このコマンドは、特定の 1 時間間隔で 1 時間の分解能で平均と合計をクエリします。*asset-id* をアセットの ID に置き換え、*property-id* をプロパティの ID に置き換えます。パラメータをクエリする集計と間隔に置き換えます。

```
aws iotsitewise get-asset-property-aggregates \
  --asset-id asset-id \
  --property-id property-id \
  --start-date 1575216000 \
  --end-date 1575219600 \
  --aggregate-types AVERAGE SUM \
  --resolution 1h
```

   オペレーションは、プロパティの履歴 TQV を含むレスポンスを次の形式で返します。レスポンスには、リクエストされた集計のみが含まれます。

```
{
  "aggregatedValues": [
    {
      "timestamp": Number,
      "quality": "String",
      "value": {
```

```
        "average": Number,
        "count": Number,
        "maximum": Number,
        "minimum": Number,
        "standardDeviation": Number,
        "sum": Number
      }
    }
  ],
  "nextToken": "String"
}
```

2. 値のエントリが他にも存在する場合は、nextTokenそのフィールドから以降のオペレーションの呼び出しにページネーショントークンを渡すことができます。GetAssetPropertyAggregates

AWS IoT SiteWise クエリ言語

AWS IoT SiteWise データ取得 ExecuteQueryAPI オペレーションでは、宣言型構造定義とそれに関連する時系列データに関する情報を以下から取得できます。

- モデル
- アセット
- 測定値
- メトリクス
- 変換する
- 集合体

これは SQL のようなクエリステートメントを使用して、1 回の API リクエストで実行できます。

> **Note**
>
> この機能は、 AWS GovCloud (米国西部) を除き、 AWS IoT SiteWise AWS IoT TwinMaker とが両方とも利用できるすべてのリージョンで利用できます。

トピック

- 前提条件

- クエリ言語リファレンス

前提条件

AWS IoT SiteWise AWS IoT TwinMaker 産業データを整理してモデル化できるようにするには、との統合権限が必要です。

モデル、資産、測定値、指標、変換、集計に関する情報を取得する前に、次の前提条件が満たされていることを確認してください。

- 両方のサービスにリンクされたロールと、での設定。 AWS IoT SiteWise AWS IoT TwinMaker AWS アカウントサービスリンクロールの詳細については、「IAM ユーザーガイド」の「サービスリンクロールの使用」を参照してください。
- IAM AWS IoT SiteWise ロールのインテグレーションが有効になっている。詳細については、「AWS IoT SiteWise と AWS IoT TwinMaker の統合」を参照してください。
- `IoTSiteWiseDefaultWorkspace`リージョンのアカウントにある ID AWS IoT TwinMaker の付いたワークスペース。詳細については、IoTSiteWiseDefaultWorkspace ユーザーガイドの「AWS IoT TwinMaker の使用」を参照してください。
- AWS IoT TwinMaker 標準または段階的なバンドル価格設定モードのいずれかが有効になっています。詳しくは、AWS IoT TwinMaker ユーザーガイドの「AWS IoT TwinMaker 価格設定モードの切り替え」を参照してください。

クエリ言語リファレンス

AWS IoT SiteWise データを扱うための豊富なクエリ言語をサポートします。使用可能なデータ型、演算子、関数、および構成については、以下のトピックで説明します。

クエリの例 AWS IoT SiteWise クエリ言語でクエリを記述するにはを参照してください。

トピック

- ビューを理解する
- サポートされている データ型
- SELECT ステートメントを使用してデータを取得します。
- 論理演算子
- 比較演算子

- クエリの例

ビューを理解する

このセクションでは、プロセスメタデータやテレメトリデータなど AWS IoT SiteWise、のビューを理解するのに役立つ情報を提供します。

以下の表は、ビューの名前とビューの説明をまとめたものです。

データモデル

ビュー名	ビューの説明
アセット	アセットとモデルの派生に関する情報が含まれます。
アセットプロパティ	アセットプロパティの構造に関する情報が含まれます。
raw_time_series	時系列の履歴データが含まれます。
最新価値時系列	時系列の最新の値が含まれます。
事前計算済み_集計	自動的に計算された集計された資産プロパティ値が含まれます。これらは複数の時間間隔で計算される基本的な指標のセットです。

以下のビューには、クエリの列名とサンプルデータが表示されます。

ビュー:アセット

アセット ID	アセット名	アセットの説明	アセットモデル ID
88898498-0b8b-42b5-bf57-16180bc3d3a0	WindTurbine A	WindTurbine アセット A	17847250-5bf0-4f74-b775-cc03f05e7cb8
17847250-5bf0-4f74-b775-cc03f05e7cb8	風力タービン資産モデル	風力発電所のタービンを表します。	

ビュー:アセットプロパティ

プロパティ ID	アセット ID	プロパティ名	プロパティ_データ_タイプ	プロパティ_エイリアス	アセット_コンポジット_モデル_ID
b29be434-b000-4d74-b809-75287d83bcd6	88898498-0b8b-42b5-bf57-16180bc3d3a0	モーター温度	ダブル	Rochester 2/44///Line-5/Bus-2/Machine-5/Temperature	
3b458f00-24e7-458a-b4e8-c6026eff654a	88898498-0b8b-42b5-bf57-16180bc3d3a0	風向き	ダブル	/company/windfarm/3/turbine/7/winddirection	2f458n00-56e7-458h-b4e8-c6026eff985g

ビュー:RAW_TIME_SERIES

アセット ID	プロパティ ID	プロパティ_エイリアス	イベント_タイムスタンプ	クオリティ	ブーリアン値	int_value	ダブルバリュー	文字列_値
88898498-0b8b-42b5-bf57-16180bc3d3a	b29be434-b000-4d74-b809-75287d83bcd	Rochester 2/44///Line-5/Bus-2/Machine-5/Temperature	1575219600	GOOD			115.0	

アセット ID	プロパティ ID	プロパティ_エイリアス	イベント_タイムスタンプ	クオリティ	ブーリアン値	int_value	ダブルバリュー	文字列_値
88898498-0b8b-42b5-bf57-16180bc3d3a	3b458f00-24e7-458a-b4e8-c6026eff654a	/company/windfarm/3/turbine/7/winddirection	1575219377	GOOD			348.75	

> **Note**
>
> ビューをクエリするには、event_timestamp列にフィルター句を含める必要があります。raw_time_seriesこれは必須のフィルターで、これがないとクエリは失敗します。
>
> Example query
>
> ```
> SELECT event_timestamp, double_value FROM raw_time_series WHERE event_timestamp
>  > 1234567890
> ```

ビュー:最新値時系列

アセット ID	プロパティ ID	プロパティ_エイリアス	イベント_タイムスタンプ	クオリティ	ブーリアン値	int_value	ダブルバリュー	文字列_値
88898498-0b8b-42b5-bf57-16180bc3d3a	3b458f00-24e7-458a-b4e8-c6026eff654a	/company/windfarm/3/	1575219600	GOOD			355.39	

アセット ID	プロパティ ID	プロパティ_エイリアス	イベント_タイムスタンプ	クオリティ	ブーリアン値	int_value	ダブルバリュー	文字列_値
		turbine/7/winddirection						

ビュー:事前計算済みアグリゲート

アセット ID	プロパティ ID	プロパティ_エイリアス	イベント_タイムスタンプ	resolution	合計値	カウント・バリュー	アベレージ・バリュー	[最大値]	[最小値]	標準開発値
888984 0b8b-42 - bf57-16 80bc3d	b29be4 b000-4c - b809-75 87d83b	Roches 2/44// Li ne-5/ Bus- 2/ Machin -5/ Temper ature	157521 0	15m	1105.48	15	73.4	80.6	68	3.64

サポートされている データ型

AWS IoT SiteWise クエリ言語は次のデータ型をサポートします。

ビュー:アセット

データタイプ	説明
STRING	最大長が 1024 バイトの文字列。
INTEGER	範囲がの符号付き 32 ビット整数。-2,147,483,648 to 2,147,483,647
DOUBLE	範囲が from –10^100 to 10^100 IEEE 754 で倍精度をもつ浮動小数点数。
BOOLEAN	true-または-false

> **Note**
>
> 倍精度データは正確ではありません。一部の値は正確に変換されず、精度が限られているためすべての実数を表すわけではありません。クエリ内の浮動小数点データは、内部で表現される値と同じではない場合があります。入力数値の精度が高すぎる場合、値は四捨五入されます。

SELECT ステートメントを使用してデータを取得します。

SELECTこのステートメントは 1 つ以上のビューからデータを取得するために使用されます。AWS IoT SiteWise JOIN暗黙のビューをサポートします。結合するビューは、カンマで区切って (FROMSELECTステートメントの節に) リストできます。

Example

SELECT次のステートメントを使用してください。

```
SELECT select_expr [, ...]
[ FROM from_item [, ...] ]
[ WHERE [LIKE condition ESCAPE condition] ]
```

前の例では、LIKEこの句はワイルドカードを使用して検索条件とフィルター条件を指定しています。 AWS IoT SiteWise percentage (%)ワイルドカード文字としてサポートします。

Example %条件で使うには:

```
Prefix search: String%
Infix search: %String%
Suffix search: %String
```

Example アセットを検索するには:

```
SELECT asset_name, asset_description FROM asset WHERE asset_name LIKE 'Wind%'
```

Example ESCAPE 条件を使用してアセットを検索するには:

```
SELECT asset_name, asset_description FROM asset WHERE asset_name LIKE 'room\%' ESCAPE
 '\'
```

論理演算子

AWS IoT SiteWise 以下の論理演算子をサポートします。

論理演算子

オペレータ	説明	例
AND	TRUE両方の値が true の場合	a AND b

a または b のどちらかがの場合FALSE、前の式は false と評価されます。AND演算子が true と評価されるには、a と b の両方が true でなければなりません。

Example

```

SELECT a.asset_name
FROM asset as a, latest_value_time_series as t
WHERE t.int_value > 30 AND t.event_timestamp > 1234567890
```

比較演算子

AWS IoT SiteWise 以下の比較演算子をサポートします。

論理演算子

オペレータ	説明
<	Less than
>	Greater than
<=	以下
>=	以上
=	等しい
!=	Not equal

クエリの例

メタデータのフィルタリング

次の例は、SELECT AWS IoT SiteWise クエリ言語を使用したステートメントによるメタデータフィルタリングの例です。

```
SELECT a.asset_name, p.property_name
FROM asset a, asset_property p
WHERE a.asset_id = p.asset_id AND a.asset_name LIKE '%windmill%'
```

値フィルタリング

以下は、SELECT AWS IoT SiteWise クエリ言語を含むステートメントを使用した値フィルタリングの例です。

```
SELECT a.asset_name FROM asset a, raw_time_series r
WHERE a.asset_id = r.asset_id AND r.int_value > 30  AND r.event_timestamp > 1234567890
 AND r.event_timestamp < 1234567891
```

AWS 他のサービスとのやり取り

AWS IoT SiteWise アセットデータを AWS IoT MQTT パブリッシュ/サブスクライブメッセージブローカーに公開できるため、他のサービスのアセットデータを操作できます。 AWS AWS IoT SiteWise 各アセットプロパティに固有の MQTT トピックを割り当てます。このトピックを使用すると、コアルールを使用してアセットデータを他のサービスにルーティングできます。 AWS AWS IoT たとえば、 AWS IoT 以下のタスクを実行するようにコアルールを設定できます。

- 機器の故障を特定し、[AWS IoT Events] にデータを送信して適切な担当者に通知します。
- データを Amazon DynamoDB に送信することにより、外部ソフトウェアソリューションで使用する選択したアセットデータを履歴化します。
- [AWS Lambda] 関数をトリガーして週次レポートを生成します。

DynamoDB にプロパティ値を保存するルールを設定するのに必要なステップを説明するチュートリアルに従います。詳細については、「プロパティ値の更新を Amazon DynamoDB に公開する」を参照してください。

ルールの設定方法の詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の [Rules] (ルール) を参照してください。

また、 AWS 他のサービスのデータをに戻すこともできます AWS IoT SiteWise。 AWS IoT SiteWise ルールアクションを通じてデータを取り込む方法については、を参照してくださいルールを使ったデータの取り込み AWS IoT Core。

トピック

- アセットプロパティの MQTT トピックについて
- アセットプロパティ通知の操作。
- アセットプロパティ通知を使用して Amazon S3 にデータをエクスポートする
- Grafana との統合。
- AWS IoT SiteWise と AWS IoT TwinMaker の統合
- Amazon Lookout for Equipment による機器の異常の検出

アセットプロパティの MQTT トピックについて

すべてのアセットプロパティには、次の形式で一意の MQTT トピックパスがあります。

```
$aws/sitewise/asset-models/assetModelId/assets/assetId/properties/propertyId
```

> **Note**
>
> AWS IoT SiteWise コアルールエンジンの # (マルチレベル) トピックフィルターワイルドカードはサポートされていません。 AWS IoT + (単数レベル) ワイルドカードを使用できます。たとえば、次のトピックフィルターを使用して、特定のアセットモデルのすべての更新を照合できます。
>
> ```
> $aws/sitewise/asset-models/assetModelId/assets/+/properties/+
> ```
>
> トピックフィルターのワイルドカードの詳細については、[AWS IoT Core Developer Guide] (Core デベロッパーガイド) の [Topics] (トピック) を参照してください。

アセットプロパティ通知の操作。

プロパティ通知を有効にしてアセットデータの更新を公開し AWS IoT Core、データに対してクエリを実行できます。 AWS IoT SiteWise アセットプロパティ通知には、 AWS IoT SiteWise データを Amazon S3 AWS CloudFormation にエクスポートするために使用できるテンプレートが用意されています。

> **Note**
>
> アセットデータは、値が変更されたかどうかに関係なく AWS IoT SiteWise、 AWS IoT Core が受信するたびにに送信されます。

トピック

- アセットプロパティの通知の有効化 (コンソール)
- アセットプロパティ通知を有効にする (AWS CLI)
- アセットプロパティ通知メッセージのクエリ

アセットプロパティの通知の有効化 (コンソール)

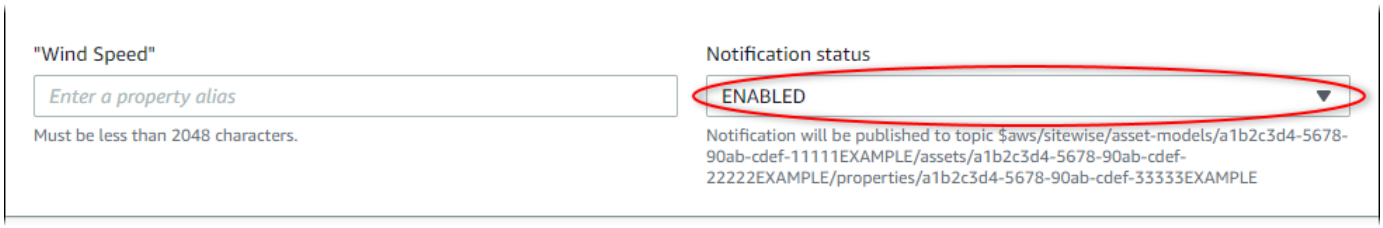
デフォルトでは、 AWS IoT SiteWise プロパティ値の更新は公開されません。 AWS IoT SiteWise コンソールを使用してアセットプロパティの通知を有効にできます。

アセットプロパティの通知を有効または無効にするには (コンソール)

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [アセット] を選択します。
3. アセットを選択して、プロパティの通知を有効にします。

> **Tip**
> 矢印アイコンを選択して、アセット階層を展開してアセットを検索できます。

4. [編集] を選択します。
5. アセットプロパティの [通知ステータス] で、[有効] を選択します。



また、[無効] を選択して、アセットプロパティの通知を無効にすることもできます。

6. [保存] を選択します。

アセットプロパティ通知を有効にする (AWS CLI)

デフォルトでは、 AWS IoT SiteWise プロパティ値の更新は公開されません。 AWS Command Line Interface (AWS CLI) を使用してアセットプロパティの通知を有効または無効にできます。

この手順を完了するには、アセットの `assetId` とプロパティの `propertyId` を知っている必要があります。外部 ID を使用することもできます。作成したアセットがわからない場合は`assetId`、ListAssetsAPI を使用して特定のモデルのすべてのアセットを一覧表示します。DescribeAssetオペレーションを使用して、プロパティ ID を含むアセットのプロパティを表示します。

UpdateAssetPropertyオペレーションを使用して、アセットプロパティの通知を有効または無効にします。以下のパラメータを指定します。

- `assetId` - アセットの ID。
- `propertyId` - アセットプロパティの ID。
- `propertyNotificationState` - プロパティ値の通知状態: `ENABLED` または `DISABLED`。
- `propertyAlias` - プロパティのエイリアス。通知状態を更新するときに、プロパティの既存のエイリアスを指定します。このパラメータを省略すると、プロパティの既存のエイリアスは削除されます。

アセットプロパティの通知を有効または無効にするには (CLI)

1. 次のコマンドを実行して、アセットプロパティのエイリアスを取得します。*asset-id* をアセットの ID に置き換え、*property-id* をプロパティの ID に置き換えます。

```
aws iotsitewise describe-asset-property \
  --asset-id asset-id \
  --property-id property-id
```

このオペレーションは、アセットプロパティの詳細を含むレスポンスを次の形式で返します。プロパティエイリアスは JSON オブジェクト `assetProperty.alias` にあります。

```
{
  "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
  "assetName": "Wind Turbine 7",
  "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
  "assetProperty": {
    "id": "a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
    "name": "Wind Speed",
    "alias": "/company/windfarm/3/turbine/7/windspeed",
    "notification": {
      "topic": "$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/
assets/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE/properties/a1b2c3d4-5678-90ab-
cdef-33333EXAMPLE",
      "state": "DISABLED"
    },
    "dataType": "DOUBLE",
    "unit": "m/s",
    "type": {
```

```
      "measurement": {}
    }
  }
}
```

2. アセットプロパティの通知を有効にするには、次のコマンドを実行します。*property-alias* は、前のコマンドのレスポンスのプロパティエイリアスに置き換えるか、--property-alias を省略して、エイリアスなしでプロパティを更新します。

```
aws iotsitewise update-asset-property \
  --asset-id asset-id \
  --property-id property-id \
  --property-notification-state ENABLED \
  --property-alias property-alias
```

また、--property-notification-state DISABLED を渡して、アセットプロパティの通知を無効にすることもできます。

アセットプロパティ通知メッセージのクエリ

アセットプロパティ通知をクエリするには、SQL AWS IoT Core ステートメントで構成されるルールを作成します。

AWS IoT SiteWise 資産プロパティデータの更新を次の形式で AWS IoT Core に公開します。

```
{
  "type": "PropertyValueUpdate",
  "payload": {
    "assetId": "String",
    "propertyId": "String",
    "values": [
      {
        "timestamp": {
          "timeInSeconds": Number,
          "offsetInNanos": Number
        },
        "quality": "String",
        "value": {
          "booleanValue": Boolean,
          "doubleValue": Number,
          "integerValue": Number,
```

```
          "stringValue": "String"
        }
      }
    ]
  }
}
```

valuesリスト内の各構造は timestamp-quality-value (TQV) 構造です。

- timestamp には、ナノ秒のオフセットがある現在の UNIX エポック時刻 (秒単位) が含まれます。
- quality には、データポイントの品質を示す、以下のいずれかの文字列が含まれます。
  - GOOD - データはいずれの問題の影響も受けません。
  - BAD - データはセンサーの障害などの問題による影響を受けます。
  - UNCERTAIN - データはセンサーの不正確さなどの問題による影響を受けます。
- value には、プロパティのタイプに応じて、以下のいずれかのフィールドが含まれます。
  - booleanValue
  - doubleValue
  - integerValue
  - stringValue

values 配列から値を解析するには、ルールの SQL ステートメントで複雑にネストされたオブジェクトクエリを使用する必要があります。詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Nested object queries] (オブジェクトクエリのネスト) を参照するか、アセットプロパティ通知メッセージの解析の具体例については、プロパティ値の更新を Amazon DynamoDB に公開する チュートリアルを参照してください。

Example 値の配列を抽出するクエリの例

次のステートメントは、そのプロパティを持つすべてのアセットの特定の Double 型プロパティについて、更新されたプロパティ値の配列をクエリする方法を示しています。

```
SELECT
  (SELECT VALUE (value.doubleValue) FROM payload.values) AS windspeed
FROM
  '$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/assets/+/
properties/a1b2c3d4-5678-90ab-cdef-33333EXAMPLE'
WHERE
```

```
  type = 'PropertyValueUpdate'
```

前のルールクエリステートメントは、次の形式でデータを出力します。

```
{
  "windspeed": [
    26.32020195042838,
    26.282584572975477,
    26.352566977372508,
    26.283084346171442,
    26.571883739599322,
    26.60684140743005,
    26.628738636715045,
    26.273486932802125,
    26.436379105473964,
    26.600590095377303
  ]
}
```

Example 単一の値を抽出するクエリの例

次のステートメントは、そのプロパティを持つすべてのアセットの特定の Double 型プロパティについて、プロパティ値の配列から最初の値をクエリする方法を示しています。

```
SELECT
  get((SELECT VALUE (value.doubleValue) FROM payload.values), 0) AS windspeed
FROM
  '$aws/sitewise/asset-models/a1b2c3d4-5678-90ab-cdef-11111EXAMPLE/assets/+/
properties/a1b2c3d4-5678-90ab-cdef-33333EXAMPLE'
WHERE
  type = 'PropertyValueUpdate'
```

前のルールクエリステートメントは、次の形式でデータを出力します。

```
{
  "windspeed": 26.32020195042838
}
```

> **Important**
>
> このルールクエリステートメントは、各バッチの最初の値以外の値の更新を無視します。各バッチには最大 10 個の値を含めることができます。残りの値を含める必要がある場合は、アセットプロパティ値を他のサービスに出力するより複雑なソリューションを設定する必要があります。たとえば、 AWS Lambda 配列内の各値を別のトピックに再公開するアクションを含むルールを設定し、そのトピックをクエリして各値を目的のルールアクションに公開する別のルールを設定できます。

アセットプロパティ通知を使用して Amazon S3 にデータをエクスポートする

受信データを からアカウントの AWS IoT SiteWise Amazon S3 バケットにエクスポートできます。履歴レポートを作成したり、複雑な方法でデータを分析したりできる形式でデータをバックアップできます。

> **Note**
>
> AWS IoT SiteWise は、カスタマー管理の Amazon S3 バケットにデータを保存できるコールド階層ストレージもサポートしています。対応するストレージ層については、ストレージの管理 を参照してください。

AWS IoT SiteWise は、この機能を AWS CloudFormation テンプレートとして提供します。テンプレートからスタックを作成すると、 は受信データを から S3 バケット AWS IoT SiteWise にストリーミングするために必要な AWS リソース AWS CloudFormation を作成します。

次に、S3 バケットは、プロパティ値の更新メッセージから送信されたすべてのアセット AWS IoT SiteWise プロパティデータを受信します。S3 バケットは、アセットとプロパティ名、およびその他の情報を含むアセットメタデータも受け取ります。

Amazon S3 にエクスポートするアセットプロパティのプロパティ値更新メッセージを有効にする方法の詳細については、AWS 他のサービスとのやり取り を参照してください。

この機能は、アセットのプロパティデータとアセットのメタデータを Apache Parquet フォーマットで Amazon S3 に保存します。Parquet は、容量を節約し、JSON のような行指向の形式に比べ、より高速なクエリを可能にするカラム型のデータ形式です。

> **Note**
>
> この機能では、アセットメタデータを取得するときに、最大約 1,500 のアセットがサポートされます。この制限は、アセットメタデータにのみ適用されます。この制限は、アセットプロパティデータをエクスポートするときにサポートされるアセットの数には適用されません。

各リソースの名前には、スタックの作成時にカスタマイズできるプレフィックスが付いています。リソースには次が含まれます。

- Amazon S3 バケット
- AWS Lambda 関数
- AWS IoT Core ルール
- AWS Identity and Access Management ロール
- Amazon Data Firehose ストリーム
- AWS Glue データベース

詳細な一覧については、「テンプレートから作成されたリソース」を参照してください。

> **Important**
>
> この AWS CloudFormation テンプレートが作成および消費するリソースに対して課金されます。これらの料金には、複数の AWS サービスのデータストレージとデータ転送が含まれます。

トピック

- AWS CloudFormation スタックを作成する
- Amazon S3 でデータを表示する
- Amazon Athena でエクスポートされたデータを分析
- テンプレートから作成されたリソース

AWS CloudFormation スタックを作成する

アセットデータを Amazon S3 にエクスポート AWS CloudFormation するには、 でスタックを作成する必要があります。 Amazon S3

Amazon S3 へデータをエクスポートするには。

1. AWS CloudFormation テンプレートを開き、 AWS Management Consoleにサインインします。
2. [スタックの作成] ページで、ページの下部にある [次へ] を選択します。
3. スタックの詳細の指定 ページで、アセットデータを受信するためにこのテンプレートが作成する BucketName S3 バケットの を入力します。このバケット名は グローバルに 一意である必要があります。詳細については、Amazon Simple Storage Service ユーザーガイドでバケットの命名規則について参照してください。
4. (オプション) テンプレートのその他のパラメータを変更します。
   - GlobalResourcePrefix – このテンプレートから作成される IAM ロールなど、グローバルリソースの名前のプレフィックス。
   - LocalResourcePrefix - 現在のリージョンでこのテンプレートから作成されるリソースの名前のプレフィックス。

   > **Note**
   > このテンプレートを複数回作成する場合は、リソース名の衝突を避けるために、バケット名とリソースプレフィックスのパラメータを変更する必要があります。
5. [Next (次へ)] を選択します。
6. [スタックオプションの設定] ページで、[Next (次へ)] を選択します。
7. ページの下部で、[I acknowledge that AWS CloudFormation might create IAM resources] (IAM リソースが作成される場合があることを承認します) チェックボックスをオンにします。
8. [スタックの作成] を選択します。

   スタックが作成されるまで数分かかります。スタックの作成に失敗した場合、アカウントに十分な権限がないか、すでに存在するバケット名を入力している可能性があります。スタックを削除して再試行するには、次の手順を実行します。

   a. 右上隅の [削除] を選択します。

スタックの削除には数分かかります。

> **Note**
>
> AWS CloudFormation は S3 バケットや CloudWatch ロググループを削除しません。これらのリソースは、それらのサービスのコンソールで削除できます。

b. スタックの削除に失敗した場合は、[削除] をもう一度選択します。

c. スタックを再度削除できない場合は、 AWS CloudFormation コンソールの手順に従って削除に失敗したリソースをスキップし、もう一度試してください。

9. AWS CloudFormation スタックが正常に作成されたら、次の手順に従って Amazon S3 のアセットプロパティデータを調べます。

> **Important**
>
> スタックを作成すると、 AWS アカウント内の新しいリソースが表示されます。これらのリソースを削除または変更すると、機能が正常に動作しなくなることがあります。バケットへのデータの送信を停止する場合や、この機能をカスタマイズする場合を除き、これらのリソースは変更しないことをお勧めします。

Amazon S3 でデータを表示する

機能を作成したら、Amazon S3 でアセットプロパティデータとアセットメタデータを表示できます。

> **Note**
>
> アセットのメタデータは 6 時間ごとに更新されます。S3 バケットにアセットメタデータが表示されるまで、最大 6 時間待つ場合があります。

この機能では、アセットプロパティデータが次の列に格納されます。各行にはデータポイントが含まれます。

- type – プロパティ通知の型 (`PropertyValueUpdate`)。
- asset_id - データポイントを受け取ったアセットの ID。

- asset_property_id - アセットのデータポイントを受け取ったプロパティの ID。
- time_in_seconds - データが受信された時刻。UNIX エポック時間で秒単位で表されます。
- offset_in_nanos - `timeInSeconds` からのナノ秒オフセット。
- asset_property_quality - データポイントの品質: `GOOD`、`UNCERTAIN`、または `BAD`。
- asset_property_value - データポイントの値。
- asset_property_data_type - アセットプロパティのデータ型: `boolean`、`double`、`integer`、または `string`。

この機能では、アセットメタデータが次の列に格納されます。各行にはアセットプロパティが含まれます。

- asset_id - アセットの ID。
- asset_name - アセットの名前。
- asset_model_id - アセットのモデルの ID。
- asset_property_id - アセットプロパティの ID。
- asset_property_name - アセットプロパティの名前。
- asset_property_data_type - アセットプロパティのデータ型: `BOOLEAN`、`DOUBLE`、`INTEGER`、または `STRING`。
- asset_property_unit - アセットプロパティの単位。
- asset_property_alias - アセットプロパティのエイリアス。

Amazon S3 で AWS IoT SiteWise データを表示するには

1. [Amazon S3 console] (Amazon S3 のコンソール) に移動します。
2. バケットのリストから、テンプレートの作成時に選択した名前のバケットを選択します。
3. バケットで、次のいずれかのフォルダを選択します。

   - `asset-property-updates` – このフォルダには、 からエクスポートされたアセットプロパティデータが含まれています AWS IoT SiteWise。
   - `asset-metadata` – このフォルダには、 からエクスポートされたアセットの詳細が含まれています AWS IoT SiteWise。
4. 表示するオブジェクトを選択します。
5. オブジェクトのページで、次の操作を行います。

a. [Select from (選択元)] タブを選択します。

   このパネルでは、Parquet ファイルのレコードをプレビューできます。

b. [File format (ファイル形式)] で [Parquet] を選択します。

c. ファイルの内容を JSON 形式で表示するには、[Show file preview] (ファイルのプレビューを表示) を選択します。

> **Note**
>
> バケットに新しいデータが表示されない場合は、アセットプロパティのプロパティ値の更新通知を有効にしていることを確認してください。詳細については、「AWS 他のサービスとのやり取り」を参照してください。

S3 バケットに保存されているアセットデータを分析する方法の詳細については、「Amazon Athena でエクスポートされたデータを分析」を参照してください。

Amazon Athena でエクスポートされたデータを分析

Amazon S3 でアセットプロパティデータを取得したら、複数の AWS サービスを使用してレポートを生成したり、データを分析およびクエリしたりできます。

- [Amazon Athena] を使用して、データの SQL クエリを実行します。
- [Amazon EMR] を使用してビッグデータ分析を実行します。
- Amazon OpenSearch Service を使用してデータを検索および分析します。

の Analytics にリストされている Amazon S3 内のデータとやり取りできる他の AWS のサービスがありますAWS Management Console。

> **Note**
>
> スタックは、アセットプロパティデータをフォーマットするための AWS Glue データベースを作成します。このデータベースではアセットデータをクエリできません。このセクションの手順に従って、クエリできる AWS Glue データベースを作成します。

このチュートリアルでは、Amazon Athena を使用するための前提条件を設定する方法と、Athena を使用してエクスポートされた AWS IoT SiteWise アセットデータに対して SQL クエリを実行する方法について説明します。Athena でデータをクエリするには、まず AWS Glue Data Catalog にアセットデータを入力する必要があります。データカタログにはデータベースとテーブルが含まれており、Athena はデータカタログ内のデータにアクセスすることができる。エクスポートしたアセットデータで Data Catalog を定期的に更新する AWS Glue クローラーを作成できます。

トピック

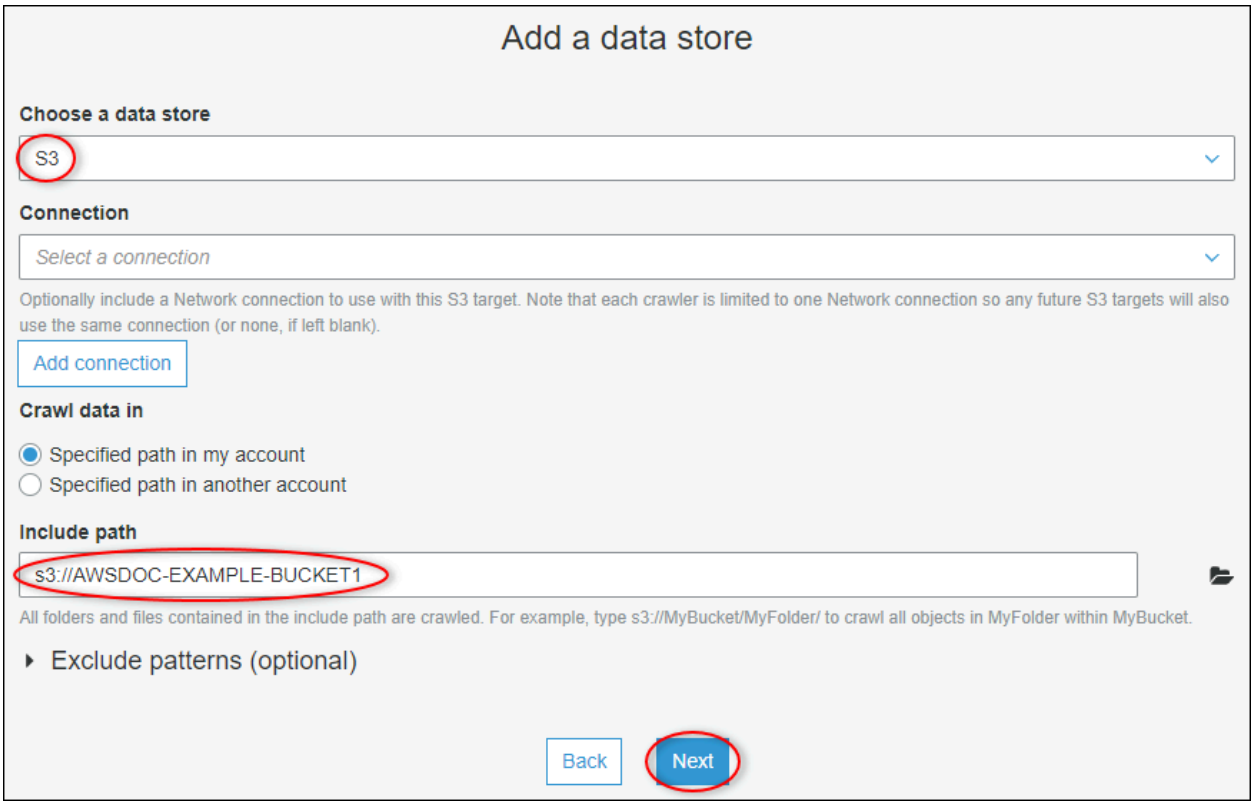
クローラを設定して AWS Glue Data Catalogを入力する

AWS Glue クローラはデータストアをクロールして、 にテーブルを入力します AWS Glue Data Catalog。この手順では、エクスポートされたアセットデータを含む S3 バケットの AWS Glue クローラーを作成して実行します。クローラは、アセットプロパティの更新用のテーブルとアセットメタデータ用のテーブルを作成します。次に、Athena を使用してこれらのテーブルに対して SQL クエリを実行できます。詳細については、[AWS Glue Developer Guide] (デベロッパーガイド) の[Populating the AWS Glue Data Catalog] (投入) と[Defining crawlers] (クローラの定義) を参照してください。

AWS Glue クローラーを作成するには

1. AWS Glue コンソールに移動します。
2. ナビゲーションペインで、[Crawlers (クローラ)] を選択します。
3. [Add crawler (クローラの追加)] を選択します。
4. [Add crawler (クローラの追加)] ページで、次の操作を行います。
    a. クローラの名前 (**`IoTSiteWiseDataCrawler`** など) を入力し、[次へ] を選択します。
    b. [クローラソースタイプ] で、[データストア] を選択し、[次へ] を選択します。
    c. [Add a data store page (データストアページの追加)] で、次の操作を行います。
        i. [Choose a data store (データストアの選択)] で [S3] を選択します。
        ii. [インクルードパス] に、アセットデータバケットをデータストアとして追加するには、**`s3://DOC-EXAMPLE-BUCKET1`** を入力します。DOC-EXAMPLE-BUCKET1 は、スタックの作成時に選択したバケット名に置き換えます。

iii. [Next (次へ)] を選択します。



d. [Add another data store (別のデータストアの追加)] ページで、[いいえ] を選択し、[次へ] を選択します。

e. [Choose an IAM role] (IAM ロールの選択) ページで、次の操作を行います。

i. が S3 バケットにアクセス AWS Glue できるようにする新しいサービスロールを作成するには、「IAM ロールの作成」を選択します。

ii. ロール名のサフィックス (**`IoTSiteWiseDataCrawler`** など) を入力します。

iii. [Next (次へ)] を選択します。

f. [頻度] で [毎時] を選択し、[次へ] を選択します。クローラは、実行するたびに新しいデータでテーブルを更新するので、ユースケースに合う任意の頻度を選択できます。

g. [Configure the crawler's output (クローラの出力を設定する)] ページで、次の操作を行います。

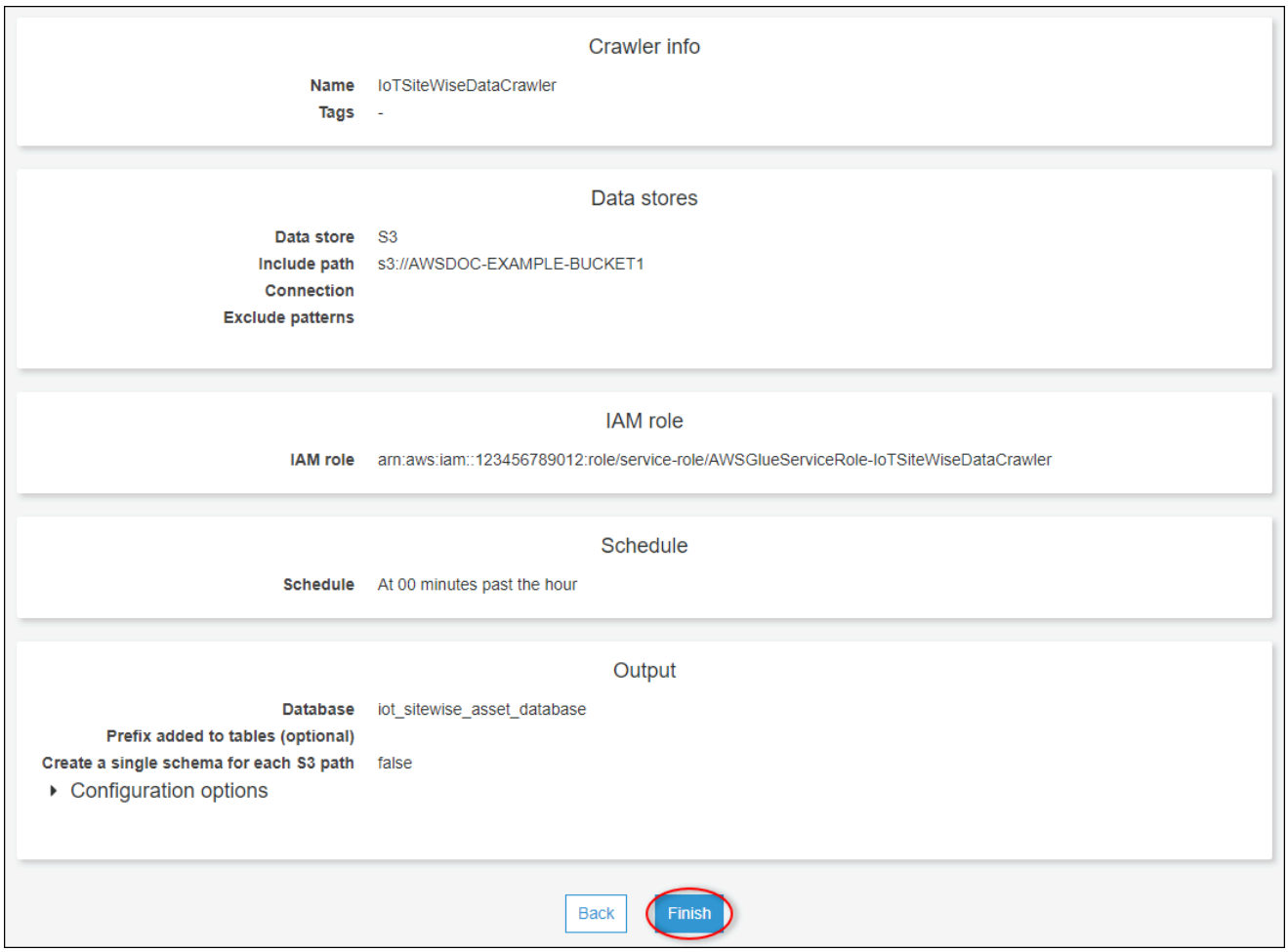
i. データベースを追加 を選択して、アセットデータの AWS Glue データベースを作成します。

ii. データベースの名前 (**`iot_sitewise_asset_database`** など) を入力します。

iii. [Create] (作成) を選択します。

iv. [Next (次へ)] を選択します。

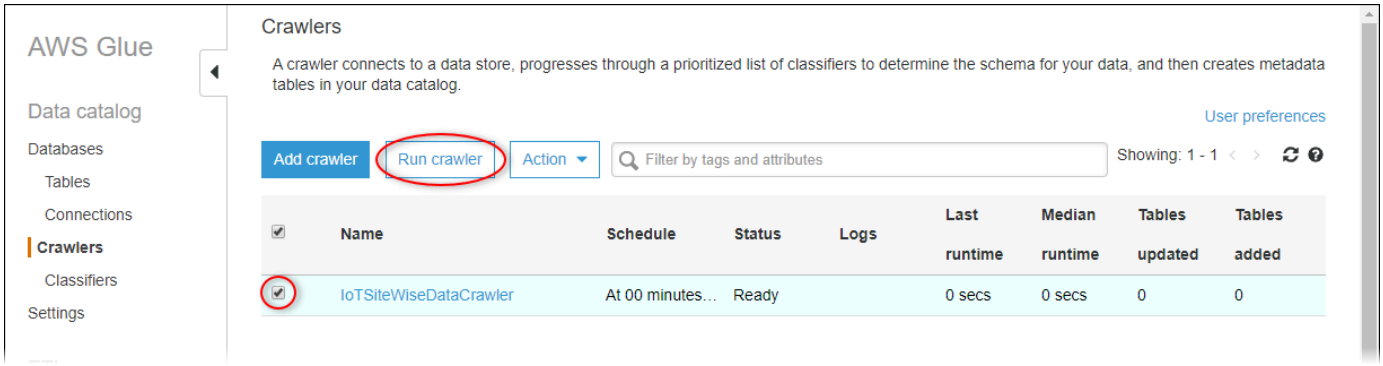
h. クローラの詳細を確認し、[完了] を選択します。



デフォルトでは、新しいクローラはすぐには実行されません。手動で実行するか、設定されたスケジュールで実行されるまで待機する必要があります。

クローラを実行するには

1. [クローラ] ページで、新しいクローラのチェックボックスをオンにし、[Run crawler (クローラの実行)] を選択します。



2. クローラが終了し、ステータスが [準備完了] になるまで待ちます。

   クローラの実行には数分かかる場合があり、クローラのステータスは自動的に更新されます。

3. ナビゲーションペインで、[Table] (テーブル) を選択します。

   asset_metadata と asset_property_updates の 2 つの新しいテーブルが表示されます。

Athena でデータをクエリする。

Athena は、 内のアセットデータテーブルを自動的に検出します AWS Glue Data Catalog。これらのテーブルの積集合に対してクエリを実行するために、論理データテーブルであるビューを作成できます。詳細については、[Amazon Athena User Guide] (Amazon Athenaユーザーガイド) の[Working with views] (ビューの操作) を参照してください。

アセットプロパティデータとメタデータを組み合わせたビューを作成した後、アセット名とプロパティ名がアタッチされたプロパティ値を出力するクエリを実行できます。詳細については、[Amazon Athena User Guide] (Amazon Athena ユーザーガイド) の[Running SQL queries using Amazon Athena] (Amazon Athena を使用した SQL クエリの実行) を参照してください。

Athena を使用してアセットデータをクエリするには

1. [Athena console] (Athena コンソール) に移動します。

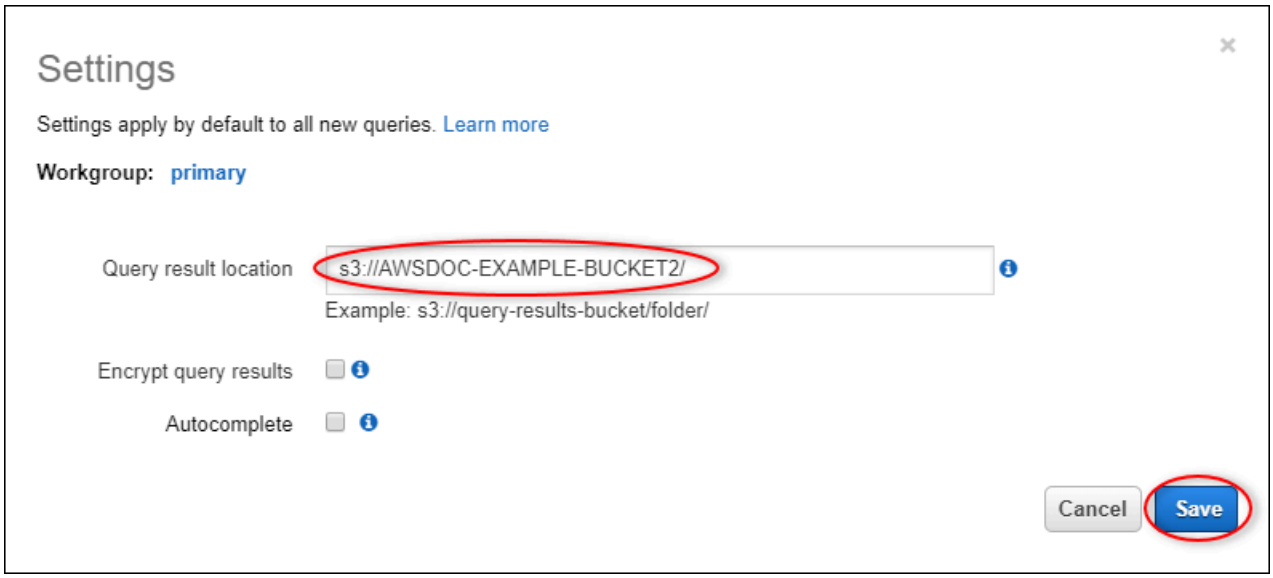
   [開始方法] のページが表示されたら、[今すぐ始める] をクリックします。

2. Athena を初めて使用する場合は、次のステップを実行して、クエリ結果に S3 バケットを設定します。Athena はこのバケットにクエリの結果を保存します。

> **Important**
>
> アセットデータバケットと異なるバケットを使用して、以前に作成したクローラがクエリ結果をクロールしないようにします。Athena クエリ結果にのみ使用するバケットを作成することをお勧めします。詳細については、[Amazon Simple Storage Service User Guide] (Amazon Simple Storage Service ユーザーガイド) の[How do I create an S3 bucket?] (S3バケットを作成する方法) をご覧ください。

a. [設定] を選択します。

b. [Query result location] (クエリ結果の場所) に、Athena クエリ結果用の S3 バケットを入力します。バケットは / で終わる必要があります。

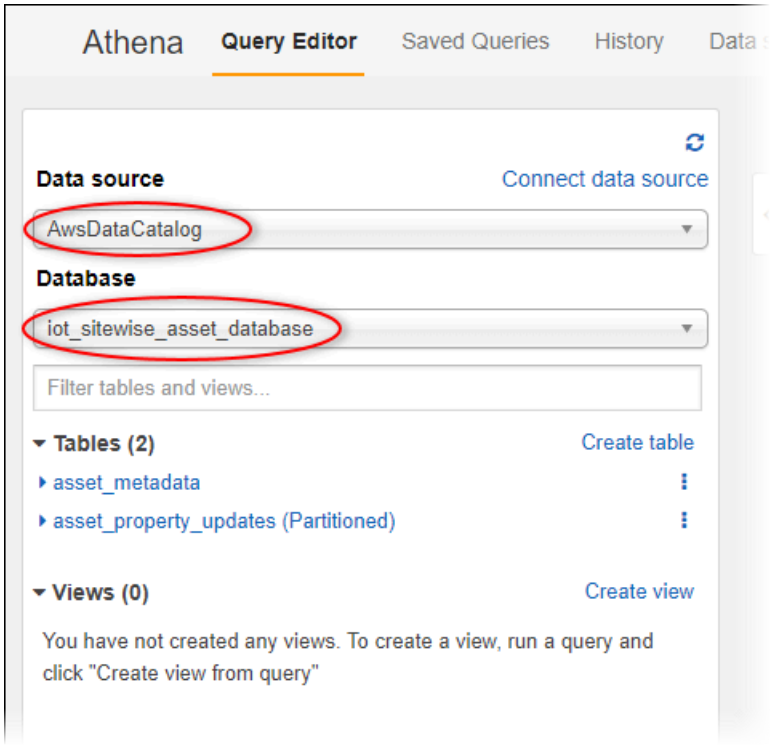


c. [保存] を選択します。

3. 左側のパネルには、クエリするデータソースが表示されます。以下の操作を実行します。

a. データソース で、 AwsDataCatalogの使用を選択します AWS Glue Data Catalog。

b. データベース で、クローラーで作成した AWS Glue データベースを選択します。



asset_metadata と asset_property_updates の 2 つのテーブルが表示されます。

4. アセットプロパティデータとメタデータの組み合わせからビューを作成するには、次のクエリを入力し、[クエリを実行] を選択します。

```
CREATE
        OR REPLACE VIEW iot_sitewise_asset_data AS
SELECT "from_unixtime"("time_in_seconds" + ("offset_in_nanos" / 1000000000))
 "timestamp",
         "metadata"."asset_name",
         "metadata"."asset_property_name",
         "data"."asset_property_value",
         "metadata"."asset_property_unit",
         "metadata"."asset_property_alias"
FROM ( "iot_sitewise_asset_database".asset_property_updates data
INNER JOIN "iot_sitewise_asset_database".asset_metadata metadata
    ON ( ("data"."asset_id" = "metadata"."asset_id")
        AND ("data"."asset_property_id" = "metadata"."asset_property_id") ) );
```

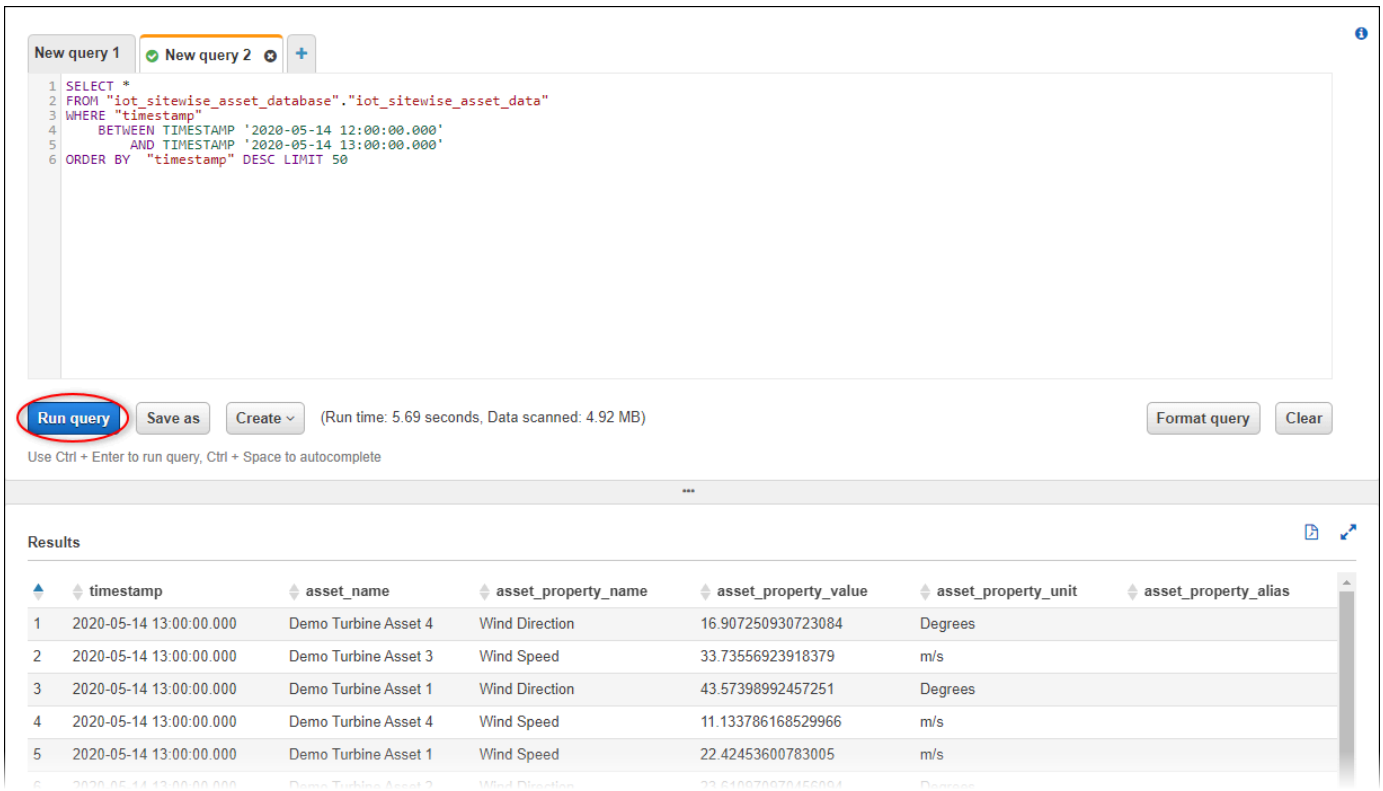
このクエリは、アセット ID とプロパティ ID のアセットプロパティデータとメタデータテーブルを結合して、ビューを作成します。このクエリは、既存のビューがすでに存在する場合は、既存のビューを置き換えるため、複数回実行できます。

5. + アイコンを選択して、新しいクエリを追加します。
6. アセットデータのサンプルを表示するには、次のクエリを入力し、[クエリを実行] を選択します。タイムスタンプを、バケットにデータがある間隔に置き換えます。

```
SELECT *
FROM "iot_sitewise_asset_database"."iot_sitewise_asset_data"
WHERE "timestamp"
    BETWEEN TIMESTAMP '2020-05-14 12:00:00.000'
        AND TIMESTAMP '2020-05-14 13:00:00.000'
ORDER BY  "timestamp" DESC LIMIT 50;
```

このクエリは、2 つのタイムスタンプ間に最大 50 個のデータポイントを出力し、最新のエントリを最初に表示します。

クエリの出力は、次のような結果になります。



	timestamp	asset_name	asset_property_name	asset_property_value	asset_property_unit	asset_property_alias
1	2020-05-14 13:00:00.000	Demo Turbine Asset 4	Wind Direction	16.907250930723084	Degrees	
2	2020-05-14 13:00:00.000	Demo Turbine Asset 3	Wind Speed	33.73556923918379	m/s	
3	2020-05-14 13:00:00.000	Demo Turbine Asset 1	Wind Direction	43.57398992457251	Degrees	
4	2020-05-14 13:00:00.000	Demo Turbine Asset 4	Wind Speed	11.133786168529966	m/s	
5	2020-05-14 13:00:00.000	Demo Turbine Asset 1	Wind Speed	22.42453600783005	m/s	

AWS IoT SiteWise アプリケーションに役立つクエリを実行できるようになりました。詳細については、[Amazon Athena User Guide] (Amazon Athena ユーザーガイド) の[SQL reference for Amazon Athena] (Amazon Athena用 SQL リファレンス) を参照してください。

テンプレートから作成されたリソース

テンプレートからスタックを作成すると、 は次のリソース AWS CloudFormation を作成します。ほとんどのリソースの名前には、スタックの作成時にカスタマイズできるプレフィックスが付いています。

リソース名パラメータ

- `BucketName` - アセットデータを受け取るこのテンプレートから作成された S3 バケットの名前。
- `GlobalResourcePrefix` - このテンプレートから作成されたグローバルリソースの名前のプレフィックス。デフォルトは `sitewise-export-to-s3` です。
- `LocalResourcePrefix` - 現在のリージョンでこのテンプレートから作成されるリソースの名前のプレフィックス。デフォルトは `sitewise_export_to_s3` です。

AWS CloudFormation テンプレートによって作成されたリソース

リソース	説明	[Name] (名前)
処理されたデータのS3 バケット	このバケットには、2 つのフォルダが含まれています。1 つのフォルダは Firehose 配信ストリームからフラット化されたフォーマット済みデータを受信し、もう 1 つのフォルダはアセットメタデータを受信します。	`${BucketName}`
データベースの AWS Glue	このデータベースには、このスタックが作成する AWS Glue テーブルが含まれています。	`${LocalResourcePrefix}_firehose_glue_database`
AWS Glue テーブル	Firehose 配信ストリームは、このテーブルを使用してデータを Parquet 形式にフォーマットします。	`${LocalResourcePrefix}_firehose_glue_table`

リソース	説明	[Name] (名前)
AWS Lambda データを変換する関数	この関数は、 から送信されるプロパティ値通知メッセージの値の配列をフラット化します AWS IoT SiteWise。	`${LocalResourcePrefix}_lambda_transform_function`
変換 Lambda 関数ロールのための IAM ポリシー。	このロールでは、変換関数のランタイムログを Lambda に保存できます。	`${GlobalResourcePrefix}-lambda-transform-role`
変換 Lambda 関数ロールのための IAM ポリシー。	このポリシーでは、変換関数の実行ログを Lambda に保存できます。	`${GlobalResourcePrefix}-lambda-transform-policy`
変換関数CloudWatch のロググループを記録します。	このロググループには、変換関数のログが含まれます。	`/aws/lambda/${LocalResourcePrefix}_lambda_transform_function`
[Lambda] アセットのメタデータを収集する関数。	この関数は、 のアセットに関する詳細を取得し、このスタックが作成する Amazon S3 バケットにその詳細 AWS IoT SiteWise を保存します。	`${LocalResourcePrefix}_lambda_metadata_function`
Lambda メタデータ関数のレイヤー。	このレイヤーは、メタデータ関数が使用する AWS IoT SiteWise オペレーションを含む AWS SDK を提供します。	`${LocalResourcePrefix}_lambda_metadata_layer`
メタデータ Lambda 関数の [IAM] ロール。	このロールにより、Lambda は のアセットに関する詳細を取得できます AWS IoT SiteWise。	`${GlobalResourcePrefix}-lambda-metadata-role`

リソース	説明	[Name] (名前)
メタデータ Lambda 機能ロールの IAM ポリシー。	このポリシーにより、Lambda は のアセットに関する詳細を取得できます AWS IoT SiteWise。	`${GlobalResourcePrefix}-lambda-metadata-policy`
EventBridge メタデータ Lambda 関数のスケジュールされたイベント	このスケジュールされたイベントは、6 時間ごとにメタデータ Lambda を実行し、アセットメタデータバケットを更新します。	`${LocalResourcePrefix}-metadata-event`
メタデータ関数CloudWatch のロググループを記録します。	このロググループには、メタデータ関数のログが含まれます。	`/aws/lambda/${LocalResourcePrefix}_lambda_metadata_function`
AWS IoT ルール	このルールは、プロパティ値通知メッセージをクエリし、アセットデータを Amazon Data Firehose 配信ストリームに送信します。	`${LocalResourcePrefix}_iot_topic_rule`
AWS IoT ルールの IAM ロール	このロールにより、 AWS IoT は Firehose 配信ストリームにデータを送信できます。	`${GlobalResourcePrefix}-core-firehose-role`
AWS IoT ルールロールの IAM ポリシー	このポリシーにより、 AWS IoT は Firehose 配信ストリームにデータを送信できます。	`${GlobalResourcePrefix}-core-firehose-policy`
Firehose 配信ストリーム	この配信ストリームは、 AWS IoT ルールからのデータを消費し、Lambda 関数を使用してデータをフラット化し、Amazon S3 にデータを配信します。	`${LocalResourcePrefix}_firehose_delivery_stream`

リソース	説明	[Name] (名前)
[IAM] 送信ストリームのロール。	このロールにより、Firehose は S3 バケット、 AWS Glue テーブル、Lambda 関数、および CloudWatch ロググループに対してオペレーションを実行できます。	`${GlobalResourcePrefix}-firehose-delivery-role`
配信ストリームCloudWatchのロググループを記録します。	このロググループには、Firehose 配信ストリームに関するログS3 Deliveryを受信するログストリーム が含まれています。	`/aws/kinesisfirehose/${LocalResourcePrefix}_firehose_delivery_stream`

Grafana との統合。

Grafana は、ダッシュボード内のデータの視覚化とモニタリングに使用できるデータ視覚化プラットフォームです。Grafana バージョン 7.3.0 以降では、AWS IoT SiteWise プラグインを使用して、Grafana ダッシュボードで AWS IoT SiteWise アセットデータを視覚化することができます。1 つの Grafana ダッシュボードで、複数のAWSソース (AWS IoT SiteWise、Amazon Timestream、Amazon など CloudWatch) やその他のデータソースのデータを視覚化できます。

AWS IoT SiteWise プラグインを使用するには、2 つの選択肢があります。

- [Local Grafana servers] (ローカル Grafana サーバー)

  お客様が管理している Grafana サーバーに AWS IoT SiteWise プラグインを設定することができます。プラグインを追加および使用する方法の詳細については、 GitHub ウェブサイトのAWS IoT SiteWise「データソースの README ファイル」を参照してください。

- [AWS Managed Service for Grafana] (マネージドサービス Grafana)

  AWS IoT SiteWise プラグインは、AWS Managed Service for Grafana (AMG) で使用することができます。AMG が Grafana サーバーを管理するため、お客様はハードウェアやその他の Grafana インフラストラクチャを構築、パッケージ化、デプロイすることなく、データを視覚化することができます。詳細については、[AWS Managed Service for Grafana] (Managed Service for Grafana ユーザーガイド) の次のトピックを参照してください。

- [What is Amazon Managed Service for Grafana (AMG) ?] (Amazon Managed Service for Grafana (AMG) とは？)
- [Using the AWS IoT SiteWise data source] (データソースを使用する)

Example Grafana ダッシュボードの例

次の Grafana ダッシュボードは、[demo wind farm] (デモの風力発電施設) を視覚したものです。このデモダッシュボードには、[Grafana Play] のウェブサイトからアクセスできます。

AWS IoT SiteWise と AWS IoT TwinMaker の統合

と統合AWS IoT TwinMakerすることで、AWS IoT SiteWiseデータ取り出し `ExecuteQuery` API や AWS IoT SiteWiseコンソールでの高度なアセット検索などAWS IoT SiteWise、 の堅牢な機能にアクセスできます。サービスを統合してこれらの機能を使用するには、まず統合を有効にする必要があります。

トピック

統合の有効化

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。JSON ポリシーの`Action`要素には、ポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。AWS IoT SiteWise がサポートするアクションの詳細については、[Service Authorization Reference] (サービス認可リファレンス) の[Actions defined by AWS IoT SiteWise] (定義するアクション) を参照してください。

AWS IoT TwinMaker サービスにリンクされたロールの詳細については、「 AWS IoT TwinMakerユーザーガイド」の「 のサービスにリンクされたロールAWS IoT TwinMaker」を参照してください。

AWS IoT SiteWise と を統合する前にAWS IoT TwinMaker、 がAWS IoT TwinMakerリンクされたワークスペースとAWS IoT SiteWise統合できるようにする次のアクセス許可を付与する必要があります。

- `iotsitewise:EnableSiteWiseIntegration` — リンクされたAWS IoT TwinMakerワークスペースとの統合AWS IoT SiteWiseを許可します。この統合によりAWS IoT TwinMaker、 は AWS IoT TwinMakerのサービスにリンクされたロールAWS IoT SiteWiseを通じて のすべてのモデリング情報を読み取ることができます。このアクセス許可を有効にするには、IAM ロールに次のポリシーを追加します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Action": [
        "iotsitewise:EnableSiteWiseIntegration"
      ],
      "Resource": "*"
    }
  ]
}
```

AWS IoT SiteWise と AWS IoT TwinMaker の統合

AWS IoT SiteWise と を統合するにはAWS IoT TwinMaker、以下が必要です。

- AWS IoT SiteWise アカウントで設定された サービスにリンクされたロール
- AWS IoT TwinMaker アカウントで設定された サービスにリンクされたロール
- AWS IoT TwinMaker リージョンIoTSiteWiseDefaultWorkspaceのアカウントで ID を持つワークスペース。

AWS IoT SiteWise コンソールを使用して を統合するには

コンソールに「 との統合AWS IoT TwinMaker」バナーが表示されたら、「アクセス許可を付与する」を選択します。前提条件はアカウントで作成されます。

を使用して を統合するには AWS CLI

AWS IoT TwinMaker を使用して AWS IoT SiteWiseと を統合するにはAWS CLI、次のコマンドを入力します。

1. AWSServiceName の CreateServiceLinkedRoleで を呼び出しますiotsitewise.amazonaws.com。

   ```
   aws iam create-service-linked-role --aws-service-name iotsitewise.amazonaws.com
   ```

2. AWSServiceName の CreateServiceLinkedRoleで を呼び出しますiottwinmaker.amazonaws.com。

   ```
   aws iam create-service-linked-role --aws-service-name iottwinmaker.amazonaws.com
   ```

3. ID の CreateWorkspaceで を呼び出しますIoTSiteWiseDefaultWorkspace。

```
aws iottwinmaker create-workspace --workspace-id IoTSiteWiseDefaultWorkspace
```

Amazon Lookout for Equipment による機器の異常の検出

> **Note**
>
> 異常検出は、Amazon Lookout for Equipment が利用可能なリージョンでのみ使用できます。

Amazon Lookout for Equipment AWS IoT SiteWise と統合すると、産業機器の異常検出と予測メンテナンスを通じて産業機器に関するインサイトを得ることができます。Lookout for Equipment は、異常な機器の動作を検出し、潜在的な障害を特定する産業機器を監視するための機械学習 (ML) サービスです。Lookout for Equipment を使用すると、予測メンテナンスプログラムを実装し、最適でない機器プロセスを特定できます。Lookout for Equipment の詳細については、「Amazon Lookout for Equipment ユーザーガイド」の「Amazon Lookout for Equipment とは」を参照してください。

異常な機器の動作を検出するように ML モデルをトレーニングする予測を作成すると、 はアセットプロパティ値を Lookout for Equipment AWS IoT SiteWise に送信して、異常な機器の動作を検出するように ML モデルをトレーニングします。アセットモデルで予測定義を定義するには、Lookout for Equipment がデータにアクセスするために必要な IAM ロールと、Lookout for Equipment に送信し、処理されたデータを Amazon S3 に送信するプロパティを指定します。詳細については、「アセットモデルを作成する」を参照してください。

AWS IoT SiteWise と Lookout for Equipment を統合するには、以下の大まかなステップを実行します。

- 追跡するプロパティの概要を示す予測定義をアセットモデルに追加します。予測定義は、そのアセットモデルに基づくアセットに予測を作成するために使用される測定値、変換、メトリクスの再利用可能なコレクションです。
- 提供した履歴データに基づいて予測をトレーニングします。
- スケジュール推論。特定の予測を実行する AWS IoT SiteWise 頻度を指定します。

推論がスケジュールされると、Lookout for Equipment モデルは機器から受信したデータをモニタリングし、機器の動作の異常を探します。 AWS IoT SiteWise GET API オペレーションまたは Lookout for Equipment コンソールを使用して、 SiteWise Monitor で結果を表示および分析できます。また、

アセットモデルのアラームディテクターを使用してアラームを作成し、異常な機器の動作を警告することもできます。

トピック

予測定義の追加 (コンソール)

によって収集されたデータを Lookout for Equipment に送信 AWS IoT SiteWise し始めるには、アセットモデルに AWS IoT SiteWise 予測定義を追加する必要があります。

AWS IoT SiteWise アセットモデルに予測定義を追加するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、モデルを選択し、予測定義を追加するアセットモデルを選択します。
3. 予測 を選択します。
4. 予測定義の追加 を選択します。
5. 予測定義の詳細を定義します。

    a. 予測定義の一意の名前と説明を入力します。予測定義の作成後は名前を変更できないため、名前は慎重に選択してください。

    b. が Amazon Lookout for Equipment とアセットデータを共有 AWS IoT SiteWise できるようにする IAM アクセス許可ロールを作成または選択します。ロールには、次の IAM ポリシーと信頼ポリシーが必要です。ロールの作成については、「カスタム信頼ポリシーを使用したロールの作成 (コンソール)」を参照してください。

    IAM ポリシー

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [{
            "Sid": "L4EPermissions",
            "Effect": "Allow",
            "Action": [
                "lookoutequipment:CreateDataset",
                "lookoutequipment:CreateModel",
                "lookoutequipment:CreateInferenceScheduler",
                "lookoutequipment:DescribeDataset",
                "lookoutequipment:DescribeModel",
                "lookoutequipment:DescribeInferenceScheduler",
                "lookoutequipment:ListInferenceExecutions",
                "lookoutequipment:StartDataIngestionJob",
                "lookoutequipment:StartInferenceScheduler",
                "lookoutequipment:UpdateInferenceScheduler",
                "lookoutequipment:StopInferenceScheduler"
            ],
            "Resource": [
                "arn:aws:lookoutequipment:Region:Account_ID:inference-
scheduler/IoTSiteWise_*",
                "arn:aws:lookoutequipment:Region:Account_ID:model/
IoTSiteWise_*",
                "arn:aws:lookoutequipment:Region:Account_ID:dataset/
IoTSiteWise_*"
            ]
        },
        {
            "Sid": "L4EPermissions2",
            "Effect": "Allow",
            "Action": [
                "lookoutequipment:DescribeDataIngestionJob"
            ],
            "Resource": "*"
        },
        {
            "Sid": "S3Permissions",
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
                "s3:ListBucket",
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": ["arn:aws:s3:::iotsitewise-*"]
        },
```

```
        {
            "Sid": "IAMPermissions",
            "Effect": "Allow",
            "Action": [
                "iam:GetRole",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::Account_ID:role/Role_name"
        }
    ]
}
```

信頼ポリシー

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Principal": {
                "Service": "iotsitewise.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "Account_ID"
                },
                "ArnEquals": {
                    "aws:SourceArn":
 "arn:aws:iotsitewise:Region:Account_ID:asset/*"
                }
            }
        },
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "lookoutequipment.amazonaws.com"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                    "aws:SourceAccount": "Account_ID"
                },
                "ArnEquals": {
```

```
                    "aws:SourceArn":
 "arn:aws:lookoutequipment:Region:Account_ID:*"
                }
            }
        }
    ]
}
```

c. [次へ] をクリックします。

6. Lookout for Equipment に送信するデータ属性 (測定、変換、メトリクス) を選択します。

   a. (オプション) 測定値を選択します。

   b. (オプション) 変換を選択します。

   c. (オプション) メトリクスを選択します。

   d. [次へ] をクリックします。

7. 選択を確認します。アセットモデルに予測定義を追加するには、概要ページで予測定義の追加を選択します。

アクティブな予測がアタッチされている既存の予測定義を編集または削除することもできます。

予測のトレーニング (コンソール)

アセットモデルに予測定義を追加したら、アセットにある予測をトレーニングできます。

で予測をトレーニングするには AWS IoT SiteWise

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、アセット を選択し、モニタリングするアセットを選択します。
3. 予測 を選択します。
4. トレーニングする予測を選択します。
5. アクション で、トレーニングの開始 を選択し、以下を実行します。

   a. 予測の詳細 で、 が Lookout for Equipment とアセットデータを共有 AWS IoT SiteWise できるようにする IAM アクセス許可ロールを選択します。新しいロールを作成する必要がある場合は、新しいロールの作成 を選択します。

   b. トレーニングデータ設定 には、トレーニングデータ時間範囲を入力して、予測のトレーニングに使用するデータを選択します。

c. （オプション) 後処理後のデータのサンプリングレートを選択します。

d. （オプション) データラベル には、ラベル付けデータを保持する Amazon S3 バケットとプレフィックスを指定します。データのラベル付けの詳細については、「Amazon Lookout for Equipment ユーザーガイド」の「データのラベル付け」を参照してください。

e. [次へ] をクリックします。

6. （オプション) トレーニングの完了直後に予測をアクティブにする場合は、詳細設定 で、トレーニング 後に予測を自動的にアクティブにして、次の操作を行います。

a. 「入力データ」で、「データのアップロード頻度」で、データのアップロード頻度を定義し、「オフセット遅延時間」で、使用するバッファの量を定義します。

b. [次へ] をクリックします。

7. 予測の詳細を確認し、保存して開始を選択します。

予測の推論を開始または停止する (コンソール)

> **Note**
>
> Lookout for Equipment の料金は、 AWS IoT SiteWise と Lookout for Equipment の間で転送されるデータを含むスケジュールされた推論に適用されます。詳細については、「Amazon Lookout for Equipment の料金」を参照してください。

予測 b"lookoutequipment:CreateDataset" を追加したが、トレーニング後にアクティブ化を選択しなかった場合は、アセットのモニタリングを開始するためにアクティブ化する必要があります。

予測の推論を開始するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、アセット を選択し、予測を追加するアセットを選択します。
3. 予測 を選択します。
4. アクティブ化する予測を選択します。
5. アクション で、推論の開始 を選択し、次の操作を行います。

a. 入力データ で、データのアップロード頻度 で、データのアップロード頻度を定義し、オフセット遅延時間 で、使用するバッファの量を定義します。

b. 保存して開始を選択します。

予測の推論を停止するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、アセット を選択し、予測を追加するアセットを選択します。
3. 予測 を選択します。
4. 停止する予測を選択します。
5. アクション で、推論を停止 を選択します。

予測定義の追加 (CLI)

新規または既存のアセットモデルで予測定義を定義するには、 AWS Command Line Interface () を使用できますAWS CLI。アセットモデルで予測定義を定義したら、Lookout for Equipment で異常検出を行うために、 のアセットの予測 AWS IoT SiteWise をトレーニングし、推論をスケジュールします。

前提条件

これらのステップを完了するには、アセットモデルと少なくとも 1 つのアセットを作成する必要があります。詳細については、「アセットモデルの作成 (AWS CLI) 」および「アセットの作成 (AWS CLI) 」を参照してください。

を初めて使用する場合は AWS IoT SiteWise、 `CreateBulkImportJob` API オペレーションを呼び出してアセットプロパティ値を にインポートする必要があります。これは AWS IoT SiteWiseモデルのトレーニングに使用されます。詳細については、「一括インポートジョブ (AWS CLI) を作成する」を参照してください。

予測定義を追加するには

1. `asset-model-payload.json` という名前のファイルを作成します。これらの他のセクションのステップに従って、アセットモデルの詳細をファイルに追加しますが、アセットモデルの作成または更新のリクエストは送信しないでください。

   - アセットモデルの作成方法の詳細については、「」を参照してください。 アセットモデルの作成 (AWS CLI)
   - 既存のアセットモデルを更新する方法の詳細については、「」を参照してください。 アセットまたはコンポーネントモデルの更新 (AWS CLI)
2. 次のコードを追加して、Lookout for Equipment 複合モデル (`assetModelCompositeModels`) をアセットモデルに追加します。

- を、含めるプロパティの ID *Property*に置き換えます。これらの IDs[DescribeAssetModel](#)。
- を、Lookout for Equipment が AWS IoT SiteWise データにアクセスできるようにする IAM ロールの ARN *RoleARN*に置き換えます。

```
{
  ...
  "assetModelCompositeModels": [
    {
      "name": "L4Epredictiondefinition",
      "type": "AWS/L4E_ANOMALY",
      "properties": [
          {
            "name": "AWS/L4E_ANOMALY_RESULT",
            "dataType": "STRUCT",
            "dataTypeSpec": "AWS/L4E_ANOMALY_RESULT",
            "unit": "none",
            "type": {
              "measurement": {}
            }
          },
          {
            "name": "AWS/L4E_ANOMALY_INPUT",
            "dataType": "STRUCT",
            "dataTypeSpec": "AWS/L4E_ANOMALY_INPUT",
            "type": {
               "attribute": {
                 "defaultValue": "{\"properties\": [\"Property1\", \"Property2\"]}"
               }
            }
          },
          {
            "name": "AWS/L4E_ANOMALY_PERMISSIONS",
            "dataType": "STRUCT",
            "dataTypeSpec": "AWS/L4E_ANOMALY_PERMISSIONS",
            "type": {
              "attribute": {
                "defaultValue": "{\"roleArn\": \"RoleARN\"}"
              }
            }
          },
```

```
            {
              "name": "AWS/L4E_ANOMALY_DATASET",
              "dataType": "STRUCT",
              "dataTypeSpec": "AWS/L4E_ANOMALY_DATASET",
              "type": {
                  "attribute": {}
              }
            },
            {
              "name": "AWS/L4E_ANOMALY_MODEL",
              "dataType": "STRUCT",
              "dataTypeSpec": "AWS/L4E_ANOMALY_MODEL",
              "type": {
                "attribute": {}
              }
            },
            {
              "name": "AWS/L4E_ANOMALY_INFERENCE",
              "dataType": "STRUCT",
              "dataTypeSpec": "AWS/L4E_ANOMALY_INFERENCE",
              "type": {
                "attribute": {}
              }
            },
            {
              "name": "AWS/L4E_ANOMALY_TRAINING_STATUS",
              "dataType": "STRUCT",
              "dataTypeSpec": "AWS/L4E_ANOMALY_TRAINING_STATUS",
              "type": {
                "attribute": {
                  "defaultValue": "{}"
                }
              }
            },
            {
              "name": "AWS/L4E_ANOMALY_INFERENCE_STATUS",
              "dataType": "STRUCT",
              "dataTypeSpec": "AWS/L4E_ANOMALY_INFERENCE_STATUS",
              "type": {
                "attribute": {
                  "defaultValue": "{}"
                }
              }
            }
```

```
    ]
}
```

3. アセットモデルを作成、または既存のアセットモデルを更新します。次のいずれかを行います。

   - アセットモデルを作成するには、次のコマンドを実行します。

     ```
     aws iotsitewise create-asset-model --cli-input-json file://asset-model-
     payload.json
     ```

   - 既存のアセットモデルを更新するには、次のコマンドを実行します。を更新するアセットモデルの ID *asset-model-id*に置き換えます。

     ```
     aws iotsitewise update-asset-model \
       --asset-model-id asset-model-id \
       --cli-input-json file://asset-model-payload.json
     ```

コマンドを実行したら、レスポンスに assetModelId に注意してください。

予測のトレーニングと推論の開始 (CLI)

予測定義が定義されたので、それに基づいてアセットをトレーニングし、推論を開始できます。予測をトレーニングしたいが推論を開始しない場合は、「」に進みます予測のトレーニング (CLI)。予測をトレーニングし、アセットの推論を開始するには、ターゲットリソースassetIdの が必要です。

予測の推論をトレーニングして開始するには

1. 次のコマンドを実行して、 assetModelCompositeModelId で を見つけますassetModelCompositeModelSummaries。を、 で作成したアセットモデルの ID *asset-model-id*に置き換えますアセットまたはコンポーネントモデルの更新 (AWS CLI) 。

   ```
   aws iotsitewise describe-asset-model \
     --asset-model-id asset-model-id \
   ```

2. 次のコマンドを実行して、 TrainingWithInferenceアクションactionDefinitionIdの を検索します。*asset-model-id* を前のステップで使用した ID *asset-model-composite-model-id*に置き換え、 を前のステップで返された ID に置き換えます。

   ```
   aws iotsitewise describe-asset-model-composite-model \
     --asset-model-id asset-model-id \
   ```

```
  --asset-model-composite-model-id asset-model-composite-model-id \
```

3. という名前のファイルtrain-start-inference-prediction.jsonを作成し、次のコードを置き換えて追加します。

   - *asset-id* ターゲットアセットの ID を含む
   - *action-definition-id* TrainingWithInference アクションの ID を含む
   - *StartTime* エポック秒で提供されるトレーニングデータの開始
   - *EndTime* エポック秒で提供されるトレーニングデータの終了
   - *TargetSamplingRate* Lookout for Equipment による後処理後のデータのサンプリングレート。使用できる値は ですPT1S | PT5S | PT10S | PT15S | PT30S | PT1M | PT5M | PT10M | PT15M | PT30M | PT1H。

```
{
  "targetResource": {
    "assetId": "asset-id"
  },
  "actionDefinitionId": "action-definition-Id",
  "actionPayload":{
    "stringValue": "{\"l4ETrainingWithInference\":{\"trainingWithInferenceMode
\":\"START\",\"trainingPayload\":{\"exportDataStartTime\":StartTime,
\"exportDataEndTime\":EndTime},\"targetSamplingRate\":\"TargetSamplingRate\"},
\"inferencePayload\":{\"dataDelayOffsetInMinutes\":0,\"dataUploadFrequency\":\"PT5M
\"}}}"
  }
}
```

4. 次のコマンドを実行して、トレーニングと推論を開始します。

```
aws iotsitewise execute-action --cli-input-json file://train-start-inference-
prediction.json
```

予測のトレーニング (CLI)

予測定義が定義されたので、それに基づいてアセットをトレーニングできます。アセットの予測をトレーニングするには、ターゲットリソースassetIdの が必要です。

予測をトレーニングするには

1. 次のコマンドを実行して、 assetModelCompositeModelId で を見つけますassetModelCompositeModelSummaries。を、 で作成したアセットモデルの ID *asset-model-id*に置き換えますアセットまたはコンポーネントモデルの更新 (AWS CLI) 。

```
aws iotsitewise describe-asset-model \
  --asset-model-id asset-model-id \
```

2. 次のコマンドを実行して、 TrainingアクションactionDefinitionIdの を検索します。*asset-model-id* を前のステップで使用した ID *asset-model-composite-model-id*に置き換え、 を前のステップで返された ID に置き換えます。

```
aws iotsitewise describe-asset-model-composite-model \
  --asset-model-id asset-model-id \
  --asset-model-composite-model-id asset-model-composite-model-id \
```

3. という名前のファイルtrain-prediction.jsonを作成し、次のコードを置き換えて追加します。

- *asset-id* ターゲットアセットの ID を含む
- *action-definition-id* トレーニングアクションの ID を含む
- *StartTime* エポック秒で提供されるトレーニングデータの開始
- *EndTime* エポック秒で提供されるトレーニングデータの終了
- (オプション) *BucketName* ラベルデータを保持する Amazon S3 バケットの名前
- (オプション) Amazon *Prefix* S3 バケットに関連付けられたプレフィックスを持つ 。Amazon S3
- *TargetSamplingRate* Lookout for Equipment による後処理後のデータのサンプリングレート。使用できる値は ですPT1S | PT5S | PT10S | PT15S | PT30S | PT1M | PT5M | PT10M | PT15M | PT30M | PT1H。

> **Note**
>
> バケット名とプレフィックスの両方を含めるか、どちらも含めないでください。

```
{
  "targetResource": {
```

```
    "assetId": "asset-id"
  },
  "actionDefinitionId": "action-definition-Id",
  "actionPayload":{ "stringValue": "{\"l4ETraining\": {\"trainingMode\":
\"START\",\"exportDataStartTime\": StartTime, \"exportDataEndTime\": EndTime,
 \"targetSamplingRate\":\"TargetSamplingRate\"}, \"labelInputConfiguration\":
 {\"bucketName\": \"BucketName\", \"prefix\": \"Prefix\"}}}"
}
}
```

4. 次のコマンドを実行してトレーニングを開始します。

```
aws iotsitewise execute-action --cli-input-json file://train-prediction.json
```

推論を開始する前に、トレーニングを完了する必要があります。トレーニングのステータスを確認するには、次のいずれかを実行します。

- コンソールから、予測対象のアセットに移動します。
- から AWS CLI、 trainingStatusプロパティpropertyIdの BatchGetAssetPropertyValueを使用して を呼び出します。

予測の推論を開始または停止する (CLI)

予測がトレーニングされたら、推論を開始して Lookout for Equipment にアセットのモニタリングを開始するように指示できます。推論を開始または停止するには、ターゲットリソースassetIdの が必要です。

推論を開始するには

1. 次のコマンドを実行して、 assetModelCompositeModelId で を見つけますassetModelCompositeModelSummaries。を、 で作成したアセットモデルの ID *asset-model-id*に置き換えますアセットまたはコンポーネントモデルの更新 (AWS CLI) 。

```
aws iotsitewise describe-asset-model \
  --asset-model-id asset-model-id \
```

2. 次のコマンドを実行して、 InferenceアクションactionDefinitionIdの を検索します。 を前のステップで使用した ID *asset-model-id*に置き換え、 を前のステップで返された ID *asset-model-composite-model-id*に置き換えます。

```
aws iotsitewise describe-asset-model-composite-model \
  --asset-model-id asset-model-id \
  --asset-model-composite-model-id asset-model-composite-model-id \
```

3. という名前のファイルstart-inference.jsonを作成し、次のコードを置き換えて追加します。

   - *asset-id* ターゲットアセットの ID を含む
   - *action-definition-id* 推論開始アクションの ID を含む
   - *Offset* 使用するバッファの量を含む
   - *Frequency* とデータのアップロード頻度

```
{
  "targetResource": {
    "assetId": "asset-id"
  },
  "actionDefinitionId": "action-definition-Id",
  "actionPayload":{ "stringValue": "{\"l4EInference\": {\"inferenceMode\":\"START
\",\"dataDelayOffsetInMinutes\": Offset, \"dataUploadFrequency\": \"Frequency\"}}"
}}
```

4. 推論を開始するには、次のコマンドを実行します。

```
aws iotsitewise execute-action --cli-input-json file://start-inference.json
```

推論を停止するには

1. 次のコマンドを実行して、 assetModelCompositeModelId で を見つけますassetModelCompositeModelSummaries。を、 で作成したアセットモデルの ID *asset-model-id*に置き換えますアセットまたはコンポーネントモデルの更新 (AWS CLI) 。

```
aws iotsitewise describe-asset-model \
  --asset-model-id asset-model-id \
```

2. 次のコマンドを実行して、 InferenceアクションactionDefinitionIdの を検索します。を前のステップで使用した ID *asset-model-id*に置き換え、 を前のステップで返された ID *asset-model-composite-model-id*に置き換えます。

```
aws iotsitewise describe-asset-model-composite-model \
  --asset-model-id asset-model-id \
  --asset-model-composite-model-id asset-model-composite-model-id \
```

3. という名前のファイルstop-inference.jsonを作成し、次のコードを置き換えて追加します。

   - *asset-id* ターゲットアセットの ID を含む
   - *action-definition-id* 推論開始アクションの ID を含む

```
{
  "targetResource": {
    "assetId": "asset-id"
  },
  "actionDefinitionId": "action-definition-Id",
  "actionPayload":{ "stringValue": "{\"l4EInference\":{\"inferenceMode\":\"STOP
\"}}"
}}
```

4. 推論を停止するには、次のコマンドを実行します。

```
aws iotsitewise execute-action --cli-input-json file://stop-inference.json
```

ストレージの管理

AWS IoT SiteWise データを次のストレージ階層に保存するように設定できます。

Hot 階層

AWS IoT SiteWise ホットストレージ階層はマネージド型の時系列ストレージです。ホット階層は、頻繁にアクセスされるデータや、 write-to-read レイテンシーが短いデータに最も効果的です。ホットティアに保存されたデータは、機器の最新の測定値にすばやくアクセスする必要がある産業用アプリケーションで使用されます。これには、インタラクティブなダッシュボードでリアルタイムのメトリックを視覚化するアプリケーションや、動作を監視してパフォーマンスの問題を特定するためのアラームを起動するアプリケーションが含まれます。

デフォルトでは、 AWS IoT SiteWise に取り込まれたデータはホット階層に保存されます。ホットティアの保持期間を定義できます。保存期間が過ぎると、ホットティアのデータは、 AWS IoT SiteWise 構成に基づいてウォーム階層またはコールド階層のストレージに移動されます。最高のパフォーマンスとコスト効率を実現するには、ホットティアの保持期間を、頻繁にデータを取得するのにかかる時間よりも長く設定します。これはリアルタイムのメトリクス、アラーム、モニタリングシナリオに使用されます。保持期間が設定されていない場合、データはホット階層に無期限に保存されます。

ウォームティア

ウォームストレージ階層は、 AWS IoT SiteWise 履歴データをコスト効率よく保存するのに効果的なマネージド階層です。 write-to-read レイテンシーが中程度の大量のデータを取得するのに最適です。Warm Tier は、大規模なワークロードに必要な履歴データを保存する場合に使用します。たとえば、分析、ビジネスインテリジェンスアプリケーション (BI)、レポートツール、機械学習 (ML) モデルのトレーニングのためのデータ取得に使用されます。コールドストレージ階層を有効にすると、ウォーム階層の保存期間を定義できます。 AWS IoT SiteWise 保持期間が終了すると、ウォーム階層からデータを削除します。

Cold 階層

コールドストレージ階層は、Amazon S3 バケットを使用して、めったに使用されないデータを保存します。コールド階層を有効にすると、測定、メトリクス、変換と集計、資産モデル定義などの時系列を 6 AWS IoT SiteWise 時間ごとに複製します。コールド階層は、履歴レポートやバックアップのための高い読み取り待ち時間を許容するデータを保存するために使用されます。

トピック

ストレージ設定の構成

サービス管理型のウォーム階層ストレージにオプトインしたり、コールド階層にデータを複製したりするようにストレージ設定を構成できます。ウォーム階層とホット階層の保持期間について詳しくは、を参照してくださいデータ保持への影響。ストレージ設定を行う際には、次のことを行ってください。

- ホット階層の保存 — データが削除され、ストレージ設定に基づいてサービス管理のウォーム階層ストレージまたはコールド階層ストレージに移動されるまでのデータをホット階層に保存する期間を設定します。 AWS IoT SiteWise 保持期間が終了する前にホット階層のデータをすべて削除します。保持期間を設定しない場合、データはホット階層に無期限に保存されます。
- ウォーム階層の保存 — AWS IoT SiteWise データがストレージから削除され、顧客が管理するコールド階層ストレージに移動される前に、データをウォーム階層に保存する期間を設定します。AWS IoT SiteWise 保存期間が終了する前に存在していたデータをウォーム階層から削除します。保存期間が設定されていない場合、データはウォーム階層に無期限に保存されます。

> **Note**
>
> クエリのパフォーマンスを向上させるには、ウォーム階層ストレージでホット階層の保持期間を設定します。

ホット階層とウォーム階層のストレージにおけるデータ保持の影響

- ホット階層ストレージの保持期間を短縮すると、データはホット階層からウォーム階層またはコールド階層に永続的に移動されます。ウォームティアの保持期間を短くすると、データはコールドティアに移動され、ウォームティアから完全に削除されます。
- ホット階層またはウォーム階層のストレージの保持期間を延長すると、 AWS IoT SiteWise その変更はそれ以降に送信されるデータに影響します。 AWS IoT SiteWise ウォームストレージまたはコールドストレージからデータを取得してホットティアに格納することはありません。たとえば、

ホット階層ストレージの保持期間を当初 30 日に設定し、その後 60 日に延長した場合、ホット階層ストレージに 60 日分のデータが保存されるまでに 30 日かかります。

トピック

ウォームティアのストレージ設定を行います (コンソール)

以下の手順は、 AWS IoT SiteWise コンソールのウォームティアにデータを複製するようにストレージ設定を構成する方法を示しています。

コンソールでストレージの設定を設定するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [設定] の [ストレージ] を選択します。
3. 右上隅の[編集] を選択します。
4. [ストレージを編集] ページで、次の作業を行います:
5. Hot Tier を設定するには、次の操作を行います。

   - データが削除されてサービス管理のウォーム階層ストレージに移動される前に、データをホット階層に保存する期間を設定する場合は、[保持期間を有効にする] を選択します。
   - 保持期間を設定するには、整数を入力して単位を選択します。保持期間は、30 日以上にする必要があります。

   AWS IoT SiteWise Hot Tier の保持期間より古いデータをすべて削除します。保持期間を設定しなかった場合、データは無期限に保存されます。
6. (推奨) ウォームティアを設定する場合は、次の操作を行います。

   - ウォーム階層ストレージにオプトインするには、[ウォーム階層ストレージのオプトインを確認] を選択してウォーム階層ストレージにオプトインします。
   - (オプション) 保存期間を設定するには、整数を入力して単位を選択します。保存期間は 365 日以上である必要があります。

AWS IoT SiteWise 保存期間より前に存在していたウォーム階層のデータを削除します。保持期間を設定しなかった場合、データは無期限に保存されます。

> **Note**
>
> - ウォームティアを選択した場合、設定は 1 回だけ表示されます。
> - ホット階層の保持を設定するには、ウォーム階層またはコールド階層のストレージが必要です。コスト効率を高め、 AWS IoT SiteWise 履歴データを取得するために、長期データはウォーム階層に保存することをお勧めします。
> - ウォーム階層の保持を設定するには、コールド階層ストレージが必要です。

7. [Save] を選択してストレージ設定を保存します。

AWS IoT SiteWise ストレージセクションでは、ウォーム階層ストレージは次のいずれかの状態になっています。

- 有効 — ホットティアの保存期間より前にデータが存在していた場合は、 AWS IoT SiteWise データをウォームティアに移動します。」
- 無効 — ウォーム階層ストレージは無効になっています。

ウォームティア (AWS CLI) のストレージ設定を行います。

AWS CLI および次のコマンドを使用して、データをウォームティアに移動するようにストレージ設定を構成できます。

既存の構成が上書きされないようにするには、以下のコマンドを実行して現在のストレージ構成情報を取得します。

```
aws iotsitewise describe-storage-configuration
```

Example コールドティア構成が存在しない場合の応答

```
{
          "storageType": "SITEWISE_DEFAULT_STORAGE",
          "disassociatedDataStorage": "ENABLED",
```

```
          "configurationStatus": {
              "state": "ACTIVE"
          },
          "lastUpdateDate": "2021-10-14T15:53:35-07:00",
          "warmTier": "DISABLED"
}
```

Example 既存のコールドティア構成での応答

```
{
      "storageType": "MULTI_LAYER_STORAGE",
          "multiLayerStorage": {
            "customerManagedS3Storage": {
            "s3ResourceArn": "arn:aws:s3:::bucket-name/prefix/",
            "roleArn": "arn:aws:iam::aws-account-id:role/role-name"
            }
          },
      "disassociatedDataStorage": "ENABLED",
      "retentionPeriod": {
      "numberOfDays": retention-in-days
      },
       "configurationStatus": {
       "state": "ACTIVE"
      },
      "lastUpdateDate": "2023-10-25T15:59:46-07:00",
      "warmTier": "DISABLED"
}
```

ウォームティアのストレージ設定は以下のように設定します。 AWS CLI

以下のコマンドを実行してストレージ設定を構成します。 AWS IoT SiteWise ストレージ設定を含むファイルの名前に置き換えますfile-name。

```
aws iotsitewise put-storage-configuration --cli-input-json file://file-name.json
```

Example AWS IoT SiteWise ホットティアとウォームティアによる設定

```
{
             "storageType": "SITEWISE_DEFAULT_STORAGE",
             "disassociatedDataStorage": "ENABLED",
             "warmTier": "ENABLED",
```

```
            "retentionPeriod": {
               "numberOfDays": hot-tier-retention-in-days
             }

}
```

hot-tier-retention-in-days30 日以上の整数でなければなりません。

Example レスポンス

```
{
            "storageType": "SITEWISE_DEFAULT_STORAGE",
            "configurationStatus": {
            "state": "UPDATE_IN_PROGRESS"
            }
}
```

コールド階層ストレージを有効にしている場合は、を参照してくださいAWS CLI 既存のコールドティアを使用してストレージ設定を行います。。

AWS CLI 既存のコールドティアを使用してストレージ設定を行います。

AWS CLI 既存のコールドティアストレージを使用してストレージ設定を行います。

- 以下のコマンドを実行してストレージ設定を構成します。*file-name* は、 AWS IoT SiteWise ストレージの設定を含むファイルの名前に置き換えてください。

  ```
  aws iotsitewise put-storage-configuration --cli-input-json file://file-name.json
  ```

  Example AWS IoT SiteWise ストレージ設定

  - *bucket-name* は Amazon S3 バケット名に置き換えます。
  - *[prefix]* (プレフィックス) は Amazon S3 のプレフィックスに置き換えてください。
  - AWS アカウント ID *aws-account-id*に置き換えてください。
  - *role-name* は、Amazon S3 AWS IoT SiteWise へのデータ送信を許可する Amazon S3 アクセスロールの名前に置き換えてください。
  - *hot-tier-retention-in-days* は 30 日以上の整数に置き換えてください。
  - *warm-tier-retention-in-days* は 365 日以上の整数に置き換えてください。

> **Note**
>
> AWS IoT SiteWise コールド階層の保持期間よりも古いウォーム階層のデータをすべて削除します。保持期間を設定しなかった場合、データは無期限に保存されます。

```
{
      "storageType": "MULTI_LAYER_STORAGE",
        "multiLayerStorage": {
          "customerManagedS3Storage": {
              "s3ResourceArn": "arn:aws:s3:::bucket-name/prefix/",
              "roleArn": "arn:aws:iam::aws-account-id:role/role-name"
              }
          },
    "disassociatedDataStorage": "ENABLED",
    "retentionPeriod": {
      "numberOfDays": hot-tier-retention-in-days
    },
    "warmTier": "ENABLED",
    "warmTierRetentionPeriod": {
      "numberOfDays": warm-tier-retention-in-days
    }
}
```

Example レスポンス

```
{
      "storageType": "MULTI_LAYER_STORAGE",
      "configurationStatus": {
        "state": "UPDATE_IN_PROGRESS"
       }
}
```

コールドティアのストレージ設定を行います (コンソール)

以下の手順は、 AWS IoT SiteWise コンソールのコールド階層にデータを複製するようにストレージ設定を行う方法を示しています。

コンソールでストレージの設定を設定するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで [設定] の [ストレージ] を選択します。
3. 右上隅の[編集] を選択します。
4. [ストレージを編集] ページで、次の作業を行います:

   a. [ストレージ設定] で [コールド階層ストレージを有効にする] を選択します。デフォルトでは、コールド階層ストレージは無効です。

   b. S3 バケットの場所には、既存の Amazon S3 バケット名とプレフィックスを入力します。

   > **Note**
   >
   > - Amazon S3 では、プレフィックスを Amazon S3 バケット内のフォルダ名として使用します。プレフィックスは1～255文字で、末尾はスラッシュ (/) とする。お客様の AWS IoT SiteWise データはこのフォルダーに保存されます。
   > - Amazon S3 バケットがない場合は、[表示] () を選択し、Amazon S3 コンソールでバケットを作成します。詳細については、Amazon S3 ユーザーガイド の 最初の S3 バケットを作成する を参照してください。

   c. S3 アクセスロールの場合、次のいずれかを実行します。

   - [AWS マネージドテンプレートからロールを作成] を選択すると、Amazon S3 AWS IoT SiteWise にデータを送信できる IAM AWS ロールが自動的に作成されます。
   - [既存のロールの使用] を選択し、リストから作成したロールを選択します。

   > **Note**
   >
   > - [S3 バケットの場所] には、前のステップで使用したのと同じ Amazon S3 バケット名をIAM ポリシーで使用する必要があります。
   > - 次の例に示すようなアクセス許可をロールが持っていることを確認してください。
   >
   > Example アクセス権限ポリシー:
   >
   > ```
   > {
   >       "Version": "2012-10-17",
   > ```

```
        "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "s3:PutObject",
                    "s3:GetObject",
                    "s3:DeleteObject",
                    "s3:GetBucketLocation",
                    "s3:ListBucket"
                ],
                "Resource": [
                    "arn:aws:s3:::bucket-name",
                    "arn:aws:s3:::bucket-name/*"
                ]
            }
        ]
    }
```

bucket-name は、Amazon S3 バケットの名前に置き換えてください。

d. ホットティアを設定するには、のステップ 5 を参照してくださいウォームティアのストレージ設定を行います (コンソール)。

e. (オプション) [AWS IoT Analytics 統合] では、以下を行います。

   i. AWS IoT Analytics を使用してデータをクエリする場合は、[AWS IoT Analytics データストアを有効にする] を選択します。

   ii. AWS IoT SiteWise データストアの名前を生成するか、別の名前を入力できます。

   AWS IoT SiteWise AWS IoT Analytics データを保存するためのデータストアを自動的に作成します。データをクエリするには、 AWS IoT Analytics を使用してデータセットを作成できます。詳細については、『AWS IoT Analytics ユーザーガイド』の「AWS IoT SiteWise データの操作」を参照してください。

f. [保存] を選択します。

[AWS IoT SiteWise ストレージ] セクションの[コールド階層ストレージ]は、次のいずれかの値に設定できます。

- 有効 — 指定した Amazon S3 AWS IoT SiteWise バケットにデータを複製します。

- 有効化 — AWS IoT SiteWise コールド階層ストレージを有効にするリクエストを処理しています。このプロセスは完了までに数分かかることがあります。
- Enable_Failed — AWS IoT SiteWise コールド階層ストレージを有効にするリクエストを処理できませんでした。Amazon CloudWatch Logs AWS IoT SiteWise への ログ送信を有効にした場合は、これらのログを使用して問題のトラブルシューティングを行うことができます。詳細については、「Amazon CloudWatch ログによるモニタリング」を参照してください。
- [無効] – コールド階層ストレージは無効です。

Cold Tier ()AWS CLIのストレージ設定を行います。

以下では、 AWS CLIを使用して、ストレージの設定でコールド階層にデータが複製されるようにする手順を示します。

を使用してストレージ設定を構成するには AWS CLI

1. アカウント内の Amazon S3 バケットにデータを書き出すには、次のコマンドを実行し、ストレージの設定を行ってください。*file-name* は、 AWS IoT SiteWise ストレージ設定を含むファイルの名前に置き換えます。

```
aws iotsitewise put-storage-configuration --cli-input-json file://file-name.json
```

Example AWS IoT SiteWise ストレージ設定

- *bucket-name* は Amazon S3 バケット名に置き換えます。
- *[prefix]* (プレフィックス) は Amazon S3 のプレフィックスに置き換えてください。
- AWS アカウント ID *aws-account-id*に置き換えてください。
- *role-name* は、Amazon S3 AWS IoT SiteWise へのデータ送信を許可する Amazon S3 アクセスロールの名前に置き換えてください。
- 30 *retention-in-days*日以上の整数に置き換えてください。

```
{
      "storageType": "MULTI_LAYER_STORAGE",
      "multiLayerStorage": {
          "customerManagedS3Storage": {
              "s3ResourceArn": "arn:aws:s3:::bucket-name/prefix/",
              "roleArn": "arn:aws:iam::aws-account-id:role/role-name"
```

```
        }
    },
    "retentionPeriod": {
        "numberOfDays": retention-in-days,
        "unlimited": false
    }
}
```

Note

- AWS IoT SiteWise ストレージ設定と IAM ポリシーでは同じ Amazon S3 バケット名を使用する必要があります。
- 次の例に示すようなアクセス許可をロールが持っていることを確認してください。

  Example アクセス権限ポリシー:

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
              "Effect": "Allow",
              "Action": [
                  "s3:PutObject",
                  "s3:GetObject",
                  "s3:DeleteObject",
                  "s3:GetBucketLocation",
                  "s3:ListBucket"
              ],
              "Resource": [
                  "arn:aws:s3:::bucket-name",
                  "arn:aws:s3:::bucket-name/*"
              ]
          }
      ]
  }
```

  bucket-name は、Amazon S3 バケットの名前に置き換えてください。

Example レスポンス

```
{
    "storageType": "MULTI_LAYER_STORAGE",
    "retentionPeriod": {
        "numberOfDays": 100,
        "unlimited": false
    },
    "configurationStatus": {
        "state": "UPDATE_IN_PROGRESS"
    }
}
```

> **Note**
>
> AWS IoT SiteWise ストレージ設定の更新には数分かかることがあります。

2. ストレージの設定情報を取得するには、次のコマンドを実行します。

```
aws iotsitewise describe-storage-configuration
```

Example レスポンス

```
{
      "storageType": "MULTI_LAYER_STORAGE",
      "multiLayerStorage": {
          "customerManagedS3Storage": {
              "s3ResourceArn": "arn:aws:s3:::DOC-EXAMPLE-BUCKET/torque/",
              "roleArn": "arn:aws:iam::123456789012:role/SWAccessS3Role"
          }
      },
      "retentionPeriod": {
          "numberOfDays": 100,
          "unlimited": false
      },
      "configurationStatus": {
          "state": "ACTIVE"
      },
      "lastUpdateDate": "2021-03-30T15:54:14-07:00"
```

```
  }
```

3. Amazon S3 バケットへのデータ書き出しを停止するには、次のコマンドを実行してストレージの設定を行ってください。

```
aws iotsitewise put-storage-configuration --storage-type SITEWISE_DEFAULT_STORAGE
```

> **Note**
>
> デフォルトでは、データはのホット階層にのみ保存されます AWS IoT SiteWise。

Example レスポンス

```
{
      "storageType": "SITEWISE_DEFAULT_STORAGE",
      "configurationStatus": {
          "state": "UPDATE_IN_PROGRESS"
      }
  }
```

4. ストレージの設定情報を取得するには、次のコマンドを実行します。

```
aws iotsitewise describe-storage-configuration
```

Example レスポンス

```
{
      "storageType": "SITEWISE_DEFAULT_STORAGE",
      "configurationStatus": {
          "state": "ACTIVE"
      },
      "lastUpdateDate": "2021-03-30T15:57:14-07:00"
  }
```

(オプション) AWS IoT Analytics データストアを作成する (AWS CLI)

AWS IoT Analytics データストアは、データを受信して保存する、スケーラブルでクエリ可能なリポジトリです。 AWS IoT SiteWise コンソールまたは AWS IoT Analytics API を使用して、 AWS

IoT Analytics データを保存するデータストアを作成できます。 AWS IoT SiteWise データをクエリするには、 AWS IoT Analyticsを使用してデータセットを作成します。詳細については、 AWS IoT Analytics ユーザーガイドの「AWS IoT SiteWise データの使用」を参照してください。

次の手順では、 AWS CLI を使用してにデータストアを作成します。 AWS IoT Analytics

データストアを作成するには、次のコマンドを実行します。*file-name* は、データストアの設定を含むファイルの名前に置き換えてください。

```
aws iotanalytics create-datastore --cli-input-json file://file-name.json
```

> **Note**
>
> - 既存の Amazon S3 バケット名を指定する必要があります。 Amazon S3 バケットを持っていない場合は、まずバケットを作成します。詳細については、Amazon S3 ユーザーガイドの 最初の S3 バケットを作成する を参照してください。
> - AWS IoT SiteWise ストレージ設定、IAM ポリシー、 AWS IoT Analytics およびデータストア設定には、同じ Amazon S3 バケット名を使用する必要があります。

Example AWS IoT Analytics データストア設定

data-store-name AWS IoT Analytics s3-############## Amazon S3 バケット名に置き換えてください。

```
{
      "datastoreName": "data-store-name",
      "datastoreStorage": {
          "iotSiteWiseMultiLayerStorage": {
              "customerManagedS3Storage": {
                  "bucket": "s3-bucket-name"
              }
          }
      },
      "retentionPeriod": {
          "numberOfDays": 90
      }
  }
```

Example レスポンス

```
{
      "datastoreName": "datastore_IoTSiteWise_demo",
      "datastoreArn": "arn:aws:iotanalytics:us-west-2:123456789012:datastore/
datastore_IoTSiteWise_demo",
      "retentionPeriod": {
          "numberOfDays": 90,
          "unlimited": false
      }
  }
```

ストレージ設定のトラブルシューティング

次の情報を使用して、ストレージ設定に関する問題のトラブルシューティングと解決を行います。

問題

- エラー:バケットは存在しません
- エラー:Amazon S3 パスへのアクセスが拒否されました
- エラー:ロール ARN を引き受けることができません
- エラー:クロスリージョン Amazon S3 バケットにアクセスできませんでした

エラー:バケットは存在しません

解決策:Amazon S3 AWS IoT SiteWise バケットが見つかりませんでした。現在のRegion にある既存の Amazon S3 バケット名を入力していることを確認してください。

エラー:Amazon S3 パスへのアクセスが拒否されました

解決策:Amazon S3 AWS IoT SiteWise バケットにアクセスできませんでした。以下の操作を実行します。

- IAM ポリシーで指定した Amazon S3 バケットと同じものを使用していることを確認してください。
- 次の例に示すようなアクセス許可をロールが持っていることを確認してください。

Example アクセス許可ポリシー

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
              "Effect": "Allow",
              "Action": [
                  "s3:PutObject",
                  "s3:GetObject",
                  "s3:DeleteObject",
                  "s3:GetBucketLocation",
                  "s3:ListBucket"
              ],
              "Resource": [
                  "arn:aws:s3:::bucket-name",
                  "arn:aws:s3:::bucket-name/*"
              ]
          }
      ]
  }
```

bucket-name は、Amazon S3 バケットの名前に置き換えてください。

エラー:ロール ARN を引き受けることができません

解決策: AWS IoT SiteWise ユーザーに代わって IAM ロールを引き受けることができませんでした。ロールが次のサービスを信頼することを確認してください: `iotsitewise.amazonaws.com`。詳細については、IAM ユーザーガイドの「ロールを引き受けることができない」を参照してください。

エラー:クロスリージョン Amazon S3 バケットにアクセスできませんでした

解決策:指定した Amazon S3 AWS バケットは別のリージョンにあります。Amazon S3 AWS IoT SiteWise バケットとアセットが同じリージョンにあることを確認してください。

Cold 階層に保存されたデータのファイルパスとスキーマ

AWS IoT SiteWise は、測定値、メトリクス、変換と集計、アセットとアセットモデルの定義などの時系列をレプリケートすることで、データをコールド階層に保存します。次に、Cold 階層に送られるデータのファイルパスとスキーマについて説明します。

トピック

機器データ (測定)

AWS IoT SiteWise は、機器データ (測定値) を 6 時間に 1 回コールド階層にエクスポートします。raw データはCold 階層にApache AVRO (`.avro`) 形式で保存されます。

ファイルパス

AWS IoT SiteWise は、次のテンプレートを使用して、コールド階層に機器データ (測定値) を保存します。

```
{keyPrefix}/raw/startYear={startYear}/startMonth={startMonth}/startDay={startDay}/
seriesBucket={seriesBucket}/raw_{timeseriesId}_{startTimestamp}_{quality}.avro
```

Amazon S3 におけるraw データのファイルパスは、すべて次の構成要素を含んでいます。

パスコンポーネント。	説明
`keyPrefix`	AWS IoT SiteWise ストレージ設定で指定した Amazon S3 プレフィックス。Amazon S3 では、プレフィックスをバケット内のフォルダ名として使用します。

パスコンポーネント。	説明
raw	機器からの時系列データ (測定値) を格納するフォルダ。raw フォルダーは、プレフィックスフォルダーに保存されます。
seriesBucket	00 から ff の間の 16 進数です。timeSeriesId から導き出される数値です。このパーティションは、コールド階層への AWS IoT SiteWise 書き込み時にスループットを向上させるために使用されます。Amazon Athena を使用してクエリを実行する場合、パーティションを使用して細かい粒度のパーティショニングを行い、クエリパフォーマンスを向上させることができます。 アセットメタデータの seriesBucket と timeSeriesBucket は同じ数字です。
startYear	時系列データに関連する排他的開始時刻の年号。
startMonth	時系列データに関連付けられている排他的開始時刻の月。
startDay	時系列データに関連付けられている排他的開始時刻の月日。

パスコンポーネント。	説明
`fileName`	ファイル名は、アンダースコア (_) をデリミターとして使用し、次のように区切ります。 • `raw` プレフィックス。 • `timeSeriesId` 値。 • 時系列データに関連付けられた排他的開始時刻のエポックタイムスタンプ。 • データの品質。有効な値: `GOOD`、`BAD`、`UNCERTAIN` 。詳細については、AWS IoT SiteWise 「 API リファレンス」のAssetProperty「値」を参照してください。 [Snappy]圧縮を利用して、`.avro`形式で保存されます。

Example Cold 階層の raw データへのファイルパス

```
keyPrefix/raw/startYear=2021/startMonth=1/startDay=2/seriesBucket=a2/
raw_7020c8e2-e6db-40fa-9845-ed0dddd4c77d_95e63da7-d34e-43e1-
bc6f-1b490154b07a_1609577700_GOOD.avro
```

フィールド

Cold 階層に書き出しされる raw データのスキーマは、次のフィールドを含んでいる。

フィールド名	サポートされている型	デフォルトの 型	説明
`seriesId`	`string`	該当なし	機器からの時系列データ (測定値) を識別するための ID。このフィールドを使用して、クエリで raw

フィールド名	サポートされている型	デフォルトの 型	説明
			データとアセットメタデータを結合することができます。
timeInSeconds	long	該当なし	Unix エポック形式のタイムスタンプの日付 (秒単位)。分数ナノ秒のデータは、offsetInNanos で提供されます。
offsetInNanos	long	該当なし	timeInSeconds からのナノ秒のオフセット。
quality	string	該当なし	時系列値の品質。
doubleValue	double または null **	null	ダブル型 (浮動小数点数) の時系列データ。
stringValue	string または null **	null	文字列型 (文字の並び) の時系列データ。
integerValue	int または null **	null	整数型 (整数) の時系列データ。
booleanValue	boolean または null **	null	ブール型 (true または false) の時系列データ。
jsonValue	string または null **	null	JSON 型の時系列データ (複雑なデータ型を文字列として保存されたもの)。

フィールド名	サポートされている型	デフォルトの 型	説明
recordVersion	long または null **	null	レコードのバージョン番号。バージョン番号を使って、最新のレコードを選択することができます。新しいレコードほどバージョン番号が大きくなっています。

Example Cold 階層の raw データ

```
{"seriesId":"e9687d2a-0dbe-4f65-9ed6-6f443cba41f7_95e63da7-d34e-43e1-
bc6f-1b490154b07a","timeInSeconds":1625675887,"offsetInNanos":0,"quality":"GOOD","doubleValue":
{"double":0.75},"stringValue":null,"integerValue":null,"booleanValue":null,"jsonValue":null,"re
  {"seriesId":"e9687d2a-0dbe-4f65-9ed6-6f443cba41f7_95e63da7-d34e-43e1-
bc6f-1b490154b07a","timeInSeconds":1625675889,"offsetInNanos":0,"quality":"GOOD","doubleValue":
{"double":0.69},"stringValue":null,"integerValue":null,"booleanValue":null,"jsonValue":null,"re
  {"seriesId":"e9687d2a-0dbe-4f65-9ed6-6f443cba41f7_95e63da7-d34e-43e1-
bc6f-1b490154b07a","timeInSeconds":1625675890,"offsetInNanos":0,"quality":"GOOD","doubleValue":
{"double":0.66},"stringValue":null,"integerValue":null,"booleanValue":null,"jsonValue":null,"re
  {"seriesId":"e9687d2a-0dbe-4f65-9ed6-6f443cba41f7_95e63da7-d34e-43e1-
bc6f-1b490154b07a","timeInSeconds":1625675891,"offsetInNanos":0,"quality":"GOOD","doubleValue":
{"double":0.92},"stringValue":null,"integerValue":null,"booleanValue":null,"jsonValue":null,"re
  {"seriesId":"e9687d2a-0dbe-4f65-9ed6-6f443cba41f7_95e63da7-d34e-43e1-
bc6f-1b490154b07a","timeInSeconds":1625675892,"offsetInNanos":0,"quality":"GOOD","doubleValue":
{"double":0.73},"stringValue":null,"integerValue":null,"booleanValue":null,"jsonValue":null,"re
```

メトリクス、変換、集計

AWS IoT SiteWise は、メトリクス、変換、集計を 6 時間に 1 回コールド階層にエクスポートします。メトリクス、変換、集計は、コールド階層にApache Avro (.avro) の形式で保存されます。

ファイルパス

AWS IoT SiteWise は、次のテンプレートを使用して、メトリクス、変換、集計を Cold 階層に保存します。

```
{keyPrefix}/agg/startYear={startYear}/startMonth={startMonth}/startDay={startDay}/
seriesBucket={seriesBucket}/agg_{timeseriesId}_{startTimestamp}_{quality}.avro
```

Amazon S3 上のメトリクスへ、変換、集計のファイルパスには、次の要素が含まれます。

パスコンポーネント。	説明
keyPrefix	AWS IoT SiteWise ストレージ設定で指定した Amazon S3 プレフィックス。Amazon S3 では、プレフィックスをバケット内のフォルダ名として使用します。
agg	メトリクスの時系列データを格納するフォルダーです。agg フォルダーは、プレフィックスフォルダーに保存されます。
seriesBucket	00 から ff の間の 16 進数です。timeSeriesId から導き出される数値です。このパーティションは、コールド階層への AWS IoT SiteWise 書き込み時にスループットを向上させるために使用されます。Amazon Athena を使用してクエリを実行する場合、パーティションを使用して細かい粒度のパーティショニングを行い、クエリパフォーマンスを向上させることができます。 アセットメタデータの seriesBucket と timeSeriesBucket は同じ数字です。
startYear	時系列データに関連する排他的開始時刻の年号。
startMonth	時系列データに関連付けられている排他的開始時刻の月。
startDay	時系列データに関連付けられている排他的開始時刻の月日。

パスコンポーネント。	説明
fileName	ファイル名は、アンダースコア (_) をデリミターとして使用し、次のように区切ります。 • raw プレフィックス。 • timeSeriesId 値。 • 時系列データに関連付けられた排他的開始時刻のエポックタイムスタンプ。 • データの品質。有効な値: GOOD、BAD、UNCERTAIN 。詳細については、AWS IoT SiteWise 「 API リファレンス」のAssetProperty「値」を参照してください。 Snappy圧縮を利用して、.avro形式で保存されます。

Example Cold 階層のメトリクスへのファイルパス

```
keyPrefix/agg/startYear=2021/startMonth=1/startDay=2/seriesBucket=a2/
agg_7020c8e2-e6db-40fa-9845-ed0dddd4c77d_95e63da7-d34e-43e1-
bc6f-1b490154b07a_1609577700_GOOD.avro
```

フィールド

コールド階層に書き出されるメトリクス、変換、集計のスキーマには、次のフィールドが含まれます。

フィールド名	サポートされている型	デフォルトの 型	説明
seriesId	string	該当なし	機器、メトリクス、変換から時系列データを識別するための ID です。このフィー

フィールド名	サポートされている型	デフォルトの 型	説明
			ルドを使用して、クエリで raw データとアセットメタデータを結合することができます。
timeInSeconds	long	該当なし	Unix エポック形式のタイムスタンプの日付 (秒単位)。分数ナノ秒のデータは、offsetInNanos で提供されます。
offsetInNanos	long	該当なし	timeInSeconds からのナノ秒のオフセット。
quality	string	該当なし	アセットデータをフィルタリングするための品質です。
resolution	string	該当なし	データを集計する時間間隔。
count	double または null **	null	現在の時間間隔における、指定された変数のデータポイントの総数。
average	double または null **	null	現在の時間間隔における指定された変数値の平均。

フィールド名	サポートされている型	デフォルトの 型	説明
min	double または null **	null	現在の時間間隔における指定された変数の最小値。
max	boolean または null **	null	現在の時間間隔における指定された変数の最大値。
sum	string または null **	null	現在の時間間隔における指定された変数値の合計。
recordVersion	long または null **	null	レコードのバージョン番号。バージョン番号を使って、最新のレコードを選択することができます。新しいレコードほどバージョン番号が大きくなっています。

Example コールド階層のメトリックデータ

```
{"seriesId":"f74c2828-5317-4df3-
ba16-6d41b5bcb531","timeInSeconds":1637334060,"offsetInNanos":0,"quality":"GOOD","resolution":"
{"double":16.0},"min":{"double":1.0},"max":{"double":31.0},"sum":
{"double":496.0},"recordVersion":null}
  {"seriesId":"f74c2828-5317-4df3-
ba16-6d41b5bcb531","timeInSeconds":1637334120,"offsetInNanos":0,"quality":"GOOD","resolution":"
{"double":46.0},"min":{"double":32.0},"max":{"double":60.0},"sum":
{"double":1334.0},"recordVersion":null}
  {"seriesId":"f74c2828-5317-4df3-
ba16-6d41b5bcb531","timeInSeconds":1637334540,"offsetInNanos":0,"quality":"GOOD","resolution":"
{"double":16.0},"min":{"double":1.0},"max":{"double":31.0},"sum":
{"double":496.0},"recordVersion":null}
```

```
  {"seriesId":"f74c2828-5317-4df3-
ba16-6d41b5bcb531","timeInSeconds":1637334600,"offsetInNanos":0,"quality":"GOOD","resolution":"
{"double":46.0},"min":{"double":32.0},"max":{"double":60.0},"sum":
{"double":1334.0},"recordVersion":null}
  {"seriesId":"f74c2828-5317-4df3-
ba16-6d41b5bcb531","timeInSeconds":1637335020,"offsetInNanos":0,"quality":"GOOD","resolution":"
{"double":16.0},"min":{"double":1.0},"max":{"double":31.0},"sum":
{"double":496.0},"recordVersion":null}
```


アセットメタデータ

AWS IoT SiteWise が初めてコールド階層にデータをエクスポートできるようにすると、アセットメタデータはコールド階層にエクスポートされます。初期設定後、 はアセットモデル定義またはアセット定義を変更した場合にのみ、アセットメタデータを階層に AWS IoT SiteWise エクスポートします。アセットメタデータは、改行で区切られた JSON (`.ndjson`) 形式でコールド階層に保存されます。

ファイルパス

AWS IoT SiteWise は、次のテンプレートを使用してアセットメタデータを Cold 階層に保存します。

```
{keyPrefix}/asset_metadata/asset_{assetId}.ndjson
```

Cold 階層におけるアセットメタデータのファイルパスを生成するために、Nは次のテンプレートを使用する。

パスコンポーネント。	説明
`keyPrefix`	の AWS IoT SiteWiseストレージ設定で指定した Amazon S3 プレフィックス。 Amazon S3 では、プレフィックスをバケット内のフォルダ名として使用します。
`asset_metadata`	アセットのメタデータを保存するフォルダです。`asset_metadata` フォルダーは、プレフィックスフォルダーに保存されます。
`fileName`	ファイル名は、アンダースコア (_) をデリミターとして使用し、次のように区切ります。

パスコンポーネント。	説明
	• asset プレフィックス。 • assetId 値。 ファイルは .ndjson 形式で保存されます。

Example Cold 階層のアセットメタデータへのファイルパス

```
keyPrefix/asset_metadata/asset_35901915-d476-4dca-8637-d9ed4df939ed.ndjson
```

フィールド

Cold 階層に書き出しされるアセットメタデータのスキーマは、次のフィールドを含んでいます。

フィールド名	説明
assetId	アセットの ID。
assetName	アセットの名前。
assetExternalId	アセットの外部 ID。
assetModelId	このアセットを作成するために使用されたアセットモデルの ID。
assetModelName	アセットモデルの名前。
assetModelExternalId	アセットモデルの外部 ID。
assetPropertyId	アセットプロパティの ID。
assetPropertyName	アセットプロパティの名前。
assetPropertyExternalId	アセットプロパティの外部 ID。
assetPropertyDataType	アセットプロパティのデータ型。
assetPropertyUnit	アセットプロパティの単位 (例:NewtonsそしてRPM)。

フィールド名	説明
assetPropertyAlias	OPC-UA サーバのデータストリームのパスなど、アセットプロパティを識別するためのエイリアス (例: /company/windfarm/3/turbine/7/temperature)。
timeSeriesId	機器、メトリクス、変換から時系列データを識別するための ID です。このフィールドを使用して、クエリで raw データとアセットメタデータを結合することができます。
timeSeriesBucket	00 から ff の間の 16 進数です。timeSeriesId から導き出される数値です。このパーティションは、コールド階層への AWS IoT SiteWise 書き込み時にスループットを向上させるために使用されます。Amazon Athena を使用してクエリを実行する場合、パーティションを使用して細かい粒度のパーティショニングを行い、クエリパフォーマンスを向上させることができます。 raw データのファイルパスの timeSeriesBucket と seriesBucket が同じ数字です。
assetCompositeModelId	複合モデルの ID。
assetCompositeModelExternalId	複合モデルの外部 ID。
assetCompositeModelDescription	複合モデルの説明。
assetCompositeModelName	複合モデルの名前。
assetCompositeModelType	複合モデルのタイプ。アラーム複合モデルの場合、この型は AWS/ALARM です。
assetCreationDate	アセットが作成された日付 (Unix エポックタイム)。

フィールド名	説明
assetLastUpdateDate	アセットが最後に更新された日付を Unix エポックタイムで表したもの。
assetStatusErrorCode	エラーコードです。
assetStatusErrorMessage	エラーメッセージです。
assetStatusState	アセットの現在の状態。

Example Cold 階層のアセットメタデータ

```
  {"assetId":"7020c8e2-e6db-40fa-9845-
ed0dddd4c77d","assetExternalId":null,"assetName":"Wind Turbine Asset
 2","assetModelId":"ec1d924f-f07d-444f-b072-
e2994c165d35","assetModelExternalId":null,"assetModelName":"Wind
 Turbine Asset Model","assetPropertyId":"95e63da7-d34e-43e1-
bc6f-1b490154b07a","assetPropertyExternalId":null,"assetPropertyName":"Temperature","assetPrope
Washington/Seattle/WT2/temp","timeSeriesId":"7020c8e2-e6db-40fa-9845-
ed0dddd4c77d_95e63da7-d34e-43e1-
bc6f-1b490154b07a","timeSeriesBucket":"f6","assetArn":null,"assetCompositeModelDescription":nul
  {"assetId":"7020c8e2-e6db-40fa-9845-
ed0dddd4c77d","assetExternalId":null,"assetName":"Wind Turbine Asset
 2","assetModelId":"ec1d924f-f07d-444f-b072-
e2994c165d35","assetModelExternalId":null,"assetModelName":"Wind Turbine Asset
 Model","assetPropertyId":"c706d54d-4c11-42dc-9a01-63662fc697b4","assetPropertyExternalId":null
Washington/Seattle/WT2/pressure","timeSeriesId":"7020c8e2-e6db-40fa-9845-
ed0dddd4c77d_c706d54d-4c11-42dc-9a01-63662fc697b4","timeSeriesBucket":"1e","assetArn":null,"ass
  {"assetId":"7020c8e2-e6db-40fa-9845-
ed0dddd4c77d","assetExternalId":null,"assetName":"Wind Turbine Asset
 2","assetModelId":"ec1d924f-f07d-444f-b072-
e2994c165d35","assetModelExternalId":null,"assetModelName":"Wind
 Turbine Asset Model","assetPropertyId":"8cf1162f-dead-4fbe-b468-
c8e24cde9f50","assetPropertyExternalId":null,"assetPropertyName":"Max
 Temperature","assetPropertyDataType":"DOUBLE","assetPropertyUnit":null,"assetPropertyAlias":nu
e6db-40fa-9845-ed0dddd4c77d_8cf1162f-dead-4fbe-b468-
c8e24cde9f50","timeSeriesBucket":"d7","assetArn":null,"assetCompositeModelDescription":null,"as

 {"assetId":"3a5f2a22-3b37-4332-9c1c-404ea1d73fab","assetExternalId":null,"assetName":"BatchAss
ebc75e75e827","assetModelExternalId":null,"assetModelName":"FlashTestAssetModelDouble","assetPr
```

```
b410-
ab401a9176ed","assetPropertyExternalId":null,"assetPropertyName":"measurementProperty","assetPr
ae89-
ff316f5ff8aa","timeSeriesBucket":"af","assetArn":null,"assetCompositeModelDescription":null,"as
```

アセット階層メタデータ

を有効に AWS IoT SiteWise して をコールド階層に初めてデータを保存すると、アセット階層のメタデータがコールド階層にエクスポートされます。初期設定後、アセットモデルまたはアセット定義を変更した場合にのみ、 はアセット階層メタデータを Cold 階層に AWS IoT SiteWise エクスポートします。アセット階層メタデータは、改行で区切られた JSON (`.ndjson`) 形式でコールド階層に保存されます。

階層、ターゲットアセット、またはソースアセットの外部識別子は、 DescribeAsset API を呼び出すことで取得されます。

ファイルパス

AWS IoT SiteWise は、次のテンプレートを使用してアセット階層メタデータを Cold 階層に保存します。

```
{keyPrefix}/asset_hierarchy_metadata/{parentAssetId}_{hierarchyId}.ndjson
```

Cold 階層のアセット階層メタデータの各ファイルパスは、次のコンポーネントを含んでいます。

パスコンポーネント。	説明
`keyPrefix`	AWS IoT SiteWise ストレージ設定で指定した Amazon S3 プレフィックス。Amazon S3 では、プレフィックスをバケット内のフォルダ名として使用します。
`asset_hierarchy_metadata`	アセット階層のメタデータを保存するフォルダです。`asset_hierarchy_metadata` フォルダーは、プレフィックスフォルダーに保存されます。

パスコンポーネント。	説明
fileName	ファイル名は、アンダースコア (_) をデリミターとして使用し、次のように区切ります。 • parentAssetId 値。 • hierarchyId 値。 ファイルは .ndjson 形式で保存されます。

Example Cold 階層におけるアセット階層メタデータへのファイルパス

keyPrefix/asset_hierarchy_metadata/35901915-d476-4dca-8637-d9ed4df939ed_c5b3ced8-589a-48c7-9998-cdccfc9747a0.ndjson

フィールド

Cold 階層にに書き出しされるアセット階層メタデータのスキーマは、次のフィールドを含んでいます。

フィールド名	説明
sourceAssetId	このアセットリレーションシップのソースアセットの ID。
targetAssetId	このアセットリレーションシップのターゲットアセットの ID。
hierarchyId	階層の ID。
associationType	このアセットリレーションシップの関連付け型。 値は CHILD でなければならない。ターゲットアセットは、ソースアセットのアセットです。

Example Cold 階層におけるアセット階層メタデータ

```
{"sourceAssetId":"80388e72-2284-44fb-9c89-
bfbaf0dfedd2","targetAssetId":"2b866c25-0c74-4750-bdf5-
b73683c8a2a2","hierarchyId":"bbed9f59-0412-4585-
a61d-6044db526aee","associationType":"CHILD"}
  {"sourceAssetId":"80388e72-2284-44fb-9c89-
bfbaf0dfedd2","targetAssetId":"6b51246e-984d-460d-
bc0b-470ea47d1e31","hierarchyId":"bbed9f59-0412-4585-
a61d-6044db526aee","associationType":"CHILD"}
```

Cold 階層でデータを表示するには

1. [Amazon S3 console] (Amazon S3 のコンソール) に移動します。
2. ナビゲーションペインで、[バケット] を選択し、Amazon S3 バケットを選択します。
3. raw データ、アセットメタデータ、またはアセット階層メタデータが含まれるフォルダに移動します。
4. ファイルを選択し、[アクション] から[ダウンロード] を選択します。

ストレージデータインデックスファイル

AWS IoT SiteWise は、これらのファイルを使用してデータクエリのパフォーマンスを最適化します。これらは Amazon S3 バケットに表示されますが、使用する必要はありません。

ファイルパス

AWS IoT SiteWise は、次のテンプレートを使用してデータインデックスファイルを Cold 階層に保存します。

```
keyPrefix/index/series=timeseriesId/startYear=startYear/startMonth=startMonth/
startDay=startDay/index_timeseriesId_startTimestamp_quality
```

Example データストレージインデックスファイルへのファイルパス

```
keyPrefix/index/series=7020c8e2-e6db-40fa-9845-ed0dddd4c77d_95e63da7-
d34e-43e1-bc6f-1b490154b07a/startYear=2022/startMonth=02/startDay=03/
index_7020c8e2-e6db-40fa-9845-ed0dddd4c77d_95e63da7-d34e-43e1-
bc6f-1b490154b07a_1643846400_GOOD
```

のセキュリティ AWS IoT SiteWise

のクラウドセキュリティが最優先事項 AWS です。 AWS のお客様は、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャからメリットを得られます。

セキュリティは、 AWS とユーザーの間で共有される責任です。責任共有モデルではこれを、クラウドのセキュリティ、およびクラウド内でのセキュリティと説明しています:

- クラウドのセキュリティ — クラウドで AWS サービスを実行するインフラストラクチャを保護する責任 AWS は AWS にあります。 AWS また、 では、安全に使用できるサービスも提供しています。コンプライアンスAWS プログラムコンプライアンスプログラムコンプライアンスプログラムの一環として、サードパーティーの監査者は定期的にセキュリティの有効性をテストおよび検証します。に適用されるコンプライアンスプログラムの詳細については AWS IoT SiteWise、「コンプライアンスプログラムAWS による対象範囲内の のサービス」、「コンプライアンスプログラム」を参照してください。
- クラウドのセキュリティ — お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

このドキュメントは、 を使用する際の責任共有モデルの適用方法を理解するのに役立ちます AWS IoT SiteWise。以下のトピックでは、セキュリティおよびコンプライアンスの目的を達成するために AWS IoT SiteWise を設定する方法を示します。また、 AWS IoT SiteWise リソースのモニタリングや保護に役立つ他の AWS のサービスの使用方法についても説明します。

トピック

- のセキュリティのベストプラクティス AWS IoT SiteWise

でのデータ保護 AWS IoT SiteWise

責任 AWS 共有モデル、 でのデータ保護に適用されます AWS IoT SiteWise。このモデルで説明されているように、 AWS はすべての を実行するグローバルインフラストラクチャを保護する責任があります AWS クラウド。お客様は、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。また、使用する AWS のサービス のセキュリティ設定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「データプライバシーのよくある質問」を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログに投稿された記事「AWS 責任共有モデルおよび GDPR」を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management (IAM) を使用して個々のユーザーを設定することをお勧めします。この方法により、それぞれのジョブを遂行するために必要な権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 は必須であり TLS 1.3 がお勧めです。
- を使用して API とユーザーアクティビティのログ記録を設定します AWS CloudTrail。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API AWS を介して にアクセスするときに FIPS 140-2 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-2」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報は、タグ、または名前フィールドなどの自由形式のテキストフィールドに配置しないことを強くお勧めします。これは、コンソール、API、AWS IoT SiteWise または AWS のサービス SDK を使用して AWS CLIまたは他の を操作する場合も同様です。 AWS SDKs 名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

トピック

インターネットトラフィックのプライバシー

SiteWise Edge ゲートウェイなどの AWS IoT SiteWise とオンプレミスアプリケーション間の接続は、Transport Layer Security (TLS) 接続で保護されます。詳細については、「転送中の暗号化」を参照してください。

AWS IoT SiteWise は、 AWS リージョン内のアベイラビリティーゾーン間の接続やアカウント間の接続をサポートしていません AWS 。

IAM Identity Center は、一度に 1 つのリージョンでしか設定できません。 SiteWise Monitor は、IAM Identity Center 用に設定したリージョンに接続します。つまり、IAM Identity Center へのアクセスには 1 つのリージョンを使用しますが、どのリージョンにもポータルを作成することができます。

データ暗号化

データ暗号化とは、転送中 (との間でデータを送受信するとき AWS IoT SiteWise、および SiteWise Edge ゲートウェイとサーバー間でデータを送受信するとき) および保管中 (ローカルデバイスまたは AWS サービスに保存されているとき) のデータを保護することです。転送中のデータは、Transport Layer Security (TLS) を使用して保護することも、クライアント側の暗号化を使用して保存することもできます。

> **Note**
>
> AWS IoT SiteWise エッジ処理は、 SiteWise エッジゲートウェイ内でホストされ、ローカルネットワーク経由でアクセス可能な APIs を公開します。これらの APIs は、 AWS IoT SiteWise エッジコネクタが所有するサーバー証明書によってバックアップされた TLS 接続を介して公開されます。これらの API では、クライアント認証にアクセス制御パスワードが使用されます。サーバー証明書のプライベートキーとアクセスコントロールパスワードはどちらもディスクに保存されます。 AWS IoT SiteWise エッジ処理は、保管中のこれらの認証情報のセキュリティのためにファイルシステムの暗号化に依存します。

サーバー側の暗号化とクライアント側の暗号化の詳細については、以下のトピックを参照してください。

トピック

保管中の暗号化

AWS IoT SiteWise は、データを AWS クラウドと AWS IoT SiteWise Edge ゲートウェイに保存します。

AWS クラウドに保管中のデータ

AWS IoT SiteWise は、デフォルトで保管中のデータを暗号化 AWS する他のサービスにデータを保存します。保管時の暗号化は AWS Key Management Service（ AWS KMS) と統合され、 のアセットプロパティ値と集計値を暗号化するために使用される暗号化キーを管理します AWS IoT SiteWise。 AWS IoT SiteWiseのアセットプロパティ値と集計値の暗号化には、カスタマーマネージドキーの使用を選択することができます。 AWS KMSを通じて、暗号化キーの作成、管理、閲覧が可能です。

AWS 所有のキー を選択してデータを暗号化するか、カスタマーマネージドキーを選択してアセットプロパティ値と集計値を暗号化できます。

仕組み

保管時の暗号化は、データの暗号化に使用される暗号化キー AWS KMS を管理するために と統合されます。

- AWS 所有のキー – デフォルトの暗号化キー。このキー AWS IoT SiteWise を所有します。 AWS アカウントではこのキーを表示できません。また、 AWS CloudTrail の口グでキーに対する操作を確認することもできません。このキーは追加料金なしで使用することができます。
- カスタマーマネージドキー - キーは、お客様が作成、所有、管理するアカウントに保存されます。ユーザーは、KMS キーに関する完全なコントロール権を持ちます。 AWS KMS 追加料金が適用されます。

AWS 所有のキー

AWS 所有のキー は アカウントに保存されません。これらは、 AWS が所有および管理する KMS キーのコレクションの一部であり、複数の AWS accounts. AWS services がデータを保護するためにを使用できます AWS 所有のキー 。

を表示、管理、使用 AWS 所有のキー、またはそれらの使用を監査することはできません。ただし、データを暗号化するキーを保護するための作業やプログラムを操作したり変更したりする必要はありません。

を使用する場合、月額料金や使用料金は請求されず AWS 所有のキー、アカウントの AWS KMS クォータにもカウントされません。

カスタマーマネージドキー

カスタマーマネージドキーは、お客様が作成、所有、管理するアカウント内の KMS キーです。これらの KMS キーはお客様が完全に制御でき、次のような操作が可能です。

- キーポリシー、IAM ポリシー、グラントの確立と維持
- 有効化と無効化
- 暗号化マテリアルのローテーション
- タグの追加
- それらを参照するエイリアスの作成
- 削除のスケジュール設定

CloudTrail および Amazon CloudWatch Logs を使用して、 が AWS KMS ユーザーに代わって AWS IoT SiteWise に送信するリクエストを追跡することもできます。

カスタマーマネージドキーを使用している場合は、 アカウントに保存されている KMS キー AWS IoT SiteWise へのアクセスを許可する必要があります。 AWS IoT SiteWise は、エンベロープ暗号化とキー階層を使用してデータを暗号化します。 AWS KMS 暗号化キーは、このキー階層のルートキーを暗号化するために使用されます。詳細については、「AWS Key Management Service デベロッパーガイド」の「エンベロープ暗号化」を参照してください。

次のポリシー例では、ユーザーに代わってカスタマーマネージドキーの作成にアクセス AWS IoT SiteWise 許可を付与します。キーを作成する際に、`kms:CreateGrant` および `kms:DescribeKey` アクションを許可する必要があります。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1603902045292",
      "Action": [
```

```
        "kms:CreateGrant",
        "kms:DescribeKey"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

作成したグラントの暗号化コンテキストは、`aws:iotsitewise:subscriberId` とアカウント ID を使用します。

SiteWise Edge ゲートウェイの保管中のデータ

AWS IoT SiteWise ゲートウェイは、ローカルファイルシステムに次のデータを保存します。

- OPC-UA ソースの設定情報
- 接続した OPC-UA ソースからの OPC-UA データストリームパスのセット
- SiteWise Edge ゲートウェイがインターネットへの接続を失ったときにキャッシュされる産業用データ

SiteWise エッジゲートウェイは . AWS IoT Greengrass AWS IoT Greengrass relies on Unix file permissions and full-disk encryption (有効になっている場合) で実行され、コア上の保管中のデータを保護します。ファイルシステムとデバイスを保護するのはお客様の責任となります。

ただし、 AWS IoT Greengrass Secrets Manager から取得した OPC-UA サーバーシークレットのローカルコピーは暗号化されます。詳細については、[AWS IoT Greengrass Version 1 Developer Guide] (デベロッパーガイド) の[Secrets encryption] (秘密暗号化) を参照してください。

AWS IoT Greengrass コアでの保管時の暗号化の詳細については、「 AWS IoT Greengrass Version 1 デベロッパーガイド」の「保管時の暗号化」を参照してください。

転送中の暗号化

AWS IoT SiteWise には、データが転送される 3 つの通信モードがあります。

- インターネット経由 — ローカルデバイス (SiteWise エッジゲートウェイを含む) と 間の通信 AWS IoT SiteWise は暗号化されます。
- ローカルネットワーク経由 — SiteWise アプリケーションゲートウェイと SiteWise エッジゲートウェイ OpsHub の 間の通信は常に暗号化されます。ブラウザ内で実行されている SiteWise モニ

ターアプリケーションと SiteWise Edge ゲートウェイ間の通信は常に暗号化されます。 SiteWise Edge ゲートウェイと OPC-UA ソース間の通信は暗号化できます。

- SiteWise Edge ゲートウェイ上のコンポーネント間 — SiteWise Edge ゲートウェイ上の AWS IoT Greengrass コンポーネント間の通信は暗号化されません。

トピック

- インターネット経由で転送されるデータ
- ローカルネットワーク経由で転送されるデータ
- SiteWise Edge ゲートウェイ上のローカルコンポーネント間で転送中のデータ

インターネット経由で転送されるデータ

AWS IoT SiteWise は、Transport Layer Security (TLS) を使用して、インターネット経由のすべての通信を暗号化します。 AWS クラウドに送信されるすべてのデータは、MQTT または HTTPS プロトコルを使用して TLS 接続を介して送信されるため、デフォルトでは安全です。 SiteWise エッジゲートウェイは で実行され AWS IoT Greengrass、プロパティ値通知は AWS IoT トランスポートセキュリティモデルを使用します。詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Transport security] (トランスポートセキュリティ) を参照してください。

ローカルネットワーク経由で転送されるデータ

SiteWise エッジゲートウェイは、ローカル OPC-UA ソースとの通信に関する OPC-UA 仕様に従います。転送されるデータを暗号化するメッセージセキュリティモードを使用するようにソースを設定するのは、お客様の責任となります。

署名メッセージセキュリティモードを選択した場合、 SiteWise エッジゲートウェイとソース間で転送中のデータは署名されますが、暗号化されません。署名を選択し、メッセージセキュリティモードを暗号化すると、 SiteWise エッジゲートウェイとソース間で転送中のデータが署名され、暗号化されます。ソースの設定の詳細については、「データソースの設定」を参照してください。

エッジコンソールアプリケーションと SiteWise エッジゲートウェイ間の通信は常に TLS によって暗号化されます。 SiteWise Edge ゲートウェイの SiteWise Edge コネクタは、 AWS IoT SiteWise アプリケーションのエッジコンソールとの TLS 接続を確立できるように、自己署名証明書を生成して保存します。 AWS IoT SiteWise アプリケーションを SiteWise Edge ゲートウェイに接続する前に、この証明書を Edge ゲートウェイからアプリケーションの SiteWise エッジコンソールにコピーする必要があります。これにより、 AWS IoT SiteWise アプリケーションのエッジコンソールが信頼できる SiteWise エッジゲートウェイに接続していることを確認できます。

秘匿性とサーバーの信頼性のための TLS に加えて、 SiteWise Edge は SigV4 プロトコルを使用してエッジコンソールアプリケーションの信頼性を確立します。 SiteWise Edge ゲートウェイの SiteWise Edge コネクタは、エッジコンソールアプリケーション、ブラウザ内で実行されている SiteWise モニタリングアプリケーション、および AWS IoT SiteWise SDK に基づく他のクライアントからの受信接続を検証するためのパスワードを受け入れて保存します。

パスワードとサーバー証明書の生成については、the section called “ SiteWise Edge ゲートウェイの管理” を参照してください。

SiteWise Edge ゲートウェイ上のローカルコンポーネント間で転送中のデータ

SiteWise エッジゲートウェイは で実行され AWS IoT Greengrass、データはデバイスから離れないため、 AWS IoT Greengrass コア上でローカルに交換されるデータは暗号化されません。これには、 AWS IoT SiteWise コネクタなどの AWS IoT Greengrass コンポーネント間の通信が含まれます。詳細については、[AWS IoT Greengrass Version 1 Developer Guide] (デベロッパーガイド) の[Data on the core device] (コアデバイスのデータ) を参照してください。

キー管理

AWS IoT SiteWise クラウドキー管理

デフォルトでは、 AWS IoT SiteWise を使用して AWS クラウド内のデータ AWS マネージドキー を保護します。 AWS IoT SiteWiseの一部のデータを暗号化するために、カスタマーマネージドキーを使用するように設定を更新することができます。 AWS Key Management Service (AWS KMS). を通じて、暗号化キーの作成、管理、閲覧が可能です。

AWS IoT SiteWise は、 に保存されているカスタマーマネージドキーによるサーバー側の暗号化をサポートし AWS KMS 、次のデータを暗号化します。

- アセットプロパティ値。
- 集計値。

> **Note**
>
> その他のデータとリソースは、 によって管理されるキーによるデフォルトの暗号化を使用して暗号化されます AWS IoT SiteWise。このキーは、 AWS IoT SiteWise アカウントに保存されます。

詳細については、「 AWS Key Management Service デベロッパーガイド」の「 とは AWS Key Management Service」を参照してください。

カスタマーマネージドキーを使用した暗号化の有効化。

でカスタマーマネージドキーを使用するには AWS IoT SiteWise、設定を更新 AWS IoT SiteWise する必要があります。

KMS キーを使用した暗号化を有効にするには。

1. AWS IoT SiteWise コンソールに移動します。
2. [Account Settings] (アカウント設定) を選び、[Edit] (編集) を選んで[Edit account settings] (アカウント設定の編集) 画面を表示します。
3. [Encryption key type] (暗号化キー型) で[Choose a different AWS KMS key] (別のキーを選択) を選択します。これにより、 AWS KMSに保存されたカスタマーマネージドキーによる暗号化が有効となります。

   > **Note**
   >
   > 現在、カスタマーマネージドキーの暗号化は、アセットのプロパティ値と集計値に対してのみ使用可能です。

4. KMS キーは、次のいずれかを選択してください。

   - 既存の KMS キーを使用するには - リストから KMS キーエイリアスを選択します。
   - 新しい KMS キーを作成するには — AWS KMS キーの作成 を選択します。

   > **Note**
   >
   > これにより、 AWS KMS ダッシュボードが開きます。詳細については、[AWS Key Management Service Developer Guide] (デベロッパーガイド) の[Creating keys] (キーの作成) を参照してください。

5. [保存] を選択して、設定を更新します。

SiteWise エッジゲートウェイのキー管理

SiteWise エッジゲートウェイは で実行され AWS IoT Greengrass、 AWS IoT Greengrass コアデバイスはパブリックキーとプライベートキーを使用して AWS クラウドで認証し、OPC-UA 認証シークレットなどのローカルシークレットを暗号化します。詳細については、[AWS IoT Greengrass Version 1 Developer Guide] (デベロッパーガイド) の[Key management] (キー管理) を参照してください。

の Identity and Access Management AWS IoT SiteWise

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスを安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰に AWS IoT SiteWise リソースの使用を承認する (アクセス許可を付与する) かを制御します。IAM は、追加料金なしで AWS のサービス 使用できる です。

トピック

- 対象者
- アイデンティティを使用した認証
- が IAM と AWS IoT SiteWise 連携する方法
- AWS の マネージドポリシー AWS IoT SiteWise
- AWS IoT SiteWiseのサービスにリンクされたロールの使用
- AWS IoT Events アラームのアクセス許可の設定
- サービス間の混乱した代理の防止
- AWS IoT SiteWise ID とアクセスのトラブルシューティング

対象者

AWS Identity and Access Management (IAM) の使用方法は、 で行う作業によって異なります AWS IoT SiteWise。

サービスユーザー – AWS IoT SiteWise サービスを使用してジョブを実行する場合、管理者から必要な認証情報とアクセス許可が与えられます。さらに多くの AWS IoT SiteWise 機能を使用して作業を行う場合は、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解すると、管理者から適切な権限をリクエストするのに役に立ちます。 AWS IoT SiteWise機能にアクセスできない場合は、「AWS IoT SiteWise ID とアクセスのトラブルシューティング」を参照してください。

サービス管理者 – 社内の AWS IoT SiteWise リソースを担当している場合は、通常、 へのフルアクセスがあります AWS IoT SiteWise。サービスユーザーがどの AWS IoT SiteWise 機能やリソースにアクセスするかを決めるのは管理者の仕事です。その後、IAM 管理者にリクエストを送信して、サービスユーザーの権限を変更する必要があります。このページの情報を点検して、IAM の基本概念を理解してください。会社で IAM を で使用する方法の詳細については、 AWS IoT SiteWise「」を参照してくださいが IAM と AWS IoT SiteWise 連携する方法。

IAM 管理者 - 管理者は、 AWS IoT SiteWiseへのアクセスを管理するポリシーの書き込み方法の詳細について確認する場合があります。IAM で使用できる AWS IoT SiteWise アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS IoT SiteWise アイデンティティベースのポリシーの例。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAM ユーザーとして AWS アカウントのルートユーザー、または IAM ロールを引き受けて認証（ にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center（ IAM Identity Center) ユーザー、会社のシングルサインオン認証、Google または Facebook の認証情報は、フェデレーティッド ID の例です。フェデレーティッドアイデンティティとしてサインインする場合、IAM ロールを使用して、前もって管理者により ID フェデレーションが設定されています。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 ユーザーガイド」の「 にサインインする方法 AWS アカウントAWS サインイン 」を参照してください。

AWS プログラムで にアクセスする場合、 は Software Development Kit (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストに暗号で署名します。 AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、IAM ユーザーガイドの API AWS リクエストの署名を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 AWS では、アカウントのセキュリティを高めるために多要素認証 (MFA) を使用することをお勧めします。詳細については、『AWS IAM Identity Center ユーザーガイド』の「Multi-factor

authentication」(多要素認証) および『IAM ユーザーガイド』の「AWSにおける多要素認証 (MFA) の使用」を参照してください。

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての AWS のサービス およびリソースへの完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、IAM ユーザーガイドの「ルートユーザー認証情報が必要なタスク」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を保有する IAM ユーザーを作成する代わりに、一時認証情報を使用することをお勧めします。ただし、IAM ユーザーでの長期的な認証情報が必要な特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、IAM ユーザーガイドの「長期的な認証情報を必要とするユースケースのためにアクセスキーを定期的にローテーションする」を参照してください。

IAM グループは、IAM ユーザーの集団を指定するアイデンティティです。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、IAMAdmins という名前のグループを設定して、そのグループに IAM リソースを管理する権限を与えることができます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時的な認証情報が提供されます。詳細については、『IAM ユーザーガイド』の「IAM ユーザー (ロールではなく) の作成が適している場合」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーに似ていますが、特定のユーザーには関連付けられていません。ロール を切り替える

AWS Management Console ことで、 で IAM ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API AWS CLI オペレーションを呼び出すか、カスタム URL を使用します。ロールを使用する方法の詳細については、「IAM ユーザーガイド」の「IAM ロールの使用」を参照してください。

IAM ロールと一時的な認証情報は、次の状況で役立ちます:

- フェデレーションユーザーアクセス – フェデレーティッドアイデンティティに権限を割り当てるには、ロールを作成してそのロールの権限を定義します。フェデレーティッドアイデンティティが認証されると、そのアイデンティティはロールに関連付けられ、ロールで定義されている権限が付与されます。フェデレーションの詳細については、『IAM ユーザーガイド』の「サードパーティーアイデンティティプロバイダー向けロールの作成」 を参照してください。IAM アイデンティティセンターを使用する場合、権限セットを設定します。アイデンティティが認証後にアクセスできるものを制御するため、IAM Identity Center は、権限セットを IAM のロールに関連付けます。権限セットの詳細については、『AWS IAM Identity Center ユーザーガイド』の「権限セット」を参照してください。
- 一時的な IAM ユーザー権限 - IAM ユーザーまたはロールは、特定のタスクに対して複数の異なる権限を一時的に IAM ロールで引き受けることができます。
- クロスアカウントアクセス - IAM ロールを使用して、自分のアカウントのリソースにアクセスすることを、別のアカウントの人物 (信頼済みプリンシパル) に許可できます。クロスアカウントアクセス権を付与する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、 (ロールをプロキシとして使用する代わりに) ポリシーをリソースに直接アタッチできます。クロスアカウントアクセスにおけるロールとリソースベースのポリシーの違いについては、『IAM ユーザーガイド』の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。
- クロスサービスアクセス — 一部の は、他の の機能 AWS のサービス を使用します AWS のサービス。例えば、あるサービスで呼び出しを行うと、通常そのサービスによって Amazon EC2 でアプリケーションが実行されたり、Amazon S3 にオブジェクトが保存されたりします。サービスでは、呼び出し元プリンシパルの権限、サービスロール、またはサービスにリンクされたロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) — IAM ユーザーまたはロールを使用して でアクションを実行する場合 AWS、ユーザーはプリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があ

るリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

- サービスロール - サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「IAM ユーザーガイド」の「AWS のサービスにアクセス許可を委任するロールの作成」を参照してください。
- サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールの権限を表示できますが、編集することはできません。

- Amazon EC2 で実行されているアプリケーション – IAM ロールを使用して、EC2 インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2 インスタンス内でのアクセスキーの保存に推奨されます。 AWS ロールを EC2 インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルにはロールが含まれ、EC2 インスタンスで実行されるプログラムは一時的な認証情報を取得できます。詳細については、『IAM ユーザーガイド』の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用して権限を付与する」を参照してください。

IAM ロールと IAM ユーザーのどちらを使用するかについては、『IAM ユーザーガイド』の「(IAM ユーザーではなく) IAM ロールをいつ作成したら良いのか?」を参照してください。

が IAM と AWS IoT SiteWise 連携する方法

AWS Identity and Access Management (IAM) を使用して へのアクセスを管理する前に AWS IoT SiteWise、 で使用できる IAM 機能を理解しておく必要があります AWS IoT SiteWise。

IAM 機能	でサポートされていますか AWS IoT SiteWis
リソースレベルのアクセス許可を持つアイデンティティベースポリシー	はい
ポリシーアクション	Yes
ポリシーリソース	Yes
ポリシー条件キー	はい
リソースベースのポリシー	いいえ
アクセスコントロールリスト (ACL)	いいえ
タグベースの認証 (ABAC)	はい
一時的な認証情報	はい

IAM 機能	でサポートされていますか AWS IoT SiteWis
転送アクセスセッション (FAS)	はい
サービスリンクロール	はい
サービスロール	はい

AWS IoT SiteWise およびその他の AWS のサービスが IAM と連携する方法の概要を把握するには、「IAM ユーザーガイド」の「IAM AWS と連携する のサービス」を参照してください。

目次

- AWS IoT SiteWise アイデンティティベースのポリシー
  - ポリシーアクション
    - BatchPutAssetPropertyValue 認証
  - ポリシーリソース
  - ポリシー条件キー
  - 例
- AWS IoT SiteWise アイデンティティベースのポリシーの例
  - ポリシーのベストプラクティス
  - AWS IoT SiteWise コンソールを使用する
  - ユーザー自身のアクセス許可を表示することをユーザーに許可する
  - ユーザーが 1 つの階層でアセットにデータを取り込むことを許可する
  - タグに基づく AWS IoT SiteWise アセットの表示
- ポリシーを使用したアクセスの管理
  - アイデンティティベースのポリシー
  - リソースベースのポリシー
  - アクセスコントロールリスト (ACL)
  - その他のポリシータイプ
  - 複数のポリシータイプ

AWS IoT SiteWise IAM ロール

IAM ロール は、特定の権限を持つ、 AWS アカウント 内のエンティティです。

での一時的な認証情報の使用 AWS IoT SiteWise

一時的な認証情報を使用して、フェデレーションでサインインする、IAM ロールを引き受ける、またはクロスアカウントロールを引き受けることができます。一時的なセキュリティ認証情報を取得するには、 AssumeRoleやGetFederationトークン などの AWS STS API オペレーションを呼び出します。

AWS IoT SiteWise では、一時的な認証情報の使用がサポートされています。

SiteWise Monitor は、フェデレーティッドユーザーによるポータルへのアクセスをサポートしています。ポータルのユーザーは、IAM Identity Center または IAM の認証情報を使って認証を受けます。

> **Important**
>
> ユーザーまたはロールがこのポータルにサインインするには `iotsitewise:DescribePortal` アクセス許可が必要です。

ユーザーがポータルにサインインすると、 SiteWise Monitor は次のアクセス許可を提供するセッションポリシーを生成します。

- そのポータルのロールがアクセスを提供するアカウントの AWS IoT SiteWise のアセットとアセットデータへの読み取り専用アクセス。
- ユーザーが管理者 (プロジェクト所有者) または読み取り専用 (プロジェクトビューワー) のアクセス権を持つ、そのポータル内のプロジェクトへのアクセス。

フェデレーティッドユーザーのアクセス許可の詳細については、「のサービスロールの使用 AWS IoT SiteWise Monitor」を参照してください。

の転送アクセスセッション (FAS) AWS IoT SiteWise

転送アクセスセッション (FAS) をサポート	はい

IAM ユーザーまたはロールを使用して でアクションを実行する場合 AWS、ユーザーはプリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するためのアクセス許可が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

サービスリンクロール

サービスにリンクされたロールを使用すると、 AWS サービスは他の サービスのリソースにアクセスして、ユーザーに代わってアクションを実行できます。サービスリンクロールは、 AWS アカウント に表示され、サービスによって所有されます。IAM 管理者は、サービスリンクロールの許可を表示できますが、編集することはできません。

AWS IoT SiteWise は、サービスにリンクされたロールをサポートします。 AWS IoT SiteWise サービスにリンクされたロールの作成または管理の詳細については、「AWS IoT SiteWiseのサービスにリンクされたロールの使用」を参照してください。

サービスロール

この機能により、ユーザーに代わってサービスがサービスロールを引き受けることが許可されます。このロールにより、サービスがお客様に代わって他のサービスのリソースにアクセスし、アクションを完了することが許可されます。サービスロールは、 AWS アカウント に表示され、そのアカウントによって所有されます。つまり、IAM 管理者は、このロールの権限を変更できます。ただし、それにより、サービスの機能が損なわれる場合があります。

AWS IoT SiteWise はサービスロールを使用して、 SiteWise Monitor ポータルユーザーがユーザーに代わって一部の AWS IoT SiteWise リソースにアクセスできるようにします。詳細については、「のサービスロールの使用 AWS IoT SiteWise Monitor」を参照してください。

で AWS IoT Events アラームモデルを作成する前に、必要なアクセス許可が必要です AWS IoT SiteWise。詳細については、「AWS IoT Events アラームのアクセス許可の設定」を参照してください。

AWS IoT SiteWiseで IAM ロールを選択

でportalリソースを作成するときは AWS IoT SiteWise、 SiteWise Monitor ポータルのフェデレーティッドユーザーが AWS IoT SiteWise ユーザーに代わって にアクセスできるようにするロールを選択する必要があります。以前にサービスロールを作成したことがある場合、 AWS IoT SiteWise は選択するロールのリストを提供します。それ以外の場合は、ポータルの作成時に必要なアクセス許可を持つロールを作成できます。アセットとアセットデータへのアクセスを許可するロールを選択することが重要です。詳細については、「のサービスロールの使用 AWS IoT SiteWise Monitor」を参照してください。

AWS IoT SiteWise タグに基づく認可

AWS IoT SiteWise リソースにタグをアタッチしたり、 へのリクエストでタグを渡すことができます AWS IoT SiteWise。タグに基づいてアクセスを管理するには、`aws:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。 AWS IoT SiteWise リソースのタグ付けの詳細については、「リソースにタグを付ける AWS IoT SiteWise」を参照してください。

リソースのタグに基づいてリソースへのアクセスを制限するためのアイデンティティベースのポリシーの例を表示するには、「タグに基づく AWS IoT SiteWise アセットの表示」を参照してください。

AWS IoT SiteWise アイデンティティベースのポリシー

IAM ポリシーを使用すると、誰が で何をできるかを制御できます AWS IoT SiteWise。どのアクションを許可するかしないかを決定し、これらのアクションに特定の条件を設定できます。例えば、で情報を表示または変更できるユーザーに関するルールを作成できます AWS IoT SiteWise。 は、特定のアクション、リソース、および条件キー AWS IoT SiteWise をサポートします。JSON ポリシーで使用するすべての要素については、『IAM ユーザーガイド』の「IAM JSON policy elements reference (IAM JSON ポリシーエレメントのリファレンス)」を参照してください。

ポリシーアクション

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

JSON ポリシーのAction要素には、ポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連付けられた AWS API オペレーションと同じです。一致する API オペレーションのない権限のみのアクションなど、いくつかの例外があります。また、ポリシーに複数アクションが必要なオペレーションもあります。これらの追加アクションは、 依存アクション と呼ばれます。

このアクションは、関連付けられたオペレーションを実行するための権限を付与するポリシーで使用されます。

のポリシーアクションは、アクションの前にプレフィックス AWS IoT SiteWise を使用しますiotsitewise:。例えば、 BatchPutAssetPropertyValue API オペレーション AWS IoT SiteWise を使用して にアセットプロパティデータをアップロードするアクセス許可を付与するには、ポリシーに iotsitewise:BatchPutAssetPropertyValueアクションを含めます。ポリシーステートメントには、 Actionまたは NotAction element. AWS IoT SiteWise defines のいずれかを含める必要があります。このサービスで実行できるタスクを記述する独自のアクションのセットを定義します。

単一のステートメントに複数の アクションを指定するには、次のようにコンマで区切ります。

```
"Action": [
  "iotsitewise:action1",
  "iotsitewise:action2"
```

```
]
```

ワイルドカード *を使用して複数のアクションを指定することができます。例えば、Describe という単語で始まるすべてのアクションを指定するには、次のアクションを含めます。

```
"Action": "iotsitewise:Describe*"
```

AWS IoT SiteWise アクションのリストを確認するには、「IAM ユーザーガイド」の「 で定義されるアクション AWS IoT SiteWise」を参照してください。

BatchPutAssetPropertyValue 認証

AWS IoT SiteWise は、異常な方法で BatchPutAssetPropertyValue アクションへのアクセスを許可します。ほとんどのアクションでは、アクセスを許可または拒否すると、アクセス許可が付与されていない場合、そのアクションはエラーを返します。を使用するとBatchPutAssetPropertyValue、1 つの API リクエストで複数のデータ入力を異なるアセットやアセットプロパティに送信できます。 AWS IoT SiteWise は各データ入力を個別に承認します。リクエストで認証に失敗した個々のエントリの場合、 はエラーの返されたリストAccessDeniedExceptionに AWS IoT SiteWise を含めます。 は、同じリクエスト内の別のエントリが失敗した場合でも、 が許可して成功するすべてのエントリのデータ AWS IoT SiteWise を受け取ります。

> **Important**
>
> データストリームにデータを取り込む前に、次の操作を行います。
>
> - プロパティエイリアスを使用してデータストリームを識別するtime-series場合は、リソースを承認します。
> - アセット ID を使用して、関連付けられたアセットプロパティを含むアセットを識別する場合は、assetリソースを承認します。

ポリシーリソース

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースにどのような条件でアクションを実行できるかということです。

Resource JSON ポリシー要素は、アクションが適用されるオブジェクトを指定します。ステートメントには、Resource または NotResource要素を含める必要があります。ベストプラクティスと

して、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの権限と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

操作のリスト化など、リソースレベルの許可をサポートしないアクションの場合は、ワイルドカード (*) を使用して、ステートメントがすべてのリソースに適用されることを示します。

```
"Resource": "*"
```

各 IAM ポリシーステートメントは、ARN を使用して指定されたリソースに適用されます。ARN には以下の一般的な構文があります。

```
arn:${Partition}:${Service}:${Region}:${Account}:${ResourceType}/${ResourcePath}
```

ARN の形式の詳細については、「Amazon リソースネーム (ARNs AWS 「サービス名前空間」を参照してください。

たとえば、ステートメントで ID a1b2c3d4-5678-90ab-cdef-22222EXAMPLE のアセットを指定するには、次の ARN を使用します。

```
"Resource": "arn:aws:iotsitewise:region:123456789012:asset/a1b2c3d4-5678-90ab-
cdef-22222EXAMPLE"
```

特定のアカウントに属するすべてのデータストリームを指定する場合は、ワイルドカード (*) を使用します。

```
"Resource": "arn:aws:iotsitewise:region:123456789012:time-series/*"
```

特定のアカウントに属するすべてのアセットを指定するには、ワイルドカード (*) を使用します。

```
"Resource": "arn:aws:iotsitewise:region:123456789012:asset/*"
```

リソースを作成するための AWS IoT SiteWise アクションなど、一部のアクションは、特定のリソースで実行できません。このような場合は、ワイルドカード (*) を使用する必要があります。

```
"Resource": "*"
```

複数リソースを単一ステートメントで指定するには、ARN をカンマで区切ります。

```
"Resource": [
```

```
  "resource1",
  "resource2"
]
```

AWS IoT SiteWise リソースタイプとその ARNs」の「 で定義されるリソース AWS IoT SiteWise」を参照してください。 どのアクションで各リソースの ARN を指定できるかについては、「AWS IoT SiteWiseで定義されるアクション」を参照してください。

ポリシー条件キー

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

Condition 要素 (または Condition ブロック) を使用すると、ステートメントが有効な条件を指定できます。Condition 要素はオプションです。イコールや未満などの 条件演算子 を使用して条件式を作成することで、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の Condition 要素を指定するか、1 つの Condition 要素に複数のキーを指定すると、 AWS は AND 論理演算子を使用してそれらを評価します。1 つの条件キーに複数の値を指定すると、 は論理ORオペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば IAM ユーザーに、IAM ユーザー名がタグ付けされている場合のみリソースにアクセスできる権限を付与することができます。詳細については、『IAM ユーザーガイド』の「IAM ポリシーの要素: 変数およびタグ」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートします。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」のAWS 「 グローバル条件コンテキストキー」を参照してください。

> **Important**
>
> 多くの条件キーはリソースに固有のものであり、一部の API アクションでは複数のリソースを使用します。条件キーを使用してポリシーステートメントを作成する場合は、ポリシーステートメントの Resource 要素で、条件キーが適用されるリソースを指定します。指定しない場合、そのポリシーはユーザーに対してすべてのアクションの実行を禁止する可能性があります。これは、条件キーが適用されないリソースに対して条件チェックが失敗するためです。リソースを指定しない場合や、ポリシーの Action 要素に複数の API アクションを含めている場合は、...IfExists 条件タイプを使用して、条件キーが適用されないリソースに

対して無視されるようにする必要があります。詳細については、「IAM ユーザーガイド」の「...IfExists conditions」を参照してください。

AWS IoT SiteWise は独自の条件キーのセットを定義し、一部のグローバル条件キーの使用もサポートします。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」のAWS「 グローバル条件コンテキストキー」を参照してください。

AWS IoT SiteWise 条件キー

条件キー	説明	型
`iotsitewise:isAssociatedWithAssetProperty`	データストリームがアセットプロパティと関連付けられているかどうか。この条件キーを使用して、データストリームの関連するアセットプロパティの存在に基づくアクセス許可を定義することができます。 値の例: `true`	文字列
`iotsitewise:assetHierarchyPath`	アセットの階層パス。各アセット ID をスラッシュで区切った文字列です。この条件キーを使用して、アカウント内のすべてのアセットの階層のサブセットに基づいたアクセス許可を定義します。 値の例: `/a1b2c3d4-5678-90ab-cdef-222222EXAMPLE/a1b2c3d4-5678-90ab-cdef-666666EXAMPLE`	文字列
`iotsitewise:propertyId`	アセットプロパティの ID。この条件キーを使用して、ア	文字列

条件キー	説明	型
	セットモデルの指定されたプロパティに基づいたアクセス許可を定義します。この条件キーは、そのモデルのすべてのアセットに適用されます。 値の例: a1b2c3d4-5678-90ab-cdef-33333EXAMPLE	
iotsitewise:childAssetId	別のアセットに子として関連付けられているアセットの ID。この条件キーを使用して、子アセットに基づいたアクセス許可を定義します。親アセットに基づいてアクセス許可を定義するには、ポリシーステートメントのリソースセクションを使用します。 値の例: a1b2c3d4-5678-90ab-cdef-66666EXAMPLE	文字列
iotsitewise:iam	アクセスポリシーを一覧表示する際の IAM アイデンティティ のARN です。この条件キーを使用して、IAM アイデンティティのアクセスポリシー許可を定義します。 値の例: arn:aws:iam::123456789012:user/JohnDoe	文字列、Null

条件キー	説明	型
iotsitewise:propertyAlias	アセットプロパティまたはデータストリームを識別するエイリアス。この条件キーを使用して、エイリアスに基づくアクセス許可を定義します。	文字列
iotsitewise:user	アクセスポリシーをリスト化するときの IAM Identity Center ユーザーの ID。この条件キーを使用して、IAM Identity Center ユーザーのアクセスポリシーアクセス許可を定義します。 値の例: a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE	文字列、Null
iotsitewise:group	アクセスポリシーをリスト化するときの IAM Identity Center グループの ID。この条件キーを使用して、IAM Identity Center グループのアクセスポリシーアクセス許可を定義します。 値の例: a1b2c3d4e5-a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE	文字列、Null

条件キー	説明	型
iotsitewise:portal	アクセスポリシー内のポータルの ID。この条件キーを使用して、ポータルに基づいたアクセスポリシー許可を定義します。 値の例: a1b2c3d4-5678-90ab-cdef-77777EXAMPLE	文字列、Null
iotsitewise:project	アクセスポリシー内のプロジェクトの ID、またはダッシュボードのプロジェクトの ID。この条件キーを使用して、ポータルに基づいたダッシュボードまたはアクセスポリシー許可を定義します。 値の例: a1b2c3d4-5678-90ab-cdef-88888EXAMPLE	文字列、Null

条件キーを使用できるアクションとリソースについては、「 で定義されるアクション AWS IoT SiteWise」を参照してください。

例

AWS IoT SiteWise アイデンティティベースのポリシーの例を表示するには、「」を参照してくださいAWS IoT SiteWise アイデンティティベースのポリシーの例。

AWS IoT SiteWise アイデンティティベースのポリシーの例

デフォルトでは、エンティティ (ユーザーとロール) にはリソースを作成または変更 AWS IoT SiteWise するアクセス許可はありません。また、、 AWS Command Line Interface （ AWS CLI ） AWS Management Console、または AWS API を使用してタスクを実行することはできません。ア

クセス許可を調整するには、 AWS Identity and Access Management (IAM) 管理者が以下を実行する必要があります。

1. 必要なリソースに対して特定の API オペレーションを実行するアクセス許可をユーザーとロールに付与する IAM ポリシーを作成します。
2. これらのアクセス許可を必要とするユーザーまたはグループに、これらのポリシーをアタッチします。

これらの JSON ポリシードキュメント例を使用して IAM のアイデンティティベースのポリシーを作成する方法については、『IAM ユーザーガイド』の「JSON タブでのポリシーの作成」を参照してください。

トピック

- ポリシーのベストプラクティス
- AWS IoT SiteWise コンソールを使用する
- ユーザー自身のアクセス許可を表示することをユーザーに許可する
- ユーザーが 1 つの階層でアセットにデータを取り込むことを許可する
- タグに基づく AWS IoT SiteWise アセットの表示

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かが AWS IoT SiteWise リソースを作成、アクセス、または削除できるどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可を付与するには、多くの一般的なユースケースにアクセス許可を付与する AWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義して、アクセス許可をさらに減らすことをお勧めします。詳細については、IAM ユーザーガイドの「AWS マネージドポリシー」または「AWS ジョブ機能の管理ポリシー」を参照してください。
- 最小特権を適用する – IAM ポリシーで権限を設定するときは、タスクの実行に必要な権限のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権権限とも呼ばれています。IAM を使用して権限を適用する方法の詳細については、『IAM ユーザーガイド』の「IAM でのポリシーと権限」を参照してください。

- IAM ポリシーで条件を使用してアクセスをさらに制限する - ポリシーに条件を追加して、アクションやリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを SSL を使用して送信するように指定できます。条件を使用して、 などの特定の を介してサービスアクションが使用される場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます AWS CloudFormation。詳細については、IAM ユーザーガイドの [IAM JSON policy elements: Condition] (IAM JSON ポリシー要素：条件) を参照してください。
- IAM Access Analyzer を使用して IAM ポリシーを検証し、安全で機能的な権限を確保する - IAM Access Analyzer は、新規および既存のポリシーを検証して、ポリシーが IAM ポリシー言語 (JSON) および IAM のベストプラクティスに準拠するようにします。IAM アクセスアナライザーは 100 を超えるポリシーチェックと実用的な推奨事項を提供し、安全で機能的なポリシーの作成をサポートします。詳細については、「IAM ユーザーガイド」の「IAM Access Analyzer ポリシーの検証」を参照してください。
- 多要素認証 (MFA) を要求する – で IAM ユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化するために MFA を有効にします。API オペレーションが呼び出されるときに MFA を必須にするには、ポリシーに MFA 条件を追加します。詳細については、「IAM ユーザーガイド」の「MFA 保護 API アクセスの設定」を参照してください。

IAM でのベストプラクティスの詳細については、「IAM ユーザーガイド」の「IAM でのセキュリティのベストプラクティス」を参照してください。

AWS IoT SiteWise コンソールを使用する

AWS IoT SiteWise コンソールにアクセスするには、基本的なアクセス許可のセットが必要です。これらのアクセス許可により、 の AWS IoT SiteWise リソースに関する詳細を表示および管理できます AWS アカウント。

制限が厳しいポリシーを作成すると、そのポリシーを持つユーザーまたはロール (エンティティ) に対してコンソールが期待どおりに動作しない可能性があります。これらのエンティティが AWS IoT SiteWise 引き続きコンソールを使用できるようにするには、 AWSIoTSiteWiseConsoleFullAccess管理ポリシーをアタッチするか、それらのエンティティに同等のアクセス許可を定義します。詳細については、「IAM ユーザーガイド」の「ユーザーへのアクセス許可の追加」を参照してください。

エンティティがコンソールではなく AWS Command Line Interface (CLI) または AWS IoT SiteWise API のみを使用している場合、これらの最小限のアクセス許可は必要ありません。その場合は、API タスクに必要な特定のアクションへのアクセスを許可するだけです。

ユーザー自身のアクセス許可を表示することをユーザーに許可する

この例では、ユーザーアイデンティティにアタッチされたインラインおよびマネージドポリシーの表示を IAM ユーザーに許可するポリシーの作成方法を示します。このポリシーには、コンソールで、または AWS CLI または AWS API を使用してプログラムでこのアクションを実行するアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

ユーザーが 1 つの階層でアセットにデータを取り込むことを許可する

この例では、ルートアセット から始まる特定のアセット階層内のすべてのアセットプロパティにデータを書き込む AWS アカウント アクセス許可を のユーザーに付与しますa1b2c3d4-5678-90ab-cdef-22222EXAMPLE。このポリシーでは、ユーザーに iotsitewise:BatchPutAssetPropertyValue アクセス許可を付与します。このポリシーは、iotsitewise:assetHierarchyPath 条件キーを使用して、階層パスがアセットまたはその子孫と一致するアセットへのアクセスを制限します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PutAssetPropertyValuesForHierarchy",
      "Effect": "Allow",
      "Action": "iotsitewise:BatchPutAssetPropertyValue",
      "Resource": "arn:aws:iotsitewise:*:*:asset/*",
      "Condition": {
        "StringLike": {
          "iotsitewise:assetHierarchyPath": [
            "/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
            "/a1b2c3d4-5678-90ab-cdef-22222EXAMPLE/*"
          ]
        }
      }
    }
  ]
}
```

タグに基づく AWS IoT SiteWise アセットの表示

アイデンティティベースのポリシーの条件を使用して、タグに基づいて AWS IoT SiteWise リソースへのアクセスを制御します。この例では、アセットの表示を許可するポリシーを作成する方法を示します。ただし、アクセス許可は、アセットタグ Owner にそのユーザーのユーザー名の値がある場合のみ、付与されます。このポリシーは、コンソールでこのアクションを実行するアクセス許可も付与します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Sid": "ListAllAssets",
      "Effect": "Allow",
      "Action": [
        "iotsitewise:ListAssets",
        "iotsitewise:ListAssociatedAssets"
      ],
      "Resource": "*"
    },
    {
      "Sid": "DescribeAssetIfOwner",
      "Effect": "Allow",
      "Action": "iotsitewise:DescribeAsset",
      "Resource": "arn:aws:iotsitewise:*:*:asset/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Owner": "${aws:username}"
        }
      }
    }
  ]
}
```

このポリシーをアカウントのユーザーにアタッチします。という名前のユーザーが AWS IoT SiteWise アセットを表示richard-roeしようとする場合は、アセットに 0wner=richard-roeまたは のタグを付ける必要がありますowner=richard-roe。それ以外の場合、Richard はアクセスを拒否されます。条件タグのキー名では、大文字と小文字は区別されません。したがって、 は 0wnerと の両方0wnerに一致しますowner。詳細については、「IAM ユーザーガイド」の「IAM JSON ポリシー要素: 条件」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、 AWS ID またはリソースにアタッチします。ポリシーは AWS 、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義する のオブジェクトです。 は、プリンシパル (ユーザー、ルートユーザー、またはロールセッション) がリクエストを行うときに、これらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーは JSON ドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「IAM ユーザーガイド」の「JSON ポリシー概要」を参照してください。

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。

デフォルトでは、ユーザーやロールに権限はありません。IAM 管理者は、リソースで必要なアクションを実行するための権限をユーザーに付与する IAM ポリシーを作成できます。その後、管理者はロールに IAM ポリシーを追加し、ユーザーはロールを引き継ぐことができます。

IAM ポリシーは、オペレーションの実行方法を問わず、アクションの権限を定義します。例えば、`iam:GetRole` アクションを許可するポリシーがあるとします。そのポリシーを持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS API からロール情報を取得できます。

アイデンティティベースのポリシー

アイデンティティベースポリシーは、IAM ユーザー、ユーザーのグループ、ロールなど、アイデンティティにアタッチできる JSON 権限ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

アイデンティティベースポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 内の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。マネージドポリシーまたはインラインポリシーのいずれかを選択する方法については、『IAM ユーザーガイド』の「マネージドポリシーとインラインポリシーの比較」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースに添付する JSON ポリシードキュメントです。リソースベースのポリシーには例として、IAM ロールの信頼ポリシーや Amazon S3 バケットポリシーがあげられます。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスを制御できます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーでは、IAM の AWS マネージドポリシーを使用できません。

アクセスコントロールリスト (ACL)

アクセスコントロールリスト (ACL) は、どのプリンシパル (アカウントメンバー、ユーザー、またはロール) がリソースにアクセスするための権限を持つかをコントロールします。ACL はリソースベースのポリシーに似ていますが、JSON ポリシードキュメント形式は使用しません。

Amazon S3、 AWS WAF、および Amazon VPC は、ACLs。ACL の詳細については、『Amazon Simple Storage Service デベロッパーガイド』の「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、一般的ではない追加のポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 - アクセス許可の境界は、アイデンティティベースのポリシーによって IAM エンティティ (IAM ユーザーまたはロール) に付与できる権限の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。許可の境界の詳細については、「IAM ユーザーガイド」の「IAM エンティティの許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPs は、 の組織または組織単位 (OU) に対する最大アクセス許可を指定する JSON ポリシーです AWS Organizations。 AWS Organizations は、 AWS アカウント ビジネスが所有する複数の をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCP) を一部またはすべてのアカウントに適用できます。SCP は、各 を含むメンバーアカウントのエンティティのアクセス許可を制限します AWS アカウントのルートユーザー。Organizations と SCP の詳細については、『AWS Organizations ユーザーガイド』の「SCP の仕組み」を参照してください。
- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「IAM ユーザーガイド」の「セッションポリシー」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、IAM ユーザーガイドの「ポリシー評価ロジック」を参照してください。

AWS の マネージドポリシー AWS IoT SiteWise

自分でポリシーを記述するのではなく、 AWS 管理ポリシーを使用して、ユーザー、グループ、ロールにアクセス許可を簡単に追加できます。チームが正確なアクセス許可を付与する IAM カスタマー管理ポリシーを作成するには、時間と専門知識が必要です。セットアップを高速化するには、一般的なユースケースに AWS マネージドポリシーを使用することを検討してください。で AWS マネージドポリシーを検索します AWS アカウント。 AWS マネージドポリシーの詳細については、IAM ユーザーガイドの「AWS マネージドポリシー」を参照してください。

AWS サービスは、 AWS マネージドポリシーの更新と保守を行います。つまり、これらのポリシーのアクセス許可を変更することはできません。場合によっては、新機能に対応するためのアクセス許可を追加し、ポリシーがアタッチされているすべての ID に影響を与える AWS IoT SiteWise ことがあります。このような更新は、新しいサービスや機能の導入に共通しています。ただし、アクセス許可は削除されず、セットアップはそのまま維持されます。

さらに、 は、複数の サービスにまたがる職務機能の マネージドポリシー AWS をサポートします。例えば、 ReadOnlyアクセス AWS 管理ポリシーは、すべての AWS サービスとリソースへの読み取り専用アクセスを提供します。サービスが新機能を起動すると、 は新しいオペレーションとリソースの読み取り専用アクセス許可 AWS を追加します。ジョブ機能ポリシーのリスト付き説明については、[IAM User Guide] (IAM ユーザーガイド) の[AWS managed policies for job functions] (ジョブ関数のマネージドポリシー) を参照してください。

AWS マネージドポリシー： AWSIoTSiteWiseReadOnlyAccess

`AWSIoTSiteWiseReadOnlyAccess` AWS 管理ポリシーを使用して、 への読み取り専用アクセスを許可します AWS IoT SiteWise。

`AWSIoTSiteWiseReadOnlyAccess` ポリシーは IAM ID にアタッチできます。

サービスレベルのアクセス許可

このポリシーは、 への読み取り専用アクセスを提供します AWS IoT SiteWise。このポリシーには、他のサービス権限は含まれていません。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:Describe*",
                "iotsitewise:List*",
                "iotsitewise:BatchGet*",
                "iotsitewise:Get*"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS マネージドポリシー： AWSServiceRoleForIoTSiteWise

AWSServiceRoleForIoTSiteWise ロールは、次のアクセス許可で AWSServiceRoleForIoTSiteWise ポリシーを使用します。このポリシー：

- AWS IoT SiteWise が SiteWise Edge ゲートウェイ (で実行される) をデプロイできるようにしますAWS IoT Greengrass。
- ログ記録の実行を AWS IoT SiteWise に許可します。
- AWS IoT SiteWise が AWS IoT TwinMaker データベースに対してメタデータ検索クエリを実行できるようにします。

単一のユーザーアカウント AWS IoT SiteWise で を使用している場合、AWSServiceRoleForIoTSiteWiseロールは IAM アカウントにAWSServiceRoleForIoTSiteWiseポリシーを作成し、 AWSServiceRoleForIoTSiteWiseのサービスにリンクされたロール AWS IoT SiteWiseにアタッチします。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowSiteWiseReadGreenGrass",
   "Effect": "Allow",
   "Action": [
```

```
    "greengrass:GetAssociatedRole",
    "greengrass:GetCoreDefinition",
    "greengrass:GetCoreDefinitionVersion",
    "greengrass:GetGroup",
    "greengrass:GetGroupVersion"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AllowSiteWiseAccessLogGroup",
   "Effect": "Allow",
   "Action": [
    "logs:CreateLogGroup",
    "logs:DescribeLogGroups"
   ],
   "Resource": "arn:aws:logs:*:*:log-group:/aws/iotsitewise*"
  },
  {
   "Sid": "AllowSiteWiseAccessLog",
   "Effect": "Allow",
   "Action": [
    "logs:CreateLogStream",
    "logs:DescribeLogStreams",
    "logs:PutLogEvents"
   ],
   "Resource": "arn:aws:logs:*:*:log-group:/aws/iotsitewise*:log-stream:*"
  },
  {
   "Sid": "AllowSiteWiseAccessSiteWiseManagedWorkspaceInTwinMaker",
   "Effect": "Allow",
   "Action": [
    "iottwinmaker:GetWorkspace",
    "iottwinmaker:ExecuteQuery"
   ],
   "Resource": "arn:aws:iottwinmaker:*:*:workspace/*",
   "Condition": {
    "ForAnyValue:StringEquals": {
     "iottwinmaker:linkedServices": [
      "IOTSITEWISE"
     ]
    }
   }
  }
 ]
```

```
}
```

AWS IoT SiteWiseAWS 管理ポリシーの更新

このサービスが変更の追跡を開始した時点から AWS IoT SiteWise、 の AWS マネージドポリシーの更新に関する詳細を表示できます。このページの変更に関する自動アラートを受け取るには、 AWS IoT SiteWise ドキュメント履歴ページの RSS フィードをサブスクライブしてください。

変更	説明	日付
AWSServiceRoleForIoTSiteWise - 既存ポリシーへの更新	AWS IoT SiteWise は、データベースに対して AWS IoT TwinMaker メタデータ検索クエリを実行できるようになりました。	2023 年 11 月 6 日
AWSIoTSiteWiseReadOnlyAccess – 既存ポリシーへの更新	AWS IoT SiteWise に新しいポリシープレフィックス が追加されました。これによりBatchGet* 、バッチ読み取りオペレーションを実行できます。	2022 年 9 月 16 日
AWSIoTSiteWiseReadOnlyAccess - 新しいポリシー	AWS IoT SiteWise は、 への読み取り専用アクセスを許可する新しいポリシーを追加しました AWS IoT SiteWise。	2021 年 11 月 24 日
AWS IoT SiteWise が変更の追跡を開始しました	AWS IoT SiteWise が AWS マネージドポリシーの変更の追跡を開始しました。	2021 年 11 月 24 日

AWS IoT SiteWiseのサービスにリンクされたロールの使用

AWS IoT SiteWise は AWS Identity and Access Management 、 (IAM) サービスにリンクされたロール を使用します。サービスにリンクされたロールは、 に直接リンクされた一意のタイプの IAM ロールです AWS IoT SiteWise。サービスにリンクされたロールは、 によって事前定義 AWS IoT

SiteWise されており、ユーザーに代わってサービスから他の AWS のサービスを呼び出すために必要なすべてのアクセス許可が含まれています。

サービスにリンクされたロールは、必要なすべてのアクセス許可を自動的に含め AWS IoT SiteWise ることで、 の設定を簡素化します。 は、サービスにリンクされたロールのアクセス許可 AWS IoT SiteWise を定義します。特に定義されている場合を除き、 のみがそのロールを引き受け AWS IoT SiteWise ることができます。定義された許可には、信頼ポリシーと許可ポリシーが含まれます。また、そのアクセス許可ポリシーを他の IAM エンティティにアタッチすることはできません。

サービスリンクロールは、まずその関連リソースを削除しなければ削除できません。これにより、AWS IoT SiteWise リソースへのアクセス許可を誤って削除することがなくなるため、リソースが保護されます。

サービスにリンクされたロールをサポートする他サービスの情報については、「IAM と連動する AWS」をご参照のうえ、サービスにリンクされたロールの欄内の「はい」と表記されたサービスをご確認ください。[はい] のリンクを選択すると、該当するサービスのサービスリンクロールに関するドキュメントが表示されます。

トピック

- AWS IoT SiteWiseのサービスリンクロールのアクセス許可
- AWS IoT SiteWiseのサービスリンクロールの作成
- AWS IoT SiteWiseのサービスにリンクされたロールの編集
- AWS IoT SiteWiseのサービスリンクロールの削除
- AWS IoT SiteWise サービスにリンクされたロールでサポートされているリージョン
- のサービスロールの使用 AWS IoT SiteWise Monitor

AWS IoT SiteWiseのサービスリンクロールのアクセス許可

AWS IoT SiteWise は、 という名前のサービスにリンクされたロールを使用しますAWSServiceRoleForIoTSiteWise。 AWS IoT SiteWise は、このサービスにリンクされたロールを使用して SiteWise Edge ゲートウェイ (で実行される AWS IoT Greengrass) をデプロイし、ログ記録を実行します。

`AWSServiceRoleForIoTSiteWise` サービスにリンクされたロールは、次のアクセス許可を持つ`AWSServiceRoleForIoTSiteWise`ポリシーを使用します。このポリシー：

- AWS IoT SiteWise が SiteWise Edge ゲートウェイ (で実行される) をデプロイできるようにしますAWS IoT Greengrass。

- ログ記録の実行を AWS IoT SiteWise に許可します。
- AWS IoT SiteWise が AWS IoT TwinMaker データベースに対してメタデータ検索クエリを実行できるようにします。

で許可されるアクションの詳細についてはAWSServiceRoleForIoTSiteWise、「 の AWS マネージドポリシー AWS IoT SiteWise」を参照してください。

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "AllowSiteWiseReadGreenGrass",
   "Effect": "Allow",
   "Action": [
    "greengrass:GetAssociatedRole",
    "greengrass:GetCoreDefinition",
    "greengrass:GetCoreDefinitionVersion",
    "greengrass:GetGroup",
    "greengrass:GetGroupVersion"
   ],
   "Resource": "*"
  },
  {
   "Sid": "AllowSiteWiseAccessLogGroup",
   "Effect": "Allow",
   "Action": [
    "logs:CreateLogGroup",
    "logs:DescribeLogGroups"
   ],
   "Resource": "arn:aws:logs:*:*:log-group:/aws/iotsitewise*"
  },
  {
   "Sid": "AllowSiteWiseAccessLog",
   "Effect": "Allow",
   "Action": [
    "logs:CreateLogStream",
    "logs:DescribeLogStreams",
    "logs:PutLogEvents"
   ],
   "Resource": "arn:aws:logs:*:*:log-group:/aws/iotsitewise*:log-stream:*"
  },
  {
```

```
    "Sid": "AllowSiteWiseAccessSiteWiseManagedWorkspaceInTwinMaker",
    "Effect": "Allow",
    "Action": [
     "iottwinmaker:GetWorkspace",
     "iottwinmaker:ExecuteQuery"
    ],
    "Resource": "arn:aws:iottwinmaker:*:*:workspace/*",
    "Condition": {
     "ForAnyValue:StringEquals": {
      "iottwinmaker:linkedServices": [
       "IOTSITEWISE"
      ]
     }
    }
   }
  ]
}
```

ログを使用して Edge SiteWise ゲートウェイをモニタリングおよびトラブルシューティングできます。詳細については、「SiteWise Edge ゲートウェイログの監視」を参照してください。

IAM エンティティ (ユーザ、グループ、ロールなど) にサービスリンクロールの作成、編集、削除を許可するには、まずアクセス許可を設定します。詳細については、IAM ユーザーガイド の「サービスリンクロールの権限」を参照してください。

AWS IoT SiteWiseのサービスリンクロールの作成

サービスリンクロールを手動で作成する必要はありません。 AWS IoT SiteWise コンソールで次のオペレーションを実行すると、 によってサービスにリンクされたロールが自動的に AWS IoT SiteWise 作成されます。

- Greengrass V1 ゲートウェイを作成します。
- ログ記録オプションを設定します。
- 実行クエリバナーでオプトインボタンを選択します。

このサービスリンクロールを削除した後で再度作成する必要が生じた場合は、同じ方法でアカウントにロールを再作成できます。 AWS IoT SiteWise コンソールでオペレーションを実行すると、 によってサービスにリンクされたロールが再度 AWS IoT SiteWise 作成されます。

IAM コンソールまたは API を使用して、 AWS IoT SiteWiseに対してサービスにリンクされたロールを作成することもできます。

- IAM コンソールでこれを行うには、 AWSServiceRoleForIoTSiteWiseポリシーと との信頼関係を持つロールを作成しますiotsitewise.amazonaws.com。
- AWS CLI または IAM API を使用してこれを行うには、 arn:aws:iam::aws:policy/aws-service-role/AWSServiceRoleForIoTSiteWiseポリシーと との信頼関係を持つロールを作成しますiotsitewise.amazonaws.com。

詳細については、『IAM ユーザーガイド』の「サービスにリンクされたロールの作成」を参照してください。

このサービスにリンクされたロールを削除する場合、この同じプロセスを使用して、もう一度ロールを作成できます。

AWS IoT SiteWiseのサービスにリンクされたロールの編集

AWS IoT SiteWise では、 AWSServiceRoleForIoTSiteWise サービスにリンクされたロールを編集することはできません。サービスリンクロールを作成すると、多くのエンティティによってロールが参照される可能性があるため、ロール名を変更することはできません。ただし、IAM を使用したロール記述の編集はできます。詳細については、「IAM ユーザーガイド」の「サービスにリンクされたロールの編集」を参照してください。

AWS IoT SiteWiseのサービスリンクロールの削除

サービスにリンクされたロールを必要とする機能またはサービスが使用されなくなった場合は、関連付けられたロールを削除することをお勧めします。これは、モニタリングまたは保守されていない非アクティブなエンティティが発生しないようにするためです。ただし、手動で削除する前に、サービスリンクロールのリソースをクリーンアップする必要があります。

> **Note**
>
> リソースを削除しようとしたときに AWS IoT SiteWise サービスがロールを使用している場合、削除が失敗する可能性があります。その場合は、数分待ってから再試行してください。

が使用する AWS IoT SiteWise リソースを削除するには AWSServiceRoleForIoTSiteWise

1. のログ記録を無効にします AWS IoT SiteWise。詳細については、「ロギングレベルの変更」を参照してください。
2. アクティブな SiteWise Edge ゲートウェイをすべて削除します。

サービスにリンクされたロールを IAM で手動削除するには

IAM コンソール、、または AWS API を使用して AWS CLI、 AWSServiceRoleForIoTSiteWise サービスにリンクされたロールを削除します。詳細については、「IAM ユーザーガイド」の「サービスリンクロールの削除」を参照してください。

AWS IoT SiteWise サービスにリンクされたロールでサポートされているリージョン

AWS IoT SiteWise は、サービスが利用可能なすべてのリージョンで、サービスにリンクされたロールの使用をサポートします。詳細については、「AWS IoT SiteWise エンドポイントとクォータ」を参照してください。

のサービスロールの使用 AWS IoT SiteWise Monitor

サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「IAM ユーザーガイド」の「AWS のサービスにアクセス許可を委任するロールの作成」を参照してください。

フェデレーション SiteWise モニターポータルユーザーが AWS IoT SiteWiseおよび AWS IAM Identity Center リソースにアクセスできるようにするには、作成する各ポータルにサービスロールをアタッチする必要があります。サービスロールは、信頼できるエンティティとして SiteWise Monitor を指定し、 AWSIoTSiteWiseMonitorPortalAccess 管理ポリシーを含めるか、同等のアクセス許可 を定義する必要があります。このポリシーは によって維持 AWS され、 SiteWise Monitor が AWS IoT SiteWise および IAM Identity Center リソースにアクセスするために使用する一連のアクセス許可を定義します。

SiteWise Monitor ポータルを作成するときは、そのポータルのユーザーが AWS IoT SiteWiseおよび IAM Identity Center リソースにアクセスできるようにするロールを選択する必要があります。 AWS IoT SiteWise コンソールでロールを作成して設定できます。ロールは IAM で後から編集できます。ロールから必要なアクセス許可を削除するか、ロールを削除すると、ポータルユーザーは SiteWise Monitor ポータルの使用に問題が発生します。

> **Note**
>
> 2020 年 4 月 29 日より前に作成されたポータルでは、サービスロールは必須ではありませんでした。この日付より前に作成されたポータルを引き続き使用する場合は、サービスロールをアタッチする必要があります。これを行うには、[AWS IoT SiteWise console] (コンソール)

で[Portals] (ポータル) ページに移動し、[Migrate all portals to use IAM roles] (すべてのポータルでIAMロールを使用できるように移行する) を選択します。

以下のセクションでは、 AWS Management Console または で SiteWise Monitor サービスロールを作成および管理する方法を説明します AWS Command Line Interface。

目次

SiteWise Monitor のサービスロールのアクセス許可

ポータルを作成すると、 は名前が で始まるロール AWS IoT SiteWise を作成しますAWSIoTSiteWiseMonitorServiceRole。このロールにより、フェデレーション SiteWise モニターユーザーはポータル設定、アセット、アセットデータ、IAM Identity Center 設定にアクセスできます。

このロールは、その前提として以下のサービスを信頼します。

- `monitor.iotsitewise.amazonaws.com`

このロールは、名前が で始まる次のアクセス許可ポリシーを使用してAWSIoTSiteWiseMonitorServicePortalPolicy、 SiteWise Monitor ユーザーがアカウント内のリソースに対してアクションを実行できるようにします。AWSIoTSiteWiseMonitorPortalAccess 管理ポリシーは、同等のアクセス許可を定義します。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribePortal",
                "iotsitewise:CreateProject",
                "iotsitewise:DescribeProject",
                "iotsitewise:UpdateProject",
                "iotsitewise:DeleteProject",
                "iotsitewise:ListProjects",
                "iotsitewise:BatchAssociateProjectAssets",
                "iotsitewise:BatchDisassociateProjectAssets",
                "iotsitewise:ListProjectAssets",
                "iotsitewise:CreateDashboard",
                "iotsitewise:DescribeDashboard",
                "iotsitewise:UpdateDashboard",
                "iotsitewise:DeleteDashboard",
                "iotsitewise:ListDashboards",
                "iotsitewise:CreateAccessPolicy",
                "iotsitewise:DescribeAccessPolicy",
                "iotsitewise:UpdateAccessPolicy",
                "iotsitewise:DeleteAccessPolicy",
                "iotsitewise:ListAccessPolicies",
                "iotsitewise:DescribeAsset",
                "iotsitewise:ListAssets",
                "iotsitewise:ListAssociatedAssets",
                "iotsitewise:DescribeAssetProperty",
                "iotsitewise:GetAssetPropertyValue",
                "iotsitewise:GetAssetPropertyValueHistory",
                "iotsitewise:GetAssetPropertyAggregates",
                "iotsitewise:BatchPutAssetPropertyValue",
                "iotsitewise:ListAssetRelationships",
                "iotsitewise:DescribeAssetModel",
                "iotsitewise:ListAssetModels",
                "iotsitewise:UpdateAssetModel",
                "iotsitewise:UpdateAssetModelPropertyRouting",
                "sso-directory:DescribeUsers",
                "sso-directory:DescribeUser",
                "iotevents:DescribeAlarmModel",
                "iotevents:ListTagsForResource"
            ],
            "Resource": "*"
        },
```

```
        {
            "Effect": "Allow",
            "Action": [
                "iotevents:BatchAcknowledgeAlarm",
                "iotevents:BatchSnoozeAlarm",
                "iotevents:BatchEnableAlarm",
                "iotevents:BatchDisableAlarm"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "iotevents:keyValue": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotevents:CreateAlarmModel",
                "iotevents:TagResource"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/iotsitewisemonitor": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotevents:UpdateAlarmModel",
                "iotevents:DeleteAlarmModel"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/iotsitewisemonitor": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "iotevents.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

アラームに必要なアクセス許可の詳細については、AWS IoT Events アラームのアクセス許可の設定を参照してください。

ポータルユーザーがサインインすると、 SiteWise Monitor はサービスロールとそのユーザーのアクセスポリシーの共通部分に基づいてセッションポリシーを作成します。アクセスポリシーによって、ポータルおよびプロジェクトへの アイデンティティのアクセスレベルが定義されます。ポータルのアクセス許可とアクセスポリシーの詳細については、 SiteWise 監視ポータルの管理「」およびCreateAccess「 ポリシー」を参照してください。

SiteWise Monitor サービスロールの管理 (コンソール)

は、 ポータルの SiteWise Monitor サービスロールの管理 AWS IoT SiteWise コンソール を容易にします。ポータルを作成すると、コンソールはアタッチメントに適した既存のロールをチェックします。使用可能なものがない場合は、コンソールでサービスロールを作成して設定できます。詳細については、「ポータルの作成」を参照してください。

トピック

- ポータルのサービスロールの確認 (コンソール)
- SiteWise Monitor サービスロールの作成AWS IoT SiteWise （ コンソール ）
- SiteWise Monitor サービスロールの作成 (IAM コンソール)
- ポータルのサービスロールの変更 (コンソール)

ポータルのサービスロールの確認 (コンソール)

SiteWise Monitor ポータルにアタッチされているサービスロールを見つけるには、次のステップに従います。

ポータルのサービスロールを確認するには

1. AWS IoT SiteWise コンソールに移動します。
2. 左のナビゲーションペインで、[ポータル] を選択します。
3. サービスロールを確認するポータルを選択します。

   ポータルにアタッチされているロールが [アクセス許可]、[サービスロール] で表示されます。

SiteWise Monitor サービスロールの作成AWS IoT SiteWise（ コンソール ）

SiteWise Monitor ポータルを作成するときに、ポータルのサービスロールを作成できます。詳細については、「ポータルの作成」を参照してください。

AWS IoT SiteWise コンソールで既存のポータルのサービスロールを作成することもできます。これは、ポータルの既存のサービスロールを置き換えるものです。

既存のポータルのサービスロールを作成するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Portals (ポータル)] を選択します。
3. 新しいサービスロールを作成するポータルを選択します。
4. [Portal details (ポータルの詳細)] で、[編集] を選択します。
5. [アクセス許可] で、リストから [Create and use a new service role (新しいサービスロールを作成および使用)] を選択します。
6. 新しいロールの名前を入力します。
7. [保存] を選択します。

SiteWise Monitor サービスロールの作成 (IAM コンソール)

IAM コンソールのサービスロールテンプレートから、サービスロールを作成することができます。このロールテンプレートには AWSIoTSiteWiseMonitorPortalAccess管理ポリシーが含まれており、信頼できるエンティティとして SiteWise Monitor を指定します。

ポータルサービスロールテンプレートからサービスロールを作成するステップ。

1. [IAM console] (IAM コンソール) に入ります。
2. ナビゲーションペインで [Roles (ロール)] を選択します。

3. [ロールの作成] を選択します。
4. ユースケースの選択 で、IoT SiteWiseを選択します。
5. [Select your use case] (ユースケースの選択) で、IoT SiteWise Monitor - Portal を選択します。
6. [Next: Permissions (次へ: アクセス許可)] を選択します。
7. [次へ: タグ] を選択します。
8. [次へ: レビュー] を選択します。
9. 新しいサービスロールの[Role name] (ロール名) を入力します。
10. [ロールの作成] を選択します。

ポータルのサービスロールの変更 (コンソール)

ポータルの別の SiteWise Monitor サービスロールを選択するには、次の手順に従います。

ポータルのサービスロールを変更するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、[Portals (ポータル)] を選択します。
3. サービスロールを変更するポータルを選択します。
4. [Portal details (ポータルの詳細)] で、[編集] を選択します。
5. [アクセス許可] で、[Use an existing role (既存のロールを使用)] を選択します。
6. そのポータルにアタッチする既存のロールを選択します。
7. [保存] を選択します。

SiteWise Monitor サービスロールの管理 (CLI)

は、次のポータルサービスロール管理タスク AWS CLI に使用できます。

トピック

- ポータルのサービスロールの確認 (CLI)
- SiteWise Monitor サービスロールの作成 (CLI)

ポータルのサービスロールの確認 (CLI)

SiteWise Monitor ポータルにアタッチされているサービスロールを見つけるには、次のコマンドを実行して、現在のリージョン内のすべてのポータルを一覧表示します。

```
aws iotsitewise list-portals
```

このオペレーションは、ポータルの概要を含むレスポンスを次の形式で返します。

```
{
  "portalSummaries": [
    {
      "id": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
      "name": "WindFarmPortal",
      "description": "A portal that contains wind farm projects for Example Corp.",
      "roleArn": "arn:aws:iam::123456789012:role/service-role/role-name",
      "startUrl": "https://a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE.app.iotsitewise.aws",
      "creationDate": "2020-02-04T23:01:52.90248068Z",
      "lastUpdateDate": "2020-02-04T23:01:52.90248078Z"
    }
  ]
}
```

ポータルの ID がわかっている場合は、 DescribePortalオペレーションを使用してポータルのロールを検索することもできます。

SiteWise Monitor サービスロールの作成 (CLI)

新しい SiteWise Monitor サービスロールを作成するには、次のステップに従います。

SiteWise Monitor サービスロールを作成するには

1. SiteWise Monitor がロールを引き受けることを許可する信頼ポリシーを持つロールを作成します。この例では、JSON 文字列に保存された信頼ポリシーから **MySiteWiseMonitorPortalRole** という名前のロールを作成します。

   Linux, macOS, or Unix

   ```
   aws iam create-role --role-name MySiteWiseMonitorPortalRole --assume-role-
   policy-document '{
     "Version": "2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": {
           "Service": "monitor.iotsitewise.amazonaws.com"
   ```

```
      },
      "Action": "sts:AssumeRole"
    }
  ]
}'
```

Windows command prompt

```
aws iam create-role --role-name MySiteWiseMonitorPortalRole --assume-role-
policy-document "{\"Version\":\"2012-10-17\",\"Statement\":[{\"Effect\":\"Allow
\",\"Principal\":{\"Service\":\"monitor.iotsitewise.amazonaws.com\"},\"Action\":
\"sts:AssumeRole\"}]}"
```

2. 出力のロールメタデータからロールの ARN をコピーします。ポータルの作成時に、この ARN を使用してロールとポータルを関連付けます。ポータルの作成の詳細については、 API リファレンスのCreatePortal「」を参照してください。 AWS IoT SiteWise
3. AWSIoTSiteWiseMonitorPortalAccess ポリシーをロールにアタッチするか、同等のアクセス許可を定義するポリシーをアタッチします。

```
aws iam attach-role-policy --role-name MySiteWiseMonitorPortalRole --policy-arn
 arn:aws:iam::aws:policy/service-role/AWSIoTSiteWiseMonitorPortalAccess
```

既存のポータルにサービスロールをアタッチするには

1. ポータルの既存の詳細を取得するには、次のコマンドを実行します。*portal-id* をポータルの ID で置き換えてください。

```
aws iotsitewise describe-portal --portal-id portal-id
```

このオペレーションは、ポータルの詳細を含むレスポンスを次の形式で返します。

```
{
    "portalId": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
    "portalArn": "arn:aws:iotsitewise:region:account-id:portal/a1b2c3d4-5678-90ab-
cdef-aaaaaEXAMPLE",
    "portalName": "WindFarmPortal",
    "portalDescription": "A portal that contains wind farm projects for Example
 Corp.",
    "portalClientId": "E-1a2b3c4d5e6f_sn6tbqHVzLWVEXAMPLE",
```

```
    "portalStartUrl": "https://a1b2c3d4-5678-90ab-cdef-
aaaaaEXAMPLE.app.iotsitewise.aws",
    "portalContactEmail": "support@example.com",
    "portalStatus": {
        "state": "ACTIVE"
    },
    "portalCreationDate": "2020-04-29T23:01:52.90248068Z",
    "portalLastUpdateDate": "2020-04-29T00:28:26.103548287Z",
    "roleArn": "arn:aws:iam::123456789012:role/service-role/
AWSIoTSiteWiseMonitorServiceRole_1aEXAMPLE"
}
```

2. サービスロールをポータルにアタッチするには、次のコマンドを実行します。*role-arn* をサービスロールの ARN で置き換え、その他のパラメータをポータルの既存の値で置き換えてください。

```
aws iotsitewise update-portal \
  --portal-id portal-id \
  --role-arn role-arn \
  --portal-name portal-name \
  --portal-description portal-description \
  --portal-contact-email portal-contact-email
```

SiteWise の更新をモニタリングする AWSIoTSiteWiseMonitorServiceRole

AWSIoTSiteWiseMonitorServiceRole for SiteWise Monitor の更新に関する詳細は、このサービスが変更の追跡を開始した時点から確認できます。このページの変更に関する自動アラートを受け取るには、 AWS IoT SiteWise ドキュメント履歴ページの RSS フィードをサブスクライブしてください。

変更	説明	日付
AWSIoTSiteWiseMonitorPortalAccess - ポリシーを更新	AWS IoT SiteWise は、アラーム機能の AWSIoTSiteWiseMonitorPortalAccess マネージドポリシーを更新しました。	2021 年 5 月 27 日

変更	説明	日付
AWS IoT SiteWise が変更の追跡を開始しました	AWS IoT SiteWise がサービスロールの変更の追跡を開始しました。	2020 年 12 月 15 日

AWS IoT Events アラームのアクセス許可の設定

AWS IoT Events アラームモデルを使用して AWS IoT SiteWise アセットプロパティをモニタリングする場合、次の IAM アクセス許可が必要です。

- が AWS IoT Events にデータを送信できるようにする AWS IoT Events サービスロール AWS IoT SiteWise。詳細については、[AWS IoT Events Developer Guide] (デベロッパーガイド) の[Identity and access management for AWS IoT Events] (アイデンティティとアクセス管理) を参照してください。
- および の AWS IoT SiteWise アクション許可が必要です`iotsitewise:DescribeAssetModel`。`iotsitewise:UpdateAssetModelPropertyRouting`これらのアクセス許可により、 AWS IoT SiteWise はアセットプロパティ値を AWS IoT Events アラームモデルに送信できます。

詳細については、[IAM User Guide] (IAM ユーザーガイド) の[Resource-based policies] (リソースベースポリシー) を参照してください。

必要なアクセス許可 (API アクション)。

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどんなリソースにどんな条件でアクションを実行できるかということです。JSON ポリシーの`Action`要素には、ポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。

AWS IoT Events アラームモデルを定義する前に、 がアラームモデル AWS IoT SiteWise にアセットプロパティ値を送信できるようにする以下のアクセス許可を付与する必要があります。

- `iotsitewise:DescribeAssetModel` – AWS IoT Events アセットプロパティが存在するかどうかの によるチェックを許可します。

- `iotsitewise:UpdateAssetModelPropertyRouting` – が AWS IoT SiteWise にデータを送信 AWS IoT SiteWise できるようにするサブスクリプションを自動的に作成できるようにします AWS IoT Events。

AWS IoT SiteWise サポートされているアクションの詳細については、「サービス認証リファレンス」の「 で定義されるアクション AWS IoT SiteWise」を参照してください。

Example アクセス許可ポリシー 1 の例。

次のポリシーでは、 AWS IoT SiteWise がアセットプロパティ値を任意の AWS IoT Events アラームモデルに送信できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotevents:CreateAlarmModel",
                "iotevents:UpdateAlarmModel"
            ],
            "Resource": "arn:aws:iotevents:us-east-1:123456789012:alarmModel/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribeAssetModel",
                "iotsitewise:UpdateAssetModelPropertyRouting"
            ],
            "Resource": "arn:aws:iotsitewise:us-east-1:123456789012:asset-model/*"
        }
    ]
}
```

Example アクセス許可ポリシー 2 の例。

次のポリシーでは AWS IoT SiteWise 、 は指定されたアセットプロパティの値を指定された AWS IoT Events アラームモデルに送信できます。

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotevents:CreateAlarmModel",
                "iotevents:UpdateAlarmModel"
            ],
            "Resource": "arn:aws:iotevents:us-east-1:123456789012:alarmModel/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribeAssetModel"
            ],
            "Resource": "arn:aws:iotsitewise:us-east-1:123456789012:asset-model/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:UpdateAssetModelPropertyRouting"
            ],
            "Resource": [
                "arn:aws:iotsitewise:us-east-1:123456789012:asset-model/12345678-90ab-
cdef-1234-567890abcdef"
            ],
            "Condition": {
                "StringLike": {
                    "iotsitewise:propertyId": "abcdef12-3456-7890-abcd-ef1234567890",
                    "iotevents:alarmModelArn": "arn:aws:iotevents:us-
east-1:123456789012:alarmModel/MyAlarmModel"
                }
            }
        }
    ]
}
```

(オプション) ListInputRoutings アクセス許可

アセットモデルを更新または削除すると、 は、 の AWS IoT Events アラームモデルがこのアセットモデルに関連付けられたアセットプロパティをモニタリングしているかどうか AWS IoT SiteWise を確認できます。これにより、 AWS IoT Events アラームが現在使用しているアセットプロパティを削除できなくなります。でこの機能を有効にするには AWS IoT SiteWise、 アクセス`iotevents:ListInputRoutings`許可が必要です。このアクセス許可により AWS IoT

SiteWise 、 は でサポートされている ListInputRoutings API オペレーションを呼び出すことができます AWS IoT Events。

> **Note**
>
> `ListInputRoutings` 許可の追加を強く推奨します。

Example アクセス許可ポリシーの例。

次のポリシーでは、アセットモデルを更新および削除し、 で `ListInputRoutings` API を使用できます AWS IoT SiteWise。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:UpdateAssetModel",
                "iotsitewise:DeleteAssetModel",
                "iotevents:ListInputRoutings"
            ],
            "Resource": "arn:aws:iotsitewise:us-east-1:123456789012:asset-model/*"
        }
    ]
}
```

SiteWise Monitor に必要なアクセス許可

SiteWise Monitor ポータルでアラーム機能を使用する場合は、次のポリシーで Monitor SiteWise サービスロールを更新する必要があります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribePortal",
                "iotsitewise:CreateProject",
                "iotsitewise:DescribeProject",
```

```
                "iotsitewise:UpdateProject",
                "iotsitewise:DeleteProject",
                "iotsitewise:ListProjects",
                "iotsitewise:BatchAssociateProjectAssets",
                "iotsitewise:BatchDisassociateProjectAssets",
                "iotsitewise:ListProjectAssets",
                "iotsitewise:CreateDashboard",
                "iotsitewise:DescribeDashboard",
                "iotsitewise:UpdateDashboard",
                "iotsitewise:DeleteDashboard",
                "iotsitewise:ListDashboards",
                "iotsitewise:CreateAccessPolicy",
                "iotsitewise:DescribeAccessPolicy",
                "iotsitewise:UpdateAccessPolicy",
                "iotsitewise:DeleteAccessPolicy",
                "iotsitewise:ListAccessPolicies",
                "iotsitewise:DescribeAsset",
                "iotsitewise:ListAssets",
                "iotsitewise:ListAssociatedAssets",
                "iotsitewise:DescribeAssetProperty",
                "iotsitewise:GetAssetPropertyValue",
                "iotsitewise:GetAssetPropertyValueHistory",
                "iotsitewise:GetAssetPropertyAggregates",
                "iotsitewise:BatchPutAssetPropertyValue",
                "iotsitewise:ListAssetRelationships",
                "iotsitewise:DescribeAssetModel",
                "iotsitewise:ListAssetModels",
                "iotsitewise:UpdateAssetModel",
                "iotsitewise:UpdateAssetModelPropertyRouting",
                "sso-directory:DescribeUsers",
                "sso-directory:DescribeUser",
                "iotevents:DescribeAlarmModel",
                "iotevents:ListTagsForResource"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotevents:BatchAcknowledgeAlarm",
                "iotevents:BatchSnoozeAlarm",
                "iotevents:BatchEnableAlarm",
                "iotevents:BatchDisableAlarm"
            ],
```

```
            "Resource": "*",
            "Condition": {
                "Null": {
                    "iotevents:keyValue": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotevents:CreateAlarmModel",
                "iotevents:TagResource"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/iotsitewisemonitor": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iotevents:UpdateAlarmModel",
                "iotevents:DeleteAlarmModel"
            ],
            "Resource": "*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/iotsitewisemonitor": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "iotevents.amazonaws.com"
                    ]
```

```
                }
            }
        }
    ]
}
```

サービス間の混乱した代理の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。サービス間でのなりすましは、あるサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスが操作され、それ自身のアクセス許可を使用して、本来アクセス許可が付与されるべきではない方法で別の顧客のリソースに対して働きかけることがあります。これを防ぐため、 AWS では、アカウント内のリソースへのアクセス許可が付与されたサービスプリンシパルですべてのサービスのデータを保護するために役立つツールを提供しています。

リソースポリシーで aws:SourceArnおよび aws:SourceAccount グローバル条件コンテキストキーを使用して、 が別のサービスに AWS IoT SiteWise 付与するアクセス許可をリソースに制限することをお勧めします。aws:SourceArn 値が、Amazon S3 バケットの Amazon リソースネーム (ARN) などのアカウント ID が含まれていない場合、アクセス許可を制限するため、両方のグローバル条件コンテキストキーを使用する必要があります。同じポリシーステートメントでこれらのグローバル条件コンテキストキーの両方を使用し、アカウント ID にaws:SourceArn の値が含まれていない場合、aws:SourceAccount 値と aws:SourceArn 値の中のアカウントには、同じアカウント ID を使用する必要があります。

- クロスサービスのアクセスにリソースを 1 つだけ関連付けたい場合は、aws:SourceArn を使用します。
- そのアカウント内のリソースをクロスサービスの使用に関連付けることを許可する場合は、aws:SourceAccount を使用します。

の値は、sts:AssumeRoleリクエストに関連付けられている AWS IoT SiteWise カスタマーリソースaws:SourceArnである必要があります。

混乱した代理問題から保護するための最も効果的な方法は、リソースの完全な ARN を指定して aws:SourceArn グローバル条件コンテキストキーを使用することです。リソースの ARN 全体が不明または複数のリソースを指定する場合、ARN の未知部分にワイルドカー

ド *が付いた aws:SourceArn グローバルコンテキスト条件キー を使用します。例えば arn:aws:*servicename*:*:*123456789012*:* です。

Example – 混乱した代理の防止

次の例は、 で aws:SourceArnおよび aws:SourceAccount グローバル条件コンテキストキーを使用して、混乱した代理問題 AWS IoT SiteWise を回避する方法を示しています。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "iotsitewise.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Resource": [
      "arn:aws:iotsitewise:::ResourceName/*"
    ],
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:iotsitewise:*:123456789012:*"
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

AWS IoT SiteWise ID とアクセスのトラブルシューティング

次の情報は、 および AWS Identity and Access Management (IAM) の使用時に発生する可能性がある一般的な問題の診断 AWS IoT SiteWise と修正に役立ちます。

トピック

- でアクションを実行する権限がない AWS IoT SiteWise
- iam:PassRole を実行する権限がない
- 自分の 以外のユーザーに自分の AWS IoT SiteWise リソース AWS アカウント へのアクセスを許可したい

でアクションを実行する権限がない AWS IoT SiteWise

がアクションを実行する権限がないと AWS Management Console 通知した場合は、管理者に連絡してサポートを依頼する必要があります。担当の管理者はお客様のユーザー名とパスワードを発行した人です。

次のエラー例は、mateojackson IAM ユーザーがコンソールを使用してアセットの詳細を表示する際に、iotsitewise:DescribeAsset アクセス許可を持っていない場合に発生します。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 iotsitewise:DescribeAsset on resource: a1b2c3d4-5678-90ab-cdef-22222EXAMPLE
```

この場合、Mateo は管理者に依頼し、iotsitewise:DescribeAsset アクションを使用して ID が a1b2c3d4-5678-90ab-cdef-22222EXAMPLE であるアセットリソースにアクセスできるようにポリシーを更新してもらいます。

iam:PassRole を実行する権限がない

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して AWS IoT SiteWiseにロールを渡すことができるようにする必要があります。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

以下の例のエラーは、marymajor という IAM ユーザーがコンソールを使用して AWS IoT SiteWise でアクションを実行しようする場合に発生します。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。Mary には、ロールをサービスに渡す権限がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン資格情報を提供した担当者が管理者です。

自分の 以外のユーザーに自分の AWS IoT SiteWise リソース AWS アカウント へのアクセスを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACL) をサポートするサービスの場合、それらのポリシーを使用して、リソースへのアクセスを付与できます。

詳細については、以下を参照してください:

- がこれらの機能 AWS IoT SiteWise をサポートしているかどうかを確認するには、「」を参照してくださいが IAM と AWS IoT SiteWise 連携する方法。
- 所有 AWS アカウント している のリソースへのアクセスを提供する方法については、IAM ユーザーガイドの「所有 AWS アカウント している別の の IAM ユーザーへのアクセスを提供する」を参照してください。
- リソースへのアクセスをサードパーティー に提供する方法については AWS アカウント、IAM ユーザーガイドの「サードパーティー AWS アカウント が所有する へのアクセスを提供する」を参照してください。
- ID フェデレーションを介してアクセスを提供する方法については、『IAM ユーザーガイド』の「外部で認証されたユーザー (ID フェデレーション) へのアクセス権限」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いの詳細については、「IAM ユーザーガイド」の「IAM ロールとリソースベースのポリシーとの相違点」を参照してください。

のコンプライアンス検証 AWS IoT SiteWise

AWS IoT SiteWise は AWS コンプライアンスプログラムの対象ではありません。

特定のコンプライアンスプログラムの対象となる AWS サービスのリストについては、「コンプライアンスプログラムAWS による対象範囲内のサービスコンプライアンスプログラム」を参照してください。一般的な情報については、AWS 「コンプライアンスプログラム」を参照してください。

を使用して、サードパーティーの監査レポートをダウンロードできます AWS Artifact。詳細については、「 でのレポート AWS Artifactのダウンロード」の」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS IoT SiteWise は、お客様のデータの機密性、貴社のコンプライアンス目的、適用される法律および規制によって決まります。 AWS では、コンプライアンスに役立つ以下のリソースを提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド – これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスに重点を置いたベースライン環境を にデプロイする手順について説明します AWS。
- 「HIPAA セキュリティとコンプライアンスの設計」ホワイトペーパー — このホワイトペーパーでは、企業が AWS を使用して HIPAA 準拠のアプリケーションを作成する方法について説明します。
- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- 「 デベロッパーガイド」のルールによるリソースの評価 — この AWS Config サービスは、リソース設定が社内プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – この AWS サービスは、 内のセキュリティ状態を包括的に把握 AWS し、セキュリティ業界標準とベストプラクティスへの準拠を確認するのに役立ちます。
- [Ten security golden rules for Industrial IoT solutions] (産業用 IoT ソリューションにおける10のセキュリティ黄金律) - このブログ記事では、産業用制御システム (ICS) 、産業用 IoT (IIoT) 、およびクラウド環境のセキュリティに役立つ 10 の黄金律を紹介します。
- 製造 OT のセキュリティのベストプラクティス — このホワイトペーパーでは、 AWS クラウド用のこれらのオンプレミスハイブリッド製造ワークロードを設計、デプロイ、設計するためのセキュリティのベストプラクティスについて説明します。

の耐障害性 AWS IoT SiteWise

AWS グローバルインフラストラクチャは、 AWS リージョンとアベイラビリティーゾーンを中心に構築されています。 AWS リージョンは、低レイテンシー、高スループット、および高度に冗長なネットワークで接続された、物理的に分離および分離された複数のアベイラビリティーゾーンを提供します。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性が高く、フォールトトレラントで、スケーラブルです。

AWS IoT SiteWise はフルマネージド型で、Amazon S3 や Amazon EC2 などの可用性と耐久性の高い AWS サービスを使用しています。 Amazon EC2 アベイラビリティーゾーンが中断した場合に可用性を確保するために、 は複数のアベイラビリティーゾーンにわたって AWS IoT SiteWise 動作します。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS 「 グローバルインフラストラクチャ」を参照してください。

グローバル AWS インフラストラクチャに加えて、 AWS IoT SiteWise には、データの耐障害性とバックアップのニーズをサポートするのに役立ついくつかの機能が用意されています。

- MQTT メッセージ AWS IoT Core を介して にプロパティ値の更新を発行し、そのデータに基づいて動作するルールを設定できます。この機能を使用すると、Amazon S3 や Amazon DynamoDB などの他の AWS のサービスでデータをバックアップできます。詳細については、「AWS 他のサービスとのやり取り」および「アセットプロパティ通知を使用して Amazon S3 にデータをエクスポートする」を参照してください。
- APIs AWS IoT SiteWise `Get*`、過去のアセットプロパティデータを取得およびバックアップできます。詳細については、「履歴アセットプロパティ値をクエリする」を参照してください。
- APIs を使用して AWS IoT SiteWise `Describe*`、アセットやモデルなどのリソースの定義を取得できます。これらの定義をバックアップし、後でそれらを使用してリソースを再作成できます。詳細については、「AWS IoT SiteWise APIリファレンス」を参照してください。

のインフラストラクチャセキュリティ AWS IoT SiteWise

マネージドサービスである AWS IoT SiteWise は、 AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「 Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

が AWS 公開している API コールを使用して、ネットワーク AWS IoT SiteWise 経由で にアクセスします。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS)。TLS 1.2 は必須で TLS 1.3 がお勧めです。
- DHE (楕円ディフィー・ヘルマン鍵共有) や ECDHE (楕円曲線ディフィー・ヘルマン鍵共有) などの完全前方秘匿性 (PFS) による暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

SiteWise で実行されるエッジゲートウェイは AWS IoT Greengrass、X.509 証明書と暗号化キーを使用して AWS クラウドに接続し、認証します。詳細については、[AWS IoT Greengrass Version 1 Developer Guide] (デベロッパーガイド) の[Device authentication and authorization for AWS IoT Greengrass] (端末認証および認可) を参照してください。

設定と脆弱性の分析

IoT フリートは、多様な機能を持ち、存続期間が長く、地理的に分散される多数のデバイスで構成されることがあります。このような特性によってフリートのセットアップが複雑になり、エラーを起こしやすくなります。通常、デバイスの処理能力、メモリ、ストレージは限られているため、暗号化やその他のセキュリティ対策を常にサポートできるとは限りません。さらに、デバイスは多く場合、既知の脆弱性を持つソフトウェアを使用しています。このような要素が原因で、IoT フリートはハッカーの魅力的なターゲットとなり、デバイスフリートを継続的に保護することが困難になっています。

AWS IoT Device Defender セキュリティの問題やベストプラクティスからの逸脱を特定するためのツールを提供することで、これらの課題に対処します。 AWS IoT Device Defender を使用して、接続されたデバイスを分析、監査、モニタリングして異常な動作を検出し、セキュリティリスクを軽減します。 AWS IoT Device Defender は、デバイスフリートを監査して、セキュリティのベストプラクティスに準拠していることを確認して、デバイス上の異常な動作を検出できます。これにより、 AWS IoT デバイスフリート全体に一貫したセキュリティポリシーを適用し、デバイスが侵害されたときに迅速に対応できます。詳細については、「AWS IoT デベロッパーガイド」の 「AWS IoT Device Defender」を参照してください。

SiteWise Edge ゲートウェイを使用してサービスにデータを取り込む場合、 SiteWise Edge ゲートウェイの環境を設定および維持するのはユーザーの責任です。この責任には、 SiteWise エッジゲートウェイのシステムソフトウェア、 AWS IoT Greengrass ソフトウェア、コネクタの最新バージョンへのアップグレードが含まれます AWS IoT SiteWise 。詳細については、「 AWS IoT Greengrass Version 1 デベロッパーガイド」の AWS IoT Greengrass 「コアの設定」および「」を参照してくださいコネクタのアップグレード。

VPC エンドポイント

インターフェイス VPC エンドポイントは、Virtual Private Cloud (VPC) と の間のプライベート接続を確立します AWS IoT SiteWise。 はインターフェイスエンドポイントAWS PrivateLinkを強化し、API オペレーションへの AWS IoT SiteWise プライベートアクセスを可能にします。インターネットゲートウェイ、NAT デバイス、VPN 接続、または の必要性を回避できます AWS Direct Connect。VPC 内のインスタンスは、パブリック IP アドレスがなくても AWS IoT SiteWise API と

通信できます。VPC と 間のトラフィック AWS IoT SiteWise は、ネットワークを AWS 離れません。

各インターフェイスエンドポイントは、サブネット内の 1 つ以上の Elastic Network Interface によって表されます。

のインターフェイス VPC エンドポイントを設定する前に AWS IoT SiteWise、「Amazon VPC ユーザーガイド」の「インターフェイスエンドポイントのプロパティと制限」を確認してください。

詳細については、「Amazon VPC ユーザーガイド」の「インターフェイス VPC エンドポイント (AWS PrivateLink)」を参照してください。

VPC エンドポイントでサポートされている API オペレーション

AWS IoT SiteWise は、VPC から次の AWS IoT SiteWise API オペレーションの呼び出しをサポートします。

- すべてのデータプレーン API オペレーションで、次のエンドポイントを使用します。 を *region*に置き換えます AWS リージョン。

  ```
  data.iotsitewise.region.amazonaws.com
  ```

  データプレーン API には、以下のオペレーションが含まれます。

  - BatchGetAssetProperty値
  - BatchGetAssetPropertyValueHistory
  - BatchPutAssetProperty値
  - GetAssetPropertyAggregates
  - GetAssetPropertyValue
  - GetAssetPropertyValue履歴
  - GetInterpolatedAssetProperty値
- アセットモデル、アセット、 SiteWise エッジゲートウェイ、タグ、アカウント設定の管理に使用するコントロールプレーン API オペレーションには、次のエンドポイントを使用します。*region* の部分はお客様の AWS リージョンに置き換えてください。

  ```
  api.iotsitewise.region.amazonaws.com
  ```

  サポートされているコントロールプレーン API オペレーションには以下が含まれます。

- AssociateAssets
- CreateAsset
- CreateAssetモデル
- DeleteAsset
- DeleteAssetモデル
- DeleteDashboard
- DescribeAsset
- DescribeAssetモデル
- DescribeAssetプロパティ
- DescribeDashboard
- DescribeLoggingオプション
- DisassociateAssets
- ListAssetモデル
- ListAsset関係
- ListAssets
- ListAssociatedアセット
- PutLoggingオプション
- UpdateAsset
- UpdateAssetモデル
- UpdateAssetプロパティ
- CreateGateway
- DeleteGateway
- DescribeDefaultEncryptionConfiguration
- DescribeGateway
- DescribeGatewayCapabilityConfiguration
- DescribeStorage設定
- ListGateways
- ListTagsForResource
- UpdateGateway
- UpdateGatewayCapabilityConfiguration

- PutDefaultEncryptionConfiguration
- PutStorage設定
- TagResource
- UntagResource

> **Note**
>
> 現在、コントロールプレーン API オペレーションのインターフェイス VPC エンドポイントは、次の SiteWise Monitor API オペレーションの呼び出しをサポートしていません。
>
> - BatchAssociateProjectAssets
> - BatchDisassociateProjectAssets
> - CreateAccessポリシー
> - CreateDashboard
> - CreatePortal
> - CreateProject
> - DeleteAccessポリシー
> - DeletePortal
> - DeleteProject
> - DescribeAccessポリシー
> - DescribePortal
> - DescribeProject
> - ListAccessポリシー
> - ListDashboards
> - ListPortals
> - ListProjects
> - ListProjectアセット
> - UpdateAccessポリシー
> - UpdateDashboard
> - UpdatePortal
> - UpdateProject

AWS IoT SiteWiseのインターフェイス VPC エンドポイントの作成

AWS IoT SiteWise サービスの VPC エンドポイントを作成するには、Amazon VPC コンソールまたは AWS Command Line Interface () を使用しますAWS CLI。詳細については、 Amazon VPC ユーザーガイド のインターフェイスエンドポイントの作成を参照してください。

次のいずれかのサービス名を使用して AWS IoT SiteWise 、 の VPC エンドポイントを作成します。

- データプレーン API オペレーションには、次のサービス名を使用します。

```
com.amazonaws.region.iotsitewise.data
```

- コントロールプレーン API オペレーションには、次のサービス名を使用します。

```
com.amazonaws.region.iotsitewise.api
```

インターフェイス VPC エンドポイント AWS IoT SiteWise を介したアクセス

インターフェイスエンドポイントを作成すると、 との通信に使用できるエンドポイント固有の DNS ホスト名が生成されます AWS IoT SiteWise。プライベート DNS のオプションはデフォルトで有効になっています。詳細については、「Amazon VPCユーザーガイド」の「プライベートホストゾーンの使用」を参照してください。

エンドポイント のプライベート DNS を有効にすると、次のいずれかの VPC エンドポイントを介して に AWS IoT SiteWise API リクエストを行うことができます。

- データプレーン API オペレーションには、次のエンドポイントを使用します。*region* を に置き換えます AWS リージョン。

```
data.iotsitewise.region.amazonaws.com
```

- コントロールプレーン API オペレーションには、次のエンドポイントを使用します: *region* を に置き換えます AWS リージョン。

```
api.iotsitewise.region.amazonaws.com
```

エンドポイント のプライベート DNS を無効にする場合は、エンドポイント を介して にアクセスするには AWS IoT SiteWise 、次の操作を行う必要があります。

1. API リクエストで VPC エンドポイント URL を指定します。
   - データプレーン API オペレーションには、次のエンドポイント URL を使用します。*#vpc-endpoint-id#* と *#######* は、VPCエンドポイント ID とリージョンに置き換えてください。

     ```
     vpc-endpoint-id.data.iotsitewise.region.vpce.amazonaws.com
     ```

   - コントロールプレーン API オペレーションには、次のエンドポイント URL を使用します。*#vpc-endpoint-id#* と *#######* は、VPCエンドポイント ID とリージョンに置き換えてください。

     ```
     vpc-endpoint-id.api.iotsitewise.region.vpce.amazonaws.com
     ```

2. ホストプレフィックスインジェクションを無効にする。 AWS CLI および AWS SDKs、各 API オペレーションを呼び出すときに、サービスエンドポイントの前にさまざまなホストプレフィックスを付加します。この機能により、 AWS CLI VPC エンドポイントを指定する AWS IoT SiteWise と、 および AWS SDKs は に対して有効でない URLs を生成します。

   > **Important**
   >
   > AWS CLI または でホストプレフィックスインジェクションを無効にすることはできません AWS Tools for PowerShell。つまり、プライベート DNS を無効にすると、これらのツールを使用して VPC エンドポイント AWS IoT SiteWise 経由で にアクセスすることはできません。プライベート DNS を有効にして、 AWS CLI または を使用してエンドポイント AWS IoT SiteWise 経由で AWS Tools for PowerShell にアクセスします。

   AWS SDKsの以下のドキュメントセクションを参照してください。

   - AWS SDK for C++
   - AWS SDK for Go
   - AWS SDK for Go v2
   - AWS SDK for Java
   - AWS SDK for Java 2.x
   - AWS SDK for JavaScript

- AWS SDK for .NET
- AWS SDK for PHP
- AWS SDK for Python (Boto3)
- AWS SDK for Ruby

詳細については、「Amazon VPC ユーザーガイド」の「インターフェイスエンドポイントを介したサービスへのアクセス」を参照してください。

の VPC エンドポイントポリシーの作成 AWS IoT SiteWise

VPC エンドポイントには、 AWS IoT SiteWiseへのアクセスを制御するエンドポイントポリシーをアタッチできます。このポリシーでは、以下の情報を指定します。

- オペレーションを実行できるプリンシパル。
- 実行できる操作。
- この操作を実行できるリソース。

詳細については、「Amazon VPC ユーザーガイド」の「VPC エンドポイントによるサービスのアクセスコントロール」を参照してください。

例: AWS IoT SiteWise アクションの VPC エンドポイントポリシー

のエンドポイントポリシーの例を次に示します AWS IoT SiteWise。エンドポイントにアタッチされると、このポリシーは、指定されたアセットに対する *123456789012* *iotsitewiseadmin*に AWS アカウント リストされた AWS IoT SiteWise アクションへのアクセスをユーザーに付与します。

```
{
    "Statement": [
        {
            "Action": [
                "iotsitewise:CreateAsset",
                "iotsitewise:ListGateways",
                "iotsitewise:ListTagsForResource"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:iotsitewise:us-west-2:123456789012:asset/
a1b2c3d4-5678-90ab-cdef-33333EXAMPLE",
            "Principal": {
                "AWS": [
```

```
                    "123456789012:user/iotsitewiseadmin"
                ]
            }
        }
    ]
}
```

のセキュリティのベストプラクティス AWS IoT SiteWise

このトピックには、 のセキュリティのベストプラクティスが含まれています AWS IoT SiteWise。

OPC-UA サーバーでの認証情報の使用

OPC-UA サーバーへの接続に認証情報を要求するようにします。これを行うには、サーバーのドキュメントを参照してください。次に、 SiteWise Edge ゲートウェイが OPC-UA サーバーに接続できるようにするには、 SiteWise Edge ゲートウェイにサーバー認証シークレットを追加します。詳細については、「ソース認証の設定」を参照してください。

OPC-UA サーバーへの暗号化通信モードの使用

SiteWise Edge ゲートウェイの OPC-UA ソースを設定するときは、非推奨で暗号化されたメッセージセキュリティモードを選択します。これにより、OPC-UA サーバーから SiteWise Edge ゲートウェイに移動するときに、産業用データが保護されます。詳細については、「ローカルネットワーク経由で転送されるデータ」および「データソースの設定」を参照してください。

コンポーネントを最新の状態に保つ。

SiteWise Edge ゲートウェイを使用してサービスにデータを取り込む場合、 SiteWise Edge ゲートウェイの環境を設定および維持するのはユーザーの責任です。これには、ゲートウェイのシステムソフトウェア、 AWS IoT Greengrass ソフトウェアとコネクタの最新バージョンへのアップグレードが含まれます。

> **Note**
>
> AWS IoT SiteWise Edge コネクタは、シークレットをファイルシステムに保存します。これらのシークレットは、 SiteWise エッジゲートウェイ内にキャッシュされたデータを表示できるユーザーを制御します。 SiteWise Edge ゲートウェイを実行しているシステムのディスクまたはファイルシステムの暗号化を有効にすることを強くお勧めします。

SiteWise Edge ゲートウェイのファイルシステムを暗号化する

SiteWise Edge ゲートウェイを暗号化して保護するため、産業データは Edge SiteWise ゲートウェイ内を移動する際に安全です。 SiteWise Edge ゲートウェイにハードウェアセキュリティモジュールがある場合は、 SiteWise エッジゲートウェイを保護する AWS IoT Greengrass ように を設定できます。詳細については、[AWS IoT Greengrass Version 1 Developer Guide] (デベロッパーガイド) の[Hardware security integration] (ハードウェアセキュリティの統合) を参照してください。ハードウェアセキュリティモジュールがない場合は、オペレーティングシステムのドキュメントを参照して、ファイルシステムを暗号化して保護する方法を確認してください。

エッジ設定へのセキュアなアクセス。

エッジコンソールアプリケーションパスワードや SiteWise Monitor アプリケーションパスワードは共有しないでください。このパスワードは、誰でも見られる場所に置かないでください。パスワードに適切な有効期限を設定することで、健全なパスワードローテーションポリシーを実施する。

最小限のアクセス許可を SiteWise Monitor ユーザーに付与する

最小アクセス許可の原則に従って、ポータルのユーザーのアクセスポリシーのアクセス許可には最小限のものを使用します。

- ポータルを作成する場合は、そのポータルに必要な最小限のアセットを許可するロールを定義します。詳細については、「のサービスロールの使用 AWS IoT SiteWise Monitor」を参照してください。
- ポータル管理者がプロジェクトを作成して共有する場合は、そのプロジェクトに必要な最小限のアセットを使用します。
- ポータルやプロジェクトにアクセスする必要がなくなった ID は、そのリソースから削除する。その ID が組織に適用されなくなった場合は、ID ストアからその ID を削除してください。

最小アクセス許可の原則のベストプラクティスは、IAM ロールにも該当します。詳細については、「ポリシーのベストプラクティス」を参照してください。

機密情報を公開しない

認証情報やその他の機密情報 (個人を特定できる情報 (PII) など) のログへの記録を防止する必要があります。 SiteWise Edge ゲートウェイのローカルログへのアクセスにはルート権限が必要で、ログへのアクセスには CloudWatch IAM 権限が必要であっても、次の保護を実装することをお勧めします。

- アセットやモデルの名前、説明、プロパティに機密情報を使用しない。
- SiteWise Edge ゲートウェイまたはソース名で機密情報を使用しないでください。
- ポータル、プロジェクト、ダッシュボードの名前や説明に機密情報を使用しない。

AWS IoT Greengrass セキュリティのベストプラクティスに従う

SiteWise Edge ゲートウェイ AWS IoT Greengrass のセキュリティのベストプラクティスに従います。詳細については、[AWS IoT Greengrass Version 1 Developer Guide] (デベロッパーガイド) の [Security best practices] (セキュリティベストプラクティス) を参照してください。

以下も参照してください。

- [AWS IoT Developer Guide] (デベロッパーガイド) の [Security best practices] (セキュリティベストプラクティス)。
- 産業用 IoT ソリューションの 10 のセキュリティゴールデンルール

ログインとモニタリング AWS IoT SiteWise

監視は、 AWS その他のソリューションの信頼性、可用性、パフォーマンスを維持する上で重要な部分です。 AWS IoT SiteWise AWS IoT SiteWise サービスを監視し、問題が発生した場合は報告し、必要に応じて自動的にアクションを実行するための以下の監視ツールをサポートしています。

- Amazon は、 CloudWatch AWS AWS お客様のリソースとお客様が実行するアプリケーションをリアルタイムで監視します。メトリックスを収集して追跡し、カスタマイズされたダッシュボードを作成し、指定したメトリックスが特定のしきい値に達したときに通知またはアクションを実行するアラームを設定します。たとえば、Amazon EC2 インスタンスの CPU CloudWatch 使用率やその他のメトリックスを追跡し、必要に応じて新しいインスタンスを自動的に起動できます。詳細については、Amazon CloudWatch ユーザーガイドを参照してください。
- Amazon CloudWatch Logs は、 SiteWise Edge ゲートウェイやその他のソースからのログファイルを監視、保存 CloudTrail、アクセスします。 CloudWatch ログはログファイル内の情報を監視し、特定のしきい値に達すると通知します。高い耐久性を備えたストレージにログデータをアーカイブすることもできます。詳細については、Amazon CloudWatch Logs ユーザーガイドを参照してください。
- AWS CloudTrailアカウントによって、 AWS またはアカウントに代わって行われた API 呼び出しと関連イベントをキャプチャします。次に、指定した Amazon S3 CloudTrail バケットにログファイルを配信します。どのユーザーとアカウント AWS、呼び出しが行われたソース IP アドレス、呼び出しがいつ発生したかを特定できます。詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

トピック

Amazon CloudWatch ログによるモニタリング

AWS IoT SiteWise 情報をログに記録して、 CloudWatch サービスの監視とトラブルシューティングを行うように設定します。

AWS IoT SiteWise コンソールを使用すると、 AWS IoT SiteWise サービスがユーザーに代わって情報を記録できるようにするサービスにリンクされたロールが作成されます。 AWS IoT SiteWise コンソールを使用しない場合は、ログを受信するためにサービスにリンクされたロールを手動で作成する必要があります。詳細については、「AWS IoT SiteWiseのサービスリンクロールの作成」を参照してください。

AWS IoT SiteWise ログイベントをストリームに入れることを許可するリソースポリシーが必要です。 CloudWatch CloudWatch Logs のリソースポリシーを作成および更新するには、以下のコマンドを実行します。*logging-policy-name*作成するポリシーの名前に置き換えます。

```
aws logs put-resource-policy --policy-name logging-policy-name --policy-
document "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Sid\":
 \"IoTSiteWiseToCloudWatchLogs\", \"Effect\": \"Allow\", \"Principal\": { \"Service\":
 [ \"iotsitewise.amazonaws.com\" ] }, \"Action\":\"logs:PutLogEvents\", \"Resource\":
 \"*\" } ] }"
```

CloudWatch Logs は aws: SourceArn と aws: SourceAccount 条件コンテキストキーもサポートします。これらの条件コンテクストキーはオプションです。

AWS IoT SiteWise AWS IoT SiteWise CloudWatch 指定したリソースに関連するログのみをストリームに入れることを許可するリソースポリシーを作成または更新するには、コマンドを実行して次の操作を行います。

- *logging-policy-name*作成するポリシーの名前に置き換えます。
- *source-ARN* は、 AWS IoT SiteWise アセットモデルやアセットなどのリソースの ARN に置き換えます。 AWS IoT SiteWise 各リソースタイプの ARN を確認するには、『サービス認証リファレンス』の「AWS IoT SiteWiseで定義されるリソースタイプ」を参照してください。
- *account-ID* は、 AWS 指定したリソースに関連付けられたアカウント ID に置き換えます。 AWS IoT SiteWise

```
aws logs put-resource-policy --policy-name logging-policy-name --policy-
document "{ \"Version\": \"2012-10-17\", \"Statement\": [ { \"Sid\":
 \"IoTSiteWiseToCloudWatchLogs\", \"Effect\": \"Allow\", \"Principal\": { \"Service
\": [ \"iotsitewise.amazonaws.com\" ] }, \"Action\":\"logs:PutLogEvents\", \"Resource
\": \"*\", \"Condition\":{\"StringLike\":{\"aws:SourceArn\":[\"source-ARN\"],
\"aws:SourceAccount\":[\"account-ID\"]}}}]}"
```

デフォルトでは、 AWS IoT SiteWise Logs CloudWatch に情報を記録しません。ロギングを有効にするには、Disabled (OFF) 以外のロギングレベルを選択します。 AWS IoT SiteWise は次のロギングレベルをサポートします。

- OFF – ログ記録はオフになります。
- ERROR - エラーがログ記録されます。
- INFO - エラーおよび情報メッセージがログ記録されます。

SiteWise Edge ゲートウェイは、 CloudWatch Logs through AWS IoT Greengrassに情報を記録するように設定できます。詳細については、「 SiteWise Edge ゲートウェイログの監視」を参照してください。

AWS IoT SiteWise ルールアクションのトラブルシューティングを行う場合は、 AWS IoT Core CloudWatch 情報をログに記録するように設定することもできます。詳細については、「AWS IoT SiteWise ルールアクションのトラブルシューティング」を参照してください。

目次

ログインの管理 AWS IoT SiteWise

AWS IoT SiteWise コンソールを使用するか AWS CLI 、以下のロギング設定タスクを行います。

ロギングレベルを調べる

Console

AWS IoT SiteWise コンソールで現在のログ記録レベルを確認するには、次の手順に従います。

AWS IoT SiteWise 現在のロギングレベルを確認するには

1. AWS IoT SiteWise コンソールに移動します。
2. 左側のナビゲーションペインで、[ログ記録オプション] を選択します。

[ロギングステータス] に、現在のログ記録ステータスが表示されます。ログ記録が有効な場合は、現在のログ記録レベルが [詳細レベル] の下に表示されます。

AWS CLI

次のコマンドを実行して、 AWS IoT SiteWise の現在のロギングレベルを調べます AWS CLI。

```
aws iotsitewise describe-logging-options
```

このオペレーションは、ログ記録レベルを次の形式で返します。

```
{
  "loggingOptions": {
    "level": "String"
  }
}
```

ロギングレベルの変更

次の手順に従って、 AWS IoT SiteWise コンソールまたはを使用してロギングレベルを変更します AWS CLI。

Console

AWS IoT SiteWise ロギングレベルを変更するには

1. AWS IoT SiteWise コンソールに移動します。
2. 左側のナビゲーションペインで、[Logging options (ログ記録オプション)] を選択します。
3. [編集] を選択します。
4. 有効にする [詳細レベル] を選択します。
5. [保存] を選択します。

AWS CLI

AWS CLI AWS IoT SiteWise 以下のコマンドを実行してロギングレベルを変更します。*logging-level* は、必要なログ記録レベルに置き換えます。

```
aws iotsitewise put-logging-options --logging-options level=logging-level
```

例: AWS IoT SiteWise ログファイルエントリ

AWS IoT SiteWise 各ログエントリにはイベント情報とそのイベントに関連するリソースが含まれているため、ログデータを理解して分析できます。

次の例は、 CloudWatch AWS IoT SiteWise アセットモデルの作成に成功したときに記録するログエントリを示しています。

```
{
  "eventTime": "2020-05-05T00:10:22.902Z",
  "logLevel": "INFO",
  "eventType": "AssetModelCreationSuccess",
  "message": "Successfully created asset model.",
  "resources": {
    "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE"
  }
}
```

SiteWise Edge ゲートウェイログの監視

Amazon CloudWatch Logs またはローカルファイルシステムに情報を記録するように AWS IoT SiteWise Edge ゲートウェイを設定できます。

トピック

- Amazon CloudWatch ログを使用する
- サービスログを使用する
- イベントログを使用する

Amazon CloudWatch ログを使用する

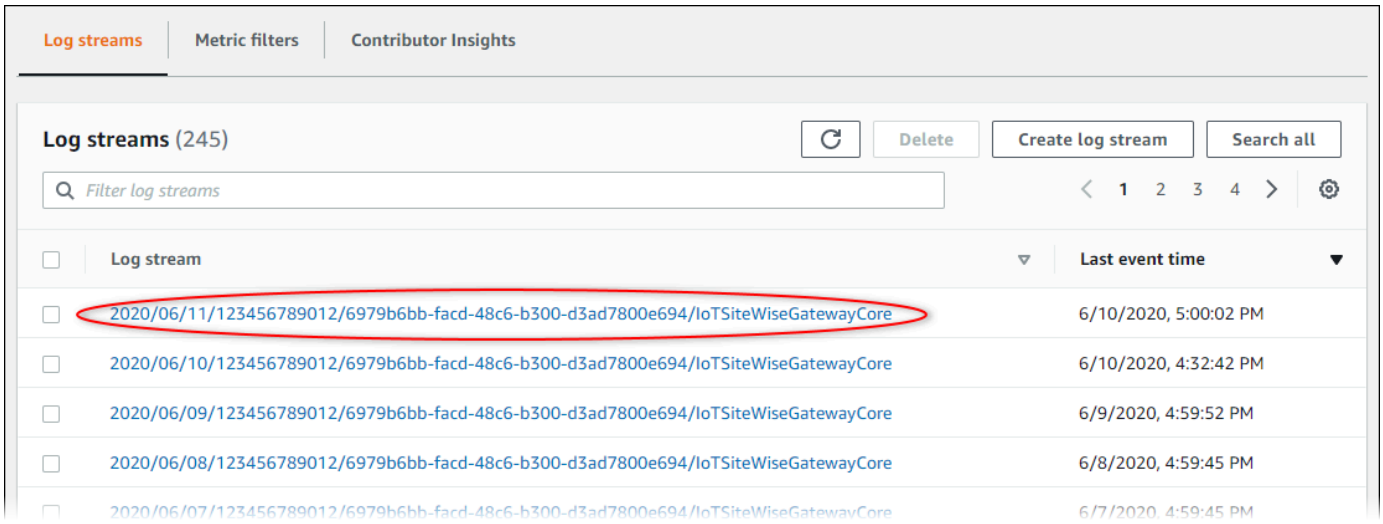
ログにログを送信するように SiteWise Edge ゲートウェイを設定できます。 CloudWatch 詳細については、『AWS IoT Greengrass Version 2 開発者ガイド』の「 CloudWatch ログのロギングを有効にする」を参照してください。

CloudWatch ログを設定してアクセスするには (コンソール)

1. CloudWatch コンソールに移動します。
2. ナビゲーションペインで、[ロググループ] を選択します。
3. AWS IoT SiteWise コンポーネントログは次のロググループにあります。

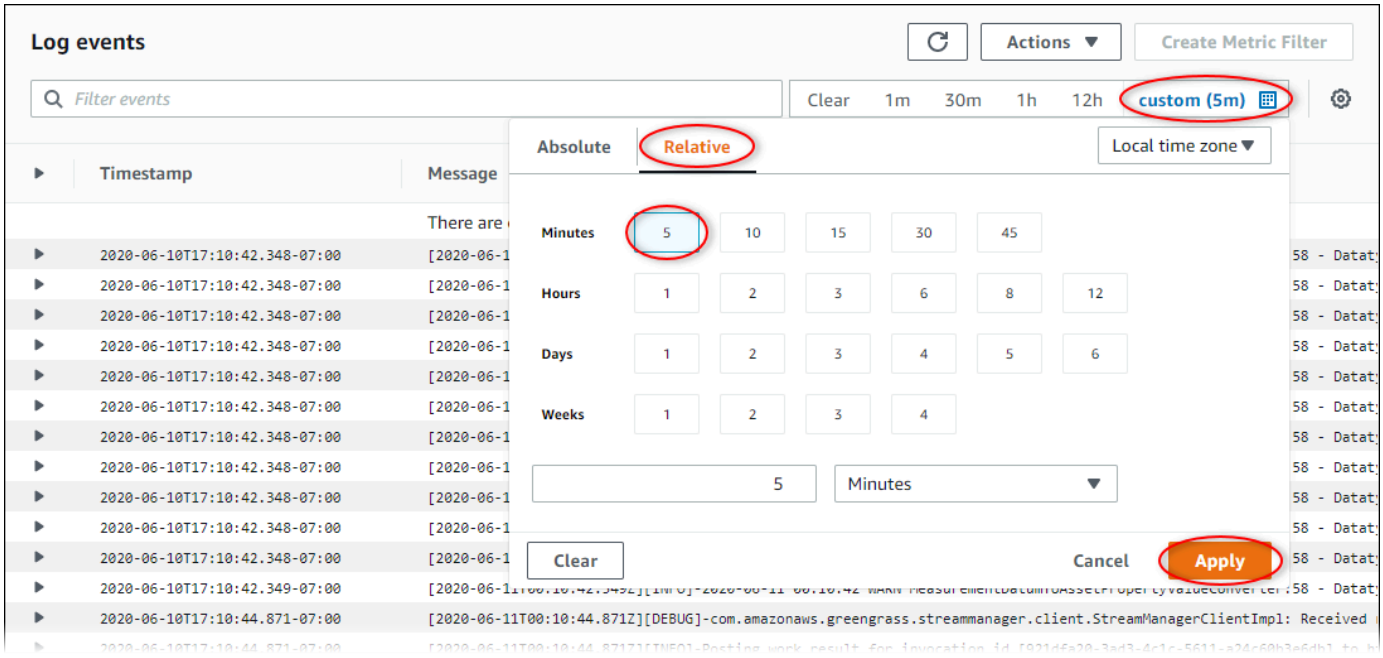
   - `/aws/greengrass/UserComponent/region/aws.iot.SiteWiseEdgeCollectorOpcua`— SiteWise Edge ゲートウェイの OPC-UA ソースからデータを収集する SiteWise Edge ゲートウェイのコンポーネントのログ。
   - `/aws/greengrass/UserComponent/region/aws.iot.SiteWiseEdgePublisher`— OPC-UA データストリームをに公開する SiteWise Edge ゲートウェイのコンポーネントのログ。 AWS IoT SiteWise

   デバッグする関数のロググループを選択します。
4. グループの名前で終わる名前のログストリームを選択してください。 AWS IoT Greengrass デフォルトでは、 CloudWatch 最新のログストリームが最初に表示されます。



5. 過去 5 分間のログを表示するには、以下を実行します。

   a. 右上隅の [カスタム] を選択します。
   b. [相対値] を選択します。
   c. [5 分] を選択します。
   d. [適用] を選択します。



6. (オプション) 表示するログの数を減らすには、右上隅で [1 分] を選択します。
7. ログエントリの一番下までスクロールして、最新のログを表示します。

サービスログを使用する

SiteWise Edge ゲートウェイデバイスには、問題のデバッグに役立つサービスログファイルが含まれています。以下のセクションは、 AWS IoT SiteWise OPC-UA Collector および AWS IoT SiteWise Publisher コンポーネントのサービスログファイルを見つけて利用するのに役立ちます。

AWS IoT SiteWise OPC-UA コレクターサービスログファイル

AWS IoT SiteWise OPC-UA コレクターコンポーネントは以下のログファイルを使用します。

Linux

```
/greengrass/v2/logs/aws.iot.SiteWiseEdgeCollectorOpcua.log
```

Windows

```
C:\greengrass\v2\logs\aws.iot.SiteWiseEdgeCollectorOpcua.log
```

このコンポーネントのログを確認するには

- コアデバイスに次のコマンドを実行して、このコンポーネントのログファイルをリアルタイムに確認します。*/greengrass/v2*or *C:\greengrass\v2* AWS IoT Greengrass をルートフォルダーへのパスに置き換えます。

  Linux

  ```
  sudo tail -f /greengrass/v2/logs/aws.iot.SiteWiseEdgeCollectorOpcua.log
  ```

  Windows (PowerShell)

  ```
  Get-Content C:\greengrass\v2\logs\aws.iot.SiteWiseEdgeCollectorOpcua.log -Tail
   10 -Wait
  ```

AWS IoT SiteWise Publisher サービスのログファイル

AWS IoT SiteWise Publisher コンポーネントは次のログファイルを使用します。

Linux

```
/greengrass/v2/logs/aws.iot.SiteWiseEdgePublisher.log
```

Windows

```
C:\greengrass\v2\logs\aws.iot.SiteWiseEdgePublisher.log
```

このコンポーネントのログを確認するには

- コアデバイスに次のコマンドを実行して、このコンポーネントのログファイルをリアルタイムに確認します。*/greengrass/v2*or *C:\greengrass\v2* AWS IoT Greengrass をルートフォルダーへのパスに置き換えます。

  Linux

  ```
  sudo tail -f /greengrass/v2/logs/aws.iot.SiteWiseEdgePublisher.log
  ```

Windows (PowerShell)

```
Get-Content C:\greengrass\v2\logs\aws.iot.SiteWiseEdgePublisher.log -Tail 10 -
Wait
```

イベントログを使用する

SiteWise Edge ゲートウェイデバイスには、問題のデバッグに役立つイベントログファイルが含まれています。以下のセクションは、 AWS IoT SiteWise OPC-UA Collector および AWS IoT SiteWise Publisher コンポーネントのイベントログファイルを見つけて利用するのに役立ちます。

AWS IoT SiteWise OPC-UA コレクターのイベントログ

AWS IoT SiteWise OPC-UA Collector コンポーネントには、お客様が問題を特定して修正するのに役立つイベントログが含まれています。ログファイルはローカルログファイルとは別のもので、次の場所にあります。*/greengrass/v2*または *C:\greengrass\v2* AWS IoT Greengrass をルートフォルダーへのパスに置き換えます。

Linux

```
/greengrass/v2/work/aws.iot.SiteWiseEdgeCollectorOpcua/logs/
IotSiteWiseOpcUaCollectorEvents.log
```

Windows

```
C:\greengrass\v2\work\aws.iot.SiteWiseEdgeCollectorOpcua\logs
\IotSiteWiseOpcUaCollectorEvents.log
```

このログには、詳細情報とトラブルシューティングの手順が含まれています。トラブルシューティングに関する情報は診断結果とともに提供され、問題の解決方法の説明と、場合によっては詳細情報へのリンクが含まれます。診断情報には次のものが含まれます。

- 重要度レベル
- タイムスタンプ
- イベント固有の追加情報

Example ログの例

```
dataSourceConnectionSuccess:
  Summary: Successfully connected to OpcUa server
  Level: INFO
  Timestamp: '2023-06-15T21:04:16.303Z'
  Description: Successfully connected to the data source.
  AssociatedMetrics:
  - Name: FetchedDataStreams
    Description: The number of fetched data streams for this data source
    Value: 1.0
    Namespace: IoTSiteWise
    Dimensions:
    - Name: SourceName
      Value: SourceName{value=OPC-UA Server}
    - Name: ThingName
      Value: test-core
  AssociatedData:
  - Name: DataSourceTrace
    Description: Name of the data source
    Data:
    - OPC-UA Server
  - Name: EndpointUri
    Description: The endpoint to which the connection was attempted.
    Data:
    - '"opc.tcp://10.0.0.1:1234"'
```

AWS IoT SiteWise パブリッシャーイベントログ

AWS IoT SiteWise Publisher コンポーネントには、顧客が問題を特定して修正するのに役立つイベントログが含まれています。ログファイルはローカルログファイルとは別のもので、次の場所にあります。*/greengrass/v2*or *C:\greengrass\v2* AWS IoT Greengrass をルートフォルダーへのパスに置き換えます。

Linux

```
/greengrass/v2/work/aws.iot.SiteWiseEdgePublisher/logs/
IotSiteWisePublisherEvents.log
```

Windows

```
C:\greengrass\v2\work\aws.iot.SiteWiseEdgePublisher\logs
\IotSiteWisePublisherEvents.log
```

このログには、詳細情報とトラブルシューティングの手順が含まれています。トラブルシューティングに関する情報は診断結果とともに提供され、問題の解決方法の説明と、場合によっては詳細情報へのリンクが含まれます。診断情報には次のものが含まれます。

- 重要度レベル
- タイムスタンプ
- イベント固有の追加情報

Example ログの例

```
accountBeingThrottled:
  Summary: Data upload speed slowed due to quota limits
  Level: WARN
  Timestamp: '2023-06-09T21:30:24.654Z'
  Description: The IoT SiteWise Publisher is limited to the "Rate of data points
 ingested"
    quota for a customers account. See the associated documentation and associated
    metric for the number of requests that were limited for more information. Note
    that this may be temporary and not require any change, although if the issue
 continues
    you may need to request an increase for the mentioned quota.
  FurtherInformation:
  - https://docs.aws.amazon.com/iot-sitewise/latest/userguide/quotas.html
  - https://docs.aws.amazon.com/iot-sitewise/latest/userguide/troubleshooting-
gateway.html#gateway-issue-data-streams
  AssociatedMetrics:
  - Name: TotalErrorCount
    Description: The total number of errors of this type that occurred.
    Value: 327724.0
  AssociatedData:
  - Name: AggregatePropertyAliases
    Description: The aggregated property aliases of the throttled data.
    FileLocation: /greengrass/v2/work/aws.iot.SiteWiseEdgePublisher/./logs/data/
AggregatePropertyAliases_1686346224654.log
```

Amazon CloudWatch メトリクス AWS IoT SiteWise によるモニタリング

AWS IoT SiteWise を使用してモニタリングできます。これにより CloudWatch、raw データを収集し、読み取り可能なほぼリアルタイムのメトリクスに加工できます。これらの統計は 15 か月間保持されるため、履歴情報にアクセスし、ウェブアプリケーションまたはサービスの動作をより的確に把握できます。また、特定のしきい値を監視するアラームを設定し、これらのしきい値に達したときに通知を送信したりアクションを実行したりできます。詳細については、「Amazon ユーザーガイド CloudWatch 」を参照してください。

AWS IoT SiteWise は、以下のセクションにリストされているメトリクスとディメンションを `AWS/IoTSiteWise`名前空間に発行します。

> **Tip**
>
> AWS IoT SiteWise は 1 分間隔でメトリクスを発行します。 CloudWatch コンソールでこれらのメトリクスをグラフで表示する場合は、期間 を 1 分にすることをお勧めします。これにより、メトリクスデータが最も高い解像度で表示されます。

トピック

AWS IoT Greengrass Version 2 ゲートウェイメトリクス

AWS IoT SiteWise は、次の SiteWise Edge ゲートウェイメトリクスを発行します。すべての SiteWise Edge ゲートウェイメトリクスは 1 分間隔で発行されます。

SiteWise エッジゲートウェイのメトリクス

メトリクス	説明
`Gateway.CpuUsage`	SiteWise Edge ゲートウェイの CPU 使用率。 単位: パーセント ディメンション: なし

メトリクス	説明
Gateway.TotalDiskSpace	SiteWise Edge ゲートウェイの合計ディスク容量。 単位: バイト ディメンション: なし
Gateway.UsedDiskSpace	SiteWise Edge ゲートウェイの使用済みディスク容量。 単位: バイト ディメンション: なし
Gateway.AvailableDiskSpace	SiteWise Edge ゲートウェイの使用可能なディスク容量。 単位: バイト ディメンション: なし
Gateway.UsedPercentageDiskSpace	SiteWise Edge ゲートウェイの使用済みディスク容量の割合。 単位: バイト ディメンション: なし
Gateway.TotalMemory	SiteWise Edge ゲートウェイの合計メモリ。 単位: バイト ディメンション: なし
Gateway.UsedMemory	SiteWise Edge ゲートウェイの使用済みメモリ。 単位: バイト ディメンション: なし

メトリクス	説明
`Gateway.AvailableMemory`	SiteWise Edge ゲートウェイの使用可能なメモリ。 単位: バイト ディメンション: なし
`Gateway.UsedPercentageMemory`	SiteWise Edge ゲートウェイの使用済みメモリの割合。 単位: バイト ディメンション: なし
`Gateway.CloudConnectivity`	SiteWise Edge ゲートウェイのクラウド接続ステータス。 単位: なし ディメンション： GatewayId
`Gateway.SWE.Component.RunningStatus`	SiteWise Edge ゲートウェイ上のコンポーネントの実行中のステータス。 単位: なし ディメンション： GatewayId

OPC-UA コレクターメトリクス

メトリクス	説明
`OpcUaCollector.Heartbeat`	Edge ゲートウェイ () に接続された各 OPC-UA ソース (`sourceName`) に対して 1 SiteWise 分ごとに生成されます`gatewayId` 。

メトリクス	説明
	単位: カウント (1 はソースが接続されていることを表し、0 はソースが切断されていることを表します) 。 ディメンション： GatewayId， SourceName
OpcUaCollector.ActiveDataStreamCount	Edge SiteWise ゲートウェイ (gatewayId) が OPC-UA ソース () 用にサブスクライブしたデータストリームの数sourceName 。 単位: 個 ディメンション： GatewayId， SourceName， PropertyGroup
OpcUaCollector.IncomingValuesCount	SiteWise Edge ゲートウェイ (gatewayId) が OPC-UA ソース (sourceName) に対して受信したデータポイントの数。1 分ごとに生成されます。 単位: 個 ディメンション： GatewayId， SourceName， PropertyGroup
OpcUaCollector.IncomingValuesError	SiteWise Edge ゲートウェイ (gatewayId) が OPC-UA ソース (sourceName) から受け取る有効な値ではないデータポイントの数。これらのデータポイントはコレクターによって取り込まれず OpcUa 、毎分生成されます。 単位: 個 ディメンション： GatewayId， SourceName， PropertyGroup

メトリクス	説明
OpcUaCollector.ConversionErrors	SiteWise Edge ゲートウェイ (gatewayId) が OPC-UA ソース (sourceName) に対して受信し、データを に送信する際に変換エラーが発生したデータポイントの数 AWS IoT SiteWise。これらのデータポイントは OpcUa コレクターによって取り込まれません。 単位: 個 ディメンション： GatewayId， SourceName

AWS IoT SiteWise プロセッサメトリクス

メトリクス	説明
Gateway.DataProcessor.IngestionSuccess	1 分ごとに正常に取り込まれたデータポイントの数。 単位: 個 ディメンション: なし
Gateway.DataProcessor.IngestionThrottled	スロットリングされ、1 分ごとに生成されたデータポイントの数。 単位: 個 ディメンション： ThrottledAt
Gateway.DataProcessor.MeasurementRejected	1 分ごとに生成された、拒否された測定値の数。 単位: 個 ディメンション: 理由
Gateway.DataProcessor.MeasurementUnmodeled	1 分ごとに生成された、モデル化されていない測定の数。

メトリクス	説明
	単位: 個 ディメンション: 理由
Gateway.DataProcessor.MessagesRemaining	1 分ごとに生成される、ストリームに残っているメッセージの数。 単位: 個 ディメンション： StreamName
Gateway.DataProcessor.ProcessingError	1 分ごとに生成される処理エラーの数。 単位: 個 ディメンション: 理由
IoTSiteWiseProcessor.IsConnectedToMqttBroker	Edge ゲートウェイのプロセッサによって 1 SiteWise 分ごとに生成されます。 単位: 1 (プロセッサが MQTT ブローカーに接続されていることを示す 1) ディメンション： GatewayId
IoTSiteWiseProcessor.NumberOfSubscriptionsToMqttBroker	プロセッサによって MQTT ブローカーにサブスクライブされ、1 分ごとに生成されるトピックの数。マルチレベルワイルドカードトピックは 1 としてカウントされます。 単位: 個 ディメンション： GatewayId

メトリクス	説明
IoTSiteWiseProcessor.NumberOfUniqueMqttTopicsReceived	プロセッサが MQTT ブローカーから受信した一意のトピックの数。1 分ごとに生成されます。 単位: 個 ディメンション： GatewayId
IoTSiteWiseProcessor.MqttMessageReceivedSuccessCount	プロセッサが MQTT ブローカーから正常に受信し、1 分ごとに生成されるメッセージの数。 単位: 個 ディメンション： GatewayId
IoTSiteWiseProcessor.MqttReceivedSuccessBytes	プロセッサが MQTT ブローカーから正常に受信したメッセージデータのバイト数。1 分ごとに生成されます。 単位: 個 ディメンション： GatewayId

AWS IoT SiteWise パブリッシャーメトリクス

メトリクス	説明
IoTSiteWisePublisher.Heartbeat	Edge ゲートウェイのパブリッシャーによって 1 SiteWise 分ごとに生成されます。 単位: 1 (パブリッシャーが実行中であり、パブリッシャーが実行中でないことを示すデータポイントがないことを示す 1) ディメンション： GatewayId

メトリクス	説明
IoTSiteWisePublisher.PublishSuccessCount	SiteWise Edge ゲートウェイ (GatewayId) がクラウドに正常に発行し、1 分ごとに生成されたデータポイントの数。 単位: 個 ディメンション： GatewayId
IoTSiteWisePublisher.PublishFailureCount	SiteWise Edge ゲートウェイ (GatewayId) がパブリッシュに失敗したデータポイントの数。1 分ごとに生成されます。 単位: 個 ディメンション： GatewayId
IoTSiteWisePublisher.PublishRejectedCount	SiteWise Edge ゲートウェイ (GatewayId) がクラウド側から拒否したデータポイントの数。1 分ごとに生成されます。 単位: 個 ディメンション： GatewayId
IoTSiteWisePublisher.DroppedCount	SiteWise Edge ゲートウェイ (GatewayId) によってドロップされ、クラウドに公開されないデータポイントの数。1 分ごとに生成されます。 単位: 個 ディメンション： GatewayId

メトリクス	説明
IoTSiteWisePublisher.IsConnectedToMqttBroker	Edge ゲートウェイのパブリッシャーによって 1 SiteWise 分ごとに生成されます。 単位: 1 (パブリッシャーが MQTT ブローカーに接続されていることを示す 1) ディメンション： GatewayId
IoTSiteWisePublisher.NumberOfSubscriptionsToMqttBroker	パブリッシャーによって MQTT ブローカーにサブスクライブされ、1 分ごとに生成されるトピックの数。マルチレベルワイルドカードトピックは 1 としてカウントされます。 単位: 個 ディメンション： GatewayId
IoTSiteWisePublisher.NumberOfUniqueMqttTopicsReceived	パブリッシャーが MQTT ブローカーから受信した一意のトピックの数。1 分ごとに生成されます。 単位: 個 ディメンション： GatewayId
IoTSiteWisePublisher.MqttMessageReceivedSuccessCount	パブリッシャーが MQTT ブローカーから正常に受信し、1 分ごとに生成されるメッセージの数。 単位: 個 ディメンション： GatewayId

メトリクス	説明
`IoTSiteWisePublisher.MqttReceivedSuccessBytes`	パブリッシャーが MQTT ブローカーから正常に受信したメッセージデータのバイト数。1 分ごとに生成されます。 単位: 個 ディメンション： GatewayId

AWS IoT Greengrass Version 1 ゲートウェイメトリクス

AWS IoT SiteWise は、次の SiteWise Edge ゲートウェイメトリクスを発行します。すべての SiteWise Edge ゲートウェイメトリクスは 1 分間隔で発行されます。

> **Important**
>
> SiteWise Edge ゲートウェイメトリクスを受信するには、 SiteWise Edge ゲートウェイで少なくともバージョン 6 の AWS IoT SiteWise コネクタを使用する必要があります。詳細については、『AWS IoT Greengrass Version 1 デベロッパーガイド』の「AWS IoT SiteWise OPC-UA コレクタ」を参照してください。

SiteWise エッジゲートウェイのメトリクス

メトリクス	説明
`Gateway.Heartbeat`	接続された各 SiteWise Edge ゲートウェイ (`gatewayId`) に対して 1 分ごとに生成されます。 単位: 1 (SiteWise エッジゲートウェイが稼働していて、 SiteWise エッジゲートウェイがクラウドから切断されていることを示すデータポイントがないことを示す 1) ディメンション： GatewayId

メトリクス	説明
`Gateway.PublishSuccessCount`	SiteWise Edge ゲートウェイ (`gatewayId`) が正常に発行したデータポイントの数。 単位: 個 ディメンション： GatewayId
`Gateway.PublishFailureCount`	SiteWise Edge ゲートウェイ (`gatewayId`) が公開に失敗したデータポイントの数。 このメトリクスは、 BatchPutAssetPropertyValueオペレーションへの SiteWise Edge ゲートウェイの呼び出しに起因するエラーをカウントします。 SiteWise Edge ゲートウェイのトラブルシューティングの詳細については、「」を参照してください SiteWise Edge ゲートウェイのトラブルシューティング。 単位: 個 ディメンション： GatewayId
`Gateway.ProcessFailureCount`	SiteWise Edge ゲートウェイ (`gatewayId`) が処理に失敗したデータポイントの数。 このメトリクスは、 SiteWise ソースによって報告されたエラーを含め、エッジゲートウェイと SiteWise エッジゲートウェイのソースの間で発生するエラーをカウントします。 SiteWise Edge ゲートウェイのトラブルシューティングの詳細については、「」を参照してください SiteWise Edge ゲートウェイのトラブルシューティング。 単位: 個 ディメンション： GatewayId

メトリクス	説明
Gateway.PublishRejectedCount	Edge ゲートウェイ (gatewayId) SiteWise から拒否されたデータポイントの数。 単位: 個 ディメンション： GatewayId

OPC-UA 関連のメトリクス

メトリクス	説明
OPCUACollector.Heartbeat	Edge ゲートウェイ () に接続された各 OPC-UA ソース (sourceName) に対して 1 SiteWise 分ごとに生成されますgatewayId 。 単位: カウント (1 はソースが接続されていることを表し、0 はソースが切断されていることを表します)。 ディメンション： GatewayId， SourceName
OPCUACollector.ActiveDataStreamCount	Edge SiteWise ゲートウェイ (gatewayId) が OPC-UA ソース () 用にサブスクライブしたデータストリームの数sourceName 。 単位: 個 ディメンション： GatewayId， SourceName， PropertyGroup
OpcUaCollector.IncomingValuesCount	Edge SiteWise ゲートウェイ (gatewayId) が OPC-UA ソース (sourceName) に対して受信したデータポイントの数。1 分ごとに生成されます。 単位: 個

メトリクス	説明
	ディメンション： GatewayId， SourceName， PropertyGroup
`OpcUaCollector.IncomingValuesError`	SiteWise Edge ゲートウェイ (`gatewayId`) が OPC-UA ソース (`sourceName`) から受信した有効な値ではないデータポイントの数。これらのデータポイントは、毎分生成される OpcUa コレクターによって取り込まれません。 単位: 個 ディメンション： GatewayId， SourceName， PropertyGroup
`OpcUaCollector.ConversionErrors`	SiteWise Edge ゲートウェイ (`gatewayId`) が OPC-UA ソース (`sourceName`) に対して受信し、 へのデータの送信中に変換エラーが発生したデータポイントの数 AWS IoT SiteWise。これらのデータポイントは OpcUa コレクターによって取り込まれません。 単位: 個 ディメンション： GatewayId， SourceName

JSON 関連メトリクス

メトリクス	説明
`EIPCollector.Heartbeat`	Edge ゲートウェイ () に接続された各 EIP ソース (`sourceName`) に対して 1 SiteWise 分ごとに生成されます`gatewayId` 。 ユニット: 1 (1 はソースが接続されていることを表し、データポイントがないことは、ソースが切断されていることを表します)

メトリクス	説明
	ディメンション： GatewayId， SourceName
EIPCollector.IncomingValuesCount	EIP ソース () に対して SiteWise Edge ゲートウェイ (gatewayId) がサブスクライブしているデータストリームの数sourceName 。 単位: 個 ディメンション： GatewayId， SourceName
EIPCollector.ActiveDataStreamCount	SiteWise Edge ゲートウェイ (gatewayId) が EIP ソース () に対して受信したデータポイントの数sourceName 。 単位: 個 ディメンション： GatewayId， SourceName

Modbus 関連のメトリクス

メトリクス	説明
ModbusTCPCollector.Heartbeat	Edge ゲートウェイ () に接続された各 Modbus ソース (sourceName) に対して 1 SiteWise 分ごとに生成されますgatewayId 。 ユニット: 1 (1 は Modbus ソースが接続されていることを表し、データポイントがないことは、ソースが切断されていることを表します) ディメンション： GatewayId， SourceName
ModbusTCPCollector.IncomingValuesCount	Edge SiteWise ゲートウェイ (gatewayId) が Modbus ソース () 用にサブスクライブされているデータストリームの数sourceName 。 単位: 個

メトリクス	説明
	ディメンション： GatewayId， SourceName
ModbusTCPCollector.ActiveDataStreamCount	SiteWise Edge ゲートウェイ (gatewayId) が Modbus ソース () に対して受信したデータポイントの数sourceName 。 単位: 個 ディメンション： GatewayId， SourceName

を使用した AWS IoT SiteWise API コールのログ記録 AWS CloudTrail

AWS IoT SiteWise は と統合されています AWS IoT SiteWise。これは AWS CloudTrail、. CloudTrail captures API コールのユーザー、ロール、または AWS サービスによって実行されたアクションをイベント AWS IoT SiteWise として記録するサービスです。キャプチャされた呼び出しには、 AWS IoT SiteWise コンソールからの呼び出しと AWS IoT SiteWise API オペレーションへのコード呼び出しが含まれます。証跡を作成する場合は、 の CloudTrail イベントなど、Amazon S3 バケットへのイベントの継続的な配信を有効にすることができます AWS IoT SiteWise。証跡を設定しない場合でも、コンソールのイベント履歴 で最新の CloudTrail イベントを表示できます。 で収集された情報を使用して CloudTrail、 に対するリクエスト AWS IoT SiteWise、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

の詳細については CloudTrail、「 AWS CloudTrail ユーザーガイド」を参照してください。

AWS IoT SiteWise 内の情報 CloudTrail

CloudTrail は AWS 、アカウントの作成時にアカウントでアクティブ化されます。でサポートされているイベントアクティビティが発生すると AWS IoT SiteWise、そのアクティビティは CloudTrail イベント履歴 の他の AWS サービスイベントとともにイベントに記録されます。 AWS アカウントで最近のイベントを表示、検索、ダウンロードできます。詳細については、「イベント履歴 を使用した CloudTrail イベントの表示」を参照してください。

のイベントなど、 AWS アカウントのイベントの継続的な記録については AWS IoT SiteWise、証跡を作成します。証跡により、 はログファイル CloudTrail を Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成すると、証跡はすべての AWS リージョンに適用されま

す。証跡は、 AWS パーティション内のすべてのリージョンからのイベントを口グに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログで収集されたデータをより詳細に分析し、それに基づく対応を行うように他の AWS サービスを設定できます。詳細については、次を参照してください:

- 追跡を作成するための概要
- CloudTrail でサポートされているサービスと統合
- の Amazon SNS 通知の設定 CloudTrail
- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからのログファイルの受信 CloudTrail

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するために役立ちます。

- リクエストがルートまたは AWS Identity and Access Management (IAM) ユーザーの認証情報のどちらを使用して送信されたか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって送信されたかどうか。

詳細については、「CloudTrail userIdentity 要素」を参照してください。

AWS IoT SiteWise での データイベント CloudTrail

データイベントでは、リソース上またはリソース内で実行されるリソースオペレーション (Amazon S3 オブジェクトの読み取りまたは書き込みなど) についての情報が得られます。これらのイベントは、データプレーンオペレーションとも呼ばれます。データイベントは、多くの場合、高ボリュームのアクティビティです。デフォルトでは、 CloudTrail はデータイベントを記録しません。 CloudTrail イベント履歴にはデータイベントは記録されません。

追加の変更がイベントデータに適用されます。 CloudTrail 料金の詳細については、AWS CloudTrail 「 の料金」を参照してください。

CloudTrail コンソール、、または CloudTrail API オペレーションを使用して AWS CLI、 AWS IoT SiteWise リソースタイプのデータイベントを口グに記録できます。このセクションの表は、 で使用できるリソースタイプを示しています AWS IoT SiteWise。

- CloudTrail コンソールを使用してデータイベントを記録するには、証跡またはイベントデータストアを作成してデータイベントを記録するか、既存の証跡またはイベントデータストアを更新してデータイベントをログに記録します。

  1. データイベントを選択して、データイベントをログに記録します。
  2. データイベントタイプのリストから、データイベントを記録するリソースタイプを選択します。
  3. 使用するログセレクタテンプレートを選択します。リソースタイプのすべてのデータイベントをログに記録したり、すべての`readOnly`イベントをログに記録したり、すべての`writeOnly`イベントをログに記録したり、カスタムログセレクタテンプレートを作成して`readOnly`、、`eventName`および `resources.ARN`フィールドでフィルタリングしたりできます。

- を使用してデータイベントをログに記録するには AWS CLI、 `eventCategory`フィールドを `Data`に、 `resources.type`フィールドをリソースタイプ値に等しく設定するように `--advanced-event-selectors`パラメータを設定します (表 を参照) 。、`readOnly`、`eventName`および `resources.ARN`フィールドの値でフィルタリングする条件を追加できます。

  - データイベントを記録するように証跡を設定するには、 AWS CloudTrail put-event-selectors コマンドを実行します。詳細については、「 を使用した証跡のデータイベントのログ記録 AWS CLI」を参照してください。
  - データイベントをログに記録するようにイベントデータストアを設定するには、 AWS CloudTrail create-event-data-store コマンドを実行してデータイベントをログに記録する新しいイベントデータストアを作成するか、 AWS CloudTrail update-event-data-store コマンドを実行して既存のイベントデータストアを更新します。詳細については、「 を使用したイベントデータストアのデータイベントのログ記録 AWS CLI」を参照してください。

次の表に、 AWS IoT SiteWise リソースタイプを示します。データイベントタイプ (コンソール) 列には、 CloudTrail コンソールのデータイベントタイプリストから選択する値が表示されます。resources.type 値列には、 AWS CLI または CloudTrail APIs を使用して高度なイベントセレクタを設定するときに指定する `resources.type`値が表示されます。列にログ記録された Data APIs CloudTrail には、リソースタイプ CloudTrail について にログ記録された API コールが表示されます。

データイベントタイプ (コンソール)	resources.type 値	ログに記録されたデータ APIs CloudTrail*
AWS IoT SiteWise アセット	`AWS::IoTSiteWise::Asset`	• BatchPutAssetPropertyValue • GetAssetPropertyValue • GetAssetPropertyValueHistory • GetAssetPropertyAggregates • GetInterpolatedAssetPropertyValues • BatchGetAssetPropertyValue • BatchGetAssetPropertyValueHistory • BatchGetAssetPropertyAggregates
AWS IoT SiteWise 時系列	`AWS::IoTSiteWise::TimeSeries`	• BatchPutAssetPropertyValue • GetAssetPropertyValue • GetAssetPropertyValueHistory • GetAssetPropertyAggregates • GetInterpolatedAssetPropertyValues • BatchGetAssetPropertyValue • BatchGetAssetPropertyValueHistory

データイベントタイプ (コンソール)	resources.type 値	ログに記録されたデータ APIs CloudTrail*
		• BatchGetAssetPropertyAggregates

> **Note**
>
> Cloudtrail イベントに記録される resources.type は、API リクエストで使用される識別子によって異なります。リクエストでアセット ID が指定されている場合、アセット resources.type がログに記録され、それ以外の場合は TimeSeries resources.type がログに記録されます。

*、`eventName`、および `resources.ARN`フィールドでフィルタリングして`readOnly`、自分にとって重要なイベントのみをログに記録するように高度なイベントセレクタを設定できます。フィールドの詳細については、「AdvancedFieldSelector」を参照してください。

AWS IoT SiteWise での 管理イベント CloudTrail

管理イベントは、 AWS アカウントのリソースで実行される管理オペレーションに関する情報を提供します。これらのイベントは、コントロールプレーンオペレーションとも呼ばれます。デフォルトでは、 は管理イベント CloudTrail を記録します。

AWS IoT SiteWise は、すべての AWS IoT SiteWise コントロールプレーンオペレーションを管理イベントとして記録します。が に記録する AWS IoT SiteWise コントロールプレーンオペレーションのリストについては CloudTrail、 AWS IoT SiteWise AWS IoT SiteWise 「 API リファレンス」を参照してください。

例: AWS IoT SiteWise ログファイルエントリ

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信できるようにする設定です。 CloudTrail ログファイルには、1 つ以上のログエントリが含まれます。イベントは任意の送信元からの単一のリクエストを表し、リクエストされたオペレーション、オペレーションの日時、リクエストパラメータなどに関する情報が含まれます。 CloudTrail ログファイルは、パブリック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

次の例は、 `CreateAsset`オペレーションを示す CloudTrail ログエントリを示しています。

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::123456789012:user/Administrator",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "Administrator",
    "sessionContext": {
      "sessionIssuer": {},
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2020-03-11T17:26:40Z"
      }
    },
    "invokedBy": "signin.amazonaws.com"
  },
  "eventTime": "2020-03-11T18:01:22Z",
  "eventSource": "iotsitewise.amazonaws.com",
  "eventName": "CreateAsset",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.0",
  "userAgent": "signin.amazonaws.com",
  "requestParameters": {
    "assetName": "Wind Turbine 1",
    "assetModelId": "a1b2c3d4-5678-90ab-cdef-11111EXAMPLE",
    "clientToken": "a1b2c3d4-5678-90ab-cdef-00000EXAMPLE"
  },
  "responseElements": {
    "assetId": "a1b2c3d4-5678-90ab-cdef-22222EXAMPLE",
    "assetArn": "arn:aws:iotsitewise:us-east-1:123456789012:asset/a1b2c3d4-5678-90ab-
cdef-22222EXAMPLE",
    "assetStatus": {
      "state": "CREATING"
    }
  },
  "requestID": "a1b2c3d4-5678-90ab-cdef-aaaaaEXAMPLE",
  "eventID": "a1b2c3d4-5678-90ab-cdef-bbbbbEXAMPLE",
  "eventType": "AwsApiCall",
  "recipientAccountId": "123456789012"
}
```

リソースにタグを付ける AWS IoT SiteWise

AWS IoT SiteWise リソースにタグを付けると、組織の資産を効率的に分類、管理、取得するための強力な方法となります。キーと値のペアで構成されるタグを割り当てることで、わかりやすいメタデータをリソースに添付できます。タグのメタデータを使用して操作を効率化できます。たとえば、風力発電所のシナリオでは、タグを使用してタービンに位置、容量、稼働状況などの特定の属性を付けて、タービン内の識別と管理を迅速に行うことができます。 AWS IoT SiteWise

タグを AWS Identity and Access Management (IAM) ポリシーと統合すると、条件付きアクセスルールを定義できるため、セキュリティと運用管理が強化されます。つまり、特定のタグを持つユーザーのみを指定できるということです。たとえば、特定のロールまたは部署にタグ付けされたユーザーのみが、特定のリソースにアクセスしたり変更したりできます。

でのタグの使用 AWS IoT SiteWise

タグを使用して、目的、所有者、環境、 AWS IoT SiteWise またはユースケースに応じたその他の分類によってリソースを分類します。同じ型のリソースが多い場合に、そのタグに基づいて特定のリソースをすばやく識別できます。

各タグは、指定するキーとオプションの値で構成されます。たとえば、資産モデルに一連のタグを設定して、そのモデルがサポートする産業プロセスに従って追跡することができます。管理するリソースの種類ごとにタグキーをカスタマイズすることをお勧めします。タグキーのセットに一貫性を持たせることで、リソースの管理が容易になります。

によるタグ付け AWS Management Console

AWS Management Console のタグエディタを使用すると、 AWS すべてのサービスのリソースのタグを一元的に作成して管理できます。詳細については、[AWS Resource Groups User Guide] (ユーザーガイド) の[Tag Editor] (タグエディタ) を参照してください。

API によるタグ付け AWS IoT SiteWise

AWS IoT SiteWise API はタグも使用します。タグを作成する前に、タグの制限に注意してください。詳細については、「AWS 全般のリファレンス」の「タグの命名規則と使用規則」を参照してください。

- リソースの作成時にタグを追加するには、リソースの `tags` プロパティでタグを定義します。

- 既存のリソースにタグを追加したり、タグ値を更新したりするには、TagResourceオペレーションを使用します。
- リソースからタグを削除するには、UntagResourceオペレーションを使用します。
- リソースに関連付けられているタグを取得するには、ListTagsForResourceオペレーションを使用するか、tagsリソースを記述してそのプロパティを調べます。

次の表は、 AWS IoT SiteWise API を使用してタグ付けできるリソースと、それに対応する Create AND Describe オペレーションの一覧です。

タグ付け可能なリソース AWS IoT SiteWise

リソース	作成オペレーション	オペレーションの説明
資産モデルまたはコンポーネントモデル	CreateAssetModel	DescribeAssetModel
アセット	CreateAsset	DescribeAsset
SiteWise エッジゲートウェイ	CreateGateway	DescribeGateway
Portal	CreatePortal	DescribePortal
プロジェクト	CreateProject	DescribeProject
ダッシュボード	CreateDashboard	DescribeDashboard
アクセスポリシー	CreateAccessPolicy	DescribeAccessPolicy
時系列	BatchPutAssetPropertyValue	DescribeTimeSeries

BatchPutAssetPropertyValueでは、 AWS IoT SiteWise アセットモデルやアセットを作成する前に、産業用データを送信するようにデータソースを設定できます。 AWS IoT SiteWise 機器から未加工データのストリームを受信するデータストリームを自動的に作成します。詳細については、「データインジェストの管理」を参照してください。

ビューで次のオペレーションを使用して、タグ付けをサポートしているリソースのタグを管理します。

- TagResource— リソースにタグを追加したり、既存のタグの値を更新したりします。

- ListTagsForResource— リソースのタグを一覧表示します。
- UntagResource— リソースからタグを削除します。

リソースへのタグの追加や削除はいつでも可能です。既存のタグキーの値を更新するには、同じキーと希望する新しい値を含む新しいタグをリソースに追加します。このアクションは、古い値を新しい値に置き換えます。空の文字列をタグ値として割り当てることはできますが、NULL 値を割り当てることはできません。

リソースを削除すると、そのリソースにリンクされているタグもすべて削除されます。

IAM ポリシーでのタグの使用

IAM ポリシーでリソースタグを使用して、ユーザーのアクセスと権限を制御します。たとえば、ポリシーでは、特定のタグがアタッチされたリソースのみを作成することをユーザーに許可できます。ポリシーにより、特定のタグを持つリソースをユーザーが作成または変更できないよう制限することもできます。

> **Note**
>
> タグを使用してリソースへのユーザーのアクセスを許可または拒否する場合は、ユーザーが同じリソースに対してそれらのタグを追加または削除する機能を拒否する必要があります。そうしないと、ユーザーはタグを変更することで制限を回避し、リソースにアクセスできるようになる可能性があります。

ポリシーステートメントの Condition 要素 (Condition ブロックとも呼ばれます) の次の条件コンテキストキーと値を使用できます。

`aws:ResourceTag/`*`tag-key`*`: `*`tag-value`*

特定のタグを持つリソースに対してアクションを許可または拒否します。

`aws:RequestTag/`*`tag-key`*`: `*`tag-value`*

タグ付け可能なリソースを作成または変更するときに、特定のタグを使用する (または使用しない) よう要求します。

`aws:TagKeys: [tag-key, ...]`

タグ付け可能なリソースを作成または変更するときに、特定のタグキーセットを使用する (または使用しない) よう要求します。

> **Note**
>
> IAM ポリシーの条件コンテキストキーと値は、必須パラメータとしてタグ付け可能なリソースを持つアクションにのみ適用されます。たとえば、にタグベースの条件付きアクセスを設定できます。ListAssetsPutLoggingOptionsリクエストではタグ付け可能なリソースが参照されていないため、タグベースの条件付きアクセスをオンに設定することはできません。

詳細については、『IAM ユーザーガイド』の「AWS リソースタグによるリソースへのアクセスの制御」と「IAM JSON ポリシーリファレンス」を参照してください。

タグを使用した IAM ポリシーの例

- タグに基づく AWS IoT SiteWise アセットの表示

トラブルシューティング AWS IoT SiteWise

これらのセクションの情報を使用して、の問題のトラブルシューティングと解決を行います。 AWS IoT SiteWise

トピック

一括インポートおよびエクスポートオペレーションのトラブルシューティング

転送ジョブ中に生成されたエラーを処理して診断するには、 API を参照してください AWS IoT TwinMaker GetMetadataTransferJob。

1. 転送ジョブを作成して実行したら、 GetMetadataTransferJob API を呼び出します。

```
aws iottwinmaker get-metadata-transfer-job \
--metadata-transfer-job-id your_metadata_transfer_job_id \
--region us-east-1
```

2. ジョブの状態は、次のいずれかの状態に変わります。
   - COMPLETED
   - CANCELLED
   - ERROR
3. GetMetadataTransferJob API は MetadataTransferJobProgress オブジェクトを返します。
4. MetadataTransferJobProgress オブジェクトには、次のパラメータが含まれます。
   - failedCount : 転送プロセス中に失敗したアセットの数を示します。
   - skippedCount: 転送プロセス中にスキップされたアセットの数を示します。
   - succeededCount : 転送プロセス中に成功したアセットの数を示します。

- totalCount : 転送プロセスに関係するアセットの総数を示します。

5. さらに、署名付き URL を含む API コールによって reportUrl 要素が返されます。転送ジョブに調査が必要なエラーがある場合は、この URL から完全なエラーレポートをダウンロードできます。

AWS IoT SiteWise ポータルのトラブルシューティング

AWS IoT SiteWise ポータルに関する一般的な問題のトラブルシューティングを行います。

ユーザー、管理者が AWS IoT SiteWise ポータルにアクセスできない

ユーザーまたは管理者が AWS IoT SiteWise ポータルにアクセスできない場合は、アタッチされた AWS Identity and Access Management (IAM) ポリシーで、ログインの成功を妨げるアクセス許可が制限されている可能性があります。

ログインが失敗する IAM ポリシーの例を以下に示します。

> **Note**
>
> アタッチされている IAM ポリシーに "Condition" 要素を含むポリシーがあると、ログイン失敗の原因となります。

例 1: IP アドレスが制限されているため、ログインに失敗する。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribePortal"
            ],
            "Resource": "*",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": [
                        "REPLACESAMPLEIP"
                    ]
```

```
                }
            }
        }
    ]
}
```

例 2: タグが含まれているため、ログインに失敗する。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iotsitewise:DescribePortal"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:ResourceTag/project": "*"
                }
            }
        }
    ]
}
```

ポータルへのユーザーまたは管理者の追加では、IP の制限などのユーザー許可を制限する IAM ポリシーを作成しないようにしてください。アクセス許可が制限されているポリシーがアタッチされていると、 AWS IoT SiteWise ポータルに接続できません。

SiteWise Edge ゲートウェイのトラブルシューティング

AWS IoT SiteWise エッジゲートウェイは一連の AWS IoT Greengrass コンポーネントを実行します。Amazon CloudWatch および SiteWise Edge ゲートウェイのローカルファイルシステムにイベントを記録するように SiteWise Edge ゲートウェイを設定できます。その後、ログファイルを表示して SiteWise Edge ゲートウェイのトラブルシューティングを行うことができます。

SiteWise Edge ゲートウェイによって報告された CloudWatch メトリクスを表示して、接続またはデータストリームに関する問題をトラブルシューティングすることもできます。詳細については、「Amazon CloudWatch メトリクス AWS IoT SiteWise によるモニタリング」を参照してください。

トピック

SiteWise Edge ゲートウェイログの設定とアクセス

SiteWise Edge ゲートウェイログを表示する前に、Amazon CloudWatch Logs にログを送信するか、ローカルファイルシステムにログを保存するように SiteWise Edge ゲートウェイを設定する必要があります。

- を使用して SiteWise Edge ゲートウェイのログファイルを表示する場合は AWS Management Console 、 CloudWatch ログを使用します。詳細については、「Amazon CloudWatch ログを使用する」を参照してください。
- コマンドラインまたはローカルソフトウェアを使用して SiteWise Edge ゲートウェイのログファイルを表示する場合は、ローカルファイルシステムログを使用します。詳細については、「サービスログを使用する」を参照してください。

SiteWise Edge ゲートウェイの問題のトラブルシューティング

SiteWise Edge ゲートウェイの問題のトラブルシューティングには、次の情報を使用します。

問題

SiteWise Edge ゲートウェイにパックをデプロイできない

AWS IoT Greengrass nucleus コンポーネント (`aws.greengrass.Nucleus`) が古い場合、SiteWise Edge ゲートウェイにパックをデプロイできない可能性があります。 AWS IoT Greengrass V2 コンソールを使用して AWS IoT Greengrass nucleus コンポーネントをアップグレードできます。

AWS IoT Greengrass nucleus コンポーネントのアップグレード (コンソール)

1. AWS IoT Greengrass コンソールに移動します。
2. ナビゲーションペインの AWS IoT Greengrass で、[デプロイ] を選択します。
3. デプロイのリストで、修正するデプロイを選択します。
4. [修正] を選択します。
5. [Specify target] (ターゲットの指定) ページで [Next] (次へ) を選択します。
6. [コンポーネントの選択] ページの [パブリックコンポーネント] の検索ボックスに **aws.greengrass.Nucleus** と入力し、aws.greengrass.Nucleus を選択します。
7. [次へ] をクリックします。
8. [コンポーネントの設定] ページで、[次へ] をクリックします。
9. [詳細設定の設定] ページで、[次へ] を選択します。
10. [Review] ページで、[デプロイ] を選択します。

AWS IoT SiteWise OPC-UA サーバーからデータを受信しない

AWS IoT SiteWise アセットが OPC-UA サーバーから送信されたデータを受信していない場合は、SiteWise エッジゲートウェイのログを検索して問題をトラブルシューティングできます。次のメッセージを含む情報レベルの `swPublisher` ログを探します。

```
Emitting diagnostic name=PublishError.SomeException
```

ログ*SomeException*内の のタイプに基づいて、次の例外タイプと対応する問題を使用して SiteWise Edge ゲートウェイのトラブルシューティングを行います。

- ResourceNotFoundException – OPC-UA サーバーは、アセットのプロパティエイリアスと一致しないデータを送信しています。この例外は、次の 2 つの場合に発生します。
  - プロパティエイリアスが、定義したソースプレフィックスを含め、OPC-UA 変数と正確に一致しません。プロパティエイリアスとソースプレフィックスが正しいことを確認します。
  - OPC-UA 変数をアセットプロパティにマッピングしていません。詳細については、「アセットプロパティへの産業データストリームのマッピング」を参照してください。

    で必要なすべての OPC-UA 変数を既にマッピングしている場合は AWS IoT SiteWise、SiteWise エッジゲートウェイが送信する OPC-UA 変数をフィルタリングできます。詳細については、「OPC-UA ノードフィルターの使用」を参照してください。

- InvalidRequestException – OPC-UA 変数のデータ型がアセットプロパティのデータ型と一致しません。たとえば、OPC-UA 変数に整数データ型がある場合、対応するアセットプロパティは整数データ型である必要があります。ダブルタイプのアセットプロパティは、OPC-UA 整数値を受け取ることができません。この問題を解決するには、正しいデータ型で新しいプロパティを定義します。
- TimestampOutOfRangeException – SiteWise Edge ゲートウェイは、 AWS IoT SiteWise Accepts. AWS IoT SiteWise rejects の範囲外のデータを送信しています。タイムスタンプが過去 7 日より前、または今後 5 分より新しいデータポイントを拒否します。 SiteWise Edge ゲートウェイの電源または AWS クラウドへの接続が失われた場合は、 SiteWise Edge ゲートウェイのキャッシュをクリアする必要がある場合があります。
- ThrottlingException または LimitExceededException- リクエストが、取り込まれたデータポイントのレートやアセットプロパティデータ API オペレーションのリクエストレートなどの AWS IoT SiteWise サービスクォータを超えました。設定が AWS IoT SiteWise クォータ を超えないことを確認します。

ダッシュボードにデータが表示されない

ダッシュボードにデータが表示されない場合、 SiteWise Edge ゲートウェイのパブリッシャー設定とデータソースが同期していない可能性があります。同期されていない場合、データソースの名前を更新すると、クラウドからエッジへの同期が速くなり、同期停止エラーが修正される可能性があります。

データソースの名前を更新するには

1. AWS IoT SiteWise コンソールに移動します。
2. ナビゲーションペインで、エッジゲートウェイ を選択します。
3. ダッシュボードに接続されている SiteWise Edge ゲートウェイを選択します。
4. [データソース] で [編集] を選択します。
5. 新しいソースの [名前] を選択し、[保存] を選択して変更を確定します。
6. データソーステーブルのデータソース名が更新されて、変更が反映されていることを確認します。

/greengrass/v2/logs エラーで aws.iot.SiteWiseEdgePublisher logs に表示される「メインクラスが見つからないかロードされませんでした」

このエラーが表示された場合は、 SiteWise エッジゲートウェイの Java バージョンを更新する必要がある場合があります。

- ターミナルから、次のコマンドを実行します。

```
java -version
```

  SiteWise Edge ゲートウェイが実行されている Java のバージョンは、 の下に表示されますOpenJDK Runtime Environment。以下のようなレスポンスが表示されます。

```
openjdk version "11.0.20" 2023-07-18 LTS
OpenJDK Runtime Environment Corretto011.0.20.8.1 (build 11.0.20+8-LTS
OpenJDK 64-Bit Server VM Corretto-11.0.20.8.1 (build 11.0.20+8-LTS, mixed node)
```

Java バージョン 11.0.20.8.1 を実行している場合は、IoT SiteWise Publisher Pack をバージョン 2.4.1 以降に更新する必要があります。影響を受けるのは java バージョン 11.0.20.8.1 のみです。他の java バージョンを使用する環境では、古いバージョンの IoT SiteWise Publisher コンポーネントを引き続き使用できます。コンポーネントパックの更新の詳細については、「SiteWise Edge ゲートウェイコンポーネントパックのバージョンの変更」を参照してください。

AWS IoT Greengrass 問題のトラブルシューティング

で SiteWise Edge ゲートウェイを設定またはデプロイする際の多くの問題の解決策については AWS IoT Greengrass、「 AWS IoT Greengrass デベロッパーガイド」の「トラブルシューティング AWS IoT Greengrass」を参照してください。

AWS IoT SiteWise ルールアクションのトラブルシューティング

AWS IoT SiteWise でルールアクションをトラブルシューティングするには AWS IoT Core、次のいずれかの手順を実行します。

- Amazon CloudWatch ログの設定
- ルールの再パブリッシュエラーアクションを設定する

次に、エラーメッセージとこのトピックのエラーを比較して、問題のトラブルシューティングを行います。

トピック

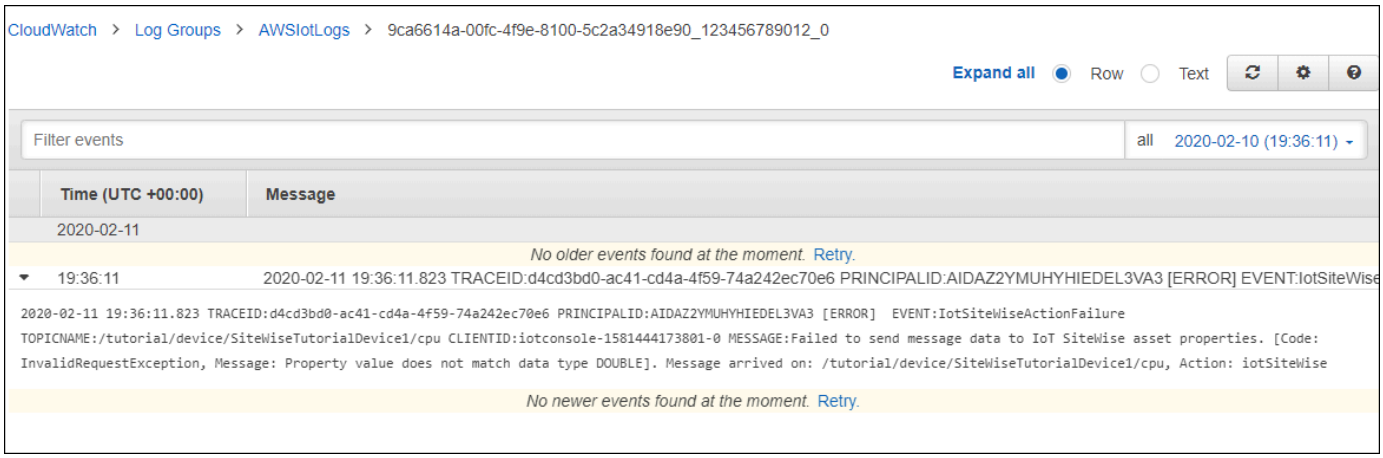
- AWS IoT Core ログを設定する
- 再パブリッシュエラーアクションの設定
- 問題のトラブルシューティング
- ルールのトラブルシューティング
- ルールのトラブルシューティング

AWS IoT Core ログを設定する

AWS IoT CloudWatch さまざまなレベルの情報をログに記録するように設定できます。

CloudWatch ログを設定してアクセスするには

1. のロギングを設定するには AWS IoT Core、『AWS IoT 開発者ガイド』の「CloudWatch ログを使った監視」を参照してください。
2. CloudWatch コンソールに移動します。
3. ナビゲーションペインで、[ロググループ] を選択します。
4. AWSIotLogsグループを選択します。
5. 最近のログストリームを選択します。デフォルトでは、 CloudWatch 最新のログストリームが最初に表示されます。
6. ログエントリを選択して、ログメッセージを展開します。ログエントリは、次のスクリーンショットのようになります。



7. エラーメッセージとこのトピックのエラーを比較して、問題のトラブルシューティングを行います。

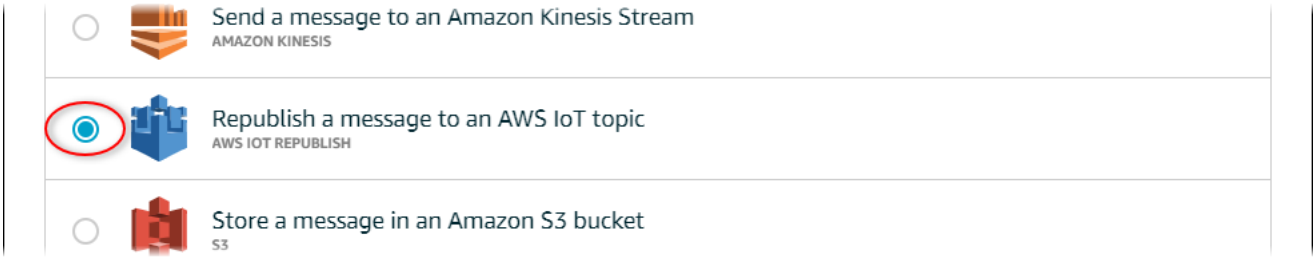
再パブリッシュエラーアクションの設定

エラーメッセージを処理するために、ルールにエラーアクションを設定できます。この手順では、MQTT テストクライアントでエラーメッセージを表示するために、エラーアクションとして再パブリッシュルールアクションを設定します。

> **Note**
>
> 再パブリッシュエラーアクションでは、ERROR レベルログと同等のログのみが出力されます。より詳細なログが必要な場合は、 CloudWatch ログを設定する必要があります。

再パブリッシュエラーアクションをルールに追加するには

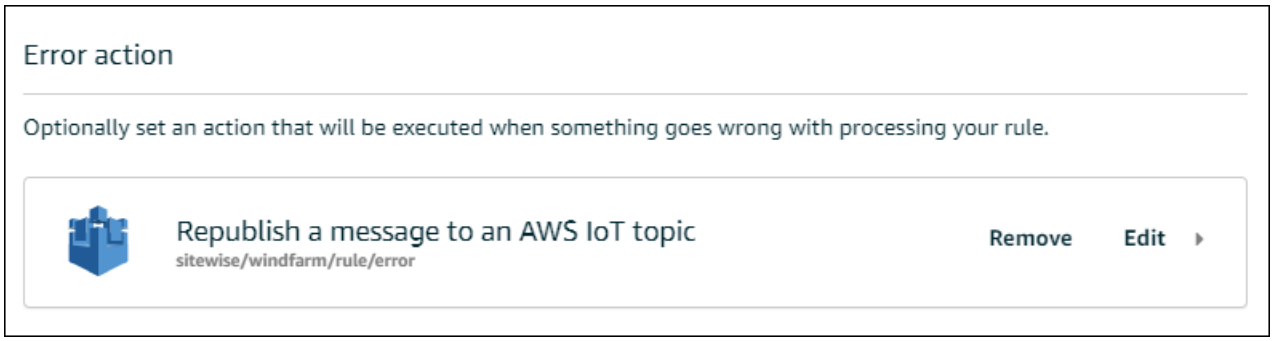
1. AWS IoT コンソールに移動します。
2. 左側のナビゲーションペインで [Act (アクト)] を選択し、[ルール] を選択します。
3. ルールを選択します。
4. [エラーアクション] で、[アクションの追加] を選択します。
5. [メッセージをトピックに再公開] を選択します。 AWS IoT



6. ページの下部にある [アクションの設定] を選択します。
7. [トピック] に、固有のトピック (例:**`sitewise/windfarm/rule/error`**) を入力します。 AWS IoT Core エラーメッセージをこのトピックに再公開します。
8. [Select] を選択すると、 AWS IoT Core エラーアクションを実行するためのアクセス権限が付与されます。
9. ルールに作成したロールの横にある [選択] を選択します。
10. [ロールの更新] を選択して、追加のアクセス許可をロールに追加します。

11. [アクションの追加] を選択します。

    ルールのエラーアクションは、次のスクリーンショットのようになります。



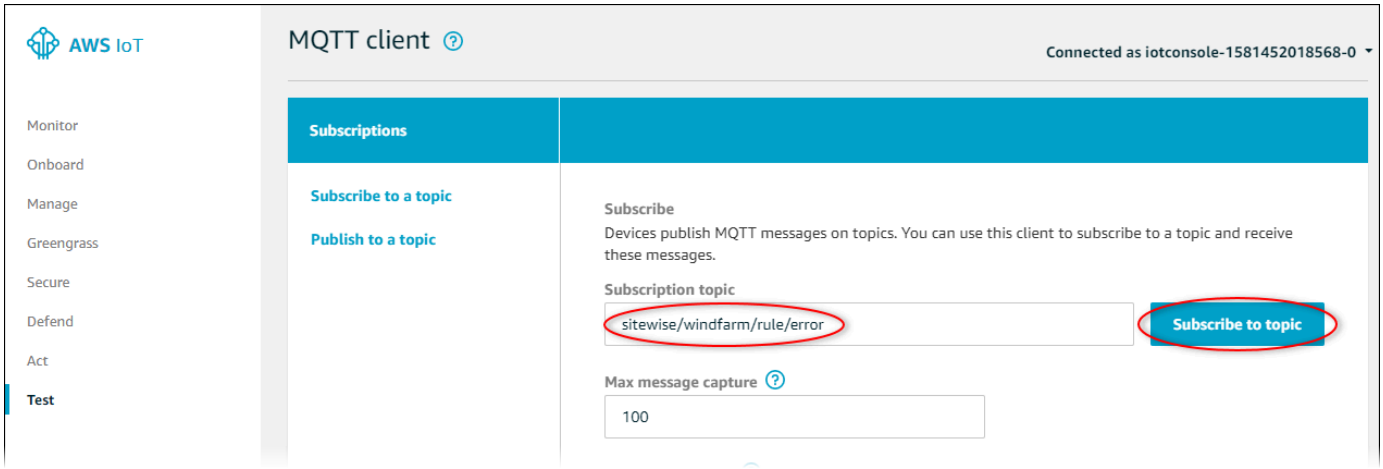
12. コンソールの左上にある戻る矢印を選択して、 AWS IoT コンソールのホームに戻ります。

再パブリッシュエラーアクションを設定したら、 AWS IoT Coreの MQTT テストクライアントでエラーメッセージを見ることができます。

次の手順では、MQTT テストクライアントでエラートピックにサブスクライブします。MQTT テストクライアントでは、問題のトラブルシューティングを行うためにルールのエラーメッセージを受信することができます。

エラーアクショントピックをサブスクライブするには

1. AWS IoT コンソールに移動します。
2. 左側のナビゲーションページで、[テスト] を選択して MQTT テストクライアントを開きます。
3. [トピックのサブスクリプション] フィールドに、前に設定したエラートピック (**sitewise/windfarm/rule/error** など) を入力し、[トピックへのサブスクライブ] を選択します。



4. エラーメッセージが表示されるまで監視し、エラーメッセージで `failures` 配列を展開します。

   次に、エラーメッセージとこのトピックのエラーを比較して、問題のトラブルシューティングを行います。

問題のトラブルシューティング

ルールの問題をトラブルシューティングするには、次の情報を使用します。

問題

- Error: Member must be within 604800 seconds before and 300 seconds after the current timestamp (エラー: メンバーは、現在のタイムスタンプ前の 604800 秒以内で、現在のタイムスタンプ後の 300 秒以内でなければなりません)
- Error: Property value does not match data type <type> (エラー: プロパティ値がデータ型 <type> と一致しません)
- エラー:ユーザー:<role-arn>リソースで iotsitewise: を実行する権限がありません BatchPutAssetPropertyValue
- エラー:iot.amazonaws.com はリソースで:sts: を実行できません:AssumeRole <role-arn>
- Info: No requests were sent. (情報: リクエストは送信されませんでした。) PutAssetPropertyValueEntries 代替テンプレートを実行した後は空になりました。

Error: Member must be within 604800 seconds before and 300 seconds after the current timestamp (エラー: メンバーは、現在のタイムスタンプ前の 604800 秒以内で、現在のタイムスタンプ後の 300 秒以内でなければなりません)

タイムスタンプが、現在の Unix エポック時間と比較して、7 日より前か、5 分より後です。次の操作を試してください：

- タイムスタンプが Unix エポック (UTC) 時間であることを確認します。別のタイムゾーンでタイムスタンプを指定すると、このエラーが表示されます。
- タイムスタンプが秒単位であることを確認してください。 AWS IoT SiteWise タイムスタンプは秒単位 (Unix エポックタイム) とオフセット (ナノ秒単位) に分割されることを想定しています。
- 過去に 7 日以内のタイムスタンプが設定されたデータをアップロードしていることを確認します。

Error: Property value does not match data type <type> (エラー: プロパティ値がデータ型 <type> と一致しません)

ルールアクションのエントリのデータ型が、ターゲットアセットプロパティとは異なります。たとえば、ターゲットアセットプロパティが DOUBLE で、選択したデータ型が [整数] であるか、値を `integerValue` に渡したとします。次の操作を試してください：

- AWS IoT コンソールからルールを設定する場合は、各エントリに正しいデータ型が選択されていることを確認してください。
- API または AWS Command Line Interface (AWS CLI) からルールを設定する場合は、`value`オブジェクトが正しいタイプフィールド (DOUBLEプロパティなど`doubleValue`) を使用していることを確認してください。

エラー:ユーザー:<role-arn>リソースで iotsitewise: を実行する権限がありません BatchPutAssetPropertyValue

ルールにターゲットアセットプロパティにアクセスする権限がないか、ターゲットアセットプロパティが存在しません。次の操作を試してください：

- プロパティエイリアスが正しいことと、指定されたプロパティエイリアスを持つアセットプロパティがあることを確認します。詳細については、「アセットプロパティへの産業データストリームのマッピング」を参照してください。
- ルールにロールがあることと、ターゲットのアセットプロパティ (ターゲットアセットの階層など) への `iotsitewise:BatchPutAssetPropertyValue` アクセス許可がロールによって許可されていることを確認します。詳細については、「AWS IoT 必要なアクセス権の付与」を参照してください。

エラー:iot.amazonaws.com はリソースで:sts: を実行できません:AssumeRole <role-arn>

ユーザーには (IAM) のルールでロールを引き受ける権限がありません。 AWS Identity and Access Management

作成されたルールで、そのロールに対する `iam:PassRole` へのアクセス許可がユーザーに許可されていることを確認します。詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Pass role permissions] (ロールアクセス許可を渡す) を参照してください。

Info: No requests were sent. (情報: リクエストは送信されませんでした。) PutAssetPropertyValueEntries 代替テンプレートを実行した後は空になりました。

> **Note**
>
> このメッセージは `INFO` レベルのログです。

リクエストに、必須パラメータをすべて含むエントリが少なくとも 1 つ含まれている必要があります。

置換テンプレートなどのルールのパラメータが空でない値になることを確認します。置換テンプレートは、ルールクエリステートメントの `AS` 句に定義されている値にアクセスできません。詳細については、[AWS IoT Developer Guide] (デベロッパーガイド) の[Substitution templates] (置換テンプレート) を参照してください。

ルールのトラブルシューティング

CPU AWS IoT SiteWise とメモリの使用状況データが期待どおりに表示されない場合は、以下の手順に従ってルールをトラブルシューティングしてください。この手順では、MQTT テストクライアントでエラーメッセージを表示するために、エラーアクションとして再パブリッシュルールアクションを設定します。また、 CloudWatch ログへのロギングを設定してトラブルシューティングすることもできます。詳細については、「AWS IoT SiteWise ルールアクションのトラブルシューティング」を参照してください。

再パブリッシュエラーアクションをルールに追加するには

1. AWS IoT コンソールに移動します。
2. 左側のナビゲーションペインで [メッセージルーティング] を選択し、[ルール] を選択します。
3. 先ほど作成したルールを選択し、[編集] を選択します。
4. [エラーアクション - オプション] で、[エラーアクションの追加] を選択します。
5. [メッセージをトピックに再公開] を選択します。 AWS IoT
6. [トピック] に、エラーへのパス (例:**`sitewise/rule/tutorial/error`**) を入力します。 AWS IoT Core エラーメッセージをこのトピックに再公開します。
7. 以前に作成したロール (例:SiteWiseTutorialDeviceRuleRole) を選択します。
8. [更新] を選択します。

再パブリッシュエラーアクションを設定したら、 AWS IoT Coreの MQTT テストクライアントでエラーメッセージを見ることができます。

次の手順では、MQTT テストクライアントでエラートピックにサブスクライブします。

エラーアクショントピックをサブスクライブするには

1. AWS IoT コンソールに移動します。
2. 左側のナビゲーションページで、[MQTT テストクライアント] を選択して MQTT テストクライアントを開きます。
3. [トピックのフィルター] で、**sitewise/rule/tutorial/error** と入力し、[サブスクリプション] を選択します。

エラーメッセージが表示された場合は、エラーメッセージで `failures` 配列を表示して問題を診断します。考えられる問題とその解決方法の詳細については、「AWS IoT SiteWise ルールアクションのトラブルシューティング」を参照してください。

エラーが表示されない場合は、ルールが有効になっていて、再パブリッシュエラーアクションで設定したトピックと同じトピックにサブスクライブしていることを確認します。それでもエラーが表示されない場合は、デバイススクリプトが実行され、デバイスのシャドウが正常に更新されていることを確認します。

> **Note**
>
> デバイスのシャドウアップデートトピックに登録して、 AWS IoT SiteWise アクションが解析するペイロードを表示することもできます。これを行うには、次のトピックをサブスクライブしてください。
>
> ```
> $aws/things/+/shadow/update/accepted
> ```

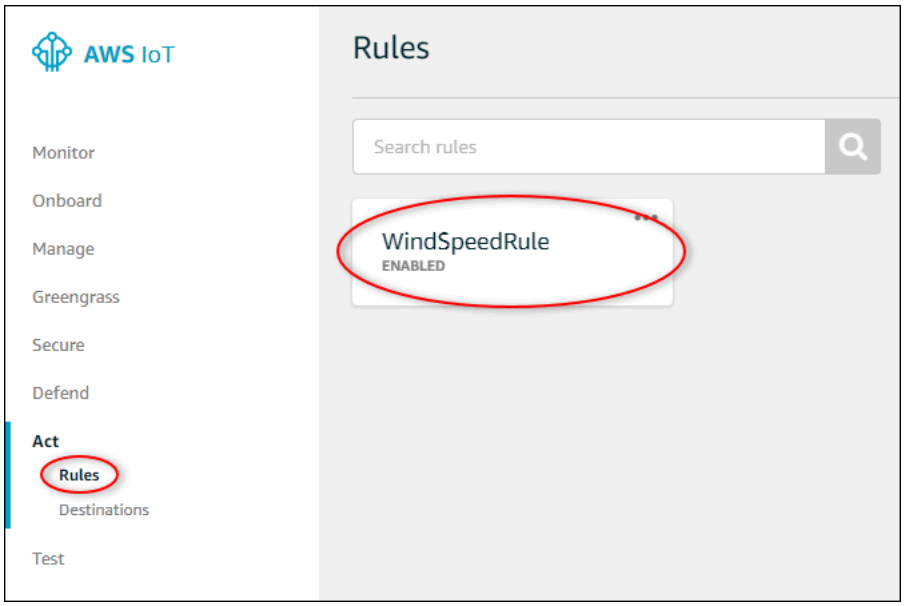
ルールのトラブルシューティング

デモのアセットデータが想定通りに DynamoDB テーブルで表示されない場合は、ここに記載されているステップに従って、ルールをトラブルシューティングします。この手順では、MQTT テストクライアントでエラーメッセージを表示するために、エラーアクションとして再パブリッシュルールアクションを設定します。 CloudWatch Logs への口ギングを設定してトラブルシューティングするこ

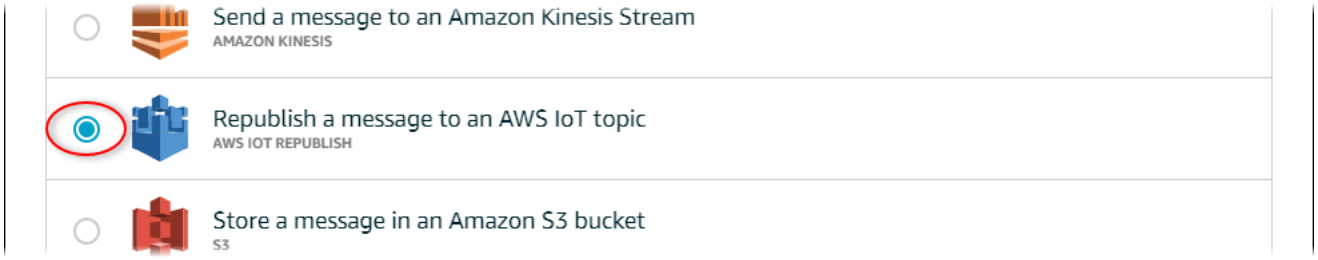
ともできます。詳細については、『AWS IoT 開発者ガイド』の「CloudWatch ログを使った監視」を参照してください。

再パブリッシュエラーアクションをルールに追加するには

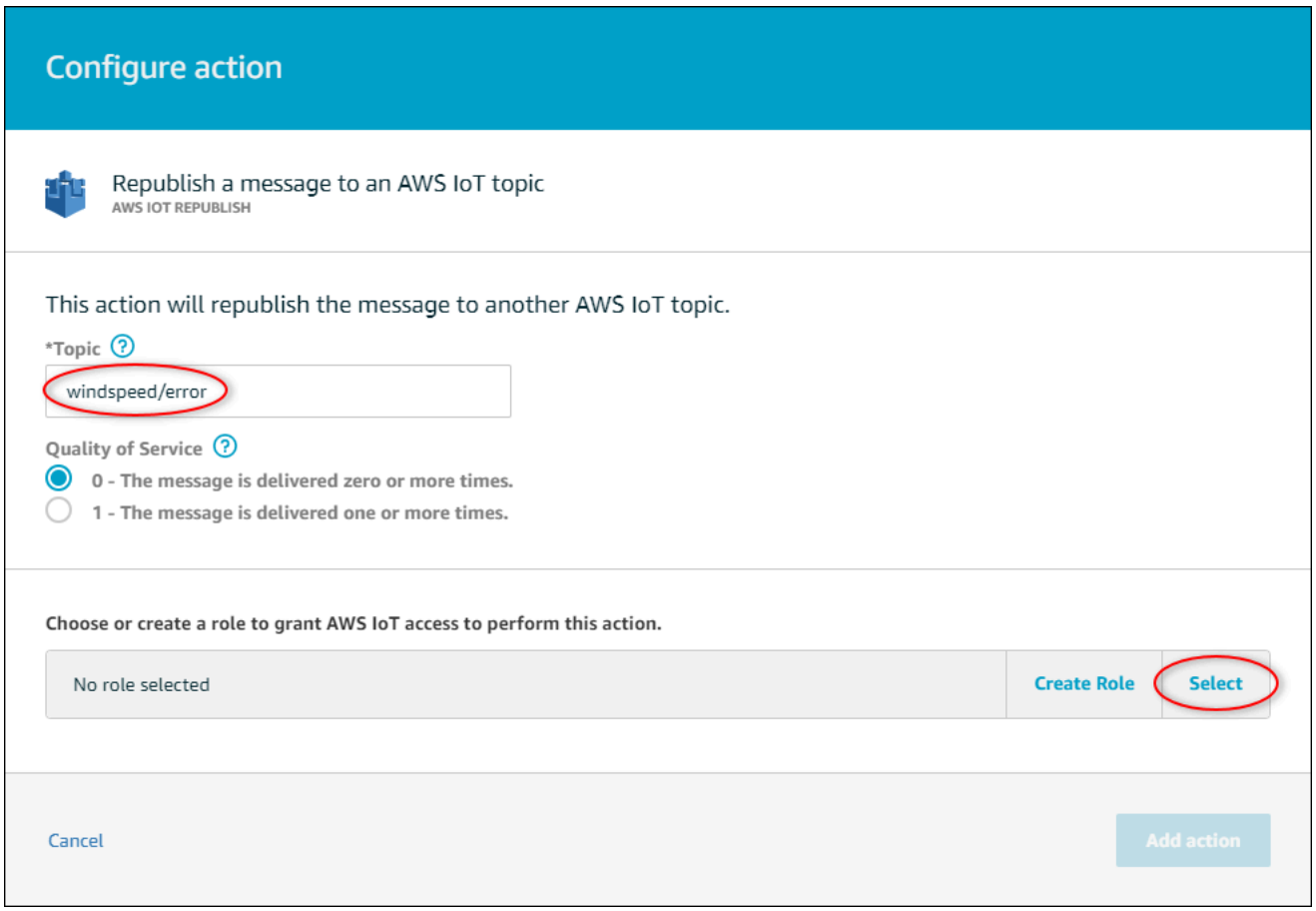
1. AWS IoT コンソールに移動します。
2. 左側のナビゲーションペインで [アクト] を選択し、[ルール] を選択します。
3. 先ほど作成したルールを選択します。



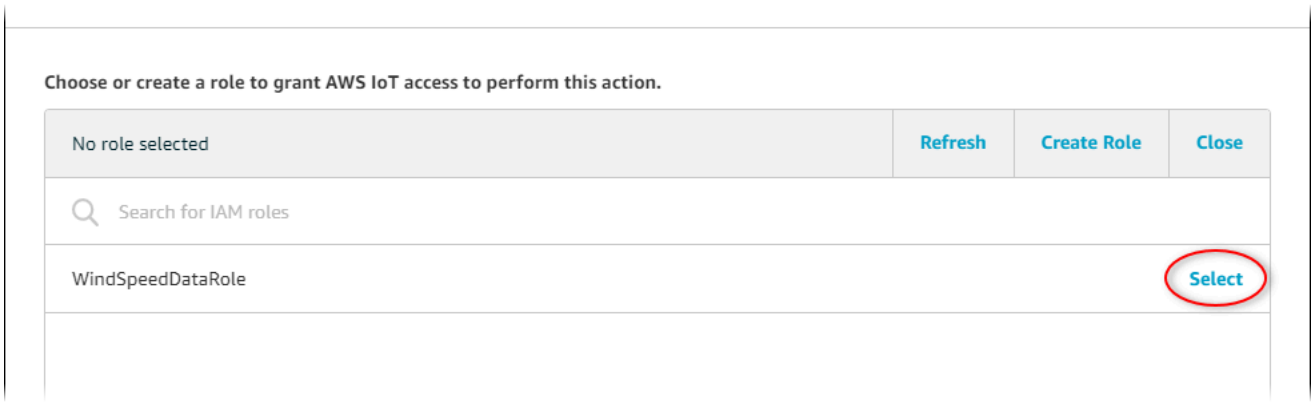
4. [エラーアクション] で、[アクションの追加] を選択します。
5. [AWS IoT メッセージをトピックに再公開] を選択します。



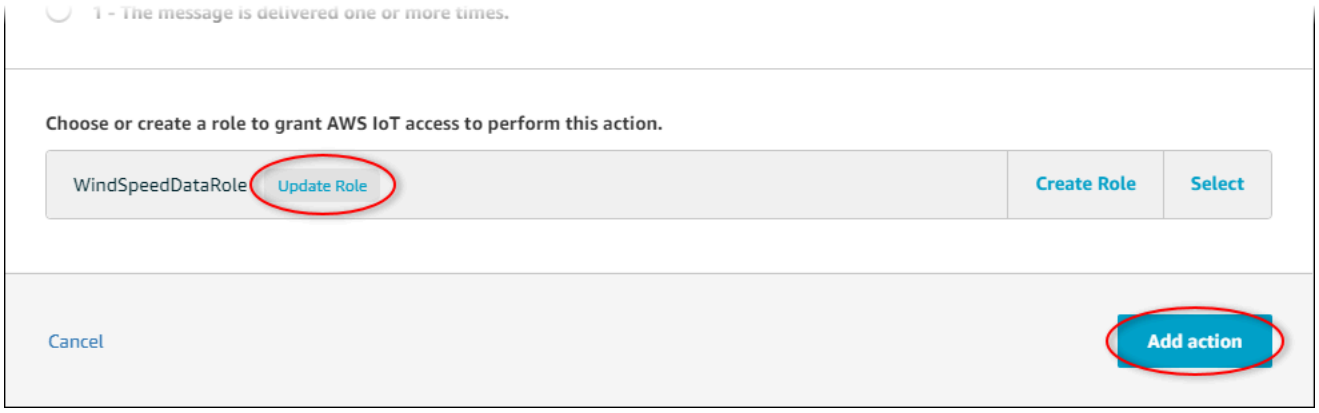
6. ページの下部にある [アクションの設定] を選択します。
7. [Topic] (トピック) で、**`windspeed/error`** と入力します。 AWS IoT Core はエラーメッセージをこのトピックに再公開します。



8. 「選択」を選択すると、前に作成したロールを使用してエラーアクションを実行するためのアクセス権限が AWS IoT Core に付与されます。
9. ロールの横にある [選択] を選択します。



10. [ロールの更新] を選択して、追加のアクセス許可をロールに追加します。



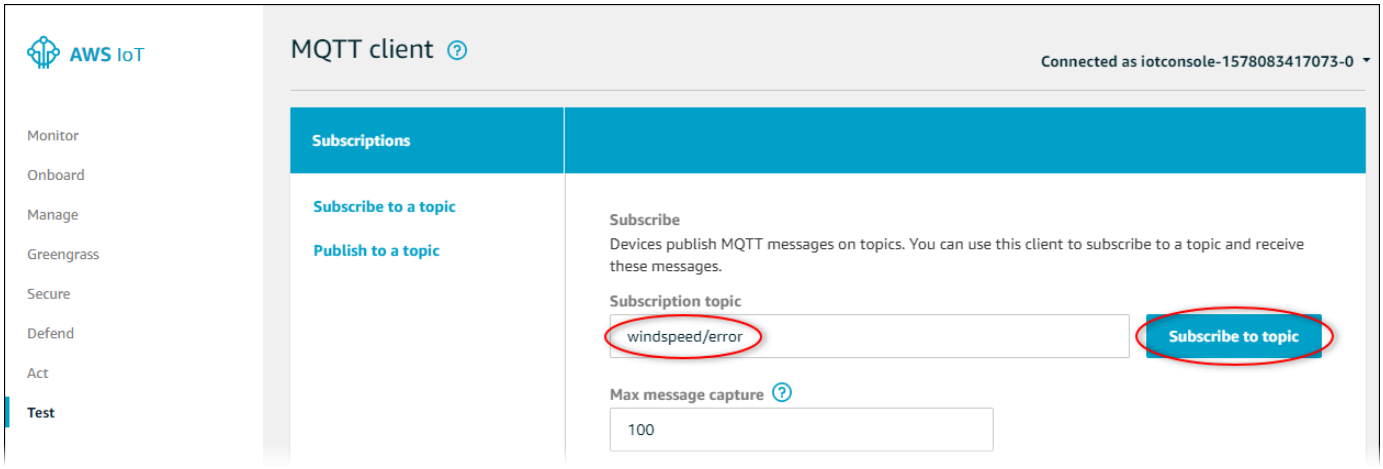
11. [アクションの追加] を選択して、エラーアクションの追加を終了します。
12. コンソールの左上にある戻る矢印を選択して AWS IoT Core コンソールのホームに戻ります。

再公開エラーアクションを設定すると、Core の MQTT AWS IoT テストクライアントにエラーメッセージが表示されます。

次の手順では、MQTT テストクライアントでエラートピックにサブスクライブします。

エラーアクショントピックをサブスクライブするには

1. AWS IoT Core コンソールの左側のナビゲーションページで [Test] を選択します。
2. [トピックのサブスクリプション] フィールドで、**windspeed/error** と入力し、[トピックへのサブスクリプション] を選びます。



3. エラーメッセージが表示されるのを監視し、エラーメッセージで `failures` 配列を調べて、次の一般的な問題を診断します。
   - ルールクエリステートメントのタイプミス
   - ロールのアクセス許可の不足

エラーが表示されない場合は、ルールが有効になっていて、再パブリッシュエラーアクションで設定したトピックと同じトピックにサブスクライブしていることを確認します。それでもエラーが表示されない場合は、デモの風力発電所アセットが存在し、風速プロパティで通知が有効になっているかを確認します。デモアセットの有効期限が切れて消えてしまった場合は AWS IoT SiteWise、新しいデモを作成し、更新されたアセットモデルとプロパティ ID を反映するようにルールクエリステートメントを更新できます。

AWS IoT SiteWise エンドポイントとクォータ

以下のセクションでは、のエンドポイントとクォータについて説明します。 AWS IoT SiteWise

コンテンツ

AWS IoT SiteWise エンドポイント

にプログラムで接続するには AWS IoT SiteWise、エンドポイントを使用します。 AWS SDK と AWS Command Line Interface (AWS CLI) は、リージョンのデフォルトエンドポイントを自動的に使用します。 AWS 利用可能なリージョンの詳細については、の AWS IoT SiteWise 「AWS IoT SiteWise エンドポイントとクォータ」を参照してください。AWS 全般のリファレンス

AWS IoT SiteWise は次のエンドポイントをサポートします。

data.iotsitewise.region.amazonaws.com

このエンドポイントを使用して、データプレーン API オペレーション (BatchPutAssetPropertyValue、、GetAssetPropertyAggregatesGetAssetPropertyValueGetAssetPropertyVal にアクセスします。 *region* AWS お住まいの地域に置き換えてください。

api.iotsitewise.region.amazonaws.com

AWS IoT SiteWise は、アセットモデル、アセット、 SiteWise Edge ゲートウェイ、タグ、アカウント設定の管理に使用するコントロールプレーン API オペレーション用の統合エンドポイントを提供します。 *region* を AWS リージョンに置き換えます。

> **Note**
>
> - デフォルトでは、サポートされているコントロールプレーン API オペレーションを呼び出すときに、 AWS IoT SiteWise 統合エンドポイントを使用します。
> - サポートされるコントロールプレーン API 操作には、統合エンドポイントを使用することを推奨します。

- 統合エンドポイントを使用して SiteWise Monitor API オペレーションにアクセスすることはできません。

サポートされているコントロールプレーン API オペレーションにはAssociateAssets、CreateAsset、CreateAssetModel、DeleteAsset、DeleteAssetModelDeleteDashboard、

コントロールプレーン API 操作のためのインタフェース VPC エンドポイントは，連結エンドポイントのみをサポートします。詳細については、「VPC エンドポイント 」を参照してください。

iotsitewise.region.amazonaws.com

このエンドポイントを使用してDescribeStorageConfiguration、、、、、PutStorageConfigurationDescribeDefaultEncryptionConfiguration よびの API UntagResourceオペレーションにアクセスします。 *region* を AWS リージョンに置き換えます。

model.iotsitewise.region.amazonaws.com

このエンドポイントを使用して、次の API オペレーションにアクセスします:AssociateAssetsCreateAsset, CreateAssetModelDeleteAsset, DeleteAssetModel, DeleteDashboard, DescribeAsset, DescribeAssetModel, DescribeAssetProperty, DescribeDashboard, DescribeLoggingOptions, DisassociateAssetsListAssetModels, ListAssetRelationships, ListAssets, ListAssociatedAssets, PutLoggingOptions, UpdateAsset、UpdateAssetModel、およびUpdateAssetProperty。 *region* AWS お住まいの地域に置き換えてください。

edge.iotsitewise.region.amazonaws.com

このエンドポイントを使用してCreateGateway、、、、、DeleteGatewayDescribeGatewayDescribeGatewayCapabilityConfigurationListG およびの API UpdateGatewayCapabilityConfigurationオペレーションにアクセスします。 *region* AWS お住まいの地域に置き換えてください。

monitor.iotsitewise.region.amazonaws.com

このエンドポイントを使用して、次の API オペレーションにアクセスします:BatchAssociateProjectAssetsBatchDisassociateProjectAssetsCreateAccessPolicy, CreateDashboard, CreatePortal, CreateProjectDeleteAccessPolicy, DeletePortal, DeleteProject, DescribeAccessPolicy, DescribePortal, DescribeProject,

ListAccessPoliciesListDashboards, ListPortals, ListProjectAssets, ListProjects, UpdateAccessPolicy、UpdateDashboard、UpdatePortal、およびUpdateProject。 *region* を AWS リージョンに置き換えます。

AWS IoT SiteWise クォータ

以下の表は、のクォータについて説明しています。 AWS IoT SiteWiseクォータおよびクォータ増加のリクエスト方法の詳細については、『AWS 全般のリファレンス』の 「AWS Service Quotas」を参照してください。 AWS IoT SiteWise クォータの詳細については、の「AWS IoT SiteWise サービスクォータ」を参照してください。AWS 全般のリファレンス

アセットおよびアセットモデルのクォータ

リソース	クォータ	引き上げ可能	メモ
1 アカウントあたりのリージョンごとのアセットモデル数 AWS	1,000	はい	
アセットモデルあたりのアセット数	10,000	はい	
親アセットあたりの子アセット数	2000	はい	
アセットモデル階層ツリーの深さ	30	はい	
アセットモデルあたりの階層定義数	30	はい	
アセットモデルごとのルートレベルのプロパティ数。	500	はい	各アセットモデルのこの最大数。assetModelProperties この数には含まれませんcompositeModelProperties 。この

リソース	クォータ	引き上げ可能	メモ
			クォータは、このアセットモデルから作成されたすべてのユニークアセットにも適用されます。
アセットモデルあたりのプロパティ数	5000	はい	ASSET_MODEL COMPONENT_MODEL タイプがまたはのアセットモデルのプロパティの最大数。この数は、component-model-based ルートアセットモデルとインクルードまたはインラインコンポジットモデルのプロパティを組み合わせて決定されます。このクォータは、このアセットモデルから作成されたすべてのユニークアセットにも適用されます。
複合モデルあたりのプロパティ数	100	はい	複合モデルに許容されるプロパティの最大数。また、1 COMPONENT_MODEL つのタイプのアセットモデルに許容されるプロパティの最大数です。

リソース	クォータ	引き上げ可能	メモ
アセットモデルあたりのプロパティツリーの深さ	10	[いいえ]	たとえば、測定プロパティ A を使用する変換プロパティ B を使用する変換プロパティ C を持つモデルの深さは、3 です。
階層ツリーあたりのアセットモデル数	100	はい	
アセットモデルあたりの直接依存プロパティ数	20	[いいえ]	このクォータは、プロパティ式の定義に従って、1 つのプロパティに直接依存できるプロパティの数を制限します。アセットモデルに対する従属プロパティの数は、アセットモデル 1 つあたりの直接依存プロパティの数より大きくなければなりません。アセットモデルに対する直接従属プロパティの上限数が、アセットモデル 1 つあたりの依存プロパティの上限数より大きい場合は、両方についてクォータ増加をリクエストすることができます。

リソース	クォータ	引き上げ可能	メモ
アセットモデルあたりの依存プロパティ数	30	[いいえ]	このクォータは、プロパティ式の定義に従って、1 つのプロパティに直接または間接的に依存できるプロパティの数を制限します。
アセットモデルあたりの複合モデル数	50	はい	1 つの資産モデルで許容される複合モデルの最大数。
複合モデルの奥行き	2	はい	インラインモデルと複合モデルを含む、component-model-based アセットモデルごとの複合モデルツリーの最大深度。
同じコンポーネントモデルを使用する独自のアセットモデルの数	20	はい	COMPONENT_MODEL component-model-based タイプの特定のアセットモデルを直接参照する複合モデルを少なくとも 1 つ含むユニークアセットモデルの最大数。

リソース	クォータ	引き上げ可能	メモ
プロパティ式あたりのプロパティ変数の数	10	[いいえ]	たとえば、式 `avg(power) + max(temp)` には `temp`および `power`の 2 つのプロパティ変数があります。これは変換の計算結果にも当てはまります。
プロパティ式ごとの関数の数	10	[いいえ]	たとえば、式 `avg(power) + max(temp)` には、`avg`および`max` という 2 つの関数があります。

アセットプロパティデータのクォータ

リソース	クォータ	引き上げ可能	メモ
アセットプロパティデータ API オペレーションのリクエストレート	1 アカウント、1 リージョンあたり 1 秒あたり 1000 リクエスト AWS	はい	このクォータは、`GetAssetPropertyValue` や `BatchPutAssetPropertyValue` などの API オペレーションに適用されます。
アセットプロパティ、データ品質ごとの 1 秒あたりのデータポイント数	10 データポイント	[いいえ]	このクォータは、各アセットプロパティのデータ品質あたりのタイムスタンプ timestamp-quality-value (秒単位) が同

リソース	クォータ	引き上げ可能	メモ
			じデータポイントの最大数 (TQV) に適用されます。各アセットプロパティについて、任意の秒に対して、品質適合、品質不明、および品質不良のデータポイントをこの数まで保存できます。
1 アカウントあたりのリージョンごとのアセットプロパティごとに 1 BatchPutAssetPropertyValue 秒あたりに取り込まれるエントリ数。 AWS	1 アセットプロパティあたり 10 エントリ	[いいえ]	このクォータは、SiteWise Edge ゲートウェイ、AWS IoT Core ルール、API BatchPutAssetPropertyValue コールを含むすべてのソースからのエントリに適用されます。
取り込まれたデータポイントの割合	1 アカウント、1 リージョンあたり 5000 データポイント/秒 AWS	はい	T imestamp-quality-value (TQV) データポイント。
BatchGetAssetPropertyAggregates のリクエストレート	200	はい	現在のリージョンで、このアカウントで実行できる1秒あたりの BatchGetAssetPropertyAggregates リクエストの最大数です。

リソース	クォータ	引き上げ可能	メモ
BatchGetAssetPropertyValue のリクエストレート	500	はい	現在のリージョンで、このアカウントで実行できる1秒あたりの BatchGetAssetPropertyValue リクエストの最大数です。
BatchGetAssetPropertyValueHistory のリクエストレート	200	はい	現在のリージョンで、このアカウントで実行できる1秒あたりの BatchGetAssetPropertyValueHistory リクエストの最大数です。
1 アカウント、1 リージョンあたりのアセットプロパティごとに 1 BatchPutAssetPropertyValue 秒あたりに取り込まれるエントリ数。 AWS	1 アセットプロパティあたり 10 エントリ	[いいえ]	このクォータは、SiteWise Edge ゲートウェイ、AWS IoT Core ルール、API BatchPutAssetPropertyValue コールを含むすべてのソースからのエントリに適用されます。

リソース	クォータ	引き上げ可能	メモ
アセットプロパティあたりの GetAssetPropertyAggregates リクエストと BatchGetAssetPropertyAggregates エントリクエリレート	50	[いいえ]	このアカウントでの現在のリージョンにおいて実行できる 1 秒あたりの GetAssetPropertyAggregates リクエストと BatchGetAssetPropertyAggregates エントリの合計最大数。
アセットプロパティあたりの GetAssetPropertyValue リクエストと BatchGetAssetPropertyValue エントリクエリレート	500	[いいえ]	このアカウントでの現在のリージョンにおいて実行できる 1 秒あたりの GetAssetPropertyValue リクエストと BatchGetAssetPropertyValue エントリの合計最大数。

リソース	クォータ	引き上げ可能	メモ
アセットプロパティあたりの GetAssetPropertyValueHistory リクエストと BatchGetAssetPropertyValueHistory エントリクエリレート	30	[いいえ]	このアカウントでの現在のリージョンにおいて実行できる 1 秒あたりの GetAssetPropertyValueHistory リクエストと BatchGetAssetPropertyValueHistory エントリの合計最大数。
GetInterpolatedAssetPropertyValues リクエストのレート。	500	はい	現在のリージョンで、このアカウントで実行できる1秒あたりの GetInterpolatedAssetPropertyValues リクエストの最大数です。
GetInterpolatedAssetPropertyValues リクエストあたりの結果数。	10	はい	ページ割りされた GetInterpolatedAssetPropertyValues リクエストごとに返す結果の最大数。

リソース	クォータ	引き上げ可能	メモ
GetAssetPropertyValueHistory および BatchGetAssetPropertyValueHistory から取得するデータポイントレート	1 アカウント 1 リージョンあたり 1 秒あたり 100 MB の読み取り応答。 AWS	はい	とでの 1 アカウントあたりのリージョンごとに 1 秒あたりに取得されるデータポイントの最大バイトレート (MB/秒)。 AWS GetAssetPropertyValueHistory BatchGetAssetPropertyValueHistory このクォータで評価されるレスポンスペイロードでは、各データポイントの Timestamp-Quality-Value (TQV) フィールドを使用しており、各 API リクエストのバイトサイズは次の 4 KB 単位で四捨五入されます。 1 秒あたりに取得される T imestamp-quality-value (TQV) データポイントは、データタイプによって異なります。 • 整数 – 1 秒あたり最大 500 万 TQV • ダブル - 1 秒あたり最大 400 万 TQV

リソース	クォータ	引き上げ可能	メモ
			• ブール値 - 1 秒あたり最大 600 万 TQV • 文字列 – 各文字列の値のサイズによって異なる。

SiteWise Edge ゲートウェイのクォータ

リソース	クォータ	調整可能
1 アカウントあたりのリージョンごとの SiteWise Edge ゲートウェイ数 AWS	100	はい
Edge ゲートウェイあたりの OPC-UA ソースの数 SiteWise	100	[いいえ]

のクォータ AWS IoT SiteWise Monitor

リソース	クォータ	調整可能
1 アカウントあたりのリージョンあたりのポータル数 AWS	100	はい
ポータルあたりのプロジェクト数	100	はい
プロジェクトあたりのダッシュボード数	100	はい
プロジェクトあたりのルートアセット数	1	いいえ
ダッシュボードあたりの可視化数	10	はい

リソース	クォータ	調整可能
ダッシュボードによる可視化あたりのメトリクス数	5	はい
ダッシュボードによる視覚化あたりのしきい値数	12	[いいえ]

AWS IoT SiteWise メタデータの一括インポートとエクスポートのクォータ

リソース	説明	Quota	調整可能
キュー内のメタデータ転送ジョブの数。	PENDINGキュー内のメタデータ転送ジョブの最大数。	10	[Yes (はい)]
メタデータ転送ジョブのインポートファイルのサイズ。	インポートされるファイルの最大サイズ (MB 単位)。	100 MB	はい
AWS IoT SiteWise メタデータ転送ジョブのリソースクォータ	1 つのジョブでインポートまたはエクスポートされるリソースの最大数。リソースにはアセットとアセットモデルが含まれます。	5000	[いいえ]

AWS IoT SiteWise データの一括インポートのクォータ

リソース	クォータ	調整可能
実行中の一括インポートジョブの数。	100	[いいえ]
CSV ファイルのサイズ。	10 GB	[いいえ]

リソース	クォータ	調整可能
圧縮されていない寄木細工ファイルのサイズ。	256 MB	[いいえ]
CSVバッファリングによる取り込み用のファイルのサイズ。	256 MB	[いいえ]
圧縮されていない寄木細工の行グループのサイズ。	64 MB	[いいえ]
寄木細工の列グループごとのユニークな測定値の数。	2000	はい
過去のタイムスタンプから今日のタイムスタンプまでのバッファ取り込み日数	30	はい
`CreateBulkImportJobs` 各アカウントの各リージョンのリクエストレート AWS	10	[Yes (はい)]
`ListBulkImportJobs` AWS 各アカウントのリージョンごとのリクエストレートです。	50	はい
`DescribeBulkImportJobs` AWS 各アカウントのリージョンごとのリクエストレートです。	50	はい

異常検出のクォータ

異常検出のクォータは Amazon Lookout for Equipment AWS IoT SiteWise との間で共有されます。詳細については、「Lookout for Equipment を使用するためのクォータ」を参照してください。

AWS IoT SiteWise ユーザーガイドのドキュメント履歴

次の表は、 の今回のリリースの内容をまとめたものです AWS IoT SiteWise。

- API バージョン: 2019-12-02

変更	説明	日付
TAK 産業用 SiteWise エッジで Edge を実行するためのサポートを追加	AWS IoT SiteWise は、TAK Industrial SiteWise Edge デバイスでの Edge の実行をサポートするようになりました。	2023 年 11 月 26 日
ウォーム階層ストレージのサポートを追加	AWS IoT SiteWise は、お客様が産業データを安全に保存およびアクセスできるようにするフルマネージドストレージ階層であるウォームストレージをサポートするようになりました。	2023 年 11 月 15 日
ユーザー定義の一意の識別子のサポートを追加	AWS IoT SiteWise では、アセット、アセットモデル、プロパティ、階層のユーザー定義の一意の識別子の使用がサポートされるようになりました。	2023 年 11 月 15 日
産業アセットの多変量異常検出のサポートを追加	AWS IoT SiteWise は、履歴およびリアルタイムの機器データを Amazon Lookout for Equipment と統合することで、産業アセットの多変量異常検出をサポートするようになりました。	2023 年 11 月 15 日

での時系列データのコスト効率とスケーラビリティに優れた取り込みのサポートを追加 AWS IoT SiteWise	AWS IoT SiteWise は、分析ユースケースに必要な時系列データのコスト効率とスケーラビリティに優れた取り込みをサポートするようになりました。	2023 年 11 月 15 日
一括インポート、エクスポート、更新のサポートを追加	AWS IoT SiteWise は、産業機器メタデータの一括インポート、エクスポート、更新をサポートするようになりました。	2023 年 11 月 15 日
アセットモデルコンポーネントのサポートを追加	AWS IoT SiteWise は、産業業界のお客様が再利用可能なコンポーネントを作成できるように、アセットモデルコンポーネントをサポートするようになりました。	2023 年 11 月 15 日
IoT ダッシュボードアプリケーションのサポートを追加	AWS IoT SiteWise は、運用データを視覚化して操作できるオープンソースのダッシュボードアプリケーションをサポートするようになりました。	2023 年 11 月 15 日
のサービスにリンクされたロールを更新しました AWS IoT SiteWise	AWS IoT SiteWise には新しいサービスにリンクされたロールがあり、 AWS IoT TwinMaker データベースに対してメタデータ検索クエリを実行できます。	2023 年 11 月 6 日
AWS IoT SiteWise データストリームリソースのタグ付けを更新	データストリームリソースのタグ付けに対するサポートを更新。	2022 年 8 月 18 日

SiteWise エッジゲートウェイの更新	パブリッシャーの設定で、エッジからクラウドに送信されるデータと、データをクラウドに送信する順序を制御できるようになりました。	2022 年 1 月 12 日
AWS IoT SiteWise デモを更新しました	これで、デモを使用して SiteWise Monitor ポータルを作成できるようになりました。	2022 年 1 月 10 日
ストレージ管理を更新しました	保存期間を定義することで、ホット階層にデータを保存する期間を制御できるようになりました。	2021 年 11 月 29 日
データストリーム管理のサポートが追加されました	アセットモデルとアセットを作成する AWS IoT SiteWise 前に、 にデータを取り込むことができるようになりました。	2021 年 11 月 24 日
アセットモデル階層を更新しました	子アセットモデルを複数の親アセットモデルに関連付けることができるようになりました。	2021 年 10 月 28 日
リージョンへの参入	が AWS GovCloud （米国西部) AWS IoT SiteWise で起動されました。	2021 年 9 月 29 日

関数を更新しました	次の機能が追加されました。 • メトリクスでは、[aggregation functions] (集計関数) や[temporal functions] (一時関数) でネスト表現を使用することができます。 • 変換では、[pretrigger () function] (pretrigger () function 関数) を使用して、現在の変換コンピューティングのトリガーとなったプロパティ更新前の変数値を取得することができます。	2021 年 8 月 10 日
カスタムメトリクスの時間間隔	メトリクスの時間間隔とオフセットのカスタムに対応しました。	2021 年 8 月 3 日
エッジ AWS IoT SiteWise での の使用	エッジ処理機能の一般提供を開始しました。	2021 年 7 月 29 日
Amazon S3 へのデータエクスポート	AWS IoT SiteWise が Amazon S3 にデータをエクスポートできるようになりました。	2021 年 7 月 27 日
VPC エンドポイント (AWS PrivateLink)	コントロールプレーン API 操作のためのインターフェース VPC エンドポイントの一般提供を開始しました。	2021 年 7 月 15 日
変換	変換で、複数のアセットプロパティ変数を入力できるようになりました。	2021 年 7 月 8 日

timestamp () 関数を更新しました	変換では、timestamp( ) 関数の引数として変数を与えることができるようになりました。	2021年6月16日
アラームの一般提供	アラーム機能は一般公開されています。	2021 年 5 月 27 日
Modbus-TCP プロトコル・アダプター バージョン 2 をリリース	Modbus-TCP プロトコルアダプターコネクター のバージョン 2 が利用可能です。このリリースでは、ASCII、UTF8、および ISO8859 エンコードされたソース文字列のサポートのサポートが追加されました。	2021 年 5 月 24 日
サービスクォータを更新	GetInterpolatedAssetPropertyValues API に次のクォータを追加しました: GetInterpolatedAssetPropertyValues リクエストのレート、GetInterpolatedAssetPropertyValues リクエストごとの結果の数、の開始日から今日までの日数GetInterpolatedAssetPropertyValues 。	2021 年 4 月 29 日

数式を更新しました	次の演算子、関数を追加しました。 • 次の [operators] (演算子) を追加しました: `<`、`>`、`<=`、`>=`、`==`、`!=`、`!`、`and`、`or`、および `not`。 • 次の [comparison function] (比較機能) を追加しました: `neq(x, y)`。 • 次の [string functions] (文字列関数) を追加しました: `join()`、`format()`、および `f''`。	2021 年 4 月 22 日
VPC エンドポイント (AWS PrivateLink)	インターフェイス VPC エンドポイントを作成して、Virtual Private Cloud (VPC) と AWS IoT SiteWise コントロールプレーン APIs の間にプライベート接続を確立する方法に関する情報を追加しました。	2021 年 3 月 16 日
IAM フェデレーション	SiteWise Monitor ポータル管理者とユーザーは、IAM 認証情報を使用して割り当てられたポータルにログインできるようになりました。	2021 年 3 月 16 日
リージョンへの参入	が中国 (北京) AWS IoT SiteWise で利用可能になりました。	2021 年 2 月 3 日

IoT SiteWise コネクタバージョン 10 のリリース	IoT SiteWise コネクタのバージョン 10 が利用可能になりました。本リリースでは、送信元接続が失われ、再確立された場合の処理を改善するよう `StreamManager` を設定しました。本バージョンでは、`SourceTimestamp` がない場合に `ServerTimestamp` で OPC-UA 値を受け付けることも可能です。	2021 年 1 月 22 日
日付および時刻関数	AWS IoT SiteWise で日付と時刻の関数がサポートされるようになりました。	2021 年 1 月 21 日
関数構文	AWS IoT SiteWise 関数に統一関数呼び出し構文 (UFCS) を使用できるようになりました。	2021 年 1 月 11 日
Grafana との統合	Grafana ダッシュボードで AWS IoT SiteWise データを視覚化する方法についての情報を追加しました。	2020 年 12 月 15 日

AWS IoT SiteWise 機能のリリース	アラームによるデータのモニタリング、エッジでの産業用データの処理、 SiteWise エッジゲートウェイへの Modbus TCP およびイーサネット/IP ソースの使用、デッドバンドによる受信データのフィルタリングなどが可能になりました。 • AWS IoT SiteWiseでのアラームの定義、設定、対応に使用できる [Monitoring data with alarms] (アラーム付きモニタリングデータ) セクションを追加しました。 • エッジデバイスで産業用データの処理を設定するために使用できる、[Edge processing] (エッジ処理) セクションを追加しました。 • Edge ゲートウェイのソースドキュメントに Modbus TCP およびイーサネット/IP SiteWise セクションを追加しました。 • 受信した産業用データの [source destination] (送信先) をカスタマイズするために使用できる送信先セクションを追加しました。 • OPC-UA フィルタリングセクションを追加しました。これを使用して、産業ローカルサーバーから SiteWise Edge ゲートウェイに送信さ	2020 年 12 月 15 日

	れるデータの頻度とタイプを制御できます。	
AWS IoT SiteWise がカスタマー管理の CMKsをサポートするようになりました。	AWS IoT SiteWise は、カスタマー管理CMKs による暗号化をサポートするようになりました。	2020 年 11 月 24 日
IoT SiteWise コネクタバージョン 8 のリリース	IoT SiteWise コネクタのバージョン 8 を利用できます。このリリースでは、コネクターのネットワーク接続が断続的に発生する場合の安定性を改善しました。	2020 年 11 月 19 日
数式で文字列と条件式を使用する	変換とメトリクスの数式式に文字列と条件関数を使用する方法に関する情報を追加しました。	2020 年 11 月 16 日
AWS IoT Greengrass ストリームマネージャーを使用したデータの取り込み	AWS IoT Greengrass エッジデバイスを使用してローカルデータソースから大量の IoT データを取り込む方法に関する情報を追加しました。	2020 年 9 月 16 日
VPC エンドポイント (AWS PrivateLink)	インターフェイス VPC エンドポイントを作成して、Virtual Private Cloud (VPC) と AWS IoT SiteWise データ APIs の間にプライベート接続を確立する方法に関する情報を追加しました。	2020 年 9 月 4 日

IoT SiteWise コネクタバージョン 7 のリリース	IoT SiteWise コネクタのバージョン 7 を利用できます。このリリースでは、 SiteWise エッジゲートウェイメトリクスの問題が修正されています。	2020 年 8 月 14 日
AWS IoT SiteWise コンソールからの IAM Identity Center ユーザーの作成	AWS IoT SiteWise コンソールで IAM Identity Center ユーザーを作成する方法に関する情報を追加しました。新規または既存のポータルにユーザーを割り当てる際に、IAM Identity Center ユーザーを作成することができるようになりました。[Visualizing and sharing wind farm data] (風力発電施設データの視覚化と共有) のチュートリアルを更新しました。この変更により、チュートリアルのステップ数が減少しました。	2020 年 8 月 4 日
SiteWise Edge ゲートウェイのトラブルシューティングの改善	SiteWise Edge ゲートウェイのトラブルシューティング方法と、ソースの OPC-UA クライアント証明書をエクスポートする方法に関する追加情報を追加しました。	2020 年 6 月 18 日

コンソールタスクのドキュメント	産業用アセットのモデリング、アセットプロパティデータのクエリ、その他のサービスの操作に関するコンソールタスクのドキュメントが追加されました。次の手順に従って、 AWS IoT SiteWise コンソールでタスクを完了できます。	2020 年 6 月 11 日
エクスポートされたデータチュートリアルの分析	Amazon Athena を使用して、エクスポート機能 AWS CloudFormation テンプレートで S3 にエクスポートしたアセットデータを分析するためのチュートリアルを追加しました。	2020 年 5 月 27 日
数式の使用の改善	AWS IoT SiteWise 式プロパティの動作に関する詳細情報を追加し、フィルタリングされたデータポイントをカウントする方法の例を追加しました。	2020 年 5 月 18 日

IoT SiteWise コネクタバージョン 6 のリリース	IoT SiteWise コネクタのバージョン 6 を利用できます。このリリースでは、 CloudWatch メトリクスのサポートと新しい OPC-UA タグの自動検出が追加されました。つまり、OPC-UA ソースのタグが変更された場合は、 SiteWise エッジゲートウェイを再起動する必要はありません。このバージョンのコネクタには、ストリームマネージャーと AWS IoT Greengrass Core ソフトウェア v1.10.0 以降が必要です。	2020 年 4 月 29 日

AWS IoT SiteWise 機能のリリース	AWS IoT SiteWise 機能のリリース。API を使用した SiteWise Edge ゲートウェイの管理、ポータルへの口ゴの追加、 SiteWise Edge ゲートウェイメトリクスの表示などを実行できるようになりました。 • 新しいデータ値を Amazon S3バケットにエクスポートするために使用できる テンプレートを含む「Amazon S3 へのデータのエクスポート」セクションを追加しました。 AWS CloudFormation • Edge ゲートウェイのソースドキュメントを改善し、新しい Edge ゲートウェイ API を含む「データソースの設定APIs 」セクションを追加しました。 SiteWise SiteWise • SiteWise Edge ゲートウェイが公開するメトリクスについて説明する SiteWise「エッジゲートウェイ CloudWatch メトリクス」セクションを追加しました。 • テンプレートを使用して Amazon EC2 で SiteWise Edge ゲートウェイを設定するセクションを追加しました。 AWS CloudFormation これを使用すると、	2020 年 4 月 29 日

	Amazon EC2 インスタンスで SiteWise Edge ゲートウェイの依存関係をすばやく設定できます。 • SiteWise Monitor ポータルの新しいアクセス許可機能について説明するポータルサービスロールセクションを追加しました。 • ポータルサービスロールとポータルロゴについて「ポータルドキュメント」を更新しました。 • AWS IoT SiteWise リソースにタグを付けるセクションを追加しました。 • 新しいダッシュボード定義構造について「ダッシュボードの作成 (CLI) 」セクションを更新しました。 • 「セキュリティ」セクションを追加しました。	
からのデータの取り込み AWS IoT Events	イベントが発生した AWS IoT Events ときに からデータを取り込む方法に関する情報を追加しました。	2020 年 4 月 20 日
SiteWise Monitor チュートリアルの「風力発電施設データの視覚化と共有」	アセットデータの視覚化と共有に を使用する方法を学習 AWS IoT SiteWise Monitor するためのチュートリアルを追加しました。	2020 年 3 月 12 日

AWS IoT SiteWise の概念	サービスとその一般的な用語について学習するために使用できる AWS IoT SiteWise 概念の用語集を追加しました。	2020 年 3 月 5 日
削除された AWS IoT Greengrass インストール手順	AWS IoT SiteWise ユーザーガイドから AWS IoT Greengrass Core ソフトウェアのインストール手順を削除しました。AWS IoT Greengrass デベロッパーガイドには、Amazon EC2 や Docker などの他のプラットフォーム AWS IoT Greengrass で をセットアップするためのデバイスセットアップスクリプトと手順が記載されています。	2020 年 2 月 14 日
AWS IoT Core ルールを使用したデータの取り込みが改善されました	の使用方法と AWS IoT SiteWise ルールアクションのトラブルシューティング方法に関する詳細情報を追加しました。これを使用して、 を介して MQTT メッセージからデータを取り込むことができます AWS IoT Core。	2020 年 2 月 14 日
IoT SiteWise コネクタバージョン 5 のリリース	IoT SiteWise コネクタのバージョン 5 を利用できます。このリリースでは、 AWS IoT Greengrass Core ソフトウェア v1.9.4 との互換性の問題が修正されています。	2020 年 2 月 12 日

IoT SiteWise コネクタバージョン 4 のリリース	IoT SiteWise コネクタのバージョン 4 を利用できます。このリリースでは、OPC-UA サーバーの再接続に関する問題を修正します。	2020 年 2 月 7 日
産業用アセットのモデリングを再編成	「アセットとモデルの更新」セクションを、産業用アセットのモデリング内の複数のトピックに再編成しました。 • アセットおよびモデルの状態 • アセットプロパティへの産業データストリームのマッピング • 属性値の更新 • アセットの関連付けと関連付け解除 • アセットとモデルの更新 • アセットとモデルの削除	2020 年 2 月 4 日
AWS IoT モノのチュートリアルからのデータの取り込み	新規または既存の AWS IoT モノのフリートからデータを取り込むように AWS IoT SiteWise ルールアクションを設定する方法を学ぶためのチュートリアルを追加しました。	2020 年 2 月 4 日
からのデータ取得の再構成 AWS IoT SiteWise	「データの取得」セクションを 2 つの最上位セクションに再構成しました。アセットプロパティ値のクエリと集計、他の AWS サービスとの対話です。	2020 年 1 月 21 日

プロパティ値の更新を Amazon DynamoDB チュートリアルにパブリッシュする	プロパティ値通知を使用してアセットデータを DynamoDB に格納する方法を学習するためのチュートリアルが追加されました。	2020 年 1 月 8 日
数式の使用	変換プロパティとメトリクスプロパティで使用できる定数と関数を整理するための数式参照を追加しました。「アセットプロパティ」をプロパティタイプごとの個別のトピックに再編成しました。	2020 年 1 月 7 日
OPC-UA ノードフィルターの使用	OPC-UA ノードフィルターを使用して、 SiteWise エッジゲートウェイソースを追加するときに SiteWise エッジゲートウェイのパフォーマンスを向上させる方法に関する情報を追加しました。	2020 年 1 月 3 日
コネクタのアップグレード	新しいコネクタバージョンがリリースされたときに SiteWise Edge ゲートウェイをアップグレードする方法についての情報を追加しました。	2019 年 12 月 30 日
IoT SiteWise コネクタバージョン 3 のリリース	IoT SiteWise コネクタのバージョン 3 を利用できます。このリリースでは、iot: * アクセス許可の要件が削除されました。	2019 年 12 月 17 日

IoT SiteWise コネクタバージョン 2 のリリース	IoT SiteWise コネクタのバージョン 2 を利用できます。このリリースでは、複数の OPC-UA シークレットリソースのサポートを追加しました。	2019 年 12 月 10 日
ダッシュボードの作成 (AWS CLI)	AWS IoT SiteWise Monitor を使用して でダッシュボードを作成する方法についての情報を追加しました AWS CLI。	2019 年 12 月 6 日

AWS IoT SiteWise バージョン 2 のリリース	のバージョン 2 のプレビューをリリースしました AWS IoT SiteWise。Monitor で、OPC-UA、MQTT、HTTP 経由でデータを取り込んで、アセット階層にデータをモデル化し、データを視覚化できるようになりました SiteWise 。 • アセット、アセットモデル、およびアセット階層の変更について、「アセットモデリング」セクションを書き直しました。 • データインジェストセクションを更新して、 AWS IoT Greengrass コネクタステップと非ゲートウェイデータインジェストセクションを含めました。 • SiteWise Monitor ウェブアプリケーションの使用方法を示すAWS IoT SiteWise Monitorセクションと、別のアプリケーションガイドを追加しました。 https://docs.aws.amazon.com/iot-sitewise/latest/appguide/ • 「からのデータのクエリ AWS IoT SiteWise」セクションと「AWS 他のサービスとのやり取り」セクションを追加しました。	2019 年 12 月 2 日

	• 更新されたデモ体験に合わせて、「開始方法」セクションを書き直しました。	
AWS IoT SiteWise バージョン 1 のリリース	のバージョン 1 の初期プレビューをリリースしました AWS IoT SiteWise。	2019 年 2 月 25 日

AWS 用語集

AWS の最新の用語については、「AWS の用語集リファレンス」の「AWS 用語集」を参照してください。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。