Entwicklerhandbuch

AWS IoT FleetWise

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS IoT FleetWise: Entwicklerhandbuch

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, durch die Kunden irregeführt werden könnten oder Amazon in schlechtem Licht dargestellt oder diskreditiert werden könnte. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist AWS IoT FleetWise? 1
Vorteile 2
Anwendungsfälle 3
Wichtiger Hinweis 4
Sind Sie neu im AWS Internet der Dinge FleetWise? 4
Zugriff auf AWS IoT FleetWise 4
Preise für AWS IoT FleetWise 5
Zugehörige Services 5
Die wichtigsten Konzepte 5
Die wichtigsten Konzepte 6
Funktionen von AWS IoT FleetWise 11
Unterstützte AWS Regionen 11
AWS IoT einrichten FleetWise 14
Richten Sie Ihre ein AWS-Konto 14
Melden Sie sich an für ein AWS-Konto 14
Erstellen eines Benutzers mit Administratorzugriff 15
Erste Schritte in der Konsole 16
Konfiguriere deine Einstellungen 17
Konfigurieren der Einstellungen (Konsole) 17
Einstellungen konfigurieren (AWS CLI) 18
Verwendung IPv6 mit AWS IoT FleetWise 20
IPv6 Voraussetzungen für Endpunkte der Steuerungsebene 20
IPv6 Unterstützung für Endgeräte AWS PrivateLink 20
Die Adresskompatibilität wird getestet IPv6 20
IPv6 Adressen in IAM-Richtlinien verwenden 21
Verwenden von Dual-Stack-Endpunkten 22
Erste Schritte 24
Einführung 24
Voraussetzungen 25
Schritt 1: Richten Sie die Edge Agent-Software für AWS IoT ein FleetWise 26
Schritt 2: Erstellen Sie ein Fahrzeugmodell 27
Schritt 3: Erstellen Sie ein Decoder-Manifest 29
Schritt 4: Konfigurieren Sie ein Decoder-Manifest 30
Schritt 5: Erstellen Sie ein Fahrzeug 31

Schritt 6: Erstelle eine Kampagne ... 32
Schritt 7: Bereinigen ... 34
Nächste Schritte ... 34
Daten aufnehmen ... 35
Fahrzeuge modellieren ... 39
Signalkataloge ... 42
Signale konfigurieren ... 45
Erstellen Sie einen Signalkatalog ... 52
Importieren Sie einen Signalkatalog ... 57
Aktualisieren Sie einen Signalkatalog ... 67
Löschen Sie einen Signalkatalog ... 70
Holen Sie sich Informationen zum Signalkatalog ... 72
Fahrzeugmodelle ... 73
Erstellen Sie ein Fahrzeugmodell ... 74
Aktualisieren Sie ein Fahrzeugmodell ... 81
Löschen Sie ein Fahrzeugmodell ... 83
Holen Sie sich Informationen zum Fahrzeugmodell ... 85
Decoder-Manifestspiele ... 86
Konfigurieren Sie Schnittstellen und Signale ... 88
Erstellen Sie ein Decoder-Manifest ... 92
Aktualisieren Sie ein Decoder-Manifest ... 102
Löschen Sie ein Decoder-Manifest ... 105
Rufen Sie Informationen zum Decoder-Manifest ab ... 107
Fahrzeuge verwalten ... 109
Fahrzeuge bereitstellen ... 110
Fahrzeuge authentifizieren ... 111
Fahrzeuge autorisieren ... 113
Reservierte Themen ... 114
Erstellen Sie ein Fahrzeug ... 120
Erstellen Sie ein Fahrzeug (Konsole) ... 120
Erstellen Sie ein Fahrzeug (AWS CLI) ... 122
Erstellen Sie mehrere Fahrzeuge ... 126
Aktualisieren Sie ein Fahrzeug ... 127
Aktualisieren Sie mehrere Fahrzeuge ... 130
Löschen Sie ein Fahrzeug ... 132
Lösche ein Fahrzeug (Konsole) ... 132

Lösche ein Fahrzeug (AWS CLI) ... 132
Holen Sie sich Fahrzeuginformationen ... 134
Flotten verwalten ... 136
Erstellen einer Flotte ... 137
Ordnen Sie ein Fahrzeug einer Flotte zu ... 138
Trennen Sie ein Fahrzeug von einer Flotte ... 139
Aktualisieren Sie eine Flotte ... 140
Löschen Sie eine Flotte ... 141
Überprüfen Sie das Löschen der Flotte ... 142
Holen Sie sich Flotteninformationen ... 143
Daten mit Kampagnen verwalten ... 146
Erstellen einer Kampagne ... 152
Erstellen Sie eine Kampagne (Konsole) ... 153
Erstelle eine Kampagne (AWS CLI) ... 162
Logische Ausdrücke für AWS FleetWise IoT-Kampagnen ... 168
Aktualisieren Sie eine Kampagne ... 169
Löschen Sie eine Kampagne ... 171
Löschen Sie eine Kampagne (Konsole) ... 171
Löschen Sie eine Kampagne (AWS CLI) ... 171
Bestätigen Sie das Löschen der Kampagne ... 172
Holen Sie sich Kampagneninformationen ... 172
Speichern und weiterleiten ... 173
Datenpartitionen erstellen ... 174
Laden Sie Kampagnendaten hoch ... 177
Laden Sie Daten mithilfe von AWS IoT Jobs hoch ... 178
Sammeln Sie diagnostische Fehlercode-Daten ... 180
Schlüsselwörter für den diagnostischen Fehlercode ... 182
Erstellen Sie eine Kampagne zur Datenerfassung für diagnostische Fehlercodes ... 184
Anwendungsfälle für diagnostische Fehlercodes ... 187
Visualisieren Sie Fahrzeugdaten ... 190
Verarbeitung von Fahrzeugdaten, die an ein MQTT-Thema gesendet wurden ... 191
Fahrzeugdaten in Timestream verarbeiten ... 192
Visualisieren Sie die in Timestream gespeicherten Fahrzeugdaten ... 193
Fahrzeugdaten in Amazon S3 verarbeiten ... 193
Amazon S3 S3-Objektformat ... 195
Analysieren Sie die in Amazon S3 gespeicherten Fahrzeugdaten ... 196

Befehle 198
Befehlskonzepte 199
Befehle und wichtige Konzepte 200
Status der Befehlsausführung 203
Fahrzeuge und Kommandos 209
Workflow-Übersicht 210
Arbeitsablauf bei Fahrzeugen 211
Arbeitsablauf für Befehle 214
(Optional) Befehle und Benachrichtigungen 215
Erstellen und Verwalten von Befehlen 217
Erstellen Sie eine Befehlsressource 218
Ruft Informationen zu einem Befehl ab 220
Befehle in Ihrem Konto auflisten 221
Eine Befehlsressource aktualisieren oder als veraltet kennzeichnen 221
Löschen Sie eine Befehlsressource 223
Starten und Überwachen von Befehlsausführungen 224
Das Ergebnis der Befehlsausführung aktualisieren 228
Holen Sie sich die Befehlsausführung 230
Listet die Befehlsausführungen in Ihrem Konto auf 232
Löscht die Ausführung eines Befehls 234
Beispiel: Befehle verwenden 235
Überblick über das Beispiel für den Fahrzeuglenkmodus 235
Voraussetzungen 236
IAM-Richtlinie für die Verwendung von Remote-Befehlen 236
AWS IoT Befehle ausführen (AWS CLI) 239
Bereinigen 244
Szenarien zur Verwendung von Befehlen 246
Einen Befehl ohne Parameter erstellen 247
Einen Befehl mit Standardwerten für Parameter erstellen 248
Einen Befehl mit Parameterwerten erstellen 249
Befehle mit Statusvorlagen verwenden 250
Letzter bekannter Status 253
Erstellen Sie eine Statusvorlage 254
Ordnen Sie einem Fahrzeug eine AWS FleetWise IoT-Statusvorlage zu 256
Aktualisieren Sie eine Statusvorlage 257
Löschen Sie eine Statusvorlage 259

Informationen zur Statusvorlage abrufen ... 259
Vorlagenoperationen in Bundesstaaten ... 261
Aktivieren und deaktivieren Sie die Erfassung staatlicher Daten ... 261
Rufen Sie einen Fahrzeugstatus-Snapshot ab ... 267
Verarbeiten Sie die letzten bekannten Statusfahrzeugdaten mithilfe von MQTT-Messaging . 270
Konfigurieren Sie die netzwerkunabhängige Datenerfassung ... 274
Einführung ... 274
Einrichtung der Umgebung ... 274
Datenmodelle ... 275
Aktualisierungen des Signalkatalogs ... 275
Fahrzeugmodell und Decoder ... 277
Befehl senden ... 280
AWS CLI und SDKs ... 281
Fehlerbehebung ... 282
Probleme mit dem Decoder-Manifest ... 282
Probleme mit dem Edge-Agenten ... 286
Problem: Die Edge Agent-Software startet nicht. ... 286
Problem: [FEHLER] [Io TFleetWiseEngine: :connect]: [Persistenzbibliothek konnte nicht initialisiert werden] ... 288
Problem: Die Edge Agent-Software erfasst keine On-Board-Diagnosen (OBD) II PIDs und Diagnose-Fehlercodes (). DTCs ... 288
Problem: Die Edge Agent for AWS FleetWise IoT-Software sammelt keine Daten aus dem Netzwerk oder kann keine Dateninspektionsregeln anwenden. ... 289
Problem: [FEHLER] [AwsIotConnectivityModule: :connect]: [Verbindung mit Fehler fehlgeschlagen] oder [WARN] [AwsIotChannel: :send]: [Keine aktive MQTT-Verbindung.] ... 290
Probleme speichern und weiterleiten ... 290
Problem: Empfangen eines `AccessDeniedException` mit allen erforderlichen IAM-Berechtigungen ... 290
Problem: Die in AWS IoT Jobs hochgeladenen Daten ignorieren die `endTime` ... 290
Problem: Der Datenupload in AWS IoT Jobs hat den Status „Ausführung“. `REJECTED` ... 291
Sicherheit ... 292
Datenschutz ... 293
Verschlüsselung im Ruhezustand im AWS Internet der Dinge FleetWise ... 294
Verschlüsselung während der Übertragung ... 294
Datenverschlüsselung im AWS IoT FleetWise ... 295
Steuern des Zugriffs ... 308

Erteilen Sie die AWS IoT FleetWise Erlaubnis zum Senden und Empfangen von Daten zu einem MQTT-Thema ... 309
AWS IoT FleetWise Zugriff auf ein Amazon S3 S3-Ziel gewähren ... 312
AWS IoT FleetWise Zugriff auf ein Amazon Timestream Timestream-Ziel gewähren ... 315
Erteilen Sie die AWS IoT Device Management Berechtigung zum Generieren der Payload für Befehle mit AWS IoT FleetWise ... 318
Identitäts- und Zugriffsverwaltung ... 323
Zielgruppe ... 324
Authentifizierung mit Identitäten ... 324
Verwalten des Zugriffs mit Richtlinien ... 326
So FleetWise funktioniert AWS IoT mit IAM ... 328
Beispiele für identitätsbasierte Richtlinien ... 336
Fehlerbehebung ... 340
Referenz für API-Berechtigungen ... 342
Aktualisierung der verwalteten Richtlinien ... 351
AWSIoTFleetwiseServiceRolePolicy ... 352
Compliance-Validierung ... 352
Ausfallsicherheit ... 353
Sicherheit der Infrastruktur ... 353
Verbindung zum AWS IoT FleetWise über eine Schnittstelle (VPC-Endpunkt) ... 354
Konfigurations- und Schwachstellenanalyse ... 358
Bewährte Methoden für die Gewährleistung der Sicherheit ... 358
Erteilen von Mindestberechtigungen ... 358
Keine Protokollierung sensibler Informationen ... 359
Wird verwendet AWS CloudTrail , um den API-Aufrufverlauf anzuzeigen ... 359
Synchronisieren der internen Uhr Ihres Geräts ... 359
Überwachung des AWS IoT FleetWise ... 360
Überwachung mit CloudWatch ... 360
Überwachen Sie mit CloudWatch Protokollen ... 367
AWS FleetWise IoT-Protokolle in der CloudWatch Konsole anzeigen ... 367
Konfigurieren der Protokollierung ... 374
CloudTrail protokolliert ... 378
AWS FleetWise IoT-Informationen in CloudTrail ... 378
Logdateieinträge verstehen ... 379
Dokumentverlauf ... 381
... cccIxxxv

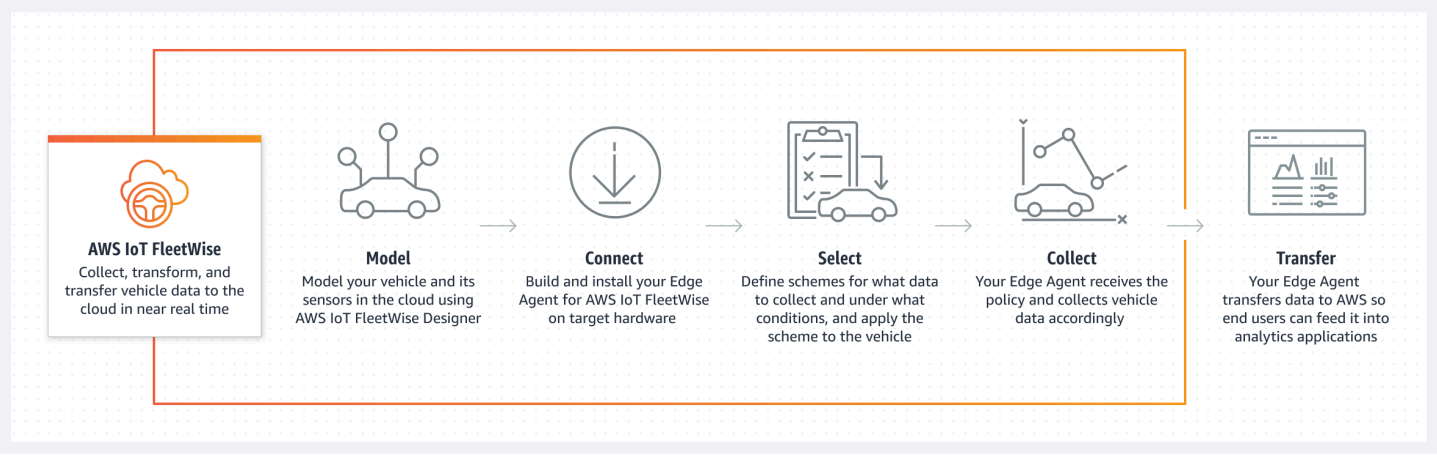
Was ist AWS IoT FleetWise?

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

AWS IoT FleetWise ist ein verwalteter Dienst, mit dem Sie Fahrzeugdaten sammeln und in der Cloud organisieren können. Sie können die gesammelten Daten verwenden, um die Fahrzeugqualität, Leistung und Autonomie zu verbessern. Mit AWS IoT FleetWise können Sie Daten von Fahrzeugen sammeln und organisieren, die unterschiedliche Protokolle und Datenformate verwenden. AWS IoT FleetWise hilft dabei, Nachrichten auf niedriger Ebene in für Menschen lesbare Werte umzuwandeln und das Datenformat in der Cloud für Datenanalysen zu standardisieren. Sie können auch Datenerfassungskampagnen definieren, um zu kontrollieren, welche Fahrzeugdaten gesammelt werden sollen und wann diese Daten in die Cloud übertragen werden sollen.

Wenn sich die Fahrzeugdaten in der Cloud befinden, können Sie sie für Anwendungen verwenden, die den Zustand der Fahrzeugflotte analysieren. Diese Daten können Ihnen helfen, potenzielle Wartungsprobleme zu erkennen, Infotainmentsysteme im Fahrzeug intelligenter zu machen und fortschrittliche Technologien wie autonomes Fahren und Fahrerassistenzsysteme mit Analysen und maschinellem Lernen (ML) zu verbessern.

Das folgende Diagramm zeigt die grundlegende Architektur von AWS IoT FleetWise.



Themen

- Vorteile
- Anwendungsfälle
- Wichtiger Hinweis
- Sind Sie neu im AWS Internet der Dinge FleetWise?
- Zugriff auf AWS IoT FleetWise
- Preise für AWS IoT FleetWise
- Zugehörige Services
- Die wichtigsten Konzepte und Funktionen von AWS IoT FleetWise
- AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise

Vorteile

Die wichtigsten Vorteile von AWS IoT FleetWise sind:

Sammeln Sie Fahrzeugdaten intelligenter

> Verbessern Sie die Datenrelevanz mit intelligenter Datenerfassung, bei der nur die Daten, die Sie benötigen, zur Analyse in die Cloud gesendet werden.

Analysieren Sie auf einfache Weise standardisierte, flottenweite Daten

> Analysieren Sie standardisierte Daten aus einer Fahrzeugflotte, ohne ein individuelles Datenerfassungs- oder Protokollierungssystem entwickeln zu müssen.

Automatische Datensynchronisierung in der Cloud

> Verschaffen Sie sich einen einheitlichen Überblick über Daten, die sowohl von Standardsensoren (Telemetriedaten) als auch von Bildverarbeitungssystemen (Daten von Kameras, Radaren und Lidaren) gesammelt wurden, und synchronisieren Sie sie automatisch in der Cloud. AWS IoT FleetWise sorgt dafür, dass sowohl strukturierte als auch unstrukturierte Bildverarbeitungssystemdaten, Metadaten und Standardsensordaten automatisch in der Cloud synchronisiert werden. Dadurch wird der Prozess optimiert, um sich ein Gesamtbild der Ereignisse zu verschaffen und Erkenntnisse zu gewinnen.

Speichern Sie Daten am Edge und leiten Sie sie unter optimalen Bedingungen weiter

> Senken Sie die Übertragungskosten, indem Sie Daten vorübergehend auf Fahrzeugen speichern. Sie können ausgewählte Daten unter bestimmten, optimalen Bedingungen an die Cloud weiterleiten, z. B. wenn Fahrzeuge eine Wi-Fi-Verbindung herstellen.

> **Note**
>
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Anwendungsfälle

Zu den Szenarien, in denen Sie AWS IoT verwenden können, FleetWise gehören die folgenden:

AI/ML Modelle trainieren

Verbessern Sie kontinuierlich Modelle für maschinelles Lernen, die für autonome und fortschrittliche Fahrerassistenzsysteme verwendet werden, indem Sie Daten von Serienfahrzeugen sammeln.

Verbessern Sie das digitale Kundenerlebnis

Nutzen Sie Daten aus Infotainmentsystemen, um audiovisuelle Inhalte im Fahrzeug und In-App-Einblicke relevanter zu machen.

Sorgen Sie für eine gesunde Fahrzeugflotte

Nutzen Sie Erkenntnisse aus Flottendaten, um den Zustand und den Ladezustand der EV-Batterie zu überwachen, Wartungspläne zu verwalten, den Kraftstoffverbrauch zu analysieren und vieles mehr.

Befehle erstellen und verwalten

Verwenden Sie Befehle, um Befehle an einem Fahrzeug aus der Cloud auszuführen. Sie können Befehle aus der Ferne an ein Fahrzeug senden, und innerhalb weniger Sekunden führt das Fahrzeug den Befehl aus. Sie können beispielsweise Befehle konfigurieren, um die Tür eines Fahrzeugs zu verriegeln oder die Temperatur einzustellen.

Statusvorlagen erstellen und verwalten

Vorlagen für Bundesstaaten bieten Fahrzeugbesitzern die Möglichkeit, den Zustand ihres Fahrzeugs nachzuverfolgen. Der AWS IoT FleetWise Edge Agent, der auf dem Fahrzeug läuft, sammelt Signalaktualisierungen und sendet sie an die Cloud.

Wichtiger Hinweis

Fahrzeugdaten, die durch Ihre Nutzung des AWS IoT FleetWise gesammelt werden, dienen ausschließlich Informationszwecken (einschließlich der Unterstützung beim Trainieren von cloudbasierten Modellen für künstliche Intelligenz und maschinelles Lernen), und Sie dürfen AWS IoT nicht FleetWise zur Steuerung oder Bedienung von Fahrzeugfunktionen verwenden. Sie sind allein verantwortlich für jegliche Haftung, die im Zusammenhang mit einer Nutzung außerhalb des vorgesehenen Zwecks des AWS IoT FleetWise und in einer Weise entstehen kann, die den geltenden Fahrzeugvorschriften widerspricht.

Fahrzeugdaten, die durch Ihre Nutzung des AWS IoT erfasst wurden, FleetWise sollten auf ihre Genauigkeit geprüft werden, die für Ihren Anwendungsfall angemessen ist, auch zum Zwecke der Erfüllung aller Compliance-Verpflichtungen, die Sie möglicherweise gemäß den geltenden Fahrzeugsicherheitsvorschriften haben (wie Sicherheitsüberwachungs- und Berichtspflichten). Eine solche Bewertung sollte die Erfassung und Überprüfung von Informationen mit anderen branchenüblichen Mitteln und Quellen (z. B. Berichte von Fahrzeugführern) umfassen. Sie und Ihre Endbenutzer tragen die alleinige Verantwortung für alle Entscheidungen, Ratschläge, ergriffenen Maßnahmen und Unterlassungen, aufgrund Ihrer Nutzung des AWS IoT Maßnahmen zu ergreifen FleetWise.

Sind Sie neu im AWS Internet der Dinge FleetWise?

Wenn Sie mit AWS IoT noch nicht vertraut sind FleetWise, empfehlen wir Ihnen, zunächst die folgenden Abschnitte zu lesen:

- Die wichtigsten Konzepte und Funktionen von AWS IoT FleetWise
- AWS IoT einrichten FleetWise
- Tutorial: Erste Schritte mit AWS IoT FleetWise
- AWS FleetWise IoT-Daten in die Cloud aufnehmen

Zugriff auf AWS IoT FleetWise

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um auf AWS IoT zuzugreifen FleetWise.

Preise für AWS IoT FleetWise

Fahrzeuge senden Daten über MQTT-Nachrichten an die Cloud. Sie zahlen am Ende eines jeden Monats für die Fahrzeuge, die Sie im AWS IoT erstellt haben FleetWise. Sie zahlen auch für Nachrichten, die Sie von Fahrzeugen sammeln. Aktuelle Informationen zur Preisgestaltung finden Sie auf der Seite mit den AWS FleetWise IoT-Preisen. Weitere Informationen zum MQTT-Messaging-Protokoll finden Sie unter MQTT im AWS IoT Core Developer Guide.

Zugehörige Services

AWS IoT FleetWise lässt sich in die folgenden AWS Dienste integrieren, um die Verfügbarkeit und Skalierbarkeit Ihrer Cloud-Lösungen zu verbessern.

- AWS IoT Core— Registrieren und steuern Sie AWS IoT Geräte, die Fahrzeugdaten in das AWS IoT FleetWise hochladen und Befehle aus der Ferne an ein Fahrzeug senden. Weitere Informationen finden Sie unter Was ist AWS IoT im AWS IoT -Entwicklerhandbuch.
- Amazon Timestream — Verwenden Sie eine Zeitreihendatenbank, um Ihre Fahrzeugdaten zu speichern und zu analysieren. Weitere Informationen finden Sie unter Was ist Amazon Timestream im Amazon Timestream Developer Guide.
- Amazon S3 — Verwenden Sie einen Objektspeicherservice, um Ihre Fahrzeugdaten zu speichern und zu verwalten. Weitere Informationen finden Sie unter Was ist Amazon S3 im Amazon Simple Storage Service-Benutzerhandbuch.

Die wichtigsten Konzepte und Funktionen von AWS IoT FleetWise

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Die folgenden Abschnitte bieten einen Überblick über AWS FleetWise IoT-Servicekomponenten und deren Zusammenspiel.

Nachdem Sie diese Einführung gelesen haben, erfahren Sie im AWS IoT einrichten FleetWise Abschnitt, wie Sie AWS IoT einrichten FleetWise.

Themen

- Die wichtigsten Konzepte
- Funktionen von AWS IoT FleetWise

Die wichtigsten Konzepte

AWS IoT FleetWise bietet ein Framework zur Fahrzeugmodellierung, mit dem Sie Ihr Fahrzeug und seine Sensoren und Aktuatoren in der Cloud modellieren können. Um die sichere Kommunikation zwischen Ihrem Fahrzeug und der Cloud zu ermöglichen, bietet AWS IoT FleetWise auch eine Referenzimplementierung, die Sie bei der Entwicklung von Edge Agent-Software unterstützt, die Sie in Ihrem Fahrzeug installieren können. Sie können Datenerfassungsschemata in der Cloud definieren und sie in Ihrem Fahrzeug bereitstellen. Die in Ihrem Fahrzeug ausgeführte Edge Agent-Software verwendet Datenerfassungsschemata, um zu steuern, welche Daten erfasst und wann sie in die Cloud übertragen werden sollen.

Im Folgenden sind die Kernkonzepte von AWS IoT aufgeführt FleetWise.

Signal

Signale sind grundlegende Strukturen, die Sie so definieren, dass sie Fahrzeugdaten und deren Metadaten enthalten. Ein Signal kann ein Attribut, ein Zweig, ein Sensor oder ein Aktuator sein. Sie können beispielsweise einen Sensor erstellen, der die Temperaturwerte im Fahrzeug empfängt und dessen Metadaten, einschließlich eines Sensornamens, eines Datentyps und einer Einheit, speichert. Weitere Informationen finden Sie unter AWS FleetWise IoT-Signalkataloge verwalten.

Attribut

Attribute stellen statische Informationen dar, die sich im Allgemeinen nicht ändern, z. B. Hersteller und Herstellungsdatum.

Verzweigung

Zweige stellen Signale in einer verschachtelten Struktur dar. Zweige zeigen Signalhierarchien. Zum Beispiel hat der `Vehicle` Zweig einen untergeordneten Zweig,`Powertrain`. Der `Powertrain` Zweig hat einen untergeordneten Zweig,`combustionEngine`. Verwenden Sie den `Vehicle.Powertrain.combustionEngine` Ausdruck, um den `combustionEngine` Zweig zu finden.

Sensor

Sensordaten geben den aktuellen Zustand des Fahrzeugs an und ändern sich im Laufe der Zeit, wenn sich der Zustand des Fahrzeugs ändert, z. B. Flüssigkeitsstand, Temperaturen, Vibrationen oder Spannung.

Aktuator

Aktuatordaten geben Auskunft über den Zustand von Fahrzeuggeräten wie Motoren, Heizungen und Türschlössern. Durch Ändern des Zustands eines Fahrzeuggeräts können Aktuatordaten aktualisiert werden. Sie können beispielsweise einen Aktuator definieren, der die Heizung darstellt. Der Aktuator empfängt neue Daten, wenn Sie die Heizung ein- oder ausschalten.

Benutzerdefinierter Aufbau

Eine benutzerdefinierte Struktur (auch als Struktur bezeichnet) stellt eine komplexe Datenstruktur oder Datenstruktur höherer Ordnung dar. Sie erleichtert das logische Binden oder Gruppieren von Daten, die aus derselben Quelle stammen. Eine Struktur wird verwendet, wenn Daten in einer atomaren Operation gelesen oder geschrieben werden, z. B. um einen komplexen Datentyp oder eine Form höherer Ordnung darzustellen.

Ein Signal vom Strukturtyp wird im Signalkatalog definiert, indem ein Verweis auf einen Strukturdatentyp anstelle eines primitiven Datentyps verwendet wird. Strukturen können für alle Arten von Signalen verwendet werden, einschließlich Sensoren, Attributen, Aktuatoren und Datentypen für Bildverarbeitungssysteme. Wenn ein Signal vom Typ Structure gesendet oder empfangen wird, FleetWise erwartet AWS IoT, dass alle enthaltenen Elemente gültige Werte haben, sodass alle Elemente obligatorisch sind. Wenn eine Struktur beispielsweise die Elemente Vehicle.Camera.Image.Height, Vehicle.Camera.Image.Width und Vehicle.Camera.Image.Data enthält, wird erwartet, dass das gesendete Signal Werte für all diese Elemente enthält.

> **Note**
>
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Benutzerdefiniertes Eigentum

Eine benutzerdefinierte Eigenschaft stellt ein Element der komplexen Datenstruktur dar. Der Datentyp der Eigenschaft kann entweder primitiv oder eine andere Struktur sein.

Bei der Darstellung einer Form höherer Ordnung mithilfe einer Struktur und einer benutzerdefinierten Eigenschaft wird die beabsichtigte Form höherer Ordnung immer als Baumstruktur definiert und betrachtet. Die benutzerdefinierte Eigenschaft wird verwendet, um alle Blattknoten zu definieren, während die Struktur verwendet wird, um alle Knoten zu definieren, die keine Blattknoten sind.

Signalkatalog

Ein Signalkatalog enthält eine Sammlung von Signalen. Signale in einem Signalkatalog können verwendet werden, um Fahrzeuge zu modellieren, die unterschiedliche Protokolle und Datenformate verwenden. Beispielsweise gibt es zwei Fahrzeuge, die von verschiedenen Autoherstellern hergestellt werden: eines verwendet das Control Area Network (CAN-Bus) -Protokoll, das andere das On-Board Diagnostics (OBD) -Protokoll. Sie können im Signalkatalog einen Sensor für den Empfang von Fahrzeugtemperaturwerten definieren. Dieser Sensor kann zur Darstellung der Thermoelemente in beiden Fahrzeugen verwendet werden. Weitere Informationen finden Sie unter AWS FleetWise IoT-Signalkataloge verwalten.

Fahrzeugmodell (Modellmanifest)

Fahrzeugmodelle sind deklarative Strukturen, mit denen Sie das Format Ihrer Fahrzeuge standardisieren und Beziehungen zwischen Signalen in den Fahrzeugen definieren können. Fahrzeugmodelle sorgen für konsistente Informationen für mehrere Fahrzeuge desselben Typs. Sie fügen Signale hinzu, um Fahrzeugmodelle zu erstellen. Weitere Informationen finden Sie unter AWS FleetWise IoT-Fahrzeugmodelle verwalten.

Decoder-Manifest

Decoder-Manifeste enthalten Dekodierungsinformationen für jedes Signal in Fahrzeugmodellen. Sensoren und Aktuatoren in Fahrzeugen übertragen Nachrichten auf niedriger Ebene (Binärdaten). Mit Decoder-Manifesten FleetWise ist AWS IoT in der Lage, Binärdaten in menschenlesbare Werte umzuwandeln. Jedes Decoder-Manifest ist einem Fahrzeugmodell zugeordnet. Weitere Informationen finden Sie unter AWS FleetWise IoT-Decoder-Manifeste verwalten.

Netzwerkschnittstelle

Enthält Informationen über das Protokoll, das das bordeigene Netzwerk verwendet. AWS IoT FleetWise unterstützt die folgenden Protokolle.

Controller Area Network (CAN-Bus)

Ein Protokoll, das definiert, wie Daten zwischen elektronischen Steuergeräten (ECUs) übertragen werden. ECUs kann das Motorsteuergerät, Airbags oder das Audiosystem sein.

On-Board-Diagnose (OBD) II

Ein weiterentwickeltes Protokoll, das definiert, wie Selbstdiagnosedaten untereinander ausgetauscht werden. ECUs Es bietet eine Reihe von Standard-Diagnosefehlercodes (DTCs), mit deren Hilfe Sie feststellen können, was mit Ihrem Fahrzeug nicht stimmt.

Middleware für Fahrzeuge

Die Fahrzeug-Middleware ist als eine Art Netzwerkschnittstelle definiert. Zu den Beispielen für Fahrzeug-Middleware gehören das Robot Operating System (ROS 2) und die skalierbare serviceorientierte Middleware over IP (SOME/IP).

> **Note**
>
> AWS IoT FleetWise unterstützt ROS 2-Middleware für Bildverarbeitungssystemdaten.

Kundenspezifische Schnittstellen

Sie können auch Ihre eigene Schnittstelle verwenden, um Signale am Edge zu dekodieren. Dies kann Ihnen Zeit sparen, da Sie keine Dekodierungsregeln in der Cloud erstellen müssen.

Signaldecoder

Bietet detaillierte Dekodierungsinformationen für ein bestimmtes Signal. Jedes im Fahrzeugmodell angegebene Signal muss mit einem Signaldecoder gepaart werden. Wenn das Decoder-Manifest CAN-Netzwerkschnittstellen enthält, muss es CAN-Decodersignale enthalten. Wenn das Decoder-Manifest OBD-Netzwerkschnittstellen enthält, muss es OBD-Signaldecoder enthalten.

Das Decoder-Manifest muss Dekoder für Nachrichtensignale enthalten, wenn es auch Middleware-Schnittstellen für Fahrzeuge enthält. Oder, wenn das Decoder-Manifest benutzerdefinierte Dekodierungsschnittstellen enthält, muss es auch benutzerdefinierte Dekodierungssignale enthalten.

Fahrzeug

Eine virtuelle Darstellung Ihres physischen Fahrzeugs, z. B. eines Autos oder eines Lastwagens. Fahrzeuge sind Beispiele für Fahrzeugmodelle. Fahrzeuge, die aus demselben Fahrzeugmodell hergestellt wurden, erben dieselbe Gruppe von Signalen. Jedes Fahrzeug entspricht einer AWS IoT Sache.

Flotte

Eine Flotte steht für eine Gruppe von Fahrzeugen. Bevor Sie eine Fahrzeugflotte einfach verwalten können, müssen Sie einzelne Fahrzeuge einer Flotte zuordnen.

Kampagne

Enthält Datenerfassungsschemata. Sie definieren eine Kampagne in der Cloud und stellen sie für ein Fahrzeug oder eine Flotte bereit. Kampagnen geben der Edge Agent-Software Anweisungen zur Auswahl, Erfassung und Übertragung von Daten in die Cloud.

Datenpartition

Konfigurieren Sie partitionierte Daten in einer Kampagne, um Signaldaten vorübergehend zu speichern. Sie konfigurieren, wann und wie die Daten an die Cloud weitergeleitet werden sollen.

Schema der Datenerfassung

Datenerfassungsschemata geben der Edge Agent-Software Anweisungen zum Sammeln von Daten. Derzeit FleetWise unterstützt AWS IoT das zustandsbasierte Sammelschema und das zeitbasierte Sammelschema.

Bedingungsabhängiges Sammelsystem

Verwenden Sie einen logischen Ausdruck, um zu erkennen, welche Daten gesammelt werden sollen. Die Edge Agent-Software sammelt Daten, wenn die Bedingung erfüllt ist. Wenn der Ausdruck beispielsweise lautet`$variable.myVehicle.InVehicleTemperature >35.0`, erfasst die Edge Agent-Software Temperaturwerte, die über 35,0 liegen.

Zeitbasiertes Erfassungsschema

Geben Sie einen Zeitraum in Millisekunden an, um zu definieren, wie oft Daten gesammelt werden sollen. Wenn der Zeitraum beispielsweise 10.000 Millisekunden beträgt, erfasst die Edge Agent-Software alle 10 Sekunden Daten.

Befehle

Befehle führen Befehle an einem Fahrzeug aus der Cloud aus. Sie können Befehle aus der Ferne an ein Fahrzeug senden, und innerhalb weniger Sekunden führt das Fahrzeug den Befehl aus. Sie können beispielsweise Befehle konfigurieren, um die Tür eines Fahrzeugs zu verriegeln oder die Temperatur einzustellen.

Der Befehl ist eine Ressource, die von verwaltet wird AWS IoT Device Management. Er enthält wiederverwendbare Konfigurationen, die angewendet werden, wenn eine Befehlsausführung an

das Fahrzeug gesendet wird. Weitere Informationen finden Sie unter AWS IoT Befehle im AWS IoT Core Entwicklerhandbuch.

Vorlagen für Bundesstaaten

Staatliche Vorlagen bieten Fahrzeugbesitzern die Möglichkeit, den Zustand ihres Fahrzeugs nachzuverfolgen. Der Edge Agent-Softwareagent, der auf dem Fahrzeug ausgeführt wird, sammelt Signalaktualisierungen und sendet sie an die Cloud. Jede Statusvorlage enthält eine Liste von Signalen, aus denen Daten gesammelt werden.

Funktionen von AWS IoT FleetWise

Im Folgenden sind die wichtigsten Funktionen von AWS IoT aufgeführt FleetWise.

Modellierung von Fahrzeugen

Erstellen Sie virtuelle Darstellungen Ihrer Fahrzeuge und wenden Sie ein einheitliches Format an, um Fahrzeugsignale zu organisieren. AWS IoT FleetWise unterstützt die Vehicle Signal Specification (VSS), mit der Sie Fahrzeugsignale standardisieren können.

Schemabasierte Datenerfassung

Definieren Sie Schemata, um nur hochwertige Fahrzeugdaten in die Cloud zu übertragen. Sie können zustandsabhängige Schemata definieren, um zu steuern, welche Daten erfasst werden sollen, z. B. Daten, die im Fahrzeug über 40 Grad liegen. Sie können auch zeitbasierte Schemata definieren, um zu steuern, wie oft Daten erfasst werden.

Edge-Agent für AWS FleetWise IoT-Software

Die in Fahrzeugen ausgeführte Edge Agent-Software erleichtert die Kommunikation zwischen Fahrzeugen und der Cloud. Während Fahrzeuge mit der Cloud verbunden sind, empfängt die Edge Agent-Software kontinuierlich Datenerfassungsschemata und sammelt Daten entsprechend.

AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise

Eine Liste der AWS Regionen, die AWS IoT unterstützen FleetWise, finden Sie unter AWS FleetWise IoT-Endpunkte und Kontingente. AWS FleetWise IoT-Funktionen unterscheiden sich in ihrer regionalen Unterstützung.

> **Note**
>
> Der Zugang zur Region Asien-Pazifik (Mumbai) und zu einigen AWS FleetWise IoT-Funktionen sind derzeit gesperrt. Um Zugriff auf diese AWS Region und alle geschützten Funktionen zu beantragen, wenden Sie sich an Ihren Account Manager oder das AWS Support Center.

Die folgende Tabelle zeigt den Feature-Support nach Region:

Funktionen/Regionen	USA Ost (Nord-Virginia)	Europa (Frankfurt)	Asien-Pazifik (Mumbai) HINWEIS: Nur bewachter Zugang
Signalkataloge	Ja	Ja	Geschützt
Fahrzeugmodelle	Ja	Ja	Geschützt
Decoder-Manifestationen	Ja	Ja	Geschützt
Fahrzeuge	Ja	Ja	Geschützt
Flotten	Ja	Ja	Umzäunt
Kampagnen	Ja	Ja	Umzäunt
Daten zum Bildverarbeitungssystem (in der Vorschauversion)	Ja	Ja	Geschützt
MQTT-Thema als Ziel für Kampagnendaten	Geschützt	Umzäunt	Umzäunt
Speichern und weiterleiten	Geschützt	Umzäunt	Umzäunt
Befehle	Umzäunt	Umzäunt	Umzäunt

Funktionen/Regionen	USA Ost (Nord-Virginia)	Europa (Frankfurt)	Asien-Pazifik (Mumbai) HINWEIS: Nur bewachter Zugang
Letzter bekannter Bundesstaat	Geschützt	Umzäunt	Umzäunt
Netzwerkunabhängige Datenerfassung mit einer benutzerdefinierten Dekodierungsschnittstelle	Geschützt	Umzäunt	Umzäunt
Abrufen des Diagnose-Fehlercodes (DTC) *	Geschützt	Umzäunt	Umzäunt

*Der DTC-Abruf bietet eine Reihe von Funktionen, die über den einfachen Abruf von DTC-Daten hinausgehen. Diese Funktionalität umfasst benutzerdefinierte Funktionen, mit denen Sie Funktionen am Netzwerkrand definieren und diese innerhalb von bedingungsbasierten Kampagnenausdrücken namentlich aufrufen können. Darüber hinaus unterstützt sie die Erfassung unbegrenzter Zeichenketten und ermöglicht so eine flexible Handhabung von Zeichenketten-Datentypen. Der Edge-Agent kann Daten entweder in regelmäßigen Abständen oder ausgelöst durch bestimmte Bedingungen abrufen, wodurch seine Anpassungsfähigkeit und Effizienz bei Datenerfassungsprozessen verbessert werden. Weitere Informationen finden Sie im Leitfaden für benutzerdefinierte Funktionen und in der Referenzimplementierung der DTC-Datenerfassung im Edge Agent Developer Guide.

AWS IoT einrichten FleetWise

Bevor Sie AWS IoT FleetWise zum ersten Mal verwenden, führen Sie die Schritte in den folgenden Abschnitten aus.

Topics

- Richten Sie Ihre ein AWS-Konto
- Erste Schritte in der Konsole
- Konfigurieren Sie Ihre AWS FleetWise IoT-Einstellungen
- Anfragen an das AWS IoT stellen FleetWise mit IPv6

Richten Sie Ihre ein AWS-Konto

Führen Sie die folgenden Aufgaben aus, um sich für einen Administratorbenutzer zu registrieren AWS und diesen zu erstellen.

Melden Sie sich an für ein AWS-Konto

Wenn Sie noch keine haben AWS-Konto, führen Sie die folgenden Schritte aus, um eine zu erstellen.

Um sich für eine anzumelden AWS-Konto

1. Öffnen Sie https://portal.aws.amazon.com/billing/die Anmeldung.
2. Folgen Sie den Online-Anweisungen.

   Während der Anmeldung erhalten Sie einen Telefonanruf oder eine Textnachricht und müssen einen Verifizierungscode über die Telefontasten eingeben.

   Wenn Sie sich für eine anmelden AWS-Konto, Root-Benutzer des AWS-Kontoswird eine erstellt. Der Root-Benutzer hat Zugriff auf alle AWS-Services und Ressourcen des Kontos. Als bewährte Sicherheitsmethode weisen Sie einem Benutzer Administratorzugriff zu und verwenden Sie nur den Root-Benutzer, um Aufgaben auszuführen, die Root-Benutzerzugriff erfordern.

AWS sendet Ihnen nach Abschluss des Anmeldevorgangs eine Bestätigungs-E-Mail. Du kannst jederzeit deine aktuellen Kontoaktivitäten einsehen und dein Konto verwalten, indem du zu https://aws.amazon.com/gehst und Mein Konto auswählst.

Erstellen eines Benutzers mit Administratorzugriff

Nachdem Sie sich für einen angemeldet haben AWS-Konto, sichern Sie Ihren Root-Benutzer des AWS-Kontos AWS IAM Identity Center, aktivieren und erstellen Sie einen Administratorbenutzer, sodass Sie den Root-Benutzer nicht für alltägliche Aufgaben verwenden.

Sichern Sie Ihre Root-Benutzer des AWS-Kontos

1. Melden Sie sich AWS-Managementkonsoleals Kontoinhaber an, indem Sie Root-Benutzer auswählen und Ihre AWS-Konto E-Mail-Adresse eingeben. Geben Sie auf der nächsten Seite Ihr Passwort ein.

   Hilfe bei der Anmeldung mit dem Root-Benutzer finden Sie unter Anmelden als Root-Benutzer im AWS-Anmeldung -Benutzerhandbuch zu.
2. Aktivieren Sie die Multi-Faktor-Authentifizierung (MFA) für den Root-Benutzer.

   Anweisungen finden Sie unter Aktivieren eines virtuellen MFA-Geräts für Ihren AWS-Konto Root-Benutzer (Konsole) im IAM-Benutzerhandbuch.

Erstellen eines Benutzers mit Administratorzugriff

1. Aktivieren Sie das IAM Identity Center.

   Anweisungen finden Sie unter Aktivieren AWS IAM Identity Center im AWS IAM Identity Center -Benutzerhandbuch.
2. Gewähren Sie einem Administratorbenutzer im IAM Identity Center Benutzerzugriff.

   Ein Tutorial zur Verwendung von IAM-Identity-Center-Verzeichnis als Identitätsquelle finden Sie IAM-Identity-Center-Verzeichnis im Benutzerhandbuch unter Benutzerzugriff mit der Standardeinstellung konfigurieren.AWS IAM Identity Center

Anmelden als Administratorbenutzer

- Um sich mit Ihrem IAM-Identity-Center-Benutzer anzumelden, verwenden Sie die Anmelde-URL, die an Ihre E-Mail-Adresse gesendet wurde, als Sie den IAM-Identity-Center-Benutzer erstellt haben.

  Hilfe bei der Anmeldung mit einem IAM Identity Center-Benutzer finden Sie im AWS-Anmeldung Benutzerhandbuch unter Anmeldung beim AWS Access-Portal.

Weiteren Benutzern Zugriff zuweisen

1. Erstellen Sie im IAM-Identity-Center einen Berechtigungssatz, der den bewährten Vorgehensweisen für die Anwendung von geringsten Berechtigungen folgt.

   Anweisungen hierzu finden Sie unter Berechtigungssatz erstellen im AWS IAM Identity Center - Benutzerhandbuch.
2. Weisen Sie Benutzer einer Gruppe zu und weisen Sie der Gruppe dann Single Sign-On-Zugriff zu.

   Eine genaue Anleitung finden Sie unter Gruppen hinzufügen im AWS IAM Identity Center - Benutzerhandbuch.

> **Note**
>
> Sie können eine serviceverknüpfte Rolle mit AWS IoT FleetWise verwenden. Servicebezogene Rollen sind von AWS IoT vordefiniert FleetWise und beinhalten die Berechtigungen, die AWS IoT FleetWise benötigt, um Metriken an Amazon CloudWatch zu senden. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für AWS IoT FleetWise.

Erste Schritte in der Konsole

Wenn Sie noch nicht bei Ihrem angemeldet sind AWS-Konto, melden Sie sich an und öffnen Sie dann die AWS FleetWise IoT-Konsole. Um mit AWS IoT zu beginnen FleetWise, erstellen Sie ein Fahrzeugmodell. Ein Fahrzeugmodell standardisiert das Format Ihrer Fahrzeuge.

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie unter Erste Schritte mit AWS IoT FleetWise die Option Erste Schritte aus.

Weitere Informationen zum Erstellen eines Fahrzeugmodells finden Sie unterErstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.

Konfigurieren Sie Ihre AWS FleetWise IoT-Einstellungen

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um Einstellungen für Amazon CloudWatch Logs-Metriken und Amazon CloudWatch Logs zu konfigurieren und Daten mit einem Von AWS verwalteter Schlüssel zu verschlüsseln.

Mit CloudWatch Metriken können Sie AWS IoT FleetWise und andere AWS Ressourcen überwachen. Sie können CloudWatch Metriken verwenden, um Messwerte zu sammeln und nachzuverfolgen, z. B. um festzustellen, ob ein Servicelimit überschritten wurde. Weitere Informationen zu CloudWatch Metriken finden Sie unterÜberwachen Sie AWS IoT FleetWise mit Amazon CloudWatch.

Mit CloudWatch Logs FleetWise sendet AWS IoT Protokolldaten an eine CloudWatch Protokollgruppe, wo Sie sie verwenden können, um Probleme zu identifizieren und zu beheben. Weitere Informationen zu CloudWatch Protokollen finden Sie unterAWS FleetWise IoT-Protokollierung konfigurieren.

Mit Datenverschlüsselung FleetWise verwendet AWS Von AWS verwaltete Schlüssel IoT Daten. Sie können sich auch dafür entscheiden, Schlüssel mit AWS KMS zu erstellen und zu verwalten. Weitere Informationen zur Verschlüsselung finden Sie unter Datenverschlüsselung im AWS IoT FleetWise.

Konfigurieren der Einstellungen (Konsole)

Wenn Sie noch nicht bei Ihrem angemeldet sind AWS-Konto, melden Sie sich an und öffnen Sie dann die AWS FleetWiseIoT-Konsole.

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im linken Bereich Einstellungen aus.
3. Wählen Sie unter Metriken die Option Aktivieren aus. AWS IoT fügt der serviceverknüpften Rolle FleetWise automatisch eine CloudWatch verwaltete Richtlinie hinzu und aktiviert CloudWatch Metriken.
4. Wählen Sie unter Protokollierung die Option Bearbeiten aus.
   a. Geben Sie im Abschnitt CloudWatch Protokollierung die Protokollgruppe ein.
   b. Um Ihre Änderungen zu speichern, wählen Sie Submit.
5. Wählen Sie im Bereich Verschlüsselung die Option Bearbeiten aus.
   a. Wählen Sie den Schlüsseltyp aus, den Sie verwenden möchten. Weitere Informationen finden Sie unter Schlüsselmanagement im AWS Internet der Dinge FleetWise.

i. AWS Schlüssel verwenden — AWS IoT FleetWise besitzt und verwaltet den Schlüssel.

ii. Wählen Sie einen anderen AWS Key Management Service Schlüssel — Sie verwalten AWS KMS keys die Schlüssel in Ihrem Konto.

b. Um Ihre Änderungen zu speichern, wählen Sie Senden.

Einstellungen konfigurieren (AWS CLI)

Registrieren Sie im das Konto AWS CLI, um die Einstellungen zu konfigurieren.

Einrichtung der IAM-Berechtigungen für die Kontoregistrierung

Um die `RegisterAccount` API erfolgreich aufzurufen, müssen Sie Folgendes `iam:CreateServiceLinkedRole` in Ihr IAM-Richtliniendokument aufnehmen. Diese API erstellt eine serviceverknüpfte Rolle in Ihrem Konto, die verwendet wird, um AWS FleetWise IoT-Metriken für Ihr CloudWatch Konto zu veröffentlichen. Um zu überprüfen, ob das Konto erfolgreich registriert wurde, rufen Sie die `GetRegisterAccountStatus` API auf und stellen Sie sicher, dass der Registrierungsstatus lautet. `REGISTRATION_SUCCESS`

Das folgende Beispiel zeigt ein Beispiel für ein Richtliniendokument zum Einrichten von Berechtigungen für `RegisterAccount` und`GetRegisterAccountStatus`:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iotfleetwise:RegisterAccount",
        "iotfleetwise:GetRegisterAccountStatus",
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

1. Führen Sie den folgenden Befehl aus, um Einstellungen zu konfigurieren.

```
aws iotfleetwise register-account
```

2. Um Ihre Einstellungen zu überprüfen, führen Sie den folgenden Befehl aus, um den Registrierungsstatus abzurufen.

> **Note**
>
> Die serviceverknüpfte Rolle wird nur verwendet, um AWS FleetWise IoT-Metriken zu CloudWatch veröffentlichen. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für AWS IoT FleetWise.

```
aws iotfleetwise get-register-account-status
```

Example response

```
{
    "accountStatus": "REGISTRATION_SUCCESS",
    "creationTime": "2022-07-28T11:31:22.603000-07:00",
    "customerAccountId": "012345678912",
    "iamRegistrationResponse": {
        "errorMessage": "",
        "registrationStatus": "REGISTRATION_SUCCESS",
        "roleArn": "arn:aws:iam::012345678912:role/AWSIoTFleetwiseServiceRole"
    },
    "lastModificationTime": "2022-07-28T11:31:22.854000-07:00",
    }
}
```

Der Registrierungsstatus kann einer der folgenden sein:

- `REGISTRATION_SUCCESS`— Die AWS Ressource wurde erfolgreich registriert.
- `REGISTRATION_PENDING`— AWS IoT bearbeitet FleetWise die Registrierungsanfrage. Dieser Vorgang dauert ungefähr fünf Minuten.
- `REGISTRATION_FAILURE`— AWS IoT FleetWise kann die AWS Ressource nicht registrieren. Bitte versuchen Sie es später erneut.

Anfragen an das AWS IoT stellen FleetWise mit IPv6

Sie können mit AWS IoT FleetWise über das Internetprotokoll Version 6 (IPv6) kommunizieren und IPv4 Ihre Ressourcen verwalten. Dual-Stack-Endpunkte unterstützen Anfragen an das AWS IoT FleetWise APIs über IPv6 und. IPv4 Für die Kommunikation über fallen keine zusätzlichen Gebühren an. IPv6

Das IPv6 Protokoll ist der IP-Standard der nächsten Generation mit zusätzlichen Sicherheitsfunktionen. Es bietet einen 128-Bit-Adressraum und IPv4 eine 32-Bit-Adresse. IPv4 kann 4,29 x 10^9 Adressen generieren und gleichzeitig 3,4 x IPv6 10^38 Adressen haben.

IPv6 Voraussetzungen für Endpunkte der Steuerungsebene

IPv6 Die Protokollunterstützung wird für Endpunkte der Steuerungsebene automatisch aktiviert. Wenn Sie die Endpunkte für Clients auf der Kontrollebene verwenden, müssen Sie die Erweiterung Server Name Indication (SNI) angeben. Clients können mithilfe der SNI-Erweiterung den Namen des Servers angeben, mit dem Kontakt aufgenommen wird, und angeben, ob er die regulären Endpunkte oder die Dual-Stack-Endpunkte verwendet. Siehe Verwenden von Dual-Stack-Endpunkten.

IPv6 Unterstützung für Endgeräte AWS PrivateLink

AWS IoT FleetWise unterstützt die IPv6 Kommunikation mit VPC-Endpunkten über Schnittstellen. AWS PrivateLink

Die Adresskompatibilität wird getestet IPv6

Wenn Sie use Linux/Unix oder Mac OS X verwenden, können Sie testen, ob Sie auf einen Dual-Stack-Endpunkt zugreifen können, IPv6 indem Sie den Befehl curl verwenden, wie im folgenden Beispiel gezeigt:

```
curl -v https://iotfleetwise.<us-east-1>.api.aws
```

Sie erhalten Informationen wie im folgenden Beispiel gezeigt zurück. Wenn Sie über verbunden sind IPv6, ist die verbundene IP-Adresse eine IPv6 Adresse.

```
* Host iotfleetwise.us-east-1.api.aws:443 was resolved.
* IPv6: ::ffff:3.82.78.135, ::ffff:54.211.220.216, ::ffff:54.211.201.157
* IPv4: (none)
*   Trying [::ffff:3.82.78.135]:443...
```

```
* Connected to iotfleetwise.us-east-1.api.aws (::ffff:3.82.78.135) port 443
* ALPN: curl offers h2,http/1.1
```

Wenn Sie Microsoft Windows 7 oder Windows 10 verwenden, können Sie testen, ob Sie über IPv6 oder IPv4 mithilfe des Ping-Befehls auf einen Dual-Stack-Endpunkt zugreifen können, wie im folgenden Beispiel gezeigt.

```
ping iotfleetwise.<us-east-1>.api.aws
```

IPv6 Adressen in IAM-Richtlinien verwenden

Bevor Sie sie IPv6 für Ihre Ressourcen verwenden, müssen Sie sicherstellen, dass alle IAM-Richtlinien, die für die IP-Adressfilterung verwendet werden, Adressbereiche enthalten IPv6 . Weitere Informationen über die Verwaltung von Zugriffsberechtigungen mit IAM finden Sie unter Identity and Access Management für AWS IoT FleetWise.

IAM-Richtlinien, die IP-Adressen filtern, verwenden Bedingungsoperatoren für IP-Adressen. Die folgende Richtlinie identifiziert den `54.240.143.*` Bereich der zulässigen IPv4 Adressen mithilfe von Operatoren für IP-Adressbedingungen. Da sich alle IPv6 Adressen außerhalb des zulässigen Bereichs befinden, verhindert diese Richtlinie die Kommunikation unter Verwendung von IPv6 Adressen.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "IPAllow",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "iotfleetwise:*",
      "Resource": "arn:aws:iotfleetwise:us-east-1:111122223333:*",
      "Condition": {
         "IpAddress": {"aws:SourceIp": "54.240.143.0/24"}
      }
    }
  ]
}
```

Um IPv6 Adressen einzubeziehen, können Sie das Condition-Element der Richtlinie so ändern, dass es sowohl IPv4 (54.240.143.0/24) als auch IPv6 (2001:: 1234:5678DB8: :/64) Adressbereiche zulässt, wie im folgenden Beispiel gezeigt.

```
"Condition": {
  "IpAddress": {
    "aws:SourceIp": [
      "54.240.143.0/24",
      "2001:DB8:1234:5678::/64"
    ]
  }
}
```

Verwenden von Dual-Stack-Endpunkten

AWS FleetWise IoT-Dual-Stack-Endpunkte unterstützen Anfragen an AWS IoT FleetWise APIs über IPv6 und. IPv4 Wenn Sie eine Anfrage an einen Dual-Stack-Endpunkt stellen, wird diese automatisch als Adresse oder als IPv4 Adresse aufgelöst. IPv6 Im Dual-Stack-Modus werden IPv4 sowohl Verbindungen als auch IPv6 Client-Verbindungen akzeptiert.

Wenn Sie die REST-API verwenden, können Sie mithilfe des FleetWise Endpunktnamens (URI) direkt auf einen AWS IoT-Endpunkt zugreifen. AWS IoT FleetWise unterstützt nur regionale Dual-Stack-Endpunktnamen, was bedeutet, dass Sie die AWS-Region als Teil des Namens angeben müssen.

Die folgende Tabelle zeigt das Format der Steuerebenen-Endpunkte für AWS IoT FleetWise bei Verwendung IPv4 und die Dual-Stack-Modi. Weitere Informationen zu diesen Endpunkten finden Sie unter AWS FleetWise IoT-Endpunkte.

Endpoint	IPv4 Adresse	Dual-Stack-Modus
Steuerebene	flottenweise. *<region>*.amazonaws.com	flottenweise. *<region>*.api.aws

Wenn Sie AWS CLI und verwenden AWS SDKs, können Sie eine `AWS_USE_DUALSTACK_ENDPOINT` Umgebungsvariable oder den `use_dualstack_endpoint` Parameter verwenden, bei dem es sich um eine Einstellung in einer gemeinsam genutzten Konfigurationsdatei handelt, um zu einem Dual-

Stack-Endpunkt zu wechseln. Sie können den Dual-Stack-Endpunkt auch direkt als Überschreibung des AWS FleetWise IoT-Endpunkts in der Konfigurationsdatei angeben. Weitere Informationen finden Sie unter Dual-Stack- und FIPS-Endpunkte.

Wenn Sie den verwenden AWS CLI, können Sie den Konfigurationswert `use_dualstack_endpoint` wie `true` in einem Profil in Ihrer AWS Konfigurationsdatei festlegen. Dadurch werden alle AWS FleetWise IoT-Anfragen, die von den Befehlen gestellt werden, an den Dual-Stack-Endpunkt für die angegebene Region weitergeleitet. Sie geben die Region in der Konfigurationsdatei oder in einem Befehl mit der Option `--region` an.

```
$ aws configure set default.iotfleetwise.use_dualstack_endpoint true
```

Anstatt die Dual-Stack-Endpunkte für alle Befehle zu verwenden, gehen Sie wie folgt vor, um diese Endpunkte für bestimmte Befehle zu verwenden:

- Sie können den Dual-Stack-Endpunkt für bestimmte Befehle verwenden, indem Sie die `--endpoint-url` Parameter für diese Befehle festlegen. Im folgenden Befehl können Sie beispielsweise das Zeichen *<endpoint-url>* to `iotfleetwise.`*<region>*`.api.aws` ersetzen.

  ```
  aws iotfleetwise list-fleets \
    --endpoint-url <endpoint-url>
  ```

- Sie können separate Profile in Ihrer AWS Config-Datei einrichten. Erstellen Sie beispielsweise ein Profil, das `use_dualstack_endpoint` auf true gesetzt ist, und ein Profil, das nicht festgelegt ist`use_dualstack_endpoint`. Wenn Sie einen Befehl ausführen, geben Sie an, welches Profil Sie verwenden wollen, abhängig davon, ob Sie den Dual-Stack-Endpunkt verwenden wollen oder nicht.

Tutorial: Erste Schritte mit AWS IoT FleetWise

Mit AWS IoT FleetWise können Sie Ihre Fahrzeugdaten sammeln, transformieren und übertragen. Verwenden Sie das Tutorial in diesem Abschnitt, um mit AWS IoT zu beginnen FleetWise.

In den folgenden Themen erfahren Sie mehr über AWS IoT FleetWise:

- AWS FleetWise IoT-Daten in die Cloud aufnehmen
- AWS FleetWise IoT-Fahrzeuge modellieren
- AWS FleetWise IoT-Fahrzeuge verwalten
- Verwaltung von Flotten im Internet der Dinge AWS FleetWise
- Sammeln Sie AWS FleetWise IoT-Daten mit Kampagnen

Einführung

Nutzen Sie AWS IoT, FleetWise um das einzigartige Datenformat nahezu in Echtzeit von automatisierten Fahrzeugen zu sammeln, zu transformieren und in die Cloud zu übertragen. Sie haben Zugang zu flottenweiten Erkenntnissen. Dies kann Ihnen helfen, Probleme mit dem Fahrzeugzustand effizient zu erkennen und zu beheben, hochwertige Datensignale zu übertragen und Probleme aus der Ferne zu diagnostizieren und gleichzeitig die Kosten zu senken.

Dieses Tutorial zeigt Ihnen, wie Sie mit AWS IoT beginnen können FleetWise. Sie lernen, wie Sie ein Fahrzeugmodell (Modellmanifest), ein Decoder-Manifest, ein Fahrzeug und eine Kampagne erstellen.

Weitere Informationen zu den wichtigsten Komponenten und Konzepten von AWS IoT FleetWise finden Sie unterDie wichtigsten Konzepte und Funktionen von AWS IoT FleetWise.

Geschätzte Zeit: Ungefähr 45 Minuten.

> **Important**
>
> Die AWS FleetWise IoT-Ressourcen, die diese Demo erstellt und verbraucht, werden Ihnen in Rechnung gestellt. Weitere Informationen finden Sie unter AWS IoT auf FleetWise der Seite mit den AWS FleetWise IoT-Preisen.

Voraussetzungen

Um dieses Tutorial „Erste Schritte“ abzuschließen, benötigen Sie zunächst Folgendes:

- Ein AWS-Konto. Falls Sie noch keinen haben AWS-Konto, finden Sie weitere Informationen unter Erstellen eines AWS-Konto im AWS -Kontenverwaltung Referenzhandbuch.
- Zugriff auf einen AWS-Region , der AWS IoT unterstützt FleetWise. Derzeit FleetWise wird AWS IoT in den USA Ost (Nord-Virginia) und Europa (Frankfurt) unterstützt. Sie können die Regionsauswahl in verwenden AWS-Managementkonsole , um zu einer dieser Regionen zu wechseln. Weitere Informationen finden Sie unter AWS FleetWise IoT-Endpunkte und Kontingente.
- Amazon Timestream Timestream-Ressourcen:
  - Eine Amazon Timestream Timestream-Datenbank. Weitere Informationen finden Sie unter Create a database im Amazon Timestream Developer Guide.
  - Eine in Amazon Timestream erstellte Amazon Timestream Timestream-Tabelle, die Ihre Daten enthält. Weitere Informationen finden Sie unter Erstellen einer Tabelle im Amazon Timestream Developer Guide.
- Die Edge Agent-Softwaredemo. (Anweisungen zum Einrichten der Demo finden Sie im nächsten Schritt.)
  - Sie können die Explore Edge Agent-Schnellstart-Demo verwenden, um sich mit AWS IoT FleetWise vertraut zu machen und zu erfahren, wie Sie Edge Agent-Software für AWS IoT entwickeln FleetWise. Diese Demo verwendet eine CloudFormation Vorlage. Es führt Sie durch die Überprüfung der Edge Agent-Referenzimplementierung, die Entwicklung Ihres Edge-Agents und die anschließende Bereitstellung Ihrer Edge-Agent-Software auf einem Amazon EC2 Graviton sowie die Generierung von Beispielfahrzeugdaten. Die Demo bietet auch ein Skript, mit dem Sie einen Signalkatalog, ein Fahrzeugmodell, ein Decoder-Manifest, ein Fahrzeug, eine Flotte und eine Kampagne erstellen können — alles in der Cloud.
  - Um die Demo herunterzuladen, navigieren Sie zur AWS FleetWise IoT-Konsole. Wählen Sie auf der Service-Startseite im FleetWise Abschnitt Erste Schritte mit AWS IoT die Option Explore Edge Agent aus.

Schritt 1: Richten Sie die Edge Agent-Software für AWS IoT ein FleetWise

> **Note**
>
> Der CloudFormation Stack in diesem Schritt verwendet Telemetriedaten. Sie können einen CloudFormation Stapel auch mithilfe von Bildverarbeitungssystemdaten erstellen. Weitere Informationen finden Sie im Vision System Data Developer Guide.
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Ihre Edge Agent-Software für AWS IoT FleetWise erleichtert die Kommunikation zwischen Fahrzeugen und der Cloud. Sie erhält von Datenerfassungsprogrammen Anweisungen zur Erfassung von Daten aus mit der Cloud verbundenen Fahrzeugen.

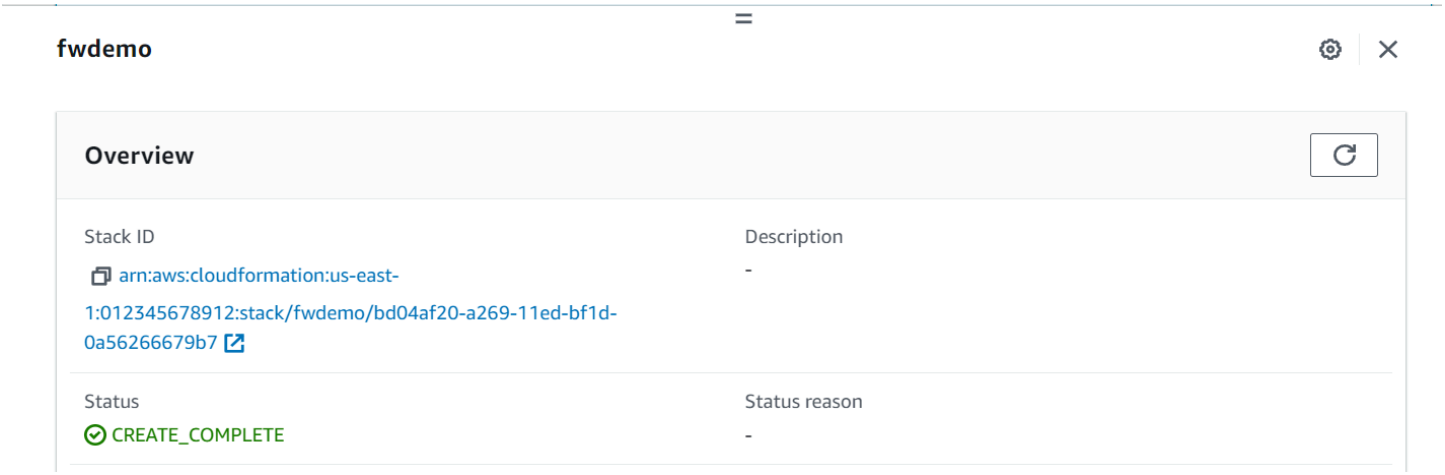
Gehen Sie unter Allgemeine Informationen wie folgt vor, um Ihre Edge Agent-Software einzurichten:

1. Öffnen Sie die CloudFormation Startvorlage.
2. Geben Sie auf der Seite Quick create stack unter Stackname den Namen Ihres Stacks von AWS FleetWise IoT-Ressourcen ein. Ein Stack ist ein benutzerfreundlicher Name, der als Präfix in den Namen der Ressourcen erscheint, die diese CloudFormation Vorlage erstellt.
3. Geben Sie unter Parameter Ihre benutzerdefinierten Werte für die Parameter ein, die sich auf Ihren Stack beziehen.

   a. Flottengröße — Sie können die Anzahl der Fahrzeuge in Ihrer Flotte erhöhen, indem Sie den Parameter Fleetsize aktualisieren.

   b. TCoreIo-Region - Sie können die Region angeben, in der das Objekt erstellt wird AWS IoT , indem Sie den Parameter Io-Region aktualisieren. TCore Sie müssen dieselbe Region verwenden, in der Sie Ihre AWS FleetWise IoT-Fahrzeuge erstellt haben. Weitere Informationen finden Sie AWS-Regionen unter Regionen und Zonen — Amazon Elastic Compute Cloud.
4. Wählen Sie im Abschnitt Funktionen das Kästchen aus, um zu bestätigen, dass CloudFormation dadurch IAM-Ressourcen erstellt werden.
5. Wählen Sie Stack erstellen und warten Sie dann etwa 15 Minuten, bis der Status des Stacks CREATE_COMPLETE anzeigt.

6. Um zu bestätigen, dass der Stack erstellt wurde, wählen Sie den Tab Stack-Info, aktualisieren Sie die Ansicht und suchen Sie nach CREATE_COMPLETE.



> **Important**
>
> Die AWS FleetWise IoT-Ressourcen, die diese Demo erstellt und verbraucht, werden Ihnen in Rechnung gestellt. Weitere Informationen finden Sie unter AWS IoT auf FleetWise der Seite mit den AWS FleetWise IoT-Preisen.

Schritt 2: Erstellen Sie ein Fahrzeugmodell

> **Important**
>
> Sie können kein Fahrzeugmodell mit Datensignalen des Bildverarbeitungssystems in der AWS FleetWise IoT-Konsole erstellen. Verwenden Sie stattdessen die AWS CLI.

Sie verwenden Fahrzeugmodelle, um das Format Ihrer Fahrzeuge zu standardisieren und die Beziehung zwischen den Signalen in den von Ihnen erstellten Fahrzeugen zu definieren. Ein Signalkatalog wird auch erstellt, wenn Sie ein Fahrzeugmodell erstellen. Ein Signalkatalog ist eine Sammlung standardisierter Signale, die zur Erstellung von Fahrzeugmodellen wiederverwendet werden können. Signale sind grundlegende Strukturen, die Sie so definieren, dass sie Fahrzeugdaten und ihre Metadaten enthalten. Derzeit unterstützt der AWS FleetWise IoT-Dienst nur einen Signalkatalog AWS-Region pro Konto. Auf diese Weise kann überprüft werden, ob die von einer Fahrzeugflotte verarbeiteten Daten konsistent sind.

Um ein Fahrzeugmodell zu erstellen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
3. Wählen Sie auf der Seite Fahrzeugmodelle die Option Fahrzeugmodell erstellen aus.
4. Geben Sie im Abschnitt Allgemeine Informationen den Namen Ihres Fahrzeugmodells, z. B. Vehicle1, und eine optionale Beschreibung ein. Klicken Sie anschließend auf Weiter.
5. Wählen Sie ein oder mehrere Signale aus dem Signalkatalog aus. Sie können Signale im Suchkatalog nach Namen filtern oder sie aus der Liste auswählen. Sie können beispielsweise Signale für Reifendruck und Bremsdruck auswählen, um Daten zu diesen Signalen zu sammeln. Wählen Sie Weiter aus.
6. Wählen Sie Ihre DBC-Dateien aus und laden Sie sie von Ihrem lokalen Gerät hoch. Wählen Sie Weiter aus.

   > **Note**
   >
   > Für dieses Tutorial können Sie eine .dbc-Beispieldatei herunterladen, um sie für diesen Schritt hochzuladen.

7. Fügen Sie Ihrem Fahrzeugmodell Attribute hinzu und wählen Sie dann Weiter.

   a. Name - Geben Sie den Namen des Fahrzeugattributs ein, z. B. den Herstellernamen oder das Herstellungsdatum.

   b. Datentyp - Wählen Sie im Menü Datentyp einen Datentyp aus.

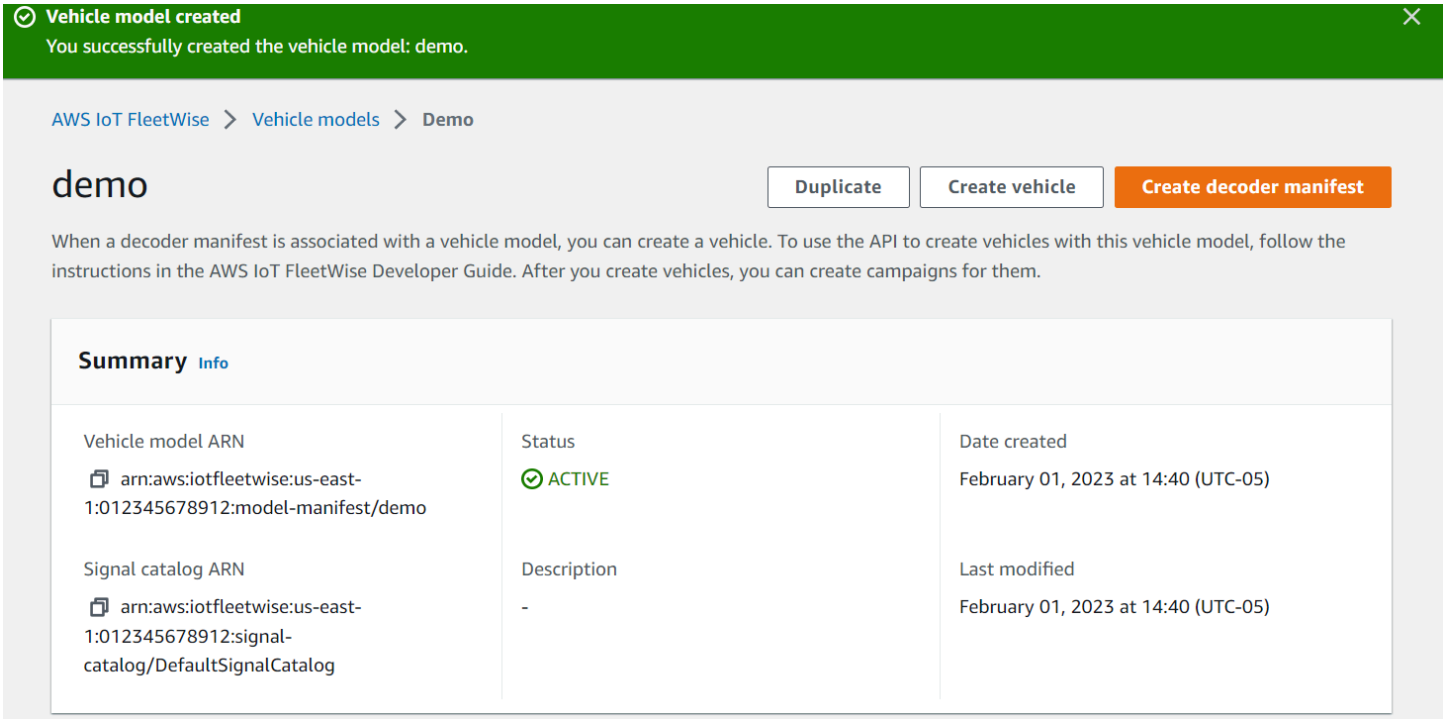
   c. Einheit - (optional) Geben Sie einen Einheitswert ein, z. B. Kilometer oder Celsius.

   d. Pfad — (Optional) Geben Sie einen Namen für den Pfad zu einem Signal ein, z. B. `Vehicle.Engine.Light.` Der Punkt (.) gibt an, dass es sich um ein untergeordnetes Signal handelt.

   e. Standardwert - (Optional) Geben Sie einen Standardwert ein.

   f. Beschreibung - (Optional) Geben Sie eine Beschreibung des Attributs ein.

8. Überprüfen Sie Ihre Konfigurationen. Sobald Sie bereit sind, klicken Sie auf Create (Erstellen). Es wird eine Benachrichtigung angezeigt, dass Ihr Fahrzeugmodell erfolgreich erstellt wurde.



Schritt 3: Erstellen Sie ein Decoder-Manifest

Decoder-Manifeste sind den Fahrzeugmodellen zugeordnet, die Sie erstellen. Sie enthalten Informationen, die dem AWS IoT helfen, Fahrzeugdaten aus einem Binärformat zu FleetWise dekodieren und in menschenlesbare Werte umzuwandeln, die analysiert werden können. Netzwerkschnittstellen und Decodersignale sind Komponenten, die bei der Konfiguration von Decoder-Manifesten helfen. Eine Netzwerkschnittstelle enthält Informationen über das CAN- oder OBD-Protokoll, das Ihr Fahrzeugnetzwerk verwendet. Das Decodersignal liefert Dekodierungsinformationen für ein bestimmtes Signal.

Um ein Decoder-Manifest zu erstellen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
3. Wählen Sie im Abschnitt Fahrzeugmodelle das Fahrzeugmodell aus, das Sie zum Erstellen eines Decoder-Manifests verwenden möchten.
4. Wählen Sie Decoder-Manifest erstellen.

Schritt 4: Konfigurieren Sie ein Decoder-Manifest

Um ein Decoder-Manifest zu konfigurieren

> **Important**
>
> Sie können mit der AWS FleetWise IoT-Konsole keine Datensignale von Bildverarbeitungssystemen in Decoder-Manifesten konfigurieren. Verwenden Sie stattdessen die AWS CLI. Weitere Informationen finden Sie unter Erstellen Sie ein Decoder-Manifest ()AWS CLI.

1. Geben Sie einen Namen und eine optionale Beschreibung ein, damit Sie Ihr Decoder-Manifest leichter identifizieren können. Wählen Sie anschließend Weiter.
2. Um eine oder mehrere Netzwerkschnittstellen hinzuzufügen, wählen Sie entweder den Typ CAN_INTERFACE oder den Typ OBD_INTERFACE.
   - On-Board-Diagnoseschnittstelle (OBD) — Wählen Sie diesen Schnittstellentyp, wenn Sie ein Protokoll benötigen, das definiert, wie Selbstdiagnosedaten zwischen elektronischen Steuergeräten () übertragen werden. ECUs Dieses Protokoll bietet eine Reihe von Standard-Diagnose-Fehlercodes (DTCs), die Ihnen bei der Behebung von Problemen mit Ihrem Fahrzeug helfen können.
   - CAN-Bus-Schnittstelle (Controller Area Network) - Wählen Sie diesen Schnittstellentyp, wenn Sie ein Protokoll benötigen, das festlegt, wie Daten untereinander ausgetauscht werden ECUs. ECUs können Motorsteuergeräte, Airbags oder das Audiosystem sein.
3. Geben Sie einen Namen für die Netzwerkschnittstelle ein.
4. Um der Netzwerkschnittstelle Signale hinzuzufügen, wählen Sie ein oder mehrere Signale aus der Liste aus.
5. Wählen Sie ein Decodersignal für das Signal, das Sie im vorherigen Schritt hinzugefügt haben. Laden Sie eine DBC-Datei hoch, um Dekodierungsinformationen bereitzustellen. Jedes Signal im Fahrzeugmodell muss mit einem Decodersignal gekoppelt werden, das Sie aus der Liste auswählen können.
6. Um eine weitere Netzwerkschnittstelle hinzuzufügen, wählen Sie Netzwerkschnittstelle hinzufügen. Wenn Sie mit dem Hinzufügen von Netzwerkschnittstellen fertig sind, wählen Sie Weiter.

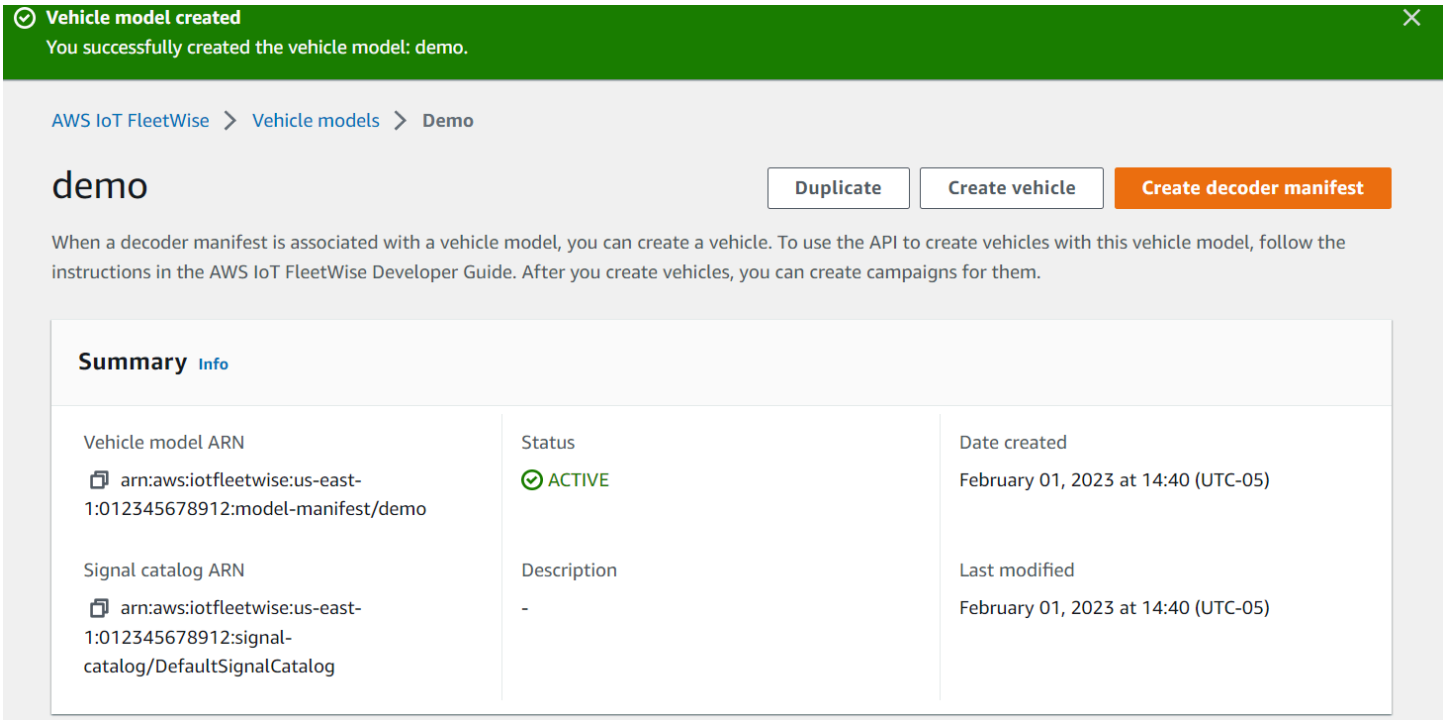
7. Überprüfen Sie Ihre Konfigurationen und wählen Sie dann Erstellen. Es wird eine Benachrichtigung angezeigt, dass Ihr Decoder-Manifest erfolgreich erstellt wurde.

Schritt 5: Erstellen Sie ein Fahrzeug

Im AWS IoT FleetWise der Dinge sind Fahrzeuge virtuelle Repräsentationen Ihres realen, physischen Fahrzeugs. Alle Fahrzeuge, die mit demselben Fahrzeugmodell erstellt wurden, erben dieselbe Gruppe von Signalen, und jedes Fahrzeug, das Sie erstellen, entspricht einem neu erstellten IoT-Ding. Sie müssen alle Fahrzeuge einem Decoder-Manifest zuordnen.

Voraussetzungen

1. Stellen Sie sicher, dass Sie das Fahrzeugmodell und das Decoder-Manifest bereits erstellt haben. Stellen Sie außerdem sicher, dass der Status des Fahrzeugmodells AKTIV ist.
   a. Um zu überprüfen, ob der Status des Fahrzeugmodells AKTIV ist, öffnen Sie die AWS FleetWise IoT-Konsole.
   b. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
   c. Überprüfen Sie im Abschnitt Zusammenfassung unter Status den Status Ihres Fahrzeugs.



Um ein Fahrzeug zu erstellen

1. Öffnen Sie die FleetWise AWS-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeuge aus.
3. Wählen Sie Fahrzeug erstellen aus.
4. Um die Fahrzeugeigenschaften zu definieren, geben Sie den Fahrzeugnamen ein und wählen Sie dann ein Modellmanifest (Fahrzeugmodell) und ein Decoder-Manifest aus.
5. (Optional) Um die Fahrzeugattribute zu definieren, geben Sie ein Schlüssel-Wert-Paar ein und wählen Sie dann Attribute hinzufügen.
6. (Optional) Um Ihre AWS-Ressource zu kennzeichnen, fügen Sie Tags hinzu und wählen Sie dann Neues Tag hinzufügen.
7. Wählen Sie Weiter aus.
8. Um das Fahrzeugzertifikat zu konfigurieren, können Sie entweder Ihr eigenes Zertifikat hochladen oder die Option Neues Zertifikat automatisch generieren wählen. Wir empfehlen, Ihr Zertifikat automatisch zu generieren, um die Einrichtung zu beschleunigen. Wenn Sie bereits über ein Zertifikat verfügen, können Sie es stattdessen verwenden.
9. Laden Sie die öffentlichen und privaten Schlüsseldateien herunter und wählen Sie dann Weiter.
10. Um dem Fahrzeugzertifikat eine Richtlinie anzuhängen, können Sie entweder einen vorhandenen Richtliniennamen eingeben oder eine neue Richtlinie erstellen. Um eine neue Richtlinie zu erstellen, wählen Sie Richtlinie erstellen und dann Weiter.
11. Überprüfen Sie Ihre Konfigurationen. Wenn Sie fertig sind, wählen Sie Fahrzeug erstellen.

Schritt 6: Erstelle eine Kampagne

Im AWS IoT werden Kampagnen verwendet FleetWise, um die Auswahl, Erfassung und Übertragung von Daten von Fahrzeugen in die Cloud zu erleichtern. Kampagnen enthalten Datenerfassungsschemata, die der Edge Agent-Software Anweisungen zur Erfassung von Daten mit einem zustandsbasierten Erfassungsschema oder einem zeitbasierten Erfassungsschema geben.

So erstellen Sie eine Kampagne

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Kampagnen aus.
3. Wählen Sie Create campaign (Kampagne erstellen).

4. Geben Sie Ihren Kampagnennamen und eine optionale Beschreibung ein.
5. Um das Datenerfassungsschema Ihrer Kampagne zu konfigurieren, können Sie das Datenerfassungsschema manuell definieren oder eine JSON-Datei von Ihrem lokalen Gerät hochladen. Durch das Hochladen einer.json-Datei wird das Datenerfassungsschema automatisch definiert.

   a. Um das Datenerfassungsschema manuell zu definieren, wählen Sie Datenerfassungsschema definieren und wählen Sie den Typ des Datenerfassungsschemas aus, das Sie für Ihre Kampagne verwenden möchten. Sie können entweder ein auf Bedingungen basierendes Erfassungsschema oder ein zeitbasiertes Erfassungsschema wählen.

   b. Wenn Sie sich für ein zeitbasiertes Erfassungsschema entscheiden, müssen Sie den Zeitraum angeben, für den Ihre Kampagne die Fahrzeugdaten erfasst.

   c. Wenn Sie sich für ein zustandsabhängiges Erfassungsschema entscheiden, müssen Sie einen Ausdruck angeben, um zu erkennen, welche Daten erfasst werden sollen. Achten Sie darauf, den Namen des Signals als Variable, als Vergleichsoperator und als Vergleichswert anzugeben.

   d. (Optional) Wählen Sie die Sprachversion Ihres Ausdrucks oder behalten Sie den Standardwert 1 bei.

   e. (Optional) Geben Sie das Triggerintervall zwischen zwei Datenerfassungsereignissen an.

   f. Um Daten zu sammeln, wählen Sie die Bedingung Triggermodus für die Edge Agent-Software. Standardmäßig sammelt die Edge Agent for AWS FleetWise IoT-Software immer dann Daten, wenn die Bedingung erfüllt ist. Oder er kann nur Daten sammeln, wenn die Bedingung zum ersten Mal erfüllt ist (beim ersten Trigger).

   g. (Optional) Sie können erweiterte Schemaoptionen wählen.
6. Um die Signale anzugeben, aus denen das Datenerfassungsschema Daten sammelt, suchen Sie im Menü nach dem Namen des Signals.
7. (Optional) Sie können eine maximale Probenanzahl oder ein minimales Probenintervall wählen. Sie können auch weitere Signale hinzufügen.
8. Wählen Sie Weiter aus.
9. Definieren Sie das Speicherziel, an das die Kampagne Daten übertragen soll. Sie können Daten in Amazon S3 oder Amazon Timestream speichern.

   a. Amazon S3 — Wählen Sie den S3-Bucket aus, der AWS IoT FleetWise über Berechtigungen verfügt.

   b. Amazon Timestream — wählen Sie die Timestream-Datenbank und den Tabellennamen aus. Geben Sie eine IAM-Rolle ein, die das Senden von Daten AWS IoT FleetWise an Timestream ermöglicht.
10. Wählen Sie Weiter aus.
11. Wählen Sie Fahrzeugattribute oder Fahrzeugnamen aus dem Suchfeld aus.
12. Geben Sie den Wert ein, der sich auf das Attribut oder den Namen bezieht, den Sie für Ihr Fahrzeug ausgewählt haben.
13. Wählen Sie die Fahrzeuge aus, von denen Ihre Kampagne Daten sammeln soll. Wählen Sie anschließend Weiter.
14. Überprüfen Sie die Konfigurationen Ihrer Kampagne und wählen Sie dann Kampagne erstellen. Sie oder Ihr Team müssen die Kampagne für Fahrzeuge bereitstellen.

Schritt 7: Bereinigen

Um weitere Gebühren für die Ressourcen zu vermeiden, die Sie in diesem Tutorial verwendet haben, löschen Sie den CloudFormation Stapel und alle Stack-Ressourcen.

Um den CloudFormation Stapel zu löschen

1. Öffnen Sie die CloudFormation -Konsole.
2. Wählen Sie aus der Liste der Stacks den Stapel aus, den Sie in Schritt 1 erstellt haben.
3. Wählen Sie Löschen aus.
4. Um die Löschung zu bestätigen, klicken Sie auf Delete (Löschen). Das Löschen des Stacks dauert etwa 15 Minuten.

Nächste Schritte

1. Sie können die Fahrzeugdaten, die Ihre Kampagne sammelt, verarbeiten und visualisieren. Weitere Informationen finden Sie unter Visualisieren Sie AWS FleetWise IoT-Fahrzeugdaten.
2. Sie können Probleme mit AWS IoT beheben und lösen FleetWise. Weitere Informationen finden Sie unter AWS IoT-Problembehebung FleetWise.

AWS FleetWise IoT-Daten in die Cloud aufnehmen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Die Edge Agent for AWS FleetWise IoT-Software ist darauf ausgelegt, die sichere Kommunikation zwischen Ihren Fahrzeugen und der Cloud zu ermöglichen, wenn sie in Fahrzeugen installiert ist und läuft.

> **Note**
>
> - AWS IoT FleetWise ist nicht für den Einsatz in oder in Verbindung mit dem Betrieb gefährlicher Umgebungen oder kritischer Systeme vorgesehen, die zu schweren Körperverletzungen oder zum Tod führen oder Umwelt- oder Sachschäden verursachen können. Fahrzeugdaten, die durch Ihre Nutzung von AWS IoT FleetWise gesammelt werden, dienen nur zu Informationszwecken, und Sie dürfen AWS IoT nicht FleetWise zur Steuerung oder Bedienung von Fahrzeugfunktionen verwenden.
> - Fahrzeugdaten, die durch Ihre Nutzung des AWS IoT erfasst wurden, FleetWise sollten auf ihre Genauigkeit geprüft werden, die für Ihren Anwendungsfall angemessen ist, auch zum Zwecke der Erfüllung aller Compliance-Verpflichtungen, die Sie möglicherweise gemäß den geltenden Fahrzeugsicherheitsvorschriften haben (wie Sicherheitsüberwachungs- und Berichtspflichten). Eine solche Bewertung sollte die Erfassung und Überprüfung von Informationen mit anderen branchenüblichen Mitteln und Quellen (z. B. Berichte von Fahrzeugführern) umfassen.

Gehen Sie wie folgt vor, um Daten in die Cloud aufzunehmen:

1. Entwickeln und installieren Sie Ihre Edge Agent for AWS FleetWise IoT-Software in Ihrem Fahrzeug. Für weitere Informationen zur Arbeit mit der Edge Agent-Software gehen Sie wie folgt vor, um das Edge Agent for AWS FleetWise IoT-Softwareentwicklerhandbuch herunterzuladen.

   1. Navigieren Sie zur AWS FleetWiseIoT-Konsole.

   2. Wählen Sie auf der Service-Startseite im FleetWise Abschnitt Erste Schritte mit AWS IoT die Option Explore Edge Agent aus.

2. Erstellen oder importieren Sie einen Signalkatalog mit Signalen, die Sie zur Erstellung eines Fahrzeugmodells verwenden werden. Weitere Informationen erhalten Sie unter Erstellen Sie einen AWS FleetWise IoT-Signalkatalog und Importieren Sie einen Signalkatalog (AWS CLI).

   > **Note**
   >
   > - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um das erste Fahrzeugmodell zu erstellen, müssen Sie keinen Signalkatalog manuell erstellen. Wenn Sie Ihr erstes Fahrzeugmodell erstellen, erstellt AWS IoT FleetWise automatisch einen Signalkatalog für Sie. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.
   > - AWS IoT unterstützt FleetWise derzeit einen Signalkatalog für jedes AWS Konto pro AWS-Region.

3. Verwenden Sie Signale im Signalkatalog, um ein Fahrzeugmodell zu erstellen. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.

   > **Note**
   >
   > - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Fahrzeugmodell zu erstellen, können Sie .dbc-Dateien hochladen, um Signale zu importieren. .dbc ist ein Dateiformat, das Controller Area Network (CAN-Bus) -Datenbanken unterstützen. Nachdem das Fahrzeugmodell erstellt wurde, werden dem Signalkatalog automatisch neue Signale hinzugefügt. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.
   > - Wenn Sie die `CreateModelManifest` API-Operation verwenden, um ein Fahrzeugmodell zu erstellen, müssen Sie die `UpdateModelManifest` API-Operation verwenden, um das Fahrzeugmodell zu aktivieren. Weitere Informationen finden Sie unter Aktualisieren Sie ein AWS FleetWise IoT-Fahrzeugmodell.
   > - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Fahrzeugmodell zu erstellen, aktiviert AWS IoT das Fahrzeugmodell FleetWise automatisch für Sie.

4. Erstellen Sie ein Decoder-Manifest. Das Decoder-Manifest enthält Dekodierungsinformationen für jedes Signal, das im Fahrzeugmodell angegeben ist, das Sie im vorherigen Schritt erstellt

haben. Das Decoder-Manifest ist dem Fahrzeugmodell zugeordnet, das Sie erstellt haben. Weitere Informationen finden Sie unter AWS FleetWise IoT-Decoder-Manifeste verwalten.

> **Note**
>
> - Wenn Sie den `CreateDecoderManifest` API-Vorgang verwenden, um ein Decoder-Manifest zu erstellen, müssen Sie den `UpdateDecoderManifest` API-Vorgang verwenden, um das Decoder-Manifest zu aktivieren. Weitere Informationen finden Sie unter Ein AWS FleetWise IoT-Decoder-Manifest aktualisieren.
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Decoder-Manifest zu erstellen, aktiviert AWS IoT das Decoder-Manifest FleetWise automatisch für Sie.

5. Erstellen Sie Fahrzeuge anhand des Fahrzeugmodells. Fahrzeuge, die mit demselben Fahrzeugmodell erstellt wurden, erben dieselbe Gruppe von Signalen. Sie müssen AWS IoT Core es für die Bereitstellung Ihres Fahrzeugs verwenden, bevor Sie Daten in die Cloud aufnehmen können. Weitere Informationen finden Sie unter AWS FleetWise IoT-Fahrzeuge verwalten.
6. (Optional) Erstellen Sie eine Flotte, die eine Gruppe von Fahrzeugen repräsentiert, und ordnen Sie dann einzelne Fahrzeuge der Flotte zu. Auf diese Weise können Sie mehrere Fahrzeuge gleichzeitig verwalten. Weitere Informationen finden Sie unter Verwaltung von Flotten im Internet der Dinge AWS FleetWise.
7. (Optional) Erstellen Sie Kampagnen. Kampagnen werden für ein Fahrzeug oder eine Fahrzeugflotte eingesetzt. Kampagnen geben der Edge Agent-Software Anweisungen zur Auswahl, Erfassung und Übertragung von Daten in die Cloud. Weitere Informationen finden Sie unter Sammeln Sie AWS FleetWise IoT-Daten mit Kampagnen. Sie können Kampagnen, Statusvorlagen (unten) oder beides erstellen, um Daten zu sammeln.

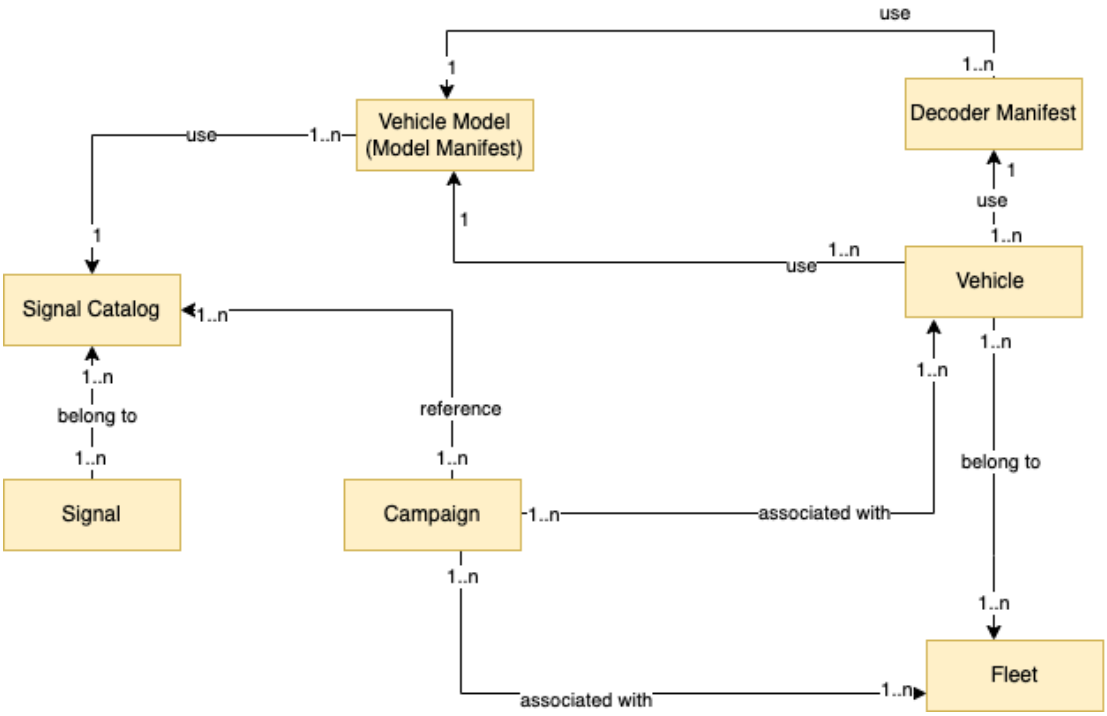
   > **Note**
   >
   > Sie müssen den `UpdateCampaign` API-Vorgang verwenden, um die Kampagne zu genehmigen, bevor AWS IoT sie für das Fahrzeug oder die Flotte bereitstellen FleetWise kann. Weitere Informationen finden Sie unter Aktualisieren Sie eine AWS FleetWise IoT-Kampagne.

8. (Optional) Erstellen Sie Statusvorlagen. Statusvorlagen werden für ein Fahrzeug bereitgestellt. Vorlagen für Bundesstaaten bieten Fahrzeugbesitzern die Möglichkeit, den Zustand ihres Fahrzeugs nachzuverfolgen. Weitere Informationen finden Sie unter Überwachen Sie den letzten bekannten Zustand Ihrer Fahrzeuge.

Die Edge Agent-Software überträgt Fahrzeugdaten AWS IoT Core mithilfe eines von Ihnen ausgewählten MQTT-Themas an. Um die Daten FleetWise für Kampagnen an AWS IoT zu senden, wird das reservierte Thema verwendet`$aws/iotfleetwise/vehicles/vehicleName/signals`. Für den letzten bekannten Status verwendet der Edge-Agent das reservierte Thema`$aws/iotfleetwise/vehicles/vehicleName/last_known_states/data`. Weitere Informationen zur Verarbeitung der aufgenommenen Daten finden Sie unterVisualisieren Sie AWS FleetWise IoT-Fahrzeugdaten.

AWS FleetWise IoT-Fahrzeuge modellieren

AWS IoT FleetWise bietet ein Framework für die Fahrzeugmodellierung, mit dem Sie virtuelle Darstellungen Ihrer Fahrzeuge in der Cloud erstellen können. Signale, Signalkataloge, Fahrzeugmodelle und Decoder-Manifeste sind die Kernkomponenten, mit denen Sie bei der Modellierung Ihrer Fahrzeuge arbeiten.



Signal

Signale sind grundlegende Strukturen, die Sie so definieren, dass sie Fahrzeugdaten und ihre Metadaten enthalten. Ein Signal kann ein Attribut, ein Zweig, ein Sensor oder ein Aktuator sein. Sie können beispielsweise einen Sensor erstellen, der die Temperaturwerte im Fahrzeug empfängt und dessen Metadaten, einschließlich eines Sensornamens, eines Datentyps und einer Einheit, speichert. Weitere Informationen finden Sie unter AWS FleetWise IoT-Signalkataloge verwalten.

Signalkatalog

Ein Signalkatalog enthält eine Sammlung von Signalen. Signale in einem Signalkatalog können verwendet werden, um Fahrzeuge zu modellieren, die unterschiedliche Protokolle und Datenformate verwenden. Beispielsweise gibt es zwei Fahrzeuge, die von verschiedenen Autoherstellern hergestellt werden: eines verwendet das Control Area Network (CAN-Bus) -

Protokoll, das andere das On-Board Diagnostics (OBD) -Protokoll. Sie können im Signalkatalog einen Sensor für den Empfang von Fahrzeugtemperaturwerten definieren. Dieser Sensor kann zur Darstellung der Thermoelemente in beiden Fahrzeugen verwendet werden. Weitere Informationen finden Sie unter AWS FleetWise IoT-Signalkataloge verwalten.

Fahrzeugmodell (Modellmanifest)

Fahrzeugmodelle sind deklarative Strukturen, mit denen Sie das Format Ihrer Fahrzeuge standardisieren und Beziehungen zwischen Signalen in den Fahrzeugen definieren können. Fahrzeugmodelle sorgen für konsistente Informationen für mehrere Fahrzeuge desselben Typs. Sie fügen Signale hinzu, um Fahrzeugmodelle zu erstellen. Weitere Informationen finden Sie unter AWS FleetWise IoT-Fahrzeugmodelle verwalten.

Decoder-Manifest

Decoder-Manifeste enthalten Dekodierungsinformationen für jedes Signal in Fahrzeugmodellen. Sensoren und Aktuatoren in Fahrzeugen übertragen Nachrichten auf niedriger Ebene (Binärdaten). Mit Decoder-Manifesten FleetWise ist AWS IoT in der Lage, Binärdaten in menschenlesbare Werte umzuwandeln. Jedes Decoder-Manifest ist einem Fahrzeugmodell zugeordnet. Weitere Informationen finden Sie unter AWS FleetWise IoT-Decoder-Manifeste verwalten.

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um Fahrzeuge auf folgende Weise zu modellieren.

1. Erstellen oder importieren Sie einen Signalkatalog mit Signalen, die Sie zur Erstellung eines Fahrzeugmodells verwenden werden. Weitere Informationen erhalten Sie unter Erstellen Sie einen AWS FleetWise IoT-Signalkatalog und Importieren Sie einen Signalkatalog (AWS CLI).

> **Note**
>
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um das erste Fahrzeugmodell zu erstellen, müssen Sie keinen Signalkatalog manuell erstellen. Wenn Sie Ihr erstes Fahrzeugmodell erstellen, erstellt AWS IoT FleetWise automatisch einen Signalkatalog für Sie. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.
> - AWS IoT unterstützt FleetWise derzeit einen Signalkatalog für jedes AWS Konto pro AWS-Region.

2. Verwenden Sie Signale im Signalkatalog, um ein Fahrzeugmodell zu erstellen. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.

> **Note**
>
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Fahrzeugmodell zu erstellen, können Sie .dbc-Dateien hochladen, um Signale zu importieren. .dbc ist ein Dateiformat, das Controller Area Network (CAN-Bus) -Datenbanken unterstützen. Nachdem das Fahrzeugmodell erstellt wurde, werden dem Signalkatalog automatisch neue Signale hinzugefügt. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.
> - Wenn Sie die `CreateModelManifest` API-Operation verwenden, um ein Fahrzeugmodell zu erstellen, müssen Sie die `UpdateModelManifest` API-Operation verwenden, um das Fahrzeugmodell zu aktivieren. Weitere Informationen finden Sie unter Aktualisieren Sie ein AWS FleetWise IoT-Fahrzeugmodell.
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Fahrzeugmodell zu erstellen, aktiviert AWS IoT das Fahrzeugmodell FleetWise automatisch für Sie.

3. Erstellen Sie ein Decoder-Manifest. Das Decoder-Manifest enthält Dekodierungsinformationen für jedes Signal, das im Fahrzeugmodell angegeben ist, das Sie im vorherigen Schritt erstellt haben. Das Decoder-Manifest ist dem Fahrzeugmodell zugeordnet, das Sie erstellt haben. Weitere Informationen finden Sie unter AWS FleetWise IoT-Decoder-Manifeste verwalten.

> **Note**
>
> - Wenn Sie den `CreateDecoderManifest` API-Vorgang verwenden, um ein Decoder-Manifest zu erstellen, müssen Sie den `UpdateDecoderManifest` API-Vorgang verwenden, um das Decoder-Manifest zu aktivieren. Weitere Informationen finden Sie unter Ein AWS FleetWise IoT-Decoder-Manifest aktualisieren.
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Decoder-Manifest zu erstellen, aktiviert AWS IoT das Decoder-Manifest FleetWise automatisch für Sie.

CAN-Bus-Datenbanken unterstützen das .dbc-Dateiformat. Sie könnten .dbc-Dateien hochladen, um Signale und Signaldecoder zu importieren. Gehen Sie wie folgt vor, um eine Beispiel-DBC-Datei zu erhalten.

Um eine.dbc-Datei abzurufen

1. Laden Sie die ZIP-Datei herunter. EngineSignals
2. Navigieren Sie zu dem Verzeichnis, in das Sie die Datei `EngineSignals.zip` heruntergeladen haben.
3. Entpacken Sie die Datei und speichern Sie sie lokal unter. `EngineSignals.dbc`

Themen

- AWS FleetWise IoT-Signalkataloge verwalten
- AWS FleetWise IoT-Fahrzeugmodelle verwalten
- AWS FleetWise IoT-Decoder-Manifeste verwalten

AWS FleetWise IoT-Signalkataloge verwalten

> **Note**
>
> Sie können ein Demo-Skript herunterladen, um ROS 2-Nachrichten in VSS-.json-Dateien zu konvertieren, die mit dem Signalkatalog kompatibel sind. Weitere Informationen finden Sie im Vision System Data Developer Guide.

Ein Signalkatalog ist eine Sammlung standardisierter Signale, die zur Erstellung von Fahrzeugmodellen wiederverwendet werden können. AWS IoT FleetWise unterstützt die Fahrzeugsignalspezifikation (VSS), anhand derer Sie Signale definieren können. Bei einem Signal kann es sich um einen der folgenden Typen handeln.

Attribut

Attribute stellen statische Informationen dar, die sich im Allgemeinen nicht ändern, wie Hersteller und Herstellungsdatum.

Verzweigung

Zweige stellen Signale in einer verschachtelten Struktur dar. Zweige zeigen Signalhierarchien. Zum Beispiel hat der `Vehicle` Zweig einen untergeordneten Zweig,`Powertrain`. Der `Powertrain` Zweig hat einen untergeordneten Zweig,`combustionEngine`. Verwenden Sie den

`Vehicle.Powertrain.combustionEngine` Ausdruck, um den `combustionEngine` Zweig zu finden.

Sensor

Sensordaten geben den aktuellen Zustand des Fahrzeugs an und ändern sich im Laufe der Zeit, wenn sich der Zustand des Fahrzeugs ändert, z. B. Flüssigkeitsstand, Temperaturen, Vibrationen oder Spannung.

Aktuator

Die Aktuatordaten geben den Zustand eines Fahrzeuggeräts wie Motoren, Heizungen und Türschlösser an. Durch Ändern des Zustands eines Fahrzeuggeräts können Aktuatordaten aktualisiert werden. Sie können beispielsweise einen Aktuator definieren, der die Heizung darstellt. Der Aktuator empfängt neue Daten, wenn Sie die Heizung ein- oder ausschalten.

Benutzerdefinierter Aufbau

Eine benutzerdefinierte Struktur (auch als Struktur bezeichnet) stellt eine komplexe Datenstruktur oder Datenstruktur höherer Ordnung dar. Sie erleichtert das logische Binden oder Gruppieren von Daten, die aus derselben Quelle stammen. Eine Struktur wird verwendet, wenn Daten in einer atomaren Operation gelesen oder geschrieben werden, z. B. um einen komplexen Datentyp oder eine Form höherer Ordnung darzustellen.

Ein Signal vom Strukturtyp wird im Signalkatalog definiert, indem ein Verweis auf einen Strukturdatentyp anstelle eines primitiven Datentyps verwendet wird. Strukturen können für alle Arten von Signalen verwendet werden, einschließlich Sensoren, Attributen, Aktuatoren und Datentypen für Bildverarbeitungssysteme. Wenn ein Signal vom Typ Structure gesendet oder empfangen wird, FleetWise erwartet AWS IoT, dass alle enthaltenen Elemente gültige Werte haben, sodass alle Elemente obligatorisch sind. Wenn eine Struktur beispielsweise die Elemente Vehicle.Camera.Image.Height, Vehicle.Camera.Image.Width und Vehicle.Camera.Image.Data enthält, wird erwartet, dass das gesendete Signal Werte für all diese Elemente enthält.

> **Note**
>
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Benutzerdefiniertes Eigentum

Eine benutzerdefinierte Eigenschaft stellt ein Element der komplexen Datenstruktur dar. Der Datentyp der Eigenschaft kann entweder primitiv oder eine andere Struktur sein.

Bei der Darstellung einer Form höherer Ordnung mithilfe einer Struktur und einer benutzerdefinierten Eigenschaft wird die beabsichtigte Form höherer Ordnung immer als Baumstruktur definiert und betrachtet. Die benutzerdefinierte Eigenschaft wird verwendet, um alle Blattknoten zu definieren, während die Struktur verwendet wird, um alle Knoten zu definieren, die keine Blattknoten sind.

> **Note**
>
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um das erste Fahrzeugmodell zu erstellen, müssen Sie keinen Signalkatalog manuell erstellen. Wenn Sie Ihr erstes Fahrzeugmodell erstellen, erstellt AWS IoT FleetWise automatisch einen Signalkatalog für Sie. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Fahrzeugmodell zu erstellen, können Sie .dbc-Dateien hochladen, um Signale zu importieren. .dbc ist ein Dateiformat, das Controller Area Network (CAN-Bus) -Datenbanken unterstützen. Nachdem das Fahrzeugmodell erstellt wurde, werden dem Signalkatalog automatisch neue Signale hinzugefügt. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.
> - AWS IoT unterstützt FleetWise derzeit einen Signalkatalog für AWS-Konto jede Region.

AWS IoT FleetWise bietet die folgenden API-Operationen, mit denen Sie Signalkataloge erstellen und verwalten können.

- CreateSignalCatalog— Erstellt einen neuen Signalkatalog.
- ImportSignalCatalog— Importiert Signale, um einen Signalkatalog zu erstellen, indem eine.json-Datei hochgeladen wird. Signale müssen gemäß VSS definiert und im JSON-Format gespeichert werden.
- UpdateSignalCatalog— Aktualisiert einen vorhandenen Signalkatalog durch Aktualisierung, Entfernung oder Hinzufügen von Signalen.

- DeleteSignalCatalog— Löscht einen vorhandenen Signalkatalog.
- ListSignalCatalogs— Ruft eine paginierte Liste mit Zusammenfassungen aller Signalkataloge ab.
- ListSignalCatalogNodes— Ruft eine paginierte Liste mit Zusammenfassungen aller Signale (Knoten) in einem bestimmten Signalkatalog ab.
- GetSignalCatalog— Ruft Informationen über einen Signalkatalog ab.

Lernprogramme

- AWS FleetWise IoT-Signale konfigurieren
- Erstellen Sie einen AWS FleetWise IoT-Signalkatalog
- Einen AWS FleetWise IoT-Signalkatalog importieren
- Aktualisieren Sie einen AWS FleetWise IoT-Signalkatalog
- Löschen Sie einen AWS FleetWise IoT-Signalkatalog
- Informationen zum AWS FleetWise IoT-Signalkatalog abrufen

AWS FleetWise IoT-Signale konfigurieren

In diesem Abschnitt erfahren Sie, wie Sie Zweige, Attribute, Sensoren und Aktoren konfigurieren.

Themen

- Zweige konfigurieren
- Attribute konfigurieren
- Sensoren oder Aktoren konfigurieren
- Konfigurieren Sie komplexe Datentypen

Zweige konfigurieren

Um einen Zweig zu konfigurieren, geben Sie die folgenden Informationen an.

- `fullyQualifiedName`— Der vollqualifizierte Name des Zweigs besteht aus dem Pfad zum Zweig plus dem Namen des Zweigs. Verwenden Sie einen Punkt (.), um auf einen untergeordneten Zweig zu verweisen. Dies `Vehicle.Chassis.SteeringWheel` ist beispielsweise der vollständig qualifizierte Name für den `SteeringWheel` Zweig. `Vehicle.Chassis.`ist der Pfad zu diesem Zweig.

Der vollqualifizierte Name kann bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9, Doppelpunkt (:) und Unterstrich (_).

- (Optional) `Description` — Die Beschreibung für den Zweig.

  Die Beschreibung kann bis zu 2048 Zeichen enthalten. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
- (Optional) `deprecationMessage` — Die veraltete Meldung für den Knoten oder Zweig, der verschoben oder gelöscht wird.

  Die DeprecationMessage kann bis zu 2048 Zeichen enthalten. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
- (Optional) `comment` — Ein Kommentar zusätzlich zur Beschreibung. Ein Kommentar kann verwendet werden, um zusätzliche Informationen über die Filiale bereitzustellen, z. B. die Begründung für die Filiale oder Verweise auf verwandte Zweige.

  Der Kommentar kann bis zu 2048 Zeichen lang sein. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).

Attribute konfigurieren

Um ein Attribut zu konfigurieren, geben Sie die folgenden Informationen an.

- `dataType`— Der Datentyp des Attributs muss einer der folgenden sein: INT8,,,,,, BOOLEAN, FLOAT UINT8 INT16, DOUBLE UINT16 INT32, STRING UINT32 INT64 UINT64, UNIX_TIMESTAMP, _ARRAY, _ARRAY, _ARRAY, _ARRAY, _ARRAY, INT8 _ARRAY, _ARRAY, UINT8 _ARRAY, INT16 _ARRAY, UINT16 _ARRAY, INT32 BOOLEAN_ARRAY, UINT32 INT64 FLOAT_ARRAY, UINT64 DOUBLE_ARRAY, STRING_ARRAY, UNIX_TIMESTAMP_ARRAY, UNKNOWN, oder ein benutzerdefinierter Struktur, die im Datentypzweig definiert ist. fullyQualifiedName
- `fullyQualifiedName`— Der vollqualifizierte Name des Attributs ist der Pfad zum Attribut plus der Name des Attributs. Verwenden Sie einen Punkt (.), um auf ein untergeordnetes Signal zu verweisen. `Vehicle.Chassis.SteeringWheel.Diameter`Ist beispielsweise der vollqualifizierte Name für das `Diameter` Attribut. `Vehicle.Chassis.SteeringWheel.`ist der Pfad zu diesem Attribut.

  Der vollqualifizierte Name kann bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich).

- (Optional) `Description` — Die Beschreibung für das Attribut.

  Die Beschreibung kann bis zu 2048 Zeichen enthalten. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
- (Optional) `unit` — Die wissenschaftliche Einheit für das Attribut, z. B. km oder Celsius.
- (Optional) `min` — Der Mindestwert des Attributs.
- (Optional) `max` — Der Höchstwert des Attributs.
- (Optional) `defaultValue` — Der Standardwert des Attributs.
- (Optional) `assignedValue` — Der dem Attribut zugewiesene Wert.
- (Optional) `allowedValues` — Eine Liste von Werten, die das Attribut akzeptiert.
- (Optional) `deprecationMessage` — Die Verfallsmeldung für den Knoten oder Zweig, der verschoben oder gelöscht wird.

  Die DeprecationMessage kann bis zu 2048 Zeichen enthalten. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
- (Optional) `comment` — Ein Kommentar zusätzlich zur Beschreibung. Ein Kommentar kann verwendet werden, um zusätzliche Informationen über das Attribut bereitzustellen, z. B. die Begründung für das Attribut oder Verweise auf verwandte Attribute.

  Der Kommentar kann bis zu 2048 Zeichen lang sein. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).

Sensoren oder Aktoren konfigurieren

Um einen Sensor oder Aktuator zu konfigurieren, geben Sie die folgenden Informationen an.

- `dataType`— Der Datentyp des Signals muss einer der folgenden sein: INT8,,,,,, BOOLEAN UINT8, FLOAT INT16 UINT16, DOUBLE INT32, STRING UINT32 INT64 UINT64, UNIX_TIMESTAMP, _ARRAY, _ARRAY, _ARRAY, _ARRAY, _ARRAY, INT8 _ARRAY, UINT8 _ARRAY, INT16 _ARRAY, _ARRAY, UINT16 _ARRAY, INT32 UINT32 BOOLEAN_ARRAY, INT64 FLOAT_ARRAY, UINT64 DOUBLE_ARRAY, STRING_ARRAY, UNIX_TIMESTAMP_ARRAY, UNKNOWN, oder ein benutzerdefinierter Struktur, die im Datentypzweig definiert ist. fullyQualifiedName
- `fullyQualifiedName`— Der vollqualifizierte Name des Signals ist der Pfad zum Signal plus der Name des Signals. Verwenden Sie einen Punkt (.), um auf ein untergeordnetes Signal zu

verweisen. Dies `Vehicle.Chassis.SteeringWheel.HandsOff.HandsOffSteeringState` ist beispielsweise der vollqualifizierte Name für den `HandsOffSteeringState` Aktuator. `Vehicle.Chassis.SteeringWheel.HandsOff.`ist der Pfad zu diesem Aktuator.

Der vollqualifizierte Name kann bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich).

- (Optional) `Description` — Die Beschreibung für das Signal.

  Die Beschreibung kann bis zu 2048 Zeichen enthalten. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
- (Optional) `unit` — Die wissenschaftliche Einheit für das Signal, z. B. km oder Celsius.
- (Optional) `min` — Der Mindestwert des Signals.
- (Optional) `max` — Der Maximalwert des Signals.
- (Optional) `assignedValue` — Der dem Signal zugewiesene Wert.
- (Optional) `allowedValues` — Liste der Werte, die das Signal akzeptiert.
- (Optional) `deprecationMessage` — Die Verfallsmeldung für den Knoten oder Zweig, der verschoben oder gelöscht wird.

  Die DeprecationMessage kann bis zu 2048 Zeichen enthalten. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
- (Optional) `comment` — Ein Kommentar zusätzlich zur Beschreibung. Ein Kommentar kann verwendet werden, um zusätzliche Informationen über den Sensor oder Aktuator bereitzustellen, z. B. dessen Begründung oder Verweise auf verwandte Sensoren oder Aktoren.

  Der Kommentar kann bis zu 2048 Zeichen lang sein. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).

Konfigurieren Sie komplexe Datentypen

Bei der Modellierung von Bildverarbeitungssystemen werden komplexe Datentypen verwendet. Diese Datentypen bestehen nicht nur aus Verzweigungen, sondern auch aus Strukturen (auch Struktur genannt) und Eigenschaften. Eine Struktur ist ein Signal, das durch mehrere Werte beschrieben wird, wie ein Bild. Eine Eigenschaft stellt ein Element der Struktur dar, z. B. einen primitiven Datentyp (wie UINT8) oder eine andere Struktur (wie Timestamp). Beispielsweise steht Vehicle.Cameras.Front für einen Zweig, Vehicle.Cameras.Front.Image für eine Struktur und Vehicle.Cameras.Timestamp für eine Eigenschaft.

Das folgende Beispiel für einen komplexen Datentyp zeigt, wie Signale und Datentypen in eine einzige JSON-Datei exportiert werden.

Example komplexer Datentyp

```
{
  "Vehicle": {
    "type": "branch"
    // Signal tree
  },
  "ComplexDataTypes": {
    "VehicleDataTypes": {
      // complex data type tree
      "children": {
        "branch": {
          "children": {
            "Struct": {
              "children": {
                "Property": {
                  "type": "property",
                  "datatype": "Data type",
                  "description": "Description",
                  //                  ...
                }
              },
              "description": "Description",
              "type": "struct"
            }
          }
          "description": "Description",
          "type": "branch"
        }
      }
    }
  }
}
```

> **Note**
>
> Sie können ein Demo-Skript herunterladen, um ROS 2-Nachrichten in VSS-.json-Dateien zu konvertieren, die mit dem Signalkatalog kompatibel sind. Weitere Informationen finden Sie im Vision System Data Developer Guide.

Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Struktur konfigurieren

Um eine benutzerdefinierte Struktur (oder Struktur) zu konfigurieren, geben Sie die folgenden Informationen an.

- `fullyQualifiedName`— Der vollqualifizierte Name der benutzerdefinierten Struktur. Der vollqualifizierte Name einer benutzerdefinierten Struktur könnte beispielsweise lauten`ComplexDataTypes.VehicleDataTypes.SVMCamera`.

  Der vollqualifizierte Name kann bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich).
- (Optional) `Description` — Die Beschreibung für das Signal.

  Die Beschreibung kann bis zu 2048 Zeichen enthalten. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
- (Optional) `deprecationMessage` — Die veraltete Meldung für den Knoten oder Zweig, der verschoben oder gelöscht wird.

  Die DeprecationMessage kann bis zu 2048 Zeichen enthalten. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
- (Optional) `comment` — Ein Kommentar zusätzlich zur Beschreibung. Ein Kommentar kann verwendet werden, um zusätzliche Informationen über den Sensor oder Aktuator bereitzustellen, z. B. dessen Begründung oder Verweise auf verwandte Sensoren oder Aktoren.

  Der Kommentar kann bis zu 2048 Zeichen lang sein. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).

Eigenschaft konfigurieren

Um eine benutzerdefinierte Eigenschaft zu konfigurieren, geben Sie die folgenden Informationen an.

- `dataType`— Der Datentyp des Signals muss einer der folgenden sein: INT8,,,,,, BOOLEAN, FLOAT UINT8 INT16, DOUBLE UINT16 INT32, STRING UINT32 INT64 UINT64, UNIX_TIMESTAMP, _ARRAY, _ARRAY, _ARRAY, _ARRAY, _ARRAY, INT8 _ARRAY, UINT8

_ARRAY, _ARRAY, INT16 _ARRAY, UINT16 INT32 BOOLEAN_ARRAY, UINT32 FLOAT_ARRAY, INT64 DOUBLE_ARRAY, UINT64 STRING_ARRAY, UNIX_TIMESTAMP_ARRAY, STRUCT, STRUCT_ARRAY, oder UNBEKANNT.

- `fullyQualifiedName`— Der vollständig qualifizierte Name der benutzerdefinierten Eigenschaft. Der vollqualifizierte Name einer benutzerdefinierten Eigenschaft könnte beispielsweise lauten`ComplexDataTypes.VehicleDataTypes.SVMCamera.FPS`.

  Der vollqualifizierte Name kann bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich)

- (Optional) `Description` — Die Beschreibung für das Signal.

  Die Beschreibung kann bis zu 2048 Zeichen enthalten. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).

- (Optional) `deprecationMessage` — Die veraltete Meldung für den Knoten oder Zweig, der verschoben oder gelöscht wird.

  Die DeprecationMessage kann bis zu 2048 Zeichen enthalten. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).

- (Optional) `comment` — Ein Kommentar zusätzlich zur Beschreibung. Ein Kommentar kann verwendet werden, um zusätzliche Informationen über den Sensor oder Aktuator bereitzustellen, z. B. dessen Begründung oder Verweise auf verwandte Sensoren oder Aktoren.

  Der Kommentar kann bis zu 2048 Zeichen lang sein. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).

- (Optional) `dataEncoding` — Gibt an, ob es sich bei der Eigenschaft um Binärdaten handelt. Die Datenkodierung der benutzerdefinierten Eigenschaft muss einer der folgenden Werte entsprechen: BINARY oder TYPED.

- (Optional) `structFullyQualifiedName` — Der vollqualifizierte Name des Strukturknotens (Struct) für die benutzerdefinierte Eigenschaft, wenn der Datentyp der benutzerdefinierten Eigenschaft Struct oder StructArray ist.

  Der vollqualifizierte Name kann bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich).

Erstellen Sie einen AWS FleetWise IoT-Signalkatalog

Sie können die CreateSignalCatalogAPI-Operation verwenden, um einen Signalkatalog zu erstellen. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um einen Signalkatalog zu erstellen.

*signal-catalog-configuration*Ersetzen Sie ihn durch den Namen der JSON-Datei, die die Konfiguration enthält.

```
aws iotfleetwise create-signal-catalog --cli-input-json file://signal-catalog-configuration.json
```

- *signal-catalog-name*Ersetzen Sie es durch den Namen des Signalkatalogs, den Sie erstellen.
- (Optional) *description* Ersetzen Sie ihn durch eine Beschreibung, die Ihnen die Identifizierung des Signalkatalogs erleichtert.

Weitere Informationen zur Konfiguration von Verzweigungen, Attributen, Sensoren und Aktoren finden Sie unterAWS FleetWise IoT-Signale konfigurieren.

```
{
    "name": "signal-catalog-name",
    "description": "description",
    "nodes": [
  {
    "branch": {
      "fullyQualifiedName": "Types"
    }
  },
  {
    "struct": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_CompressedImage"
    }
  },
  {
    "struct": {
      "fullyQualifiedName": "Types.std_msgs_Header"
    }
  },
  {
```

```
    "struct": {
      "fullyQualifiedName": "Types.builtin_interfaces_Time"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.builtin_interfaces_Time.sec",
      "dataType": "INT32",
      "dataEncoding": "TYPED"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.builtin_interfaces_Time.nanosec",
      "dataType": "UINT32",
      "dataEncoding": "TYPED"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.std_msgs_Header.stamp",
      "dataType": "STRUCT",
      "structFullyQualifiedName": "Types.builtin_interfaces_Time"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.std_msgs_Header.frame_id",
      "dataType": "STRING",
      "dataEncoding": "TYPED"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_CompressedImage.header",
      "dataType": "STRUCT",
      "structFullyQualifiedName": "Types.std_msgs_Header"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_CompressedImage.format",
      "dataType": "STRING",
      "dataEncoding": "TYPED"
```

```
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_CompressedImage.data",
      "dataType": "UINT8_ARRAY",
      "dataEncoding": "BINARY"
    }
  },
  {
    "branch": {
      "fullyQualifiedName": "Vehicle",
      "description": "Vehicle"
    }
  },
  {
    "branch": {
      "fullyQualifiedName": "Vehicle.Cameras"
    }
  },
  {
    "branch": {
      "fullyQualifiedName": "Vehicle.Cameras.Front"
    }
  },
  {
    "sensor": {
      "fullyQualifiedName": "Vehicle.Cameras.Front.Image",
      "dataType": "STRUCT",
      "structFullyQualifiedName": "Types.sensor_msgs_msg_CompressedImage"
    }
  },
  {
    "struct": {
      "fullyQualifiedName": "Types.std_msgs_msg_Float64"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.std_msgs_msg_Float64.data",
      "dataType": "DOUBLE",
      "dataEncoding": "TYPED"
    }
  },
```

```
  {
    "sensor": {
      "fullyQualifiedName": "Vehicle.Velocity",
      "dataType": "STRUCT",
      "structFullyQualifiedName": "Types.std_msgs_msg_Float64"
    }
  },
  {
    "struct": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_RegionOfInterest"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_RegionOfInterest.x_offset",
      "dataType": "UINT32",
      "dataEncoding": "TYPED"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_RegionOfInterest.y_offset",
      "dataType": "UINT32",
      "dataEncoding": "TYPED"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_RegionOfInterest.height",
      "dataType": "UINT32",
      "dataEncoding": "TYPED"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_RegionOfInterest.width",
      "dataType": "UINT32",
      "dataEncoding": "TYPED"
    }
  },
  {
    "property": {
      "fullyQualifiedName": "Types.sensor_msgs_msg_RegionOfInterest.do_rectify",
      "dataType": "BOOLEAN",
```

```
      "dataEncoding": "TYPED"
    }
  },
  {
    "branch": {
      "fullyQualifiedName": "Vehicle.Perception"
    }
  },
  {
    "sensor": {
      "fullyQualifiedName": "Vehicle.Perception.Obstacle",
      "dataType": "STRUCT",
      "structFullyQualifiedName": "Types.sensor_msgs_msg_RegionOfInterest"
    }
  }
]
}
```

> **Note**
>
> Sie können ein Demoskript herunterladen, um ROS 2-Nachrichten in VSS-.json-Dateien zu konvertieren, die mit dem Signalkatalog kompatibel sind. Weitere Informationen finden Sie im Vision System Data Developer Guide.
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `CreateSignalCatalog` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
```

```
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Einen AWS FleetWise IoT-Signalkatalog importieren

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um einen Signalkatalog zu importieren.

Themen

- Importieren Sie einen Signalkatalog (Konsole)
- Importieren Sie einen Signalkatalog (AWS CLI)

Importieren Sie einen Signalkatalog (Konsole)

Sie können die AWS FleetWise IoT-Konsole verwenden, um einen Signalkatalog zu importieren.

> **Important**
>
> Sie können maximal einen Signalkatalog haben. Wenn Sie bereits über einen Signalkatalog verfügen, wird die Option zum Importieren eines Signalkatalogs in der Konsole nicht angezeigt.

Um einen Signalkatalog zu importieren

1. Öffnen Sie die AWS FleetWiseIoT-Konsole.
2. Wählen Sie im Navigationsbereich Signal-Katalog aus.
3. Wählen Sie auf der Übersichtsseite des Signalkatalogs die Option Signalkatalog importieren aus.
4. Importieren Sie die Datei, die die Signale enthält.
   - Um eine Datei aus einem S3-Bucket hochzuladen:
     a. Wählen Sie Import aus S3 aus.

b. Wählen Sie S3 durchsuchen.

c. Geben Sie für Buckets den Bucket-Namen oder das Objekt ein, wählen Sie es aus der Liste aus und wählen Sie dann die Datei aus der Liste aus. Wählen Sie die Schaltfläche „Datei auswählen“.

Oder geben Sie für S3-URI eine Amazon Simple Storage Service-URI ein. Weitere Informationen finden Sie unter Methoden für den Zugriff auf einen Bucket im Amazon S3 S3-Benutzerhandbuch.

- Um eine Datei von Ihrem Computer hochzuladen:

a. Wählen Sie Aus Datei importieren.

b. Laden Sie eine JSON-Datei im Format Vehicle Signal Specification (VSS) hoch.

5. Überprüfen Sie den Signalkatalog und wählen Sie dann Datei importieren.

Importieren Sie einen Signalkatalog (AWS CLI)

Sie können den ImportSignalCatalogAPI-Vorgang verwenden, um eine JSON-Datei hochzuladen, die bei der Erstellung eines Signalkatalogs hilft. Sie müssen die Vehicle Signal Specification (VSS) befolgen, um Signale in der JSON-Datei zu speichern. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um einen Signalkatalog zu importieren.

- *signal-catalog-name*Ersetzen Sie ihn durch den Namen des Signalkatalogs, den Sie erstellen.
- (Optional) Ersetzen Sie die Beschreibung durch eine*description*, damit Sie den Signalkatalog leichter identifizieren können.
- *signal-catalog-configuration-vss*Ersetzen Sie es durch den Namen der JSON-Zeichenfolgendatei, die in VSS definierte Signale enthält.

Weitere Informationen zur Konfiguration von Verzweigungen, Attributen, Sensoren und Aktuatoren finden Sie unter. AWS FleetWise IoT-Signale konfigurieren

```
aws iotfleetwise import-signal-catalog \
                --name signal-catalog-name \
                --description  description \
                --vss file://signal-catalog-configuration-vss.json
```

Das JSON muss stringifiziert und durch das Feld übergeben werden. `vssJson` Das Folgende ist ein Beispiel für Signale, die in VSS definiert sind.

```
{
 "Vehicle": {
  "type": "branch",
  "children": {
   "Chassis": {
    "type": "branch",
    "description": "All data concerning steering, suspension, wheels, and brakes.",
    "children": {
     "SteeringWheel": {
      "type": "branch",
      "description": "Steering wheel signals",
      "children": {
       "Diameter": {
        "type": "attribute",
        "description": "The diameter of the steering wheel",
        "datatype": "float",
        "unit": "cm",
        "min": 1,
        "max": 50
       },
       "HandsOff": {
        "type": "branch",
        "children": {
         "HandsOffSteeringState": {
          "type": "actuator",
          "description": "HndsOffStrWhlDtSt. Hands Off Steering State",
          "datatype": "boolean"
         },
         "HandsOffSteeringMode": {
          "type": "actuator",
          "description": "HndsOffStrWhlDtMd. Hands Off Steering Mode",
          "datatype": "int8",
          "min": 0,
          "max": 2
         }
        }
       }
      }
     },
     "Accelerator": {
```

```
      "type": "branch",
      "description": "",
      "children": {
       "AcceleratorPedalPosition": {
        "type": "sensor",
        "description": "Throttle__Position. Accelerator pedal position as percent. 0 =
 Not depressed. 100 = Fully depressed.",
        "datatype": "uint8",
        "unit": "%",
        "min": 0,
        "max": 100.000035
       }
      }
     }
    }
   },
   "Powertrain": {
    "type": "branch",
    "description": "Powertrain data for battery management, etc.",
    "children": {
     "Transmission": {
      "type": "branch",
      "description": "Transmission-specific data, stopping at the drive shafts.",
      "children": {
       "VehicleOdometer": {
        "type": "sensor",
        "description": "Vehicle_Odometer",
        "datatype": "float",
        "unit": "km",
        "min": 0,
        "max": 67108863.984375
       }
      }
     },
     "CombustionEngine": {
      "type": "branch",
      "description": "Engine-specific data, stopping at the bell housing.",
      "children": {
       "Engine": {
        "type": "branch",
        "description": "Engine description",
        "children": {
         "timing": {
          "type": "branch",
```

```
          "description": "timing description",
          "children": {
           "run_time": {
            "type": "sensor",
            "description": "Engine run time",
            "datatype": "int16",
            "unit": "ms",
            "min": 0,
            "max": 10000
           },
           "idle_time": {
            "type": "sensor",
            "description": "Engine idle time",
            "datatype": "int16",
            "min": 0,
            "unit": "ms",
            "max": 10000
           }
          }
         }
        }
       }
      }
     }
    }
   },
   "Axle": {
    "type": "branch",
    "description": "Axle signals",
    "children": {
     "TireRRPrs": {
      "type": "sensor",
      "description": "TireRRPrs. Right rear Tire pressure in kilo-Pascal",
      "datatype": "float",
      "unit": "kPaG",
      "min": 0,
      "max": 1020
     }
    }
   }
  }
 },
 "Cameras": {
  "type": "branch",
```

```
  "description": "Branch to aggregate all cameras in the vehicle",
  "children": {
   "FrontViewCamera": {
    "type": "sensor",
    "datatype": "VehicleDataTypes.SVMCamera",
    "description": "Front view camera"
   },
   "RearViewCamera": {
    "type": "sensor",
    "datatype": "VehicleDataTypes.SVMCamera",
    "description": "Rear view camera"
   },
   "LeftSideViewCamera": {
    "type": "sensor",
    "datatype": "VehicleDataTypes.SVMCamera",
    "description": "Left side view camera"
   },
   "RightSideViewCamera": {
    "type": "sensor",
    "datatype": "VehicleDataTypes.SVMCamera",
    "description": "Right side view camera"
   }
  }
 },
 "ComplexDataTypes": {
  "VehicleDataTypes": {
   "type": "branch",
   "description": "Branch to aggregate all camera related higher order data types",
   "children": {
    "SVMCamera": {
     "type": "struct",
     "description": "This data type represents Surround View Monitor (SVM) camera
system in a vehicle",
     "comment": "Test comment",
     "deprecation": "Test deprecation message",
     "children": {
      "Make": {
       "type": "property",
       "description": "Make of the SVM camera",
       "datatype": "string",
       "comment": "Test comment",
       "deprecation": "Test deprecation message"
      },
      "Description": {
```

```
        "type": "property",
        "description": "Description of the SVM camera",
        "datatype": "string",
        "comment": "Test comment",
        "deprecation": "Test deprecation message"
       },
       "FPS": {
        "type": "property",
        "description": "FPS of the SVM camera",
        "datatype": "double",
        "comment": "Test comment",
        "deprecation": "Test deprecation message"
       },
       "Orientation": {
        "type": "property",
        "description": "Orientation of the SVM camera",
        "datatype": "VehicleDataTypes.Orientation",
        "comment": "Test comment",
        "deprecation": "Test deprecation message"
       },
       "Range": {
        "type": "property",
        "description": "Range of the SVM camera",
        "datatype": "VehicleDataTypes.Range",
        "comment": "Test comment",
        "deprecation": "Test deprecation message"
       },
       "RawData": {
        "type": "property",
        "description": "Represents binary data of the SVM camera",
        "datatype": "uint8[]",
        "dataencoding": "binary",
        "comment": "Test comment",
        "deprecation": "Test deprecation message"
       },
       "CapturedFrames": {
        "type": "property",
        "description": "Represents selected frames captured by the SVM camera",
        "datatype": "VehicleDataTypes.Frame[]",
        "dataencoding": "typed",
        "comment": "Test comment",
        "deprecation": "Test deprecation message"
       }
      }
```

```
    },
    "Range": {
     "type": "struct",
     "description": "Range of a camera in centimeters",
     "comment": "Test comment",
     "deprecation": "Test deprecation message",
     "children": {
      "Min": {
       "type": "property",
       "description": "Minimum range of a camera in centimeters",
       "datatype": "uint32",
       "comment": "Test comment",
       "deprecation": "Test deprecation message"
      },
      "Max": {
       "type": "property",
       "description": "Maximum range of a camera in centimeters",
       "datatype": "uint32",
       "comment": "Test comment",
       "deprecation": "Test deprecation message"
      }
     }
    },
    "Orientation": {
     "type": "struct",
     "description": "Orientation of a camera",
     "comment": "Test comment",
     "deprecation": "Test deprecation message",
     "children": {
      "Front": {
       "type": "property",
       "description": "Indicates whether the camera is oriented to the front of the
vehicle",
       "datatype": "boolean",
       "comment": "Test comment",
       "deprecation": "Test deprecation message"
      },
      "Rear": {
       "type": "property",
       "description": "Indicates whether the camera is oriented to the rear of the
vehicle",
       "datatype": "boolean",
       "comment": "Test comment",
       "deprecation": "Test deprecation message"
```

```
      },
      "Side": {
       "type": "property",
       "description": "Indicates whether the camera is oriented to the side of the
 vehicle",
       "datatype": "boolean",
       "comment": "Test comment",
       "deprecation": "Test deprecation message"
      }
     }
    },
    "Frame": {
     "type": "struct",
     "description": "Represents a camera frame",
     "comment": "Test comment",
     "deprecation": "Test deprecation message",
     "children": {
      "Data": {
       "type": "property",
       "datatype": "string",
       "dataencoding": "binary",
       "comment": "Test comment",
       "deprecation": "Test deprecation message"
      }
     }
    }
   }
  }
 }

}
```

Das folgende Beispiel zeigt dieselben Signale, die in VSS definiert sind, in einer JSON-Zeichenfolge.

```
{
   "vssJson": "{\"Vehicle\":{\"type\":\"branch\",\"children\":{\"Chassis\":{\"type
\":\"branch\",\"description\":\"All data concerning steering, suspension, wheels,
 and brakes.\",\"children\":{\"SteeringWheel\":{\"type\":\"branch\",\"description
\":\"Steering wheel signals\",\"children\":{\"Diameter\":{\"type\":\"attribute\",
\"description\":\"The diameter of the steering wheel\",\"datatype\":\"float\",\"unit
\":\"cm\",\"min\":1,\"max\":50},\"HandsOff\":{\"type\":\"branch\",\"children\":
{\"HandsOffSteeringState\":{\"type\":\"actuator\",\"description\":\"HndsOffStrWhlDtSt.
 Hands Off Steering State\",\"datatype\":\"boolean\"},\"HandsOffSteeringMode\":
```

```
{\"type\":\"actuator\",\"description\":\"HndsOffStrWhlDtMd. Hands Off Steering Mode
\",\"datatype\":\"int8\",\"min\":0,\"max\":2}}}}},\"Accelerator\":{\"type\":\"branch
\",\"description\":\"\",\"children\":{\"AcceleratorPedalPosition\":{\"type\":\"sensor
\",\"description\":\"Throttle__Position. Accelerator pedal position as percent. 0
 = Not depressed. 100 = Fully depressed.\",\"datatype\":\"uint8\",\"unit\":\"%\",
\"min\":0,\"max\":100.000035}}}}},\"Powertrain\":{\"type\":\"branch\",\"description
\":\"Powertrain data for battery management, etc.\",\"children\":{\"Transmission\":
{\"type\":\"branch\",\"description\":\"Transmission-specific data, stopping at the
 drive shafts.\",\"children\":{\"VehicleOdometer\":{\"type\":\"sensor\",\"description
\":\"Vehicle_Odometer\",\"datatype\":\"float\",\"unit\":\"km\",\"min\":0,\"max
\":67108863.984375}}},\"CombustionEngine\":{\"type\":\"branch\",\"description\":
\"Engine-specific data, stopping at the bell housing.\",\"children\":{\"Engine\":
{\"type\":\"branch\",\"description\":\"Engine description\",\"children\":{\"timing\":
{\"type\":\"branch\",\"description\":\"timing description\",\"children\":{\"run_time\":
{\"type\":\"sensor\",\"description\":\"Engine run time\",\"datatype\":\"int16\",\"unit
\":\"ms\",\"min\":0,\"max\":10000},\"idle_time\":{\"type\":\"sensor\",\"description
\":\"Engine idle time\",\"datatype\":\"int16\",\"min\":0,\"unit\":\"ms\",\"max
\":10000}}}}}}}},\"Axle\":{\"type\":\"branch\",\"description\":\"Axle signals\",
\"children\":{\"TireRRPrs\":{\"type\":\"sensor\",\"description\":\"TireRRPrs. Right
 rear Tire pressure in kilo-Pascal\",\"datatype\":\"float\",\"unit\":\"kPaG\",\"min
\":0,\"max\":1020}}}}}}"
}
```


> **Note**
>
> Sie können ein Demo-Skript herunterladen, um ROS 2-Nachrichten in VSS-JSON-Dateien zu konvertieren, die mit dem Signalkatalog kompatibel sind. Weitere Informationen finden Sie im Vision System Data Developer Guide.
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ImportSignalCatalog` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Aktualisieren Sie einen AWS FleetWise IoT-Signalkatalog

Sie können den UpdateSignalCatalogAPI-Vorgang verwenden, um einen vorhandenen Signalkatalog zu aktualisieren. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um einen vorhandenen Signalkatalog zu aktualisieren.

*signal-catalog-configuration*Ersetzen Sie ihn durch den Namen der JSON-Datei, die die Konfiguration enthält.

```
aws iotfleetwise update-signal-catalog --cli-input-json file://signal-catalog-configuration.json
```

*signal-catalog-name*Ersetzen Sie ihn durch den Namen des Signalkatalogs, den Sie aktualisieren.

Weitere Informationen zur Konfiguration von Verzweigungen, Attributen, Sensoren und Aktoren finden Sie unterAWS FleetWise IoT-Signale konfigurieren.

> **Important**
>
> Benutzerdefinierte Strukturen sind unveränderlich. Wenn Sie Eigenschaften einer vorhandenen benutzerdefinierten Struktur (Struktur) neu anordnen oder in sie einfügen müssen, löschen Sie die Struktur und erstellen Sie eine brandneue Struktur mit der gewünschten Reihenfolge der Eigenschaften.
> Um eine benutzerdefinierte Struktur zu löschen, fügen Sie den vollständig qualifizierten Namen der Struktur hinzu. `nodesToRemove` Eine Struktur kann nicht gelöscht werden,

wenn sie durch Signale referenziert wird. Alle Signale, die sich auf die Struktur beziehen (ihr Datentyp ist als Zielstruktur definiert), müssen vor der Anforderung zur Aktualisierung des Signalkatalogs aktualisiert oder gelöscht werden.

```
{
     "name": "signal-catalog-name",
     "nodesToAdd": [{
       "branch": {
        "description": "Front left of vehicle specific data.",
        "fullyQualifiedName": "Vehicle.Front.Left"
       }
      },
      {
       "branch": {
        "description": "Door-specific data for the front left of vehicle.",
        "fullyQualifiedName": "Vehicle.Front.Left.Door"
       }
      },
      {
       "actuator": {
        "fullyQualifiedName": "Vehicle.Front.Left.Door.Lock",
        "description": "Whether the front left door is locked.",
        "dataType": "BOOLEAN"
       }
      },
      {
       "branch": {
        "fullyQualifiedName": "Vehicle.Camera"
       }
      },
      {
       "struct": {
        "fullyQualifiedName": "Vehicle.Camera.SVMCamera"
       }
      },
      {
       "property": {
        "fullyQualifiedName": "Vehicle.Camera.SVMCamera.ISO",
        "dataType": "STRING"
       }
      }
```

```
    ],
    "nodesToRemove": ["Vehicle.Chassis.SteeringWheel.HandsOffSteeringState"],
    "nodesToUpdate": [{
     "attribute": {
      "dataType": "FLOAT",
      "fullyQualifiedName": "Vehicle.Chassis.SteeringWheel.Diameter",
      "max": 55
     }
    }]
   }
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `UpdateSignalCatalog` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Überprüfen Sie die Aktualisierung des Signalkatalogs

Sie können den ListSignalCatalogNodesAPI-Vorgang verwenden, um zu überprüfen, ob ein Signalkatalog aktualisiert wurde. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Signale (Knoten) in einem bestimmten Signalkatalog abzurufen.

*signal-catalog-name*Ersetzen Sie es durch den Namen des Signalkatalogs, den Sie überprüfen.

```
aws iotfleetwise list-signal-catalog-nodes --name signal-catalog-name
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ListSignalCatalogNodes` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Löschen Sie einen AWS FleetWise IoT-Signalkatalog

Sie können die DeleteSignalCatalogAPI-Operation verwenden, um einen Signalkatalog zu löschen. Das folgende Beispiel verwendet AWS CLI.

> **Important**
>
> Stellen Sie vor dem Löschen eines Signalkatalogs sicher, dass ihm keine Fahrzeugmodelle, Decoderlisten, Fahrzeuge, Flotten oder Kampagnen zugeordnet sind. Detaillierte Informationen finden Sie hier:
>
> - Löschen Sie ein AWS FleetWise IoT-Fahrzeugmodell
> - Löschen Sie ein AWS FleetWise IoT-Decoder-Manifest
> - Löschen Sie ein AWS FleetWise IoT-Fahrzeug

- Löschen Sie eine AWS FleetWise IoT-Flotte
- Eine AWS FleetWise IoT-Kampagne löschen

Führen Sie den folgenden Befehl aus, um einen vorhandenen Signalkatalog zu löschen. *signal-catalog-name*Ersetzen Sie ihn durch den Namen des Signalkatalogs, den Sie löschen möchten.

```
aws iotfleetwise delete-signal-catalog --name signal-catalog-name
```

Überprüfen Sie das Löschen des Signalkatalogs

Sie können den ListSignalCatalogsAPI-Vorgang verwenden, um zu überprüfen, ob ein Signalkatalog gelöscht wurde. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Signalkataloge abzurufen.

```
aws iotfleetwise list-signal-catalogs
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den API-Vorgang aufrufen kann. `ListSignalCatalogs`

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Informationen zum AWS FleetWise IoT-Signalkatalog abrufen

Sie können die GetSignalCatalogAPI-Operation verwenden, um Signalkataloginformationen abzurufen. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um Informationen über einen Signalkatalog abzurufen.

*signal-catalog-name*Ersetzen Sie ihn durch den Namen des Signalkatalogs, den Sie abrufen möchten.

```
aws iotfleetwise get-signal-catalog --name signal-catalog-name
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `GetSignalCatalog` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

> **Note**
>
> Diese Operation ist letztlich konsistent. Mit anderen Worten, Änderungen am Signalkatalog werden möglicherweise nicht sofort übernommen.

AWS FleetWise IoT-Fahrzeugmodelle verwalten

Sie verwenden Signale, um Fahrzeugmodelle zu erstellen, die dabei helfen, das Format Ihrer Fahrzeuge zu standardisieren. Fahrzeugmodelle sorgen für konsistente Informationen für mehrere Fahrzeuge desselben Typs, sodass Sie Daten von Fahrzeugflotten verarbeiten können. Fahrzeuge, die mit demselben Fahrzeugmodell erstellt wurden, erben dieselbe Gruppe von Signalen. Weitere Informationen finden Sie unter AWS FleetWise IoT-Fahrzeuge verwalten.

Jedes Fahrzeugmodell hat ein Statusfeld, das den Status des Fahrzeugmodells enthält. Der Status kann einer der folgenden Werte sein:

- `ACTIVE`— Das Fahrzeugmodell ist aktiv.
- `DRAFT`— Die Konfiguration des Fahrzeugmodells wird gespeichert.

> **Important**
>
> - Sie benötigen einen Signalkatalog, bevor Sie mithilfe der `CreateModelManifest` API-Operation ein Fahrzeugmodell erstellen können. Weitere Informationen finden Sie unter Erstellen Sie einen AWS FleetWise IoT-Signalkatalog.
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Fahrzeugmodell zu erstellen, aktiviert AWS IoT das Fahrzeugmodell FleetWise automatisch für Sie.
> - Wenn Sie den `CreateModelManifest` API-Vorgang verwenden, um ein Fahrzeugmodell zu erstellen, bleibt das Fahrzeugmodell im `DRAFT` Status.
> - Sie können keine Fahrzeuge anhand von Fahrzeugmodellen erstellen, die sich in diesem `DRAFT` Bundesstaat befinden. Verwenden Sie den `UpdateModelManifest` API-Vorgang, um Fahrzeugmodelle auf den jeweiligen `ACTIVE` Status umzustellen.
> - Sie können keine Fahrzeugmodelle bearbeiten, die sich im `ACTIVE` Bundesstaat befinden.

Themen

- Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell
- Aktualisieren Sie ein AWS FleetWise IoT-Fahrzeugmodell
- Löschen Sie ein AWS FleetWise IoT-Fahrzeugmodell
- Holen Sie sich Informationen zum AWS FleetWise IoT-Fahrzeugmodell

Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um Fahrzeugmodelle zu erstellen.

Themen

- Erstellen Sie ein Fahrzeugmodell (Konsole)
- Erstellen Sie ein Fahrzeugmodell (AWS CLI)

Erstellen Sie ein Fahrzeugmodell (Konsole)

In der AWS FleetWise IoT-Konsole können Sie auf folgende Weise ein Fahrzeugmodell erstellen:

- Verwenden Sie eine Vorlage von AWS
- Manuelles Erstellen Sie ein Fahrzeugmodell
- Duplizieren Sie ein Fahrzeugmodell

Verwenden Sie eine Vorlage von AWS

AWS IoT FleetWise bietet eine On-Board Diagnostic (OBD) II, J1979-Vorlage, die automatisch einen Signalkatalog, ein Fahrzeugmodell und ein Decoder-Manifest für Sie erstellt. Die Vorlage fügt dem Decoder-Manifest auch OBD-Netzwerkschnittstellen hinzu. Weitere Informationen finden Sie unter AWS FleetWise IoT-Decoder-Manifeste verwalten.

Um ein Fahrzeugmodell mithilfe einer Vorlage zu erstellen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
3. Wählen Sie auf der Seite Fahrzeugmodelle die Option Bereitgestellte Vorlage hinzufügen aus.
4. Wählen Sie On-Board-Diagnose (OBD) II aus.
5. Geben Sie einen Namen für die OBD-Netzwerkschnittstelle ein, die AWS IoT FleetWise erstellt.
6. Wählen Sie Hinzufügen aus.

Manuelles Erstellen Sie ein Fahrzeugmodell

Sie können Signale aus dem Signalkatalog hinzufügen oder Signale importieren, indem Sie eine oder mehrere DBC-Dateien hochladen. Eine.dbc-Datei ist ein Dateiformat, das CAN-Bus-Datenbanken (Controller Area Network) unterstützen.

> **Important**
>
> Mit der AWS FleetWise IoT-Konsole können Sie kein Fahrzeugmodell mit Datensignalen des Bildverarbeitungssystems erstellen. Verwenden Sie stattdessen die, AWS CLI um ein Fahrzeugmodell zu erstellen.
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Um ein Fahrzeugmodell manuell zu erstellen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
3. Wählen Sie auf der Seite Fahrzeugmodelle die Option Fahrzeugmodell erstellen aus, und gehen Sie dann wie folgt vor.

Themen

- Schritt 1: Fahrzeugmodell konfigurieren
- Schritt 2: Signale hinzufügen
- Schritt 3: Signale importieren
- (Optional) Schritt 4: Attribute hinzufügen
- Schritt 5: Überprüfen und Erstellen

Schritt 1: Fahrzeugmodell konfigurieren

Gehen Sie unter Allgemeine Informationen wie folgt vor.

1. Geben Sie einen Namen für das Fahrzeugmodell ein.
2. (Optional) Geben Sie eine Beschreibung ein.

3. Wählen Sie Weiter aus.

Schritt 2: Signale hinzufügen

> **Note**
>
> - Wenn Sie AWS IoT zum ersten Mal verwenden FleetWise, ist dieser Schritt erst verfügbar, wenn Sie einen Signalkatalog haben. Wenn das erste Fahrzeugmodell erstellt wird, erstellt AWS IoT FleetWise automatisch einen Signalkatalog mit Signalen, die dem ersten Fahrzeugmodell hinzugefügt wurden.
> - Wenn Sie Erfahrung mit AWS IoT haben FleetWise, können Sie Ihrem Fahrzeugmodell Signale hinzufügen, indem Sie Signale aus dem Signalkatalog auswählen oder .dbc-Dateien hochladen, um Signale zu importieren.
> - Sie benötigen mindestens ein Signal, um ein Fahrzeugmodell zu erstellen.

Um Signale hinzuzufügen

1. Wählen Sie ein oder mehrere Signale aus dem Signalkatalog aus, den Sie dem Fahrzeugmodell hinzufügen möchten. Sie können die ausgewählten Signale im rechten Bereich überprüfen.

   > **Note**
   >
   > Nur ausgewählte Signale werden dem Fahrzeugmodell hinzugefügt.

2. Wählen Sie Weiter aus.

Schritt 3: Signale importieren

> **Note**
>
> - Wenn Sie AWS IoT zum ersten Mal verwenden FleetWise, müssen Sie mindestens eine .dbc-Datei hochladen, um Signale zu importieren.
> - Wenn Sie Erfahrung mit AWS IoT haben FleetWise, können Sie Ihrem Fahrzeugmodell Signale hinzufügen, indem Sie Signale aus dem Signalkatalog auswählen oder .dbc-Dateien hochladen, um Signale zu importieren.

- Sie benötigen mindestens ein Signal, um ein Fahrzeugmodell zu erstellen.

Um Signale zu importieren

1. Wählen Sie Dateien auswählen.
2. Wählen Sie im Dialogfeld die .dbc-Datei aus, die Signale enthält. Sie können mehrere .dbc-Dateien hochladen.
3. AWS IoT FleetWise analysiert Ihre DBC-Dateien, um Signale abzurufen.

   Geben Sie im Abschnitt Signale die folgenden Metadaten für jedes Signal an.

   - Name — Der Name des Signals.

     Der Signalname muss eindeutig sein. Der Signalname und der Pfad können bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich).
   - Datentyp — Der Datentyp des Signals muss einer der folgenden sein: INT8,,,,,,,, UINT8, BOOLEAN INT16 UINT16, FLOAT INT32 UINT32, DOUBLE INT64 UINT64, STRING, UNIX_TIMESTAMP, _ARRAY, _ARRAY, _ARRAY, INT8 _ARRAY, UINT8 _ARRAY, INT16 _ARRAY, UINT16 _ARRAY, INT32 _ARRAY, UINT32 _ARRAY, INT64 BOOLEAN_ARRAY, UINT64 FLOAT_ARRAY, DOUBLE_ARRAY, STRING_ARRAY, UNIX_TIMESTAMP_ARRAY oder UNKNOWN.
   - Signaltyp — Der Typ des Signals, bei dem es sich um einen Sensor oder einen Aktuator handeln kann.
   - (Optional) Einheit — Die wissenschaftliche Einheit für das Signal, z. B. km oder Celsius.
   - (Optional) Pfad — Der Pfad zum Signal. Ähnlich wie JSONPath, verwenden Sie einen Punkt (.), um auf ein untergeordnetes Signal zu verweisen. Beispiel, **`Vehicle.Engine.Light`**.

     Der Signalname und der Pfad können bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich).
   - (Optional) Min — Der Mindestwert des Signals.
   - (Optional) Max — Der Maximalwert des Signals.
   - (Optional) Beschreibung — Die Beschreibung für das Signal.

     Die Beschreibung kann bis zu 2048 Zeichen enthalten. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).

4. Wählen Sie Weiter aus.

(Optional) Schritt 4: Attribute hinzufügen

Sie können bis zu 100 Attribute hinzufügen, einschließlich der vorhandenen Attribute im Signalkatalog.

Um Attribute hinzuzufügen

1. Geben Sie unter Attribute hinzufügen die folgenden Metadaten für jedes Attribut an.

   - Name — Der Name des Attributs.

     Der Signalname muss eindeutig sein. Der Signalname und der Pfad können bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich)
   - Datentyp — Der Datentyp des Attributs muss einer der folgenden sein: INT8,,,,,,, BOOLEAN UINT8 INT16, FLOAT UINT16 INT32, DOUBLE UINT32 INT64, STRING UINT64, UNIX_TIMESTAMP, _ARRAY, _ARRAY, _ARRAY, _ARRAY, INT8 _ARRAY, UINT8 _ARRAY, INT16 _ARRAY, UINT16 _ARRAY, INT32 _ARRAY, _ARRAY, UINT32 INT64 BOOLEAN_ARRAY, UINT64 FLOAT_ARRAY, DOUBLE_ARRAY, STRING_ARRAY, UNIX_TIMESTAMP_ARRAY oder UNKNOWN
   - (Optional) Einheit — Die wissenschaftliche Einheit für das Attribut, z. B. km oder Celsius.
   - (Optional) Pfad — Der Pfad zum Signal. Ähnlich wie JSONPath, verwenden Sie einen Punkt (.), um auf ein untergeordnetes Signal zu verweisen. Beispiel, **`Vehicle.Engine.Light`**.

     Der Signalname und der Pfad können bis zu 150 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt) und _ (Unterstrich)
   - (Optional) Min — Der Mindestwert des Attributs.
   - (Optional) Max — Der Höchstwert des Attributs.
   - (Optional) Beschreibung — Die Beschreibung für das Attribut.

     Die Beschreibung kann bis zu 2048 Zeichen enthalten. Zulässige Zeichen: a—z, A—Z, 0—9,: (Doppelpunkt), _ (Unterstrich) und - (Bindestrich).
2. Wählen Sie Weiter aus.

Schritt 5: Überprüfen und Erstellen

Überprüfen Sie die Konfigurationen für das Fahrzeugmodell und wählen Sie dann Create.

Duplizieren Sie ein Fahrzeugmodell

AWS IoT FleetWise kann die Konfigurationen eines vorhandenen Fahrzeugmodells kopieren, um ein neues Modell zu erstellen. Die im ausgewählten Fahrzeugmodell angegebenen Signale werden in das neue Fahrzeugmodell kopiert.

Um ein Fahrzeugmodell zu duplizieren

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
3. Wählen Sie ein Modell aus der Fahrzeugmodellliste und dann Modell duplizieren aus.

Folgen Sie der Manuelles Erstellen Sie ein Fahrzeugmodell Anleitung, um das Fahrzeugmodell zu konfigurieren.

Es kann einige Minuten dauern, FleetWise bis AWS IoT Ihre Anfrage zur Erstellung des Fahrzeugmodells bearbeitet hat. Nachdem das Fahrzeugmodell erfolgreich erstellt wurde, wird auf der Seite Fahrzeugmodelle in der Spalte Status der Wert AKTIV angezeigt. Wenn das Fahrzeugmodell aktiv wird, können Sie es nicht bearbeiten.

Erstellen Sie ein Fahrzeugmodell (AWS CLI)

Sie können den CreateModelManifestAPI-Vorgang verwenden, um Fahrzeugmodelle (Modellmanifeste) zu erstellen. Das folgende Beispiel verwendet die AWS CLI.

> **Important**
>
> Sie benötigen einen Signalkatalog, bevor Sie mithilfe der `CreateModelManifest` API-Operation ein Fahrzeugmodell erstellen können. Weitere Informationen zum Erstellen eines Signalkatalogs finden Sie unterErstellen Sie einen AWS FleetWise IoT-Signalkatalog.

Führen Sie den folgenden Befehl aus, um ein Fahrzeugmodell zu erstellen.

*`vehicle-model-configuration`*Ersetzen Sie es durch den Namen der JSON-Datei, die die Konfiguration enthält.

```
aws iotfleetwise create-model-manifest --cli-input-json file://vehicle-model-configuration.json
```

- *vehicle-model-name*Ersetzen Sie es durch den Namen des Fahrzeugmodells, das Sie erstellen.
- *signal-catalog-ARN*Ersetzen Sie durch den Amazon-Ressourcennamen (ARN) des Signalkatalogs.
- (Optional) *description* Ersetzen Sie ihn durch eine Beschreibung, die Ihnen hilft, das Fahrzeugmodell zu identifizieren.

Weitere Informationen zur Konfiguration von Verzweigungen, Attributen, Sensoren und Aktuatoren finden Sie unterAWS FleetWise IoT-Signale konfigurieren.

```
{
    "name": "vehicle-model-name",
    "signalCatalogArn": "signal-catalog-ARN",
    "description": "description",
    "nodes": ["Vehicle.Chassis"]
}
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `CreateModelManifest` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
```

```
        }
    ]
}
```

Aktualisieren Sie ein AWS FleetWise IoT-Fahrzeugmodell

Sie können den UpdateModelManifestAPI-Vorgang verwenden, um ein vorhandenes Fahrzeugmodell (Modellmanifeste) zu aktualisieren. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um ein vorhandenes Fahrzeugmodell zu aktualisieren.

*update-vehicle-model-configuration*Ersetzen Sie ihn durch den Namen der JSON-Datei, die die Konfiguration enthält.

```
aws iotfleetwise update-model-manifest --cli-input-json file://update-vehicle-model-configuration.json
```

- *vehicle-model-name*Ersetzen Sie es durch den Namen des Fahrzeugmodells, das Sie aktualisieren möchten.
- (Optional) Um das Fahrzeugmodell zu aktivieren, *vehicle-model-status* ersetzen Sie es durch`ACTIVE`.

> **Important**
>
> Nachdem das Fahrzeugmodell aktiviert wurde, können Sie das Fahrzeugmodell nicht mehr ändern.

- (Optional) *description* Ersetzen Sie es durch eine aktualisierte Beschreibung, um das Fahrzeugmodell leichter identifizieren zu können.

```
{
    "name": "vehicle-model-name",
    "status": "vehicle-model-status",
    "description": "description",
    "nodesToAdd": ["Vehicle.Front.Left"],
    "nodesToRemove": ["Vehicle.Chassis.SteeringWheel"],
}
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `UpdateModelManifest` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Überprüfen Sie die Aktualisierung des Fahrzeugmodells

Sie können den ListModelManifestNodesAPI-Vorgang verwenden, um zu überprüfen, ob ein Fahrzeugmodell aktualisiert wurde. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Signale (Knoten) in einem bestimmten Fahrzeugmodell abzurufen.

*`vehicle-model-name`*Ersetzen Sie es durch den Namen des Fahrzeugmodells, das Sie überprüfen.

```
aws iotfleetwise list-model-manifest-nodes /
                --name vehicle-model-name
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ListModelManifestNodes` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Löschen Sie ein AWS FleetWise IoT-Fahrzeugmodell

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um Fahrzeugmodelle zu löschen.

> **Important**
>
> Fahrzeuge und Decoder-Manifeste, die dem Fahrzeugmodell zugeordnet sind, müssen zuerst gelöscht werden. Weitere Informationen erhalten Sie unter Löschen Sie ein AWS FleetWise IoT-Fahrzeug und Löschen Sie ein AWS FleetWise IoT-Decoder-Manifest.

Löschen Sie ein Fahrzeugmodell (Konsole)

Verwenden Sie die AWS FleetWise IoT-Konsole, um ein Fahrzeugmodell zu löschen.

Um ein Fahrzeugmodell zu löschen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
3. Wählen Sie auf der Seite Fahrzeugmodelle das Zielfahrzeugmodell aus.
4. Wählen Sie Löschen aus.

5. Unter Löschen**`vehicle-model-name`**? , geben Sie den Namen des Fahrzeugmodells ein, das gelöscht werden soll, und wählen Sie dann Bestätigen.

Löschen Sie ein Fahrzeugmodell (AWS CLI)

Sie können den DeleteModelManifestAPI-Vorgang verwenden, um ein vorhandenes Fahrzeugmodell (Modellmanifeste) zu löschen. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um ein Fahrzeugmodell zu löschen.

*`model-manifest-name`*Ersetzen Sie es durch den Namen des Fahrzeugmodells, das Sie löschen möchten.

```
aws iotfleetwise delete-model-manifest --name model-manifest-name
```

Überprüfen Sie das Löschen des Fahrzeugmodells

Sie können den ListModelManifestsAPI-Vorgang verwenden, um zu überprüfen, ob ein Fahrzeugmodell gelöscht wurde. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Fahrzeugmodelle abzurufen.

```
aws iotfleetwise list-model-manifests
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ListModelManifests` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
```

```
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Holen Sie sich Informationen zum AWS FleetWise IoT-Fahrzeugmodell

Sie können den GetModelManifestAPI-Vorgang verwenden, um Informationen über ein Fahrzeugmodell abzurufen. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um Informationen zu einem Fahrzeugmodell abzurufen.

*vehicle-model*Ersetzen Sie es durch den Namen des Fahrzeugmodells, das Sie abrufen möchten.

```
aws iotfleetwise get-model-manifest --name vehicle-model
```

> **Note**
>
> Diese Operation ist letztlich konsistent. Mit anderen Worten, Änderungen am Fahrzeugmodell werden möglicherweise nicht sofort übernommen.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `GetModelManifest` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
```

```
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

AWS FleetWise IoT-Decoder-Manifeste verwalten

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Decoder-Manifeste enthalten Dekodierungsinformationen, die das AWS IoT FleetWise verwendet, um Fahrzeugdaten (Binärdaten) in für Menschen lesbare Werte umzuwandeln und Ihre Daten für Datenanalysen aufzubereiten. Netzwerkschnittstellen- und Signaldecoder sind die Kernkomponenten, mit denen Sie bei der Konfiguration von Decoder-Manifesten arbeiten.

Netzwerkschnittstelle

Enthält Informationen über das Protokoll, das das bordeigene Netzwerk verwendet. AWS IoT FleetWise unterstützt die folgenden Protokolle.

Controller Area Network (CAN-Bus)

Ein Protokoll, das definiert, wie Daten zwischen elektronischen Steuergeräten (ECUs) übertragen werden. ECUs kann das Motorsteuergerät, Airbags oder das Audiosystem sein.

On-Board-Diagnose (OBD) II

Ein weiterentwickeltes Protokoll, das definiert, wie Selbstdiagnosedaten untereinander ausgetauscht werden. ECUs Es bietet eine Reihe von Standard-Diagnosefehlercodes (DTCs), mit deren Hilfe Sie feststellen können, was mit Ihrem Fahrzeug nicht stimmt.

Middleware für Fahrzeuge

Die Fahrzeug-Middleware ist als eine Art Netzwerkschnittstelle definiert. Zu den Beispielen für Fahrzeug-Middleware gehören das Robot Operating System (ROS 2) und die skalierbare serviceorientierte Middleware over IP (SOME/IP).

> **Note**
>
> AWS IoT FleetWise unterstützt ROS 2-Middleware für Bildverarbeitungssystemdaten.

Kundenspezifische Schnittstellen

Sie können auch Ihre eigene Schnittstelle verwenden, um Signale am Edge zu dekodieren. Dies kann Ihnen Zeit sparen, da Sie keine Dekodierungsregeln in der Cloud erstellen müssen.

Signaldecoder

Bietet detaillierte Dekodierungsinformationen für ein bestimmtes Signal. Jedes im Fahrzeugmodell angegebene Signal muss mit einem Signaldecoder gekoppelt werden. Wenn das Decoder-Manifest CAN-Netzwerkschnittstellen enthält, muss es CAN-Decodersignale enthalten. Wenn das Decoder-Manifest OBD-Netzwerkschnittstellen enthält, muss es OBD-Signaldecoder enthalten.

Das Decoder-Manifest muss Dekoder für Nachrichtensignale enthalten, wenn es auch Middleware-Schnittstellen für Fahrzeuge enthält. Oder, wenn das Decoder-Manifest benutzerdefinierte Dekodierungsschnittstellen enthält, muss es auch benutzerdefinierte Dekodierungssignale enthalten.

Jedes Decoder-Manifest muss einem Fahrzeugmodell zugeordnet sein. AWS IoT FleetWise verwendet das zugehörige Decoder-Manifest, um Daten von Fahrzeugen zu dekodieren, die auf der Grundlage des Fahrzeugmodells erstellt wurden.

Jedes Decoder-Manifest hat ein Statusfeld, das den Status des Decoder-Manifests enthält. Der Status kann einer der folgenden Werte sein:

- `ACTIVE`— Das Decoder-Manifest ist aktiv.
- `DRAFT`— Die Konfiguration des Decoder-Manifests wurde nicht gespeichert.
- `VALIDATING`— Das Decoder-Manifest wird derzeit auf seine Eignung überprüft. Dies gilt nur für Decoder-Manifeste, die mindestens ein Datensignal des Bildverarbeitungssystems enthalten.
- `INVALID`— Das Decoder-Manifest konnte nicht validiert werden und kann noch nicht aktiviert werden. Dies gilt nur für Decoder-Manifeste, die mindestens ein Datensignal des Bildverarbeitungssystems enthalten. Sie können das ListDecoderManifests und verwenden GetDecoderManifest APIs , um den Grund für eine fehlgeschlagene Validierung zu überprüfen.

> **Important**
>
> - Wenn Sie die AWS FleetWise IoT-Konsole verwenden, um ein Decoder-Manifest zu erstellen, aktiviert AWS IoT das Decoder-Manifest FleetWise automatisch für Sie.
> - Wenn Sie den `CreateDecoderManifest` API-Vorgang verwenden, um ein Decoder-Manifest zu erstellen, bleibt das Decoder-Manifest im Status. `DRAFT`
> - Sie können keine Fahrzeuge anhand von Fahrzeugmodellen erstellen, die mit einem `DRAFT` Decoder-Manifest verknüpft sind. Verwenden Sie den `UpdateDecoderManifest` API-Vorgang, um das Decoder-Manifest in den `ACTIVE` Status zu ändern.
> - Sie können keine Decoder-Manifeste bearbeiten, die sich im `ACTIVE` Status befinden.

Themen

- Konfiguration von AWS FleetWise IoT-Netzwerkschnittstellen und Decodersignalen
- Erstellen Sie ein AWS FleetWise IoT-Decoder-Manifest
- Ein AWS FleetWise IoT-Decoder-Manifest aktualisieren
- Löschen Sie ein AWS FleetWise IoT-Decoder-Manifest
- Informationen zum AWS FleetWise IoT-Decoder-Manifest abrufen

Konfiguration von AWS FleetWise IoT-Netzwerkschnittstellen und Decodersignalen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Jedes Decoder-Manifest hat mindestens eine Netzwerkschnittstelle und Signaldecoder, die mit Signalen gepaart sind, die im zugehörigen Fahrzeugmodell spezifiziert sind.

Wenn das Decoder-Manifest CAN-Netzwerkschnittstellen enthält, muss es CAN-Signaldecoder enthalten. Wenn das Decoder-Manifest OBD-Netzwerkschnittstellen enthält, muss es OBD-Signaldecoder enthalten.

Themen

- Netzwerkschnittstellen konfigurieren
- Konfigurieren Sie Signaldecoder

Netzwerkschnittstellen konfigurieren

Um eine CAN-Netzwerkschnittstelle zu konfigurieren, geben Sie die folgenden Informationen an.

- `name`— Der Name der CAN-Schnittstelle.

  Der Schnittstellenname muss eindeutig sein und kann 1—100 Zeichen lang sein.
- (Optional) `protocolName` — Der Name des Protokolls.

  Gültige Werte: `CAN-FD` und `CAN`
- (Optional) `protocolVersion` — AWS IoT unterstützt FleetWise derzeit CAN-FD und CAN 2.0b.

  Gültige Werte: und `1.0` `2.0b`

Um eine OBD-Netzwerkschnittstelle zu konfigurieren, geben Sie die folgenden Informationen an.

- `name`— Der Name der OBD-Schnittstelle.

  Der Schnittstellenname muss eindeutig sein und kann 1—100 Zeichen lang sein.
- `requestMessageId`— Die ID der Nachricht, die Daten anfordert.
- (Optional) `dtcRequestIntervalSeconds` — Wie oft innerhalb von Sekunden Diagnose-Fehlercodes (DTCs) vom Fahrzeug angefordert werden sollen. Wenn der angegebene Wert beispielsweise 120 ist, erfasst die Edge Agent-Software die gespeicherten DTCs Daten alle 2 Minuten.
- (Optional) `hasTransmissionEcu` — Gibt an, ob das Fahrzeug über ein Getriebesteuergerät (TCM) verfügt.

  Gültige Werte: `true` und `false`
- (Optional) `obdStandard` — Der OBD-Standard, den AWS IoT FleetWise unterstützt. AWS IoT unterstützt FleetWise derzeit den Standard World Wide Harmonization On-Board Diagnostics (WWH-OBD) -4. ISO15765

- (Optional) `pidRequestIntervalSeconds` — Wie oft OBD II vom Fahrzeug angefordert werden soll. PIDs Wenn der angegebene Wert beispielsweise 120 ist, erfasst die Edge Agent-Software OBD II PIDs einmal alle 2 Minuten.
- (Optional) `useExtendedIds` — Gibt an, ob IDs in der Nachricht „Extended“ verwendet werden soll.

  Gültige Werte: `true` und `false`

Geben Sie die folgenden Informationen an, um eine Fahrzeug-Middleware-Netzwerkschnittstelle zu konfigurieren.

- `name`— Der Name der Middleware-Schnittstelle des Fahrzeugs.

  Der Schnittstellenname muss eindeutig sein und kann 1—100 Zeichen lang sein.
- `protocolName`— Der Name des Protokolls.

  Zulässige Werte: `ROS_2`

Um eine benutzerdefinierte Dekodierungsschnittstelle zu konfigurieren, geben Sie die folgenden Informationen an.

- `name`— Der Name Ihres Decoders, mit dem Sie Signale am Edge dekodieren.

  Der Name der Decoder-Schnittstelle kann 1—100 Zeichen lang sein.

Konfigurieren Sie Signaldecoder

Um einen CAN-Signaldecoder zu konfigurieren, geben Sie die folgenden Informationen an.

- `factor`— Der Multiplikator, der zur Dekodierung der Nachricht verwendet wurde.
- `isBigEndian`— Ob die Byte-Reihenfolge der Nachricht Big-Endian ist. Wenn es Big-Endian ist, wird der signifikanteste Wert in der Sequenz zuerst gespeichert, und zwar an der niedrigsten Speicheradresse.
- `isSigned`— Ob die Nachricht signiert ist. Wenn sie signiert ist, kann die Nachricht sowohl positive als auch negative Zahlen enthalten.
- `length`— Die Länge der Nachricht in Bits.
- `messageId`— Die ID der Nachricht.

- `offset`— Der Offset, der zur Berechnung des Signalwerts verwendet wurde. In Kombination mit dem Faktor ist die Berechnung`value = raw_value * factor + offset`.
- `startBit`— Gibt die Position des ersten Bits der Nachricht an.
- (Optional) `name` — Der Name des Signals.
- (Optional) `signalValueType` — Der Wertetyp des Signals. Integer ist der Standardwerttyp.

Um einen OBD-Signaldecoder zu konfigurieren, geben Sie die folgenden Informationen an.

- `byteLength`— Die Länge der Nachricht in Byte.
- `offset`— Der Offset, der zur Berechnung des Signalwerts verwendet wurde. In Kombination mit der Skalierung ist die Berechnung`value = raw_value * scaling + offset`.
- `pid`— Der Diagnosecode, der verwendet wird, um von einem Fahrzeug eine Nachricht für dieses Signal anzufordern.
- `pidResponseLength`— Die Länge der angeforderten Nachricht.
- `scaling`— Der Multiplikator, der zur Dekodierung der Nachricht verwendet wurde.
- `serviceMode`— Der Betriebsmodus (Diagnosedienst) in einer Nachricht.
- `startByte`— Zeigt den Anfang der Nachricht an.
- (Optional) `bitMaskLength` — Die Anzahl der Bits, die in einer Nachricht maskiert sind.
- (Optional) `bitRightShift` — Die Anzahl der nach rechts verschobenen Positionen.
- (Optional) `isSigned` — Ob die Nachricht signiert ist. Wenn sie signiert ist, kann die Nachricht sowohl positive als auch negative Zahlen enthalten. Die Nachricht ist standardmäßig nicht signiert (`false`).
- (Optional) `signalValueType` — Der Wertetyp des Signals. Integer ist der Standardwerttyp.

Um einen Nachrichtensignal-Decoder zu konfigurieren, geben Sie die folgenden Informationen an.

- `topicName`— Der Themenname für das Nachrichtensignal. Er entspricht den Themen in ROS 2. Weitere Hinweise zum strukturierten Nachrichtenobjekt finden Sie unter StructuredMessage.
- `structuredMessage`— Die strukturierte Nachricht für das Nachrichtensignal. Sie kann entweder mit einer primitiveMessageDefinition structuredMessageList Definition oder structuredMessageDefinition rekursiv definiert werden.

Um ein benutzerdefiniertes Dekodierungssignal zu konfigurieren, geben Sie die folgenden Informationen an.

- (Optional) `id` — Die ID des Signals, das Sie selbst mit Ihrer Decoder-Schnittstelle dekodieren. Die Signal-ID kann 1—150 Zeichen lang sein. Wenn nicht angegeben, wird `id` standardmäßig der Wert `fullyQualifiedName` des Signals verwendet.

Erstellen Sie ein AWS FleetWise IoT-Decoder-Manifest

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um ein Decoder-Manifest für Ihr Fahrzeugmodell zu erstellen.

Themen

- Erstellen Sie ein Decoder-Manifest (Konsole)
- Erstellen Sie ein Decoder-Manifest ()AWS CLI

Erstellen Sie ein Decoder-Manifest (Konsole)

Sie können die AWS FleetWise IoT-Konsole verwenden, um ein Decoder-Manifest zu erstellen, das Ihrem Fahrzeugmodell zugeordnet ist.

> **Important**
>
> Sie können mit der AWS FleetWise IoT-Konsole keine Datensignale von Bildverarbeitungssystemen in Decoder-Manifesten konfigurieren. Verwenden Sie stattdessen die AWS CLI. Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Um ein Decoder-Manifest zu erstellen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
3. Wählen Sie das Zielfahrzeugmodell aus.
4. Wählen Sie auf der Seite mit der Zusammenfassung des Fahrzeugmodells die Option Decoder-Manifest erstellen aus, und gehen Sie dann wie folgt vor.

Themen

- Schritt 1: Decoder-Manifest konfigurieren
- Schritt 2: CAN-Schnittstelle zuordnen
- Schritt 3: Überprüfen und Erstellen

Schritt 1: Decoder-Manifest konfigurieren

Gehen Sie unter Allgemeine Informationen wie folgt vor.

1. Geben Sie einen eindeutigen Namen für das Decoder-Manifest ein.
2. (Optional) Geben Sie eine Beschreibung ein.
3. Wählen Sie Weiter aus.

Fügen Sie Netzwerkschnittstellen hinzu

Jedes Decoder-Manifest muss mindestens eine Netzwerkschnittstelle haben. Sie können einem Decoder-Manifest mehrere Netzwerkschnittstellen hinzufügen.

Um eine Netzwerkschnittstelle hinzuzufügen

1. Laden Sie eine Netzwerkschnittstellendatei hoch. Sie können eine .dbc-Datei für CAN-Protokolle oder eine .json-Datei für ROS 2 oder benutzerdefinierte Schnittstellen hochladen.
2. Geben Sie einen Namen für Ihre Netzwerkschnittstelle ein. Wenn Sie eine benutzerdefinierte Schnittstelle hochgeladen haben, ist der Name bereits angegeben.

Fehlende Signale zuordnen

Wenn das Fahrzeugmodell Signale enthält, bei denen gekoppelte Signaldecoder in den hochgeladenen Netzwerkschnittstellen fehlen, können Sie einen benutzerdefinierten Standarddecoder erstellen, der die fehlenden Signale abbildet. Dies ist optional, da Sie die Signale im nächsten Schritt manuell zuordnen können.

Um einen benutzerdefinierten Standarddecoder zu erstellen

1. Wählen Sie Benutzerdefinierten Standarddecoder für fehlende Signale erstellen aus.
2. Wählen Sie Weiter aus.

Schritt 2: CAN-Schnittstelle zuordnen

Sie können die CAN-Signale mit CAN-Signal-Decodern abbilden. Wenn Sie das Kontrollkästchen Benutzerdefinierten Standarddecoder für fehlende Signale erstellen aktiviert haben, werden alle Signale, denen ein Decodersignal fehlt, automatisch den standardmäßigen benutzerdefinierten Signaldecodern zugeordnet.

Um CAN-Signale zuzuordnen

1. Wählen Sie unter CAN-Signal-Mapping einen Signaldecoder aus.
2. Wählen Sie Weiter aus.

> **Note**
>
> Wenn Sie ein ROS 2 oder eine benutzerdefinierte Schnittstelle hinzugefügt haben, können Sie die Zuordnungen überprüfen, bevor Sie das Decoder-Manifest erstellen.

Schritt 3: Überprüfen und Erstellen

Überprüfen Sie die Konfigurationen für das Decoder-Manifest und wählen Sie dann Create.

Erstellen Sie ein Decoder-Manifest ()AWS CLI

Sie können die CreateDecoderManifestAPI-Operation verwenden, um Decoder-Manifeste zu erstellen. Das folgende Beispiel verwendet die AWS CLI.

> **Important**
>
> Sie benötigen ein Fahrzeugmodell, bevor Sie ein Decoder-Manifest erstellen können. Jedes Decoder-Manifest muss einem Fahrzeugmodell zugeordnet sein. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeugmodell.

Führen Sie den folgenden Befehl aus, um ein Decoder-Manifest zu erstellen.

*decoder-manifest-configuration*Ersetzen Sie es durch den Namen der JSON-Datei, die die Konfiguration enthält.

```
aws iotfleetwise create-decoder-manifest --cli-input-json file://decoder-manifest-configuration.json
```

- *decoder-manifest-name*Ersetzen Sie es durch den Namen des Decoder-Manifests, das Sie erstellen.
- *vehicle-model-ARN*Ersetzen Sie es durch den Amazon-Ressourcennamen (ARN) des Fahrzeugmodells.
- (Optional) *description* Ersetzen Sie es durch eine Beschreibung, die Ihnen hilft, das Decoder-Manifest zu identifizieren.

Weitere Informationen zur Konfiguration von Verzweigungen, Attributen, Sensoren und Aktuatoren finden Sie unter. Konfiguration von AWS FleetWise IoT-Netzwerkschnittstellen und Decodersignalen

```
{
    "name": "decoder-manifest-name",
    "modelManifestArn": "vehicle-model-arn",
    "description": "description",
    "networkInterfaces": [
        {
            "canInterface": {
                "name": "myNetworkInterface",
                "protocolName": "CAN",
                "protocolVersion": "2.0b"
            },
            "interfaceId": "Qq1acaenByOB3sSM39SYm",
            "type": "CAN_INTERFACE"
        }
```

```
    ],
    "signalDecoders": [
        {
            "canSignal": {
                "name": "Engine_Idle_Time",
                "factor": 1,
                "isBigEndian": true,
                "isSigned": false,
                "length": 24,
                "messageId": 271343712,
                "offset": 0,
                "startBit": 16
            },
            "fullyQualifiedName": "Vehicle.EngineIdleTime",
            "interfaceId": "Qq1acaenByOB3sSM39SYm",
            "type": "CAN_SIGNAL"
        },
        {
            "canSignal": {
                "name": "Engine_Run_Time",
                "factor": 1,
                "isBigEndian": true,
                "isSigned": false,
                "length": 24,
                "messageId": 271343712,
                "offset": 0,
                "startBit": 40
            },
            "fullyQualifiedName": "Vehicle.EngineRunTime",
            "interfaceId": "Qq1acaenByOB3sSM39SYm",
            "type": "CAN_SIGNAL"
        }
    ]
}
```

- *decoder-manifest-name*Ersetzen Sie es durch den Namen des Decoder-Manifests, das Sie erstellen.
- *vehicle-model-ARN*Ersetzen Sie es durch den Amazon-Ressourcennamen (ARN) des Fahrzeugmodells.
- (Optional) *description* Ersetzen Sie es durch eine Beschreibung, die Ihnen hilft, das Decoder-Manifest zu identifizieren.

Die Reihenfolge der Eigenschaftsknoten innerhalb einer Struktur (Struktur) muss konsistent bleiben, wie sie im Signalkatalog und im Fahrzeugmodell (Modellmanifest) definiert ist. Weitere Informationen zur Konfiguration von Verzweigungen, Attributen, Sensoren und Aktuatoren finden Sie unterKonfiguration von AWS FleetWise IoT-Netzwerkschnittstellen und Decodersignalen.

```
{
 "name": "decoder-manifest-name",
 "modelManifestArn": "vehicle-model-arn",
 "description": "description",
 "networkInterfaces": [{
  "canInterface": {
   "name": "myNetworkInterface",
   "protocolName": "CAN",
   "protocolVersion": "2.0b"
  },
  "interfaceId": "Qq1acaenByOB3sSM39SYm",
  "type": "CAN_INTERFACE"
 }, {
  "type": "VEHICLE_MIDDLEWARE",
  "interfaceId": "G1KzxkdnmV5Hn7wkV3ZL9",
  "vehicleMiddleware": {
   "name": "ROS2_test",
   "protocolName": "ROS_2"
  }
 }],
 "signalDecoders": [{
   "canSignal": {
    "name": "Engine_Idle_Time",
    "factor": 1,
    "isBigEndian": true,
    "isSigned": false,
    "length": 24,
    "messageId": 271343712,
    "offset": 0,
    "startBit": 16
   },
   "fullyQualifiedName": "Vehicle.EngineIdleTime",
   "interfaceId": "Qq1acaenByOB3sSM39SYm",
   "type": "CAN_SIGNAL"
  },
  {
   "canSignal": {
    "name": "Engine_Run_Time",
```

```
    "factor": 1,
    "isBigEndian": true,
    "isSigned": false,
    "length": 24,
    "messageId": 271343712,
    "offset": 0,
    "startBit": 40
   },
   "fullyQualifiedName": "Vehicle.EngineRunTime",
   "interfaceId": "Qq1acaenByOB3sSM39SYm",
   "type": "CAN_SIGNAL"
  },
  {
   "fullyQualifiedName": "Vehicle.CompressedImageTopic",
   "type": "MESSAGE_SIGNAL",
   "interfaceId": "G1KzxkdnmV5Hn7wkV3ZL9",
   "messageSignal": {
    "topicName": "CompressedImageTopic:sensor_msgs/msg/CompressedImage",
    "structuredMessage": {
     "structuredMessageDefinition": [{
       "fieldName": "header",
       "dataType": {
        "structuredMessageDefinition": [{
          "fieldName": "stamp",
          "dataType": {
           "structuredMessageDefinition": [{
             "fieldName": "sec",
             "dataType": {
              "primitiveMessageDefinition": {
               "ros2PrimitiveMessageDefinition": {
                "primitiveType": "INT32"
               }
              }
             }
            },
            {
             "fieldName": "nanosec",
             "dataType": {
              "primitiveMessageDefinition": {
               "ros2PrimitiveMessageDefinition": {
                "primitiveType": "UINT32"
               }
              }
             }
```

```
            }
           ]
          }
         },
         {
          "fieldName": "frame_id",
          "dataType": {
           "primitiveMessageDefinition": {
            "ros2PrimitiveMessageDefinition": {
             "primitiveType": "STRING"
            }
           }
          }
         }
        ]
       }
      },
      {
       "fieldName": "format",
       "dataType": {
        "primitiveMessageDefinition": {
         "ros2PrimitiveMessageDefinition": {
          "primitiveType": "STRING"
         }
        }
       }
      },
      {
       "fieldName": "data",
       "dataType": {
        "structuredMessageListDefinition": {
         "name": "listType",
         "memberType": {
          "primitiveMessageDefinition": {
           "ros2PrimitiveMessageDefinition": {
            "primitiveType": "UINT8"
           }
          }
         },
         "capacity": 0,
         "listType": "DYNAMIC_UNBOUNDED_CAPACITY"
        }
       }
      }
```

```
      ]
     }
    }
   }
  ]
}
```

- *decoder-manifest-name*Ersetzen Sie es durch den Namen des Decoder-Manifests, das Sie erstellen.
- *vehicle-model-ARN*Ersetzen Sie es durch den Amazon-Ressourcennamen (ARN) des Fahrzeugmodells.
- (Optional) *description* Ersetzen Sie es durch eine Beschreibung, die Ihnen hilft, das Decoder-Manifest zu identifizieren.

Weitere Informationen zur Konfiguration von Verzweigungen, Attributen, Sensoren und Aktuatoren finden Sie unter. Konfiguration von AWS FleetWise IoT-Netzwerkschnittstellen und Decodersignalen

```
{
 "name": "decoder-manifest-name",
 "modelManifestArn": "vehicle-model-arn",
 "description": "description",
 "networkInterfaces": [
        {
      "interfaceId": "myCustomInterfaceId",
      "type": "CUSTOM_DECODING_INTERFACE",
            "customDecodingInterface": {
                "name": "myCustomInterface"
            }
        }
    ],
    "signalDecoders": [
        {
            "customDecodingSignal": {
                "fullyQualifiedName": "Vehicle.actuator1",
                "interfaceId": "myCustomInterfaceId",
                "type": "CUSTOM_DECODING_SIGNAL",
                "customDecodingSignal": {
                    "id": "Vehicle.actuator1"
                }
            }
```

```
        },
        {
            "customDecodingSignal": {
                "fullyQualifiedName": "Vehicle.actuator2",
                "interfaceId": "myCustomInterfaceId",
                "type": "CUSTOM_DECODING_SIGNAL",
                "customDecodingSignal": {
                    "id": "Vehicle.actuator2"
                }
            }
        }
    ]
}
```

> **Note**
>
> Sie können ein Demo-Skript herunterladen, um ein Decoder-Manifest mit Signalen des Bildverarbeitungssystems zu erstellen. Weitere Informationen finden Sie im Vision System Data Developer Guide.
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `CreateDecoderManifest` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
```

```
        }
    ]
}
```

Ein AWS FleetWise IoT-Decoder-Manifest aktualisieren

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können den UpdateDecoderManifestAPI-Vorgang verwenden, um ein Decoder-Manifest zu aktualisieren. Sie können Netzwerkschnittstellen und Signaldecoder hinzufügen, entfernen und aktualisieren. Sie können auch den Status des Decoder-Manifests ändern. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um ein Decoder-Manifest zu aktualisieren.

*decoder-manifest-name*Ersetzen Sie es durch den Namen des Decoder-Manifests, das Sie aktualisieren.

```
aws iotfleetwise update-decoder-manifest /
                --name decoder-manifest-name /
                --status ACTIVE
```

Wenn für die Signale keine festgelegten Dekodierungsregeln gelten, können Sie Standard-Dekodierungsregeln erstellen. Die Signale werden einer benutzerdefinierten dekodierten Schnittstelle hinzugefügt, wobei der vollständig qualifizierte Name des Signals `CustomDecodingSignal$id` festgelegt wird. Führen Sie den folgenden Befehl aus, um ein Decoder-Manifest mit den Standard-Dekodierungsregeln zu aktualisieren.

*decoder-manifest-name*Ersetzen Sie es durch den Namen des Decoder-Manifests, das Sie aktualisieren.

```
aws iotfleetwise update-decoder-manifest /
                --name decoder-manifest-name /
                --status ACTIVE
```

```
                --default-for-unmapped-signals CUSTOM_DECODING
```

> **Important**
>
> Nachdem Sie das Decoder-Manifest aktiviert haben, können Sie es nicht mehr bearbeiten.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `UpdateDecoderManifest` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Überprüfen Sie das Decoder-Manifest-Update

Sie können den ListDecoderManifestSignalsAPI-Vorgang verwenden, um zu überprüfen, ob die Decodersignale im Decoder-Manifest aktualisiert wurden. Das folgende Beispiel verwendet. AWS CLI

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Decodersignale (Knoten) in einem bestimmten Decoder-Manifest abzurufen.

Ersetzen Sie es *`decoder-manifest-name`* durch den Namen des Decoder-Manifests, das Sie überprüfen.

```
aws iotfleetwise list-decoder-manifest-signals /
                --name decoder-manifest-name
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ListDecoderManifestSignals` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Sie können den ListDecoderManifestNetworkInterfacesAPI-Vorgang verwenden, um zu überprüfen, ob die Netzwerkschnittstellen im Decoder-Manifest aktualisiert wurden. Im folgenden Beispiel wird AWS CLI verwendet.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Netzwerkschnittstellen in einem bestimmten Decoder-Manifest abzurufen.

*decoder-manifest-name*Ersetzen Sie es durch den Namen des Decoder-Manifests, das Sie überprüfen.

```
aws iotfleetwise list-decoder-manifest-network-interfaces /
                --name decoder-manifest-name
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ListDecoderManifestNetworkInterfaces` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Löschen Sie ein AWS FleetWise IoT-Decoder-Manifest

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um ein Decoder-Manifest zu löschen.

> **Important**
>
> Fahrzeuge, die dem Decoder-Manifest zugeordnet sind, müssen zuerst gelöscht werden. Weitere Informationen finden Sie unter Löschen Sie ein AWS FleetWise IoT-Fahrzeug.

Themen

- Löschen Sie ein Decoder-Manifest (Konsole)
- Löschen Sie ein Decoder-Manifest ()AWS CLI

Löschen Sie ein Decoder-Manifest (Konsole)

Sie können die AWS FleetWise IoT-Konsole verwenden, um ein Decoder-Manifest zu löschen.

Um ein Decoder-Manifest zu löschen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeugmodelle aus.
3. Wählen Sie das Zielfahrzeugmodell aus.
4. Wählen Sie auf der Seite mit der Zusammenfassung des Fahrzeugmodells die Registerkarte Decoder-Manifeste aus.
5. Wählen Sie das Ziel-Decoder-Manifest aus und klicken Sie dann auf Löschen.
6. Unter Löschen? **`decoder-manifest-name`** , geben Sie den Namen des Decoder-Manifests ein, das gelöscht werden soll, und wählen Sie dann Confirm.

Löschen Sie ein Decoder-Manifest ()AWS CLI

Sie können den DeleteDecoderManifestAPI-Vorgang verwenden, um ein Decoder-Manifest zu löschen. Das folgende Beispiel verwendet AWS CLI.

> **Important**
>
> Bevor Sie das Decoder-Manifest löschen, löschen Sie zuerst die zugehörigen Fahrzeuge. Weitere Informationen finden Sie unter Löschen Sie ein AWS FleetWise IoT-Fahrzeug.

Führen Sie den folgenden Befehl aus, um ein Decoder-Manifest zu löschen.

*`decoder-manifest-name`*Ersetzen Sie es durch den Namen des Decoder-Manifests, das Sie löschen möchten.

```
aws iotfleetwise delete-decoder-manifest --name decoder-manifest-name
```

Überprüfen Sie, ob das Decoder-Manifest

Sie können den ListDecoderManifestsAPI-Vorgang verwenden, um zu überprüfen, ob ein Decoder-Manifest gelöscht wurde. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Decoder-Manifeste abzurufen.

```
aws iotfleetwise list-decoder-manifests
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den API-Vorgang aufrufen kann. `ListDecoderManifests`

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Informationen zum AWS FleetWise IoT-Decoder-Manifest abrufen

Sie können den GetDecoderManifestAPI-Vorgang verwenden, um zu überprüfen, ob Netzwerkschnittstellen und Signaldecoder im Decoder-Manifest aktualisiert wurden. Das folgende Beispiel verwendet. AWS CLI

Führen Sie den folgenden Befehl aus, um Informationen über ein Decoder-Manifest abzurufen.

*decoder-manifest*Ersetzen Sie es durch den Namen des Decoder-Manifests, das Sie abrufen möchten.

```
aws iotfleetwise get-decoder-manifest --name decoder-manifest
```

> **Note**
>
> Diese Operation ist letztlich konsistent. Mit anderen Worten, Änderungen am Decoder-Manifest werden möglicherweise nicht sofort übernommen.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `GetDecoderManifest` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

AWS FleetWise IoT-Fahrzeuge verwalten

Fahrzeuge sind Beispiele für Fahrzeugmodelle. Fahrzeuge müssen anhand eines Fahrzeugmodells erstellt und einem Decoder-Manifest zugeordnet werden. Vehicles lädt einen oder mehrere Datenströme in die Cloud hoch. Beispielsweise kann ein Fahrzeug Daten zu Kilometerstand, Motortemperatur und Zustand der Heizung an die Cloud senden. Jedes Fahrzeug enthält die folgenden Informationen:

`vehicleName`

Eine ID, die das Fahrzeug identifiziert.

Fügen Sie Ihrem Fahrzeugnamen keine personenbezogenen Daten (PII) oder andere vertrauliche oder sensible Informationen hinzu. Fahrzeugnamen sind für andere AWS Dienste, einschließlich Amazon, zugänglich CloudWatch. Fahrzeugnamen sind nicht dazu bestimmt, für private oder sensible Daten verwendet zu werden.

`modelManifestARN`

Der Amazon-Ressourcenname (ARN) eines Fahrzeugmodells (Modellmanifest). Jedes Fahrzeug wird aus einem Fahrzeugmodell erstellt. Fahrzeuge, die mit demselben Fahrzeugmodell erstellt wurden, bestehen aus derselben Gruppe von Signalen, die vom Fahrzeugmodell übernommen wurden. Diese Signale sind im Signalkatalog definiert und standardisiert.

`decoderManifestArn`

Der ARN des Decoder-Manifests. Ein Decoder-Manifest bietet Dekodierungsinformationen, die das AWS IoT verwenden FleetWise kann, um Rohsignaldaten (Binärdaten) in für Menschen lesbare Werte umzuwandeln. Ein Decoder-Manifest muss einem Fahrzeugmodell zugeordnet sein. AWS IoT FleetWise verwendet dasselbe Decoder-Manifest, um Rohdaten von Fahrzeugen zu dekodieren, die auf der Grundlage desselben Fahrzeugmodells erstellt wurden.

`attributes`

Attribute sind Schlüssel-Wert-Paare, die statische Informationen enthalten. Fahrzeuge können Attribute enthalten, die vom Fahrzeugmodell übernommen wurden. Sie können zusätzliche Attribute hinzufügen, um ein einzelnes Fahrzeug von anderen Fahrzeugen zu unterscheiden, die mit demselben Fahrzeugmodell erstellt wurden. Wenn Sie beispielsweise ein schwarzes Auto haben, können Sie den folgenden Wert für ein Attribut angeben:`{"color": "black"}`.

> **Important**
>
> Attribute müssen im zugehörigen Fahrzeugmodell definiert werden, bevor Sie sie einzelnen Fahrzeugen hinzufügen können.

Weitere Informationen zu Fahrzeugmodellen, Decoder-Manifesten und Attributen finden Sie unterAWS FleetWise IoT-Fahrzeuge modellieren.

AWS IoT FleetWise bietet die folgenden API-Operationen, mit denen Sie Fahrzeuge erstellen und verwalten können.

- CreateVehicle— Erzeugt ein neues Fahrzeug.
- BatchCreateVehicle— Erzeugt ein oder mehrere neue Fahrzeuge.
- UpdateVehicle— Aktualisiert ein vorhandenes Fahrzeug.
- BatchUpdateVehicle— Aktualisiert ein oder mehrere bestehende Fahrzeuge.
- DeleteVehicle— Löscht ein vorhandenes Fahrzeug.
- ListVehicles— Ruft eine paginierte Liste mit Zusammenfassungen aller Fahrzeuge ab.
- GetVehicle— Ruft Informationen über ein Fahrzeug ab.

Lernprogramme

- Bereitstellung von AWS FleetWise IoT-Fahrzeugen
- Reservierte Themen im AWS Internet der Dinge FleetWise
- Erstellen Sie ein AWS FleetWise IoT-Fahrzeug
- Erstellen Sie mehrere AWS FleetWise IoT-Fahrzeuge
- Aktualisieren Sie ein AWS FleetWise IoT-Fahrzeug
- Aktualisieren Sie mehrere AWS FleetWise IoT-Fahrzeuge
- Löschen Sie ein AWS FleetWise IoT-Fahrzeug
- Holen Sie sich AWS FleetWise IoT-Fahrzeuginformationen

Bereitstellung von AWS FleetWise IoT-Fahrzeugen

Die Edge Agent for AWS FleetWise IoT-Software, die in Ihrem Fahrzeug ausgeführt wird, sammelt und überträgt Daten in die Cloud. AWS IoT FleetWise lässt sich integrieren AWS IoT Core , um

die sichere Kommunikation zwischen der Edge Agent-Software und der Cloud über MQTT zu unterstützen. Jedes Fahrzeug entspricht einer AWS IoT Sache. Sie können ein vorhandenes Objekt verwenden AWS IoT , um ein Fahrzeug zu erstellen, oder AWS IoT so einstellen FleetWise , dass automatisch ein Objekt AWS IoT für Ihr Fahrzeug erstellt wird. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeug.

AWS IoT Core unterstützt Authentifizierung und Autorisierung, mit deren Hilfe der Zugriff auf AWS FleetWise IoT-Ressourcen sicher gesteuert werden kann. Fahrzeuge können X.509-Zertifikate verwenden, um sich zu authentifizieren (anzumelden), um AWS IoT zu nutzen, FleetWise und AWS IoT Core Richtlinien, um autorisiert zu werden (über Berechtigungen zu verfügen), um bestimmte Aktionen auszuführen.

Fahrzeuge authentifizieren

Sie können AWS IoT Core Richtlinien zur Authentifizierung Ihrer Fahrzeuge erstellen.

Um Ihr Fahrzeug zu authentifizieren

- Führen Sie den folgenden Befehl aus, um eine AWS IoT Core Richtlinie zu erstellen.
  - *policy-name*Ersetzen Sie es durch den Namen der Richtlinie, die Sie erstellen möchten.
  - *file-name*Ersetzen Sie es durch den Namen der JSON-Datei, die die AWS IoT Core Richtlinie enthält.

```
aws iot create-policy --policy-name policy-name --policy-document file://file-
name.json
```

  Bevor Sie die Beispielrichtlinie verwenden, gehen Sie wie folgt vor:

  - *us-east-1*Ersetzen Sie durch die AWS Region, in der Sie AWS FleetWise IoT-Ressourcen erstellt haben.
  - *111122223333*Ersetzen Sie es durch Ihre AWS Konto-ID.

  Dieses Beispiel beinhaltet Themen, die dem AWS Internet der Dinge vorbehalten sind FleetWise. Sie müssen die Themen zur Richtlinie hinzufügen. Weitere Informationen finden Sie unter Reservierte Themen im AWS Internet der Dinge FleetWise.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iot:Connect"
            ],
            "Resource": [
                "arn:aws:iot:us-east-1:111122223333:client/
${iot:Connection.Thing.ThingName}"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iot:Publish"
            ],
            "Resource": [
                "arn:aws:iot:us-east-1:111122223333:topic/$aws/iotfleetwise/
vehicles/${iot:Connection.Thing.ThingName}/checkins",
                "arn:aws:iot:us-east-1:111122223333:topic/$aws/iotfleetwise/
vehicles/${iot:Connection.Thing.ThingName}/signals"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "iot:Subscribe"
            ],
            "Resource": [
                "arn:aws:iot:us-east-1:111122223333:topicfilter/$aws/
iotfleetwise/vehicles/${iot:Connection.Thing.ThingName}/collection_schemes",
                "arn:aws:iot:us-east-1:111122223333:topicfilter/$aws/
iotfleetwise/vehicles/${iot:Connection.Thing.ThingName}/decoder_manifests"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "iot:Receive"
            ],
            "Resource": [
                "arn:aws:iot:us-east-1:111122223333:topic/$aws/iotfleetwise/
vehicles/${iot:Connection.Thing.ThingName}/collection_schemes",
                "arn:aws:iot:us-east-1:111122223333:topic/$aws/iotfleetwise/
vehicles/${iot:Connection.Thing.ThingName}/decoder_manifests"
            ]
        }
    ]
}
```

Fahrzeuge autorisieren

Sie können X.509-Zertifikate erstellen, um Ihre Fahrzeuge zu autorisieren.

Um Ihr Fahrzeug zu autorisieren

> **Important**
>
> Wir empfehlen Ihnen, für jedes Fahrzeug ein neues Zertifikat zu erstellen.

1. Führen Sie den folgenden Befehl aus, um ein RSA-Schlüsselpaar zu erstellen und ein X.509-Zertifikat auszustellen.

   - *cert*Ersetzen Sie es durch den Namen der Datei, die den Inhalt der Befehlsausgabe von CertificatePEM speichert.
   - *public-key*Ersetzen Sie es durch den Namen der Datei, die den Inhalt der Befehlsausgabe von KeyPair speichert. PublicKey.
   - *private-key*Ersetzen Sie es durch den Namen der Datei, die den Inhalt der Befehlsausgabe von KeyPair speichert. PrivateKey.

```
aws iot create-keys-and-certificate \
    --set-as-active \
    --certificate-pem-outfile cert.pem \
    --public-key-outfile public-key.key" \
    --private-key-outfile  private-key.key"
```

2. Kopieren Sie den Amazon-Ressourcennamen (ARN) des Zertifikats aus der Ausgabe.
3. Führen Sie den folgenden Befehl aus, um die Richtlinie an das Zertifikat anzuhängen.

   - *policy-name*Ersetzen Sie es durch den Namen der AWS IoT Core Richtlinie, die Sie erstellt haben.
   - *certificate-arn*Ersetzen Sie es durch den ARN des Zertifikats, das Sie kopiert haben.

   ```
   aws iot attach-policy \
       --policy-name policy-name\
       --target "certificate-arn"
   ```

4. Führen Sie den folgenden Befehl aus, um das Zertifikat an das Ding anzuhängen.

   - *thing-name*Ersetze es durch den Namen deiner AWS IoT Sache oder die ID deines Fahrzeugs.
   - *certificate-arn*Ersetzen Sie es durch den ARN des Zertifikats, das Sie kopiert haben.

   ```
   aws iot attach-thing-principal \
       --thing-name thing-name \
       --principal "certificate-arn"
   ```

Reservierte Themen im AWS Internet der Dinge FleetWise

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

AWS IoT FleetWise behält sich die Verwendung der folgenden Themen vor. Wenn das reservierte Thema es zulässt, können Sie es abonnieren oder veröffentlichen. Sie können jedoch keine neuen Themen erstellen, die mit einem Dollarzeichen ($) beginnen. Wenn Sie nicht unterstützte Veröffentlichungs- oder Abonnementvorgänge mit reservierten Themen verwenden, kann dies dazu führen, dass die Verbindung beendet wird.

Thema	Client-Betrieb zulässig	Description
`$aws/iotfleetwise/vehicles/vehicleName/checkins`	Veröffentlichen	Die Edge Agent-Software veröffentlicht Fahrzeugstatusinformationen zu diesem Thema. Fahrzeugstatusinformationen werden im Protokollpufferformat (Protobuf) ausgetauscht. Weitere Informationen finden Sie im Edge Agent for AWS FleetWise IoT-Softwareentwicklerhandbuch.
`$aws/iotfleetwise/vehicles/vehicleName/signals`	Veröffentlichen	Die Edge Agent-Software veröffentlicht Signale zu diesem Thema. Signalinformationen werden im Protokollpufferformat (Protobuf) ausgetauscht. Weitere Informationen finden Sie im Edge Agent for AWS FleetWise IoT-Softwareentwicklerhandbuch.
`$aws/iotfleetwise/`	Abonnieren	AWS IoT FleetWise veröffentlicht

Thema	Client-Betrieb zulässig	Description	
vehicles/ *vehicleName* /collection_schemes		Datenerfassungsschemata zu diesem Thema. Fahrzeuge nutzen diese Datenerfassungsschemata.	
$aws/iotfleetwise/vehicles/ *vehicleName* /decoder_manifests	Abonnieren	AWS IoT FleetWise veröffentlicht Decoder-Manifeste zu diesem Thema. Fahrzeuge verbrauchen diese Decoder-Manifeste.	
$aws/iotfleetwise/vehicles/ *vehicleName* /command/request	Abonnieren	AWS IoT FleetWise veröffentlicht Anfragen zur Ausführung von Befehlen zu diesem Thema. Fahrzeuge verarbeiten diese Befehlsanforderungen dann.	

Thema	Client-Betrieb zulässig	Description
`$aws/iotfleetwise/vehicles/vehicleName/command/response`	Veröffentlichen	Die Edge Agent-Software veröffentlicht Befehlsantworten des Fahrzeugs zu diesem Thema. Befehlsantworten werden im Protokollpufferformat (Protobuf) ausgetauscht. Weitere Informationen finden Sie im Edge Agent for AWS FleetWise IoT-Softwareentwicklerhandbuch.
`$aws/iotfleetwise/vehicles/vehicleName/command/notification`	Abonnieren	AWS IoT FleetWise veröffentlicht Befehlsstatusaktualisierungen zu diesem Thema. Die Benachrichtigungen werden in einem JSON-Format gesendet.
`$aws/iotfleetwise/vehicles/$vehicle_name/last_known_states/config`	Abonnieren	AWS IoT FleetWise veröffentlicht Statusvorlagenkonfigurationen zu diesem Thema. Fahrzeuge verwenden diese Statusvorlagenkonfigurationen.

Thema	Client-Betrieb zulässig	Description	
`$aws/iotfleetwise/vehicles/ $vehicle_name /last_known_states/data`	Veröffentlichen	Die Edge Agent-Software veröffentlicht die aus den Signalen gesammelten Daten zu diesem Thema.	

Thema	Client-Betrieb zulässig	Description	
`$aws/iotfleetwise/vehicles/` *`$vehicle_name`* `/last_known_state/` *`$state_template_name`* `/data`	Abonnieren	AWS IoT FleetWise veröffentlicht Daten, die anhand der im Abschnitt *`$state_template_name`* zu diesem Thema konfigurierten Signale gesammelt wurden. Die Aktualisierungen können teilweise sein. Wenn beispielsweise eine Zustandsvorlagenzuordnung mehrere Signale mit der Aktualisierungsstrategie bei Änderung enthält, dann sind nur die Signale, die sich geändert haben, in einer bestimmten Nachricht enthalten. Signalinformationen werden im Protokollpufferformat (Protobuf) ausgetauscht. Weitere Informationen finden Sie im Edge Agent for AWS FleetWise IoT-Softwareentwicklerhandbuch.	

Erstellen Sie ein AWS FleetWise IoT-Fahrzeug

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um ein Fahrzeug zu erstellen.

> **Important**
>
> Bevor Sie beginnen, überprüfen Sie Folgendes:
>
> - Sie müssen über ein Fahrzeugmodell verfügen und der Status des Fahrzeugmodells muss lauten`ACTIVE`. Weitere Informationen finden Sie unter AWS FleetWise IoT-Fahrzeugmodelle verwalten.
> - Ihr Fahrzeugmodell muss mit einem Decoder-Manifest verknüpft sein, und der Status des Decoder-Manifests muss lauten. `ACTIVE` Weitere Informationen finden Sie unter AWS FleetWise IoT-Decoder-Manifeste verwalten.

Themen

- Erstellen Sie ein Fahrzeug (Konsole)
- Erstellen Sie ein Fahrzeug (AWS CLI)

Erstellen Sie ein Fahrzeug (Konsole)

Sie können die AWS FleetWise IoT-Konsole verwenden, um ein Fahrzeug zu erstellen.

Um ein Fahrzeug zu erstellen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeuge aus.
3. Wählen Sie auf der Seite mit der Fahrzeugübersicht die Option Fahrzeug erstellen aus, und führen Sie dann die folgenden Schritte aus.

Themen

- Schritt 1: Definieren Sie die Fahrzeugeigenschaften
- Schritt 2: Fahrzeugzertifikat konfigurieren
- Schritt 3: Richtlinien an das Zertifikat anhängen
- Schritt 4: Überprüfen und Erstellen

Schritt 1: Definieren Sie die Fahrzeugeigenschaften

In diesem Schritt benennen Sie das Fahrzeug und verknüpfen es mit dem Modellmanifest und dem Decoder-Manifest.

1. Geben Sie einen eindeutigen Namen für das Fahrzeug ein.

   > **Important**
   >
   > Ein Fahrzeug entspricht einer AWS IoT Sache. Wenn bereits ein Objekt mit diesem Namen existiert, wählen Sie Fahrzeug mit einem IoT-Ding verknüpfen, um das Ding mit dem Fahrzeug zu aktualisieren. Oder wählen Sie einen anderen Fahrzeugnamen und AWS IoT erstellt FleetWise automatisch eine neue Sache für das Fahrzeug.

2. Wählen Sie ein Fahrzeugmodell (Modellmanifest) aus der Liste aus.
3. Wählen Sie ein Decoder-Manifest aus der Liste aus. Das Decoder-Manifest ist dem Fahrzeugmodell zugeordnet.
4. (Optional) Um Fahrzeugattribute zuzuordnen, wählen Sie Attribute hinzufügen. Wenn Sie diesen Schritt überspringen, müssen Sie nach der Erstellung des Fahrzeugs Attribute hinzufügen, bevor Sie es für Kampagnen einsetzen können.
5. (Optional) Um Tags mit dem Fahrzeug zu verknüpfen, wählen Sie Neues Tag hinzufügen. Sie können Tags auch hinzufügen, nachdem das Fahrzeug erstellt wurde.
6. Wählen Sie Weiter aus.

Schritt 2: Fahrzeugzertifikat konfigurieren

Um Ihr Fahrzeug als Objekt verwenden zu AWS IoT können, müssen Sie ein Fahrzeugzertifikat mit einer angehängten Richtlinie konfigurieren. Wenn Sie diesen Schritt überspringen, müssen Sie nach der Erstellung des Fahrzeugs ein Zertifikat konfigurieren, bevor Sie es für Kampagnen einsetzen können.

1. Wählen Sie Neues Zertifikat automatisch generieren (empfohlen).
2. Wählen Sie Weiter aus.

Schritt 3: Richtlinien an das Zertifikat anhängen

Hängen Sie eine Richtlinie an das Zertifikat an, das Sie im vorherigen Schritt konfiguriert haben.

1. Geben Sie unter Richtlinien einen vorhandenen Richtliniennamen ein. Um eine neue Richtlinie zu erstellen, wählen Sie Richtlinie erstellen aus.
2. Wählen Sie Weiter aus.

Schritt 4: Überprüfen und Erstellen

Überprüfen Sie die Konfigurationen für das Fahrzeug und wählen Sie dann Fahrzeug erstellen aus.

> **Important**
>
> Nachdem das Fahrzeug erstellt wurde, müssen Sie das Zertifikat und die Schlüssel herunterladen. Sie verwenden das Zertifikat und den privaten Schlüssel, um das Fahrzeug in der Edge Agent for AWS FleetWise IoT-Software zu verbinden.

Erstellen Sie ein Fahrzeug (AWS CLI)

Wenn Sie ein Fahrzeug erstellen, müssen Sie ein Fahrzeugmodell verwenden, das einem Decoder-Manifest zugeordnet ist. Sie können den CreateVehicleAPI-Vorgang verwenden, um ein Fahrzeug zu erstellen. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um ein Fahrzeug zu erstellen.

*`file-name`*Ersetzen Sie ihn durch den Namen der JSON-Datei, die die Fahrzeugkonfiguration enthält.

```
aws iotfleetwise create-vehicle --cli-input-json file://file-name.json
```

Example— Fahrzeugkonfiguration

- (Optional) Der `associationBehavior` Wert kann einer der folgenden sein:

- `CreateIotThing`— Wenn Ihr Fahrzeug erstellt wird, erstellt AWS IoT FleetWise automatisch eine AWS IoT Sache mit dem Namen Ihrer Fahrzeug-ID für Ihr Fahrzeug.
- `ValidateIotThingExists`— Verwenden Sie ein vorhandenes AWS IoT Objekt, um ein Fahrzeug zu erstellen.

  Um ein AWS IoT Ding zu erstellen, führen Sie den folgenden Befehl aus. *`thing-name`*Ersetzen Sie es durch den Namen des Dings, das Sie erstellen möchten.

  ```
  aws iot create-thing --thing-name thing-name
  ```

Wenn es nicht spezifiziert ist, erstellt AWS IoT FleetWise automatisch AWS IoT etwas für Ihr Fahrzeug.

> **Important**
>
> Stellen Sie sicher, dass das AWS IoT Ding nach der Erstellung des Fahrzeugs bereitgestellt wird. Weitere Informationen finden Sie unter Bereitstellung von AWS FleetWise IoT-Fahrzeugen.

- *`vehicle-name`*Ersetzen Sie es durch eine der folgenden Optionen.
  - Der Name Ihres Dings AWS IoT , falls `associationBehavior` es so konfiguriert ist`ValidateIotThingExists`.
  - Die ID des Fahrzeugs, für das es erstellt werden soll, `associationBehavior` ist konfiguriert`CreateIotThing`.

    Die Fahrzeug-ID kann 1—100 Zeichen lang sein. Gültige Zeichen: a—z, A—Z, 0—9, Bindestrich (-), Unterstrich (_) und Doppelpunkt (:).
- *`model-manifest-ARN`*Ersetzen Sie es durch den ARN Ihres Fahrzeugmodells (Modellmanifest).
- Durch den *`decoder-manifest-ARN`* ARN des Decoder-Manifests ersetzen, das dem angegebenen Fahrzeugmodell zugeordnet ist.
- (Optional) Sie können zusätzliche Attribute hinzufügen, um dieses Fahrzeug von anderen Fahrzeugen zu unterscheiden, die mit demselben Fahrzeugmodell erstellt wurden. Wenn Sie beispielsweise ein Elektroauto haben, können Sie den folgenden Wert für ein Attribut angeben:`{"fuelType": "electric"}`.

> **Important**
>
> Attribute müssen im zugehörigen Fahrzeugmodell definiert werden, bevor Sie sie einzelnen Fahrzeugen hinzufügen können.

```
{
    "associationBehavior": "associationBehavior",
    "vehicleName": "vehicle-name",
    "modelManifestArn": "model-manifest-ARN",
    "decoderManifestArn": "decoder-manifest-ARN",
    "attributes": {
        "key": "value"
    }
}
```

Example— Ordnen Sie dem Fahrzeug eine Statusvorlage zu

Mithilfe des `stateTemplates` Felds können Sie dem Fahrzeug Statusvorlagen zuordnen, um die Erfassung von Statusaktualisierungen des Fahrzeugs in der Cloud zu ermöglichen.

In diesem Beispiel *`stateTemplateUpdateStrategy`* kann es sich um eines der folgenden handeln:

- `periodic`: ermöglicht es Ihnen, eine feste Rate anzugeben, mit der die Edge Agent-Software Signalaktualisierungen an die Cloud sendet (die Edge-Agent-Software sendet Updates auch dann, wenn sich der Signalwert zwischen den Updates nicht geändert hat).
- `onChange`: Die Edge Agent-Software sendet Signalaktualisierungen, wenn sich das Signal ändert.

```
aws iotfleetwise create-vehicle --cli-input-json file://create-vehicle.json
```

Wo die *`create-vehicle.json`* Datei enthält (zum Beispiel):

```
{
    "associationBehavior": "associationBehavior",
    "vehicleName": "vehicle-name",
    "modelManifestArn": "model-manifest-ARN",
    "decoderManifestArn": "decoder-manifest-ARN",
```

```
    "attributes": {
        "key": "value"
    },
    "stateTemplates": [
        {
            "identifier": "state-template-name",
            "stateTemplateUpdateStrategy": {
                "periodic": {
                    "stateTemplateUpdateRate": {
                        "unit": "SECOND",
                        "value": 10
                    }
                }
            }
        }
    ]
}
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `CreateVehicle` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Erstellen Sie mehrere AWS FleetWise IoT-Fahrzeuge

Sie können den BatchCreateVehicleAPI-Vorgang verwenden, um mehrere Fahrzeuge gleichzeitig zu erstellen. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um mehrere Fahrzeuge zu erstellen.

*file-name*Ersetzen Sie ihn durch den Namen der JSON-Datei, die die Konfigurationen mehrerer Fahrzeuge enthält.

```
aws iotfleetwise batch-create-vehicle --cli-input-json file://file-name.json
```

Example— Fahrzeugkonfigurationen

```
{
    "vehicles": [
        {
                "associationBehavior": "associationBehavior",
                "vehicleName": "vehicle-name",
                "modelManifestArn": "model-manifest-ARN",
                "decoderManifestArn": "decoder-manifest-ARN",
                "attributes": {
                    "key": "value"
                }
        },
        {
                "associationBehavior": "associationBehavior",
                "vehicleName": "vehicle-name",
                "modelManifestArn": "model-manifest-ARN",
                "decoderManifestArn": "decoder-manifest-ARN",
                "attributes": {
                    "key": "value"
                }
        }
    ]
}
```

Sie können bis zu 10 Fahrzeuge für jeden Batch-Vorgang erstellen. Weitere Informationen zur Fahrzeugkonfiguration finden Sie unterErstellen Sie ein AWS FleetWise IoT-Fahrzeug.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `BatchCreateVehicle` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Aktualisieren Sie ein AWS FleetWise IoT-Fahrzeug

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können den UpdateVehicleAPI-Vorgang verwenden, um ein vorhandenes Fahrzeug zu aktualisieren. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um ein Fahrzeug zu aktualisieren.

*file-name*Ersetzen Sie ihn durch den Namen der JSON-Datei, die die Konfiguration Ihres Fahrzeugs enthält.

```
aws iotfleetwise update-vehicle --cli-input-json file://file-name.json
```

Example— Fahrzeugkonfiguration

- *vehicle-name*Ersetzen Sie es durch die ID des Fahrzeugs, das Sie aktualisieren möchten.
- (Optional) *model-manifest-ARN* Ersetzen Sie es durch den ARN des Fahrzeugmodells (Modellmanifest), mit dem Sie das verwendete Fahrzeugmodell ersetzen.
- (Optional) *decoder-manifest-ARN* Ersetzen Sie es durch den ARN Ihres Decoder-Manifests, das dem von Ihnen angegebenen neuen Fahrzeugmodell zugeordnet ist.
- (Optional) Durch Fahrzeugattribute *attribute-update-mode* ersetzen.
  - `Merge`— Führen Sie neue Attribute mit vorhandenen Attributen zusammen, indem Sie bestehende Attribute mit neuen Werten aktualisieren und neue Attribute hinzufügen, falls sie nicht existieren.

    Wenn ein Fahrzeug beispielsweise die folgenden Attribute hat:`{"color": "black", "fuelType": "electric"}`, und Sie aktualisieren das Fahrzeug mit den folgenden Attributen:`{"color": "", "fuelType": "gasoline", "model": "x"}`, hat das aktualisierte Fahrzeug die folgenden Attribute:`{"fuelType": "gasoline", "model": "x"}`.
  - `Overwrite`— Ersetzt bestehende Attribute durch neue Attribute.

    Wenn ein Fahrzeug beispielsweise die folgenden Attribute hat:`{"color": "black", "fuelType": "electric"}`, und Sie das Fahrzeug mit dem `{"model": "x"}` Attribut aktualisieren, hat das aktualisierte Fahrzeug das `{"model": "x"}` Attribut.

  Dies ist erforderlich, wenn Attribute in der Eingabe vorhanden sind.
- (Optional) Um neue Attribute hinzuzufügen oder bestehende mit neuen Werten zu aktualisieren, konfigurieren Sie`attributes`. Wenn Sie beispielsweise ein Elektroauto haben, können Sie den folgenden Wert für ein Attribut angeben:`{"fuelType": "electric"}`.

  Um Attribute zu löschen, konfigurieren Sie `attributeUpdateMode` auf`Merge`.

> **Important**
>
> Attribute müssen im zugehörigen Fahrzeugmodell definiert werden, bevor Sie sie einzelnen Fahrzeugen hinzufügen können.

```
 {
         "vehicleName": "vehicle-name",
         "modelManifestArn": "model-manifest-arn",
         "decoderManifestArn": "decoder-manifest-arn",
         "attributeUpdateMode": "attribute-update-mode"
         }
}
```

Example— dem Fahrzeug zugeordnete Statusvorlagen hinzufügen oder entfernen

Mithilfe der folgenden Felder können Sie zusätzliche Statusvorlagen zuordnen oder bestehende Zuordnungen aus dem Fahrzeug entfernen:

- `stateTemplatesToAdd`
- `stateTemplatesToRemove`

```
aws iotfleetwise update-vehicle --cli-input-json file://update-vehicle.json
```

Wo die *`update-vehicle.json`* Datei enthält (zum Beispiel):

```
{
    "vehicleName": "vehicle-name",
    "modelManifestArn": "model-manifest-arn",
    "decoderManifestArn": "decoder-manifest-arn",
    "attributeUpdateMode": "attribute-update-mode",
    "stateTemplatesToAdd": [
        {
            "identifier": "state-template-name",
            "stateTemplateUpdateStrategy": {
                "onChange": {}
            }
        }
    ],
    "stateTemplatesToRemove": ["state-template-name"]
}
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `UpdateVehicle` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Aktualisieren Sie mehrere AWS FleetWise IoT-Fahrzeuge

Sie können den BatchUpdateVehicleAPI-Vorgang verwenden, um mehrere vorhandene Fahrzeuge gleichzeitig zu aktualisieren. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um mehrere Fahrzeuge zu aktualisieren.

*file-name*Ersetzen Sie ihn durch den Namen der JSON-Datei, die die Konfigurationen mehrerer Fahrzeuge enthält.

```
aws iotfleetwise batch-update-vehicle --cli-input-json file://file-name.json
```

Example— Fahrzeugkonfigurationen

```
{
   "vehicles": [
      {
         "vehicleName": "vehicle-name",
         "modelManifestArn": "model-manifest-arn",
         "decoderManifestArn": "decoder-manifest-arn",
         "mergeAttributes": true,
```

```
         "attributes": {
         "key": "value"
         }
      },
      {
         "vehicleName": "vehicle-name",
         "modelManifestArn": "model-manifest-arn",
         "decoderManifestArn": "decoder-manifest-arn",
         "mergeAttributes": true,
         "attributes": {
         "key": "value"
         }
      }
   ]
}
```

Sie können bis zu 10 Fahrzeuge für jeden Batch-Vorgang aktualisieren. Weitere Informationen zur Konfiguration der einzelnen Fahrzeuge finden Sie unterAktualisieren Sie ein AWS FleetWise IoT-Fahrzeug.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `BatchUpdateVehicle` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Löschen Sie ein AWS FleetWise IoT-Fahrzeug

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um Fahrzeuge zu löschen.

> **Important**
>
> Nachdem ein Fahrzeug gelöscht wurde, entfernt AWS IoT das Fahrzeug FleetWise automatisch aus den zugehörigen Flotten und Kampagnen. Weitere Informationen erhalten Sie unter Verwaltung von Flotten im Internet der Dinge AWS FleetWise und Sammeln Sie AWS FleetWise IoT-Daten mit Kampagnen. Das Fahrzeug existiert jedoch immer noch als Ding oder ist immer noch mit einem Ding in AWS IoT Core Verbindung gebracht. Anweisungen zum Löschen eines Objekts finden Sie unter Löschen eines Objekts im AWS IoT Core Entwicklerhandbuch.

Lösche ein Fahrzeug (Konsole)

Sie können die AWS FleetWise IoT-Konsole verwenden, um ein Fahrzeug zu löschen.

Um ein Fahrzeug zu löschen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeuge aus.
3. Wählen Sie auf der Seite Fahrzeuge die Schaltfläche neben dem Fahrzeug aus, das Sie löschen möchten.
4. Wählen Sie Löschen aus.
5. Geben Sie **`vehicle-name`** unter Löschen den Namen des Fahrzeugs ein und wählen Sie dann Löschen.

Lösche ein Fahrzeug (AWS CLI)

Sie können den DeleteVehicleAPI-Vorgang verwenden, um ein Fahrzeug zu löschen. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um ein Fahrzeug zu löschen.

*`vehicle-name`*Ersetzen Sie es durch die ID des Fahrzeugs, das Sie löschen möchten.

```
aws iotfleetwise delete-vehicle --vehicle-name vehicle-name
```

Überprüfen Sie das Löschen des Fahrzeugs

Sie können den ListVehiclesAPI-Vorgang verwenden, um zu überprüfen, ob ein Fahrzeug gelöscht wurde. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Fahrzeuge abzurufen.

```
aws iotfleetwise list-vehicles
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ListVehicles` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Holen Sie sich AWS FleetWise IoT-Fahrzeuginformationen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können den GetVehicleAPI-Vorgang verwenden, um Fahrzeuginformationen abzurufen. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um die Metadaten eines Fahrzeugs abzurufen.

*vehicle-name*Ersetzen Sie es durch die ID des Fahrzeugs, das Sie abrufen möchten.

```
aws iotfleetwise get-vehicle --vehicle-name vehicle-name
```

> **Note**
>
> Diese Operation ist letztlich konsistent. Mit anderen Worten, Änderungen am Fahrzeug werden möglicherweise nicht sofort übernommen.

Sie können den GetVehicleStatusAPI-Vorgang verwenden, um den Status von Ressourcen abzurufen, die einem Fahrzeug zugeordnet sind. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um den Status von Ressourcen abzurufen, die einem Fahrzeug zugeordnet sind.

- *vehicle-name*Ersetzen Sie ihn durch die ID des Fahrzeugs, dem die Ressourcen zugeordnet sind.
- *type*Ersetzen Sie durch den Typ der Ressource, deren Status Sie abrufen möchten. Gültige Werte für `type` sind `CAMPAIGN`, `STATE_TEMPLATE` und `DECODER`.

```
aws iotfleetwise get-vehicle-status --vehicle-name vehicle-name --type type
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle die `GetVehicle` `GetVehicleStatus` API-Operationen aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Verwaltung von Flotten im Internet der Dinge AWS FleetWise

Eine Flotte steht für eine Gruppe von Fahrzeugen. Eine Flotte ohne zugehörige Fahrzeuge ist eine leere Einheit. Bevor Sie die Flotte verwenden können, um mehrere Fahrzeuge gleichzeitig zu verwalten, müssen Sie der Flotte Fahrzeuge zuordnen. Ein Fahrzeug kann mehreren Flotten angehören. Durch den Einsatz einer Kampagne können Sie steuern, welche Daten aus einer Fahrzeugflotte erfasst werden sollen und wann Daten erfasst werden sollen. Weitere Informationen finden Sie unter Sammeln Sie AWS FleetWise IoT-Daten mit Kampagnen.

Eine Flotte enthält die folgenden Informationen.

`fleetId`

Die ID der Flotte.

(Optional) `description`

Eine Beschreibung, die Ihnen hilft, die Flotte zu finden.

`signalCatalogArn`

Der Amazon-Ressourcenname (ARN) des Signalkatalogs.

AWS IoT FleetWise bietet die folgenden API-Operationen, mit denen Sie Flotten erstellen und verwalten können.

- CreateFleet— Erstellt eine Gruppe von Fahrzeugen, die dieselbe Gruppe von Signalen enthalten.
- AssociateVehicleFleet— Ordnet ein Fahrzeug einer Flotte zu.
- DisassociateVehicleFleet— Trennt ein Fahrzeug von einer Flotte.
- UpdateFleet— Aktualisiert die Beschreibung für eine bestehende Flotte.
- DeleteFleet— Löscht eine bestehende Flotte.
- ListFleets— Ruft eine paginierte Liste mit Zusammenfassungen aller Flotten ab.
- ListFleetsForVehicle— Ruft eine paginierte Liste aller Flotten ab, zu IDs denen das Fahrzeug gehört.
- ListVehiclesInFleet— Ruft eine paginierte Liste mit Zusammenfassungen aller Fahrzeuge einer Flotte ab.

- GetFleet— Ruft Informationen über eine Flotte ab.

Themen

- Erstellen Sie eine AWS FleetWise IoT-Flotte
- Ordnen Sie ein AWS FleetWise IoT-Fahrzeug einer Flotte zu
- Trennen Sie ein AWS FleetWise IoT-Fahrzeug von einer Flotte
- Aktualisieren Sie eine AWS FleetWise IoT-Flotte
- Löschen Sie eine AWS FleetWise IoT-Flotte
- Holen Sie sich AWS FleetWise IoT-Flotteninformationen

Erstellen Sie eine AWS FleetWise IoT-Flotte

Sie können den CreateFleetAPI-Vorgang verwenden, um eine Fahrzeugflotte zu erstellen. Das folgende Beispiel verwendet AWS CLI.

> **Important**
>
> Sie benötigen einen Signalkatalog, bevor Sie eine Flotte erstellen können. Weitere Informationen finden Sie unter Erstellen Sie einen AWS FleetWise IoT-Signalkatalog.

Führen Sie den folgenden Befehl aus, um eine Flotte zu erstellen.

- Ersetze es *`fleet-id`* durch die ID der Flotte, die du gerade erstellst.

  Die Flotten-ID muss eindeutig sein und 1—100 Zeichen lang sein. Gültige Zeichen: Buchstaben (A-Z und a-z), Zahlen (0-9), Doppelpunkte (:), Bindestriche (-) und Unterstriche (_).
- (Optional) Durch eine Beschreibung ersetzen. *`description`*

  Die Beschreibung kann 1—2048 Zeichen lang sein.
- Durch *`signal-catalog-arn`* den ARN des Signalkatalogs ersetzen.

```
aws iotfleetwise create-fleet \
   --fleet-id fleet-id \
```

```
    --description description \
    --signal-catalog-arn signal-catalog-arn
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `CreateFleet` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Ordnen Sie ein AWS FleetWise IoT-Fahrzeug einer Flotte zu

Sie können den AssociateVehicleFleetAPI-Vorgang verwenden, um ein Fahrzeug einer Flotte zuzuordnen. Das folgende Beispiel verwendet AWS CLI.

> **Important**
>
> - Sie müssen über ein Fahrzeug und eine Flotte verfügen, bevor Sie ein Fahrzeug einer Flotte zuordnen können. Weitere Informationen finden Sie unter AWS FleetWise IoT-Fahrzeuge verwalten.
> - Wenn Sie ein Fahrzeug mit einer Flotte verknüpfen, auf die eine Kampagne abzielt, stellt AWS IoT die Kampagne FleetWise automatisch für das Fahrzeug bereit.

Führen Sie den folgenden Befehl aus, um ein Fahrzeug einer Flotte zuzuordnen.

- *fleet-id*Ersetzen Sie es durch die ID der Flotte.
- *vehicle-name*Durch die ID des Fahrzeugs ersetzen.

```
aws iotfleetwise associate-vehicle-fleet --fleet-id fleet-id --vehicle-name vehicle-name
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `AssociateVehicleFleet` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Trennen Sie ein AWS FleetWise IoT-Fahrzeug von einer Flotte

Sie können den DisassociateVehicleFleetAPI-Vorgang verwenden, um ein Fahrzeug von einer Flotte zu trennen. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um die Zuordnung eines Fahrzeugs zu einer Flotte aufzuheben.

- *fleet-id*Ersetzen Sie es durch die ID der Flotte.
- *vehicle-name*Durch die ID des Fahrzeugs ersetzen.

```
aws iotfleetwise disassociate-vehicle-fleet --fleet-id fleet-id --vehicle-name vehicle-name
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `DisassociateVehicleFleet` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Aktualisieren Sie eine AWS FleetWise IoT-Flotte

Sie können den UpdateFleetAPI-Vorgang verwenden, um die Beschreibung für eine Flotte zu aktualisieren. Im folgenden Beispiel wird AWS CLI verwendet.

Führen Sie den folgenden Befehl aus, um eine Flotte zu aktualisieren.

- *fleet-id*Ersetzen Sie es durch die ID der Flotte, die Sie aktualisieren möchten.
- *description*Durch eine neue Beschreibung ersetzen.

Die Beschreibung kann 1—2048 Zeichen lang sein.

```
aws iotfleetwise update-fleet --fleet-id fleet-id --description description
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `UpdateFleet` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Löschen Sie eine AWS FleetWise IoT-Flotte

Sie können den DeleteFleetAPI-Vorgang verwenden, um eine Flotte zu löschen. Das folgende Beispiel verwendet AWS CLI.

> **Important**
>
> Bevor Sie eine Flotte löschen, stellen Sie sicher, dass ihr keine Fahrzeuge zugeordnet sind. Anweisungen zum Trennen der Zuordnung eines Fahrzeugs zu einer Flotte finden Sie unterTrennen Sie ein AWS FleetWise IoT-Fahrzeug von einer Flotte.

Führen Sie den folgenden Befehl aus, um eine Flotte zu löschen.

*fleet-id*Ersetzen Sie es durch die ID der Flotte, die Sie löschen möchten.

```
aws iotfleetwise delete-fleet --fleet-id fleet-id
```

Überprüfen Sie das Löschen der Flotte

Sie können den ListFleetsAPI-Vorgang verwenden, um zu überprüfen, ob eine Flotte gelöscht wurde. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Flotten abzurufen.

```
aws iotfleetwise list-fleets
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den API-Vorgang aufrufen kann. `ListFleets`

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Holen Sie sich AWS FleetWise IoT-Flotteninformationen

Sie können den ListFleetsForVehicleAPI-Vorgang verwenden, um eine paginierte Liste IDs aller Flotten abzurufen, zu denen das Fahrzeug gehört. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste IDs aller Flotten abzurufen, zu denen das Fahrzeug gehört.

*vehicle-name*Ersetzen Sie es durch die ID des Fahrzeugs.

```
aws iotfleetwise list-fleets-for-vehicle \
            --vehicle-name vehicle-name
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ListFleetsForVehicle` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Sie können den ListVehiclesInFleetAPI-Vorgang verwenden, um eine paginierte Liste mit Zusammenfassungen aller Fahrzeuge in einer Flotte abzurufen. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen aller Fahrzeuge in einer Flotte abzurufen.

*fleet-id*Ersetzen Sie ihn durch die ID der Flotte.

```
aws iotfleetwise list-vehicles-in-fleet \
            --fleet-id fleet-id
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `ListVehiclesInFleet` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Sie können den GetFleetAPI-Vorgang verwenden, um Flotteninformationen abzurufen. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um die Metadaten einer Flotte abzurufen.

*fleet-id*Ersetzen Sie es durch die ID der Flotte.

```
aws iotfleetwise get-fleet \
            --fleet-id fleet-id
```

> **Note**
>
> Diese Operation ist letztlich konsistent. Mit anderen Worten, Änderungen an der Flotte werden möglicherweise nicht sofort übernommen.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `GetFleet` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Sammeln Sie AWS FleetWise IoT-Daten mit Kampagnen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Eine Kampagne ist eine Orchestrierung von Regeln für die Datenerfassung. Kampagnen geben der Edge Agent for AWS FleetWise IoT-Software Anweisungen zur Auswahl, Erfassung und Übertragung von Daten in die Cloud.

Sie erstellen Kampagnen in der Cloud. Nachdem Sie oder Ihr Team eine Kampagne genehmigt haben, FleetWise stellt AWS IoT die Kampagne als einsatzbereit fest und sie wird beim nächsten Fahrzeug-Check-in bereitgestellt. Sie können wählen, ob Sie eine Kampagne für ein Fahrzeug oder eine Fahrzeugflotte einsetzen möchten. Die Edge Agent-Software beginnt erst mit der Erfassung von Daten, wenn eine laufende Kampagne für das Fahrzeug bereitgestellt wird.

> **Important**
>
> Kampagnen funktionieren erst, wenn Sie über Folgendes verfügen.
>
> - Die Edge Agent-Software wird in Ihrem Fahrzeug ausgeführt. Gehen Sie wie folgt vor, um weitere Informationen zur Entwicklung, Installation und Arbeit mit der Edge Agent-Software zu erhalten.
>
>   1. Öffnen Sie die AWS FleetWise IoT-Konsole.
>   2. Wählen Sie auf der Service-Startseite im FleetWise Abschnitt Erste Schritte mit AWS IoT die Option Explore Edge Agent aus.
>
> - Sie haben die Einrichtung AWS IoT Core für die Bereitstellung Ihres Fahrzeugs eingerichtet. Weitere Informationen finden Sie unter Bereitstellung von AWS FleetWise IoT-Fahrzeugen.

> **Note**
>
> Sie können außerdem Überwachen Sie den letzten bekannten Zustand Ihrer Fahrzeuge (nicht Flotten) nahezu in Echtzeit Statusvorlagen verwenden, mit denen Sie Telemetriedaten streamen können, entweder mit einer Aktualisierungsstrategie „bei Änderung“ oder „in regelmäßigen Abständen“. Diese Funktion bietet auch „On Demand“ -Funktionen, mit denen Sie zuvor bereitgestellte Vorlagen aktivieren oder deaktivieren oder den aktuellen Fahrzeugstatus einmalig anfordern können (Abrufen).
> Der Zugriff auf den letzten bekannten Status ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Jede Kampagne enthält die folgenden Informationen.

`signalCatalogArn`

Der Amazon-Ressourcenname (ARN) des Signalkatalogs, der mit der Kampagne verknüpft ist.

(Optional) `tags`

Tags sind Metadaten, die zur Verwaltung der Kampagne verwendet werden können. Sie können Ressourcen aus verschiedenen Services dasselbe Tag zuweisen, um anzugeben, dass die Ressourcen miteinander verbunden sind.

`TargetArn`

Der ARN eines Fahrzeugs oder einer Flotte, in der die Kampagne bereitgestellt wird.

`name`

Ein eindeutiger Name, anhand dessen die Kampagne identifiziert werden kann.

`collectionScheme`

Die Datenerfassungsschemata geben der Edge Agent-Software Anweisungen darüber, welche Daten gesammelt werden sollen oder wann sie gesammelt werden sollen. AWS IoT unterstützt FleetWise derzeit das zustandsbasierte Sammelschema und das zeitbasierte Sammelschema.

- `conditionBasedCollectionScheme`— Das auf Bedingungen basierende Erfassungssystem verwendet einen logischen Ausdruck, um zu erkennen, welche Daten gesammelt werden sollen. Die Edge Agent-Software sammelt Daten, wenn die Bedingung erfüllt ist.

- `expression`— der logische Ausdruck, der verwendet wird, um zu erkennen, welche Daten gesammelt werden sollen. Wenn der `$variable.`myVehicle.InVehicleTemperature` > 50.0` Ausdruck beispielsweise angegeben ist, erfasst die Edge Agent-Software Temperaturwerte, die über 50,0 liegen. Anweisungen zum Schreiben von Ausdrücken finden Sie unterLogische Ausdrücke für AWS FleetWise IoT-Kampagnen.
- (Optional) `conditionLanguageVersion` — die Version der Sprache für bedingte Ausdrücke.
- (Optional) `minimumTriggerIntervalMs` — die Mindestdauer zwischen zwei Datenerfassungsereignissen in Millisekunden. Wenn sich ein Signal häufig ändert, erfassen Sie Daten möglicherweise langsamer.
- (Optional) `triggerMode` — kann einer der folgenden Werte sein:
  - `RISING_EDGE`— Die Edge Agent-Software erfasst Daten nur, wenn die Bedingung zum ersten Mal erfüllt ist. Beispiel, `$variable.`myVehicle.AirBagDeployed` == true`.
  - `ALWAYS`— Die Edge Agent-Software sammelt Daten, wenn die Bedingung erfüllt ist.

- `timeBasedCollectionScheme`— Wenn Sie ein zeitbasiertes Erfassungsschema definieren, geben Sie einen Zeitraum in Millisekunden an. Die Edge Agent-Software entscheidet anhand des Zeitraums, wie oft Daten erfasst werden sollen. Wenn der Zeitraum beispielsweise 120.000 Millisekunden beträgt, erfasst die Edge Agent-Software alle zwei Minuten Daten.
  - `periodMs`— der Zeitraum (in Millisekunden), in dem entschieden wird, wie oft Daten gesammelt werden sollen.

(Optional) `compression`

Um drahtlose Bandbreite zu sparen und den Netzwerkverkehr zu reduzieren, können Sie SNAPPY angeben, um Daten in Fahrzeugen zu komprimieren.

Standardmäßig (`OFF`) komprimiert die Edge Agent-Software keine Daten.

`dataDestinationConfigs`

Wählen Sie das einzige Ziel aus, an das die Kampagne Fahrzeugdaten übertragen soll. Sie können die Daten an ein MQTT-Thema senden oder sie in Amazon S3 oder Amazon Timestream speichern.

MQTT (Message Queuing Telemetry Transport) ist ein leichtes und weit verbreitetes Messaging-Protokoll. Sie können Daten zu einem MQTT-Thema veröffentlichen, um mithilfe von Regeln Ihre eigenen ereignisgesteuerten Architekturen aufzubauen. AWS IoT AWS IoT Die Unterstützung für

MQTT basiert auf der MQTT-Spezifikation v3.1.1 und der MQTT-Spezifikation v5.0, mit einigen Unterschieden. Weitere Informationen finden Sie unter MQTT-Unterschiede.

S3 kann ein kostengünstiger Datenspeichermechanismus sein, der dauerhafte Datenverwaltungsfunktionen und nachgelagerte Datendienste bietet. Sie können S3 für Daten zum Fahrverhalten oder zur Analyse langfristiger Wartungsarbeiten verwenden.

Timestream ist ein Mechanismus zur Datenpersistenz, mit dem Sie Trends und Muster nahezu in Echtzeit erkennen können. Sie können Timestream für Zeitreihendaten verwenden, z. B. um historische Trends bei der Fahrzeuggeschwindigkeit oder beim Bremsen zu analysieren.

> **Note**
>
> Amazon Timestream ist in der Region Asien-Pazifik (Mumbai) nicht verfügbar.

(Optional) `dataExtraDimensions`

Sie können ein oder mehrere Attribute hinzufügen, um zusätzliche Informationen für ein Signal bereitzustellen.

(Optional) `dataPartitions`

Erstellen Sie eine Datenpartition, um Signaldaten vorübergehend auf einem Fahrzeug zu speichern. Sie konfigurieren, wann und wie die Daten an die Cloud weitergeleitet werden sollen.

- Geben Sie an, wie AWS IoT die Daten in einem Fahrzeug oder einer Flotte FleetWise speichert, indem Sie die maximale Speichergröße, die Mindestlebensdauer und den Speicherort definieren.
- Die Kampagne `spoolingMode` muss sein`TO_DISK`.
- Das Hochladen von Konfigurationen umfasst die Definition der Version der Bedingungssprache und des logischen Ausdrucks.

(Optional) `description`

Fügen Sie eine Beschreibung hinzu, um den Zweck der Kampagne besser zu identifizieren.

(Optional) `diagnosticsMode`

Wenn der Diagnosemodus so konfiguriert ist`SEND_ACTIVE_DTCS`, sendet die Kampagne gespeicherte Standarddiagnose-Fehlercodes (DTCs), anhand derer festgestellt werden kann,

was mit Ihrem Fahrzeug nicht stimmt. Beispielsweise bedeutet P0097, dass das Motorsteuergerät (ECM) festgestellt hat, dass der Eingang des Ansauglufttemperatursensors 2 (IAT2) unter dem normalen Sensorbereich liegt.

Standardmäßig (`OFF`) sendet die Edge Agent-Software keine Diagnosecodes.

(Optional) `expiryTime`

Definieren Sie das Ablaufdatum für Ihre Kampagne. Wenn die Kampagne abläuft, beendet die Edge Agent-Software die Erfassung von Daten, wie in dieser Kampagne angegeben. Wenn mehrere Kampagnen für das Fahrzeug bereitgestellt werden, verwendet die Edge Agent-Software andere Kampagnen, um Daten zu sammeln.

Standardwert: `253402243200` (31. Dezember 9999, 00:00:00 UTC)

(Optional) `postTriggerCollectionDuration`

Sie können eine Dauer für die Erfassung nach dem Auslösen definieren, sodass die Edge Agent-Software nach dem Aufrufen eines Schemas für einen bestimmten Zeitraum weiterhin Daten sammelt. Wenn beispielsweise ein zustandsbasiertes Erfassungsschema mit dem folgenden Ausdruck aufgerufen wird:`$variable.`myVehicle.Engine.RPM` > 7000.0`, erfasst die Edge Agent-Software weiterhin Werte für Umdrehungen pro Minute (U/min) für den Motor. Selbst wenn die Drehzahl nur einmal höher als 7000 ist, kann dies auf ein mechanisches Problem hinweisen. In diesem Fall möchten Sie möglicherweise, dass die Edge Agent-Software weiterhin Daten sammelt, um den Zustand zu überwachen.

Standardwert: `0`

(Optional) `priority`

Geben Sie eine Ganzzahl an, um die Prioritätsstufe der Kampagne anzugeben. Kampagnen mit einer kleineren Zahl haben eine höhere Priorität. Wenn Sie mehrere Kampagnen für ein Fahrzeug bereitstellen, werden die Kampagnen mit höherer Priorität zuerst initiiert.

Standardwert: `0`

(Optional) `signalsToCollect`

Eine Liste von Signalen, aus denen Daten gesammelt werden, wenn das Datenerfassungsschema aufgerufen wird.

- `name`— der Name des Signals, aus dem Daten gesammelt werden, wenn das Datenerfassungsschema aufgerufen wird.

- `dataPartitionId`— die ID der Datenpartition, die im Signal verwendet werden soll. Die ID muss mit einer der in IDs angegebenen übereinstimmen`dataPartitions`. Wenn Sie ein Signal als Bedingung in Ihre Datenpartition hochladen, müssen dieselben Signale darin enthalten sein`signalsToCollect`.
- (Optional) `maxSampleCount` — Die maximale Anzahl von Datenproben, die die Edge Agent-Software sammelt und in die Cloud überträgt, wenn das Datenerfassungsschema aufgerufen wird.
- (Optional) `minimumSamplingIntervalMs` — die Mindestdauer zwischen zwei Datenprobenerfassungsereignissen in Millisekunden. Wenn sich ein Signal häufig ändert, können Sie diesen Parameter verwenden, um Daten langsamer zu sammeln.

  Gültiger Bereich: 0-4294967295

(Optional) `spoolingMode`

Wenn dies konfiguriert `spoolingMode` ist`TO_DISK`, speichert die Edge Agent-Software vorübergehend Daten lokal, wenn ein Fahrzeug nicht mit der Cloud verbunden ist. Nachdem die Verbindung wiederhergestellt wurde, werden die lokal gespeicherten Daten automatisch in die Cloud übertragen.

Standardwert: `OFF`

(Optional) `startTime`

Eine genehmigte Kampagne wird zum Startzeitpunkt aktiviert.

Standardwert: `0`

Der Status einer Kampagne kann einer der folgenden Werte sein.

- `CREATING`— AWS IoT FleetWise bearbeitet Ihre Anfrage zur Erstellung der Kampagne.
- `WAITING_FOR_APPROVAL`— Nachdem eine Kampagne erstellt wurde, geht sie in den `WAITING_FOR_APPROVAL` Status über. Verwenden Sie den `UpdateCampaign` API-Vorgang, um die Kampagne zu genehmigen. Nachdem die Kampagne genehmigt wurde, stellt AWS IoT die Kampagne FleetWise automatisch für das Zielfahrzeug oder die Zielflotte bereit. Weitere Informationen finden Sie unter Aktualisieren Sie eine AWS FleetWise IoT-Kampagne.
- `RUNNING` — Die Kampagne ist aktiv.
- `SUSPENDED`— Die Kampagne ist ausgesetzt. Verwenden Sie den `UpdateCampaign` API-Vorgang, um die Kampagne fortzusetzen.

AWS IoT FleetWise bietet die folgenden API-Operationen, mit denen Sie Kampagnen erstellen und verwalten können.

- CreateCampaign— Erstellt eine neue Kampagne.
- UpdateCampaign— Aktualisiert eine bestehende Kampagne. Nachdem eine Kampagne erstellt wurde, müssen Sie diese API-Operation verwenden, um die Kampagne zu genehmigen.
- DeleteCampaign— Löscht eine bestehende Kampagne.
- ListCampaigns— Ruft eine paginierte Liste mit Zusammenfassungen für alle Kampagnen ab.
- GetCampaign— Ruft Informationen über eine Kampagne ab.

Lernprogramme

- Erstellen Sie eine AWS FleetWise IoT-Kampagne
- Aktualisieren Sie eine AWS FleetWise IoT-Kampagne
- Eine AWS FleetWise IoT-Kampagne löschen
- Informationen zu AWS FleetWise IoT-Kampagnen abrufen
- Kampagnendaten speichern und weiterleiten
- Sammeln Sie diagnostische Fehlercode-Daten mithilfe von AWS IoT FleetWise
- Visualisieren Sie AWS FleetWise IoT-Fahrzeugdaten

Erstellen Sie eine AWS FleetWise IoT-Kampagne

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um Kampagnen zur Erfassung von Fahrzeugdaten zu erstellen.

> **Important**
>
> Damit Ihre Kampagne funktioniert, müssen Sie über Folgendes verfügen:

- Die Edge Agent-Software wird in Ihrem Fahrzeug ausgeführt. Gehen Sie wie folgt vor, um weitere Informationen zur Entwicklung, Installation und Arbeit mit der Edge Agent-Software zu erhalten:

  1. Öffnen Sie die AWS FleetWise IoT-Konsole.
  2. Wählen Sie auf der Service-Startseite im FleetWise Abschnitt Erste Schritte mit AWS IoT die Option Explore Edge Agent aus.
- Sie haben die Einrichtung AWS IoT Core für die Bereitstellung Ihres Fahrzeugs eingerichtet. Weitere Informationen finden Sie unter Bereitstellung von AWS FleetWise IoT-Fahrzeugen.

Themen

- Erstellen Sie eine Kampagne (Konsole)
- Erstelle eine Kampagne (AWS CLI)
- Logische Ausdrücke für AWS FleetWise IoT-Kampagnen

Erstellen Sie eine Kampagne (Konsole)

Verwenden Sie die AWS FleetWise IoT-Konsole, um eine Kampagne zur Auswahl, Erfassung und Übertragung von Fahrzeugdaten in die Cloud zu erstellen.

So erstellen Sie eine Kampagne

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Kampagnen aus.
3. Wählen Sie auf der Seite Kampagnen die Option Kampagne erstellen aus, und führen Sie dann die Schritte in den folgenden Themen aus.

Themen

- Schritt 1: Kampagne konfigurieren
- Schritt 2: Geben Sie die Speicher- und Upload-Bedingungen an
- Schritt 3: Datenziel konfigurieren
- Schritt 4: Fahrzeuge hinzufügen

- Schritt 5: Überprüfen und Erstellen
- Schritt 6: Implementieren Sie eine Kampagne

> **Important**
>
> - Sie benötigen einen Signalkatalog und ein Fahrzeug, bevor Sie eine Kampagne erstellen können. Weitere Informationen erhalten Sie unter AWS FleetWise IoT-Signalkataloge verwalten und AWS FleetWise IoT-Fahrzeuge verwalten.
> - Nachdem eine Kampagne erstellt wurde, müssen Sie die Kampagne genehmigen. Weitere Informationen finden Sie unter Aktualisieren Sie eine AWS FleetWise IoT-Kampagne.

Schritt 1: Kampagne konfigurieren

Gehen Sie unter Allgemeine Informationen wie folgt vor:

1. Geben Sie einen Namen für die Kampagne ein.
2. (Optional) Geben Sie eine Beschreibung ein.

Konfigurieren Sie das Datenerfassungsschema der Kampagne. Ein Datenerfassungsschema gibt der Edge Agent-Software Anweisungen darüber, welche Daten gesammelt werden sollen oder wann sie gesammelt werden sollen. In der AWS FleetWise IoT-Konsole können Sie ein Datenerfassungsschema auf folgende Weise konfigurieren:

- Definieren Sie das Datenerfassungsschema manuell.
- Laden Sie eine Datei hoch, um das Datenerfassungsschema automatisch zu definieren.

Wählen Sie unter Konfigurationsoption eine der folgenden Optionen aus:

- Um den Typ des Datenerfassungsschemas manuell anzugeben und Optionen zur Anpassung des Schemas zu definieren, wählen Sie Datenerfassungsschema definieren aus.

  Geben Sie manuell den Typ des Datenerfassungsschemas an und definieren Sie Optionen zur Anpassung des Schemas.

1. Wählen Sie im Abschnitt Details zum Datenerfassungsschema die Art des Datenerfassungsschemas aus, das für diese Kampagne verwendet werden soll. Wenn Sie anhand eines logischen Ausdrucks erkennen möchten, welche Fahrzeugdaten erfasst werden sollen, wählen Sie „Zustandsbasiert“. Wenn Sie anhand eines bestimmten Zeitraums entscheiden möchten, wie oft Fahrzeugdaten erfasst werden sollen, wählen Sie Zeitbasiert.
2. Definieren Sie die Dauer, für die die Kampagne Daten sammelt.

   > **Note**
   >
   > Standardmäßig wird eine genehmigte Kampagne sofort aktiviert und hat keine festgelegte Endzeit. Um zusätzliche Gebühren zu vermeiden, müssen Sie einen Zeitraum angeben.

3. Wenn Sie ein auf Bedingungen basierendes Datenerfassungsschema angegeben haben, müssen Sie einen logischen Ausdruck definieren, um zu erkennen, welche Daten gesammelt werden sollen. AWS IoT FleetWise verwendet einen logischen Ausdruck, um zu erkennen, welche Daten für ein zustandsorientiertes Schema gesammelt werden müssen. Der Ausdruck muss den vollständig qualifizierten Namen eines Signals als Variable, einen Vergleichsoperator und einen Vergleichswert angeben.

   Wenn Sie beispielsweise den `$variable.`myVehicle.InVehicleTemperature` > 50.0` Ausdruck angeben, FleetWise erfasst AWS IoT Temperaturwerte, die größer als 50,0 sind. Anweisungen zum Schreiben von Ausdrücken finden Sie unterLogische Ausdrücke für AWS FleetWise IoT-Kampagnen.

   Geben Sie den logischen Ausdruck ein, anhand dessen erkannt wird, welche Daten erfasst werden sollen.
4. (Optional) Geben Sie die Sprachversion des bedingten Ausdrucks an. Der Standardwert ist 1.
5. (Optional) Geben Sie das minimale Triggerintervall an. Dabei handelt es sich um die kleinste Zeitspanne zwischen zwei Datenerfassungsereignissen. Wenn sich ein Signal beispielsweise häufig ändert, möchten Sie möglicherweise Daten langsamer erfassen.
6. Geben Sie die Bedingung für den Triggermodus an, unter der die Edge Agent-Software Daten sammelt. Standardmäßig sammelt die Edge Agent for AWS FleetWise IoT-Software immer dann Daten, wenn die Bedingung erfüllt ist. Oder er kann nur Daten sammeln, wenn die Bedingung zum ersten Mal erfüllt ist (beim ersten Trigger).

7. Wenn Sie ein zeitbasiertes Datenerfassungsschema angegeben haben, müssen Sie einen Zeitraum in Millisekunden zwischen 10.000 und 60.000 Millisekunden angeben. Die Edge Agent-Software entscheidet anhand des Zeitraums, wie oft Daten erfasst werden sollen.
8. (Optional) Bearbeiten Sie die erweiterten Schemaoptionen des Schemas.

   a. Um drahtlose Bandbreite zu sparen und den Netzwerkverkehr durch Komprimieren von Daten zu reduzieren, wählen Sie Snappy.

   b. (Optional) Um zu definieren, wie lange (in Millisekunden) die Datenerfassung nach einem Datenerfassungsereignis fortgesetzt werden soll, können Sie die Dauer der Erfassung nach dem Trigger angeben.

   c. (Optional) Um die Prioritätsstufe der Kampagne anzugeben, geben Sie die Kampagnenpriorität an. Kampagnen mit einer kleineren Prioritätsnummer werden zuerst bereitgestellt und haben dann eine höhere Priorität.

   d. Die Edge Agent-Software kann Daten vorübergehend lokal speichern, wenn ein Fahrzeug nicht mit der Cloud verbunden ist. Nach der Wiederherstellung der Verbindung werden die lokal gespeicherten Daten automatisch in die Cloud übertragen. Geben Sie an, ob der Edge-Agent Daten während einer unterbrochenen Verbindung lokal speichern soll.

   e. (Optional) Um zusätzliche Informationen für ein Signal bereitzustellen, fügen Sie bis zu fünf Attribute als Zusätzliche Datendimensionen hinzu.

- Um eine Datei zur Definition des Datenerfassungsschemas hochzuladen, wählen Sie „.json-Datei von Ihrem lokalen Gerät hochladen“. AWS IoT definiert FleetWise automatisch, welche Optionen Sie in der Datei definieren können. Sie können die ausgewählten Optionen überprüfen und aktualisieren.

  Laden Sie eine JSON-Datei mit Details zum Datenerfassungsschema hoch.

  1. Um Informationen über das Datenerfassungsschema zu importieren, wählen Sie Dateien auswählen. Weitere Informationen zum erforderlichen Dateiformat finden Sie in der CreateCampaignAPI-Dokumentation.

     > **Note**
     >
     > AWS IoT unterstützt FleetWise derzeit die Dateiformaterweiterung „.json“.

  2. AWS IoT definiert das Datenerfassungsschema FleetWise automatisch auf der Grundlage der Informationen in Ihrer Datei. Sehen Sie sich die Optionen an, die AWS IoT für Sie FleetWise ausgewählt hat. Sie können die Optionen bei Bedarf aktualisieren.

Schritt 2: Geben Sie die Speicher- und Upload-Bedingungen an

Um auszuwählen, ob die Edge Agent-Software Daten vorübergehend lokal speichert, wenn ein Fahrzeug nicht mit der Cloud verbunden ist, geben Sie den Spooling-Modus an.

- Wählen Sie im Datenspooling-Modus eine der folgenden Optionen:
  - Nicht gespeichert — Die Edge Agent-Software sammelt Daten, speichert sie aber nicht vorübergehend lokal, wenn ein Fahrzeug offline ist. Die Edge Agent-Software überträgt Daten in die Cloud, wenn das Fahrzeug wieder eine Verbindung herstellt.
  - Auf Festplatte gespeichert — Die Edge Agent-Software sammelt Daten und speichert sie vorübergehend lokal, wenn ein Fahrzeug offline ist. Die gesammelten Daten werden vorübergehend an einem Ort gespeichert, der im Abschnitt „Persistenz“ der Edge-Agent-Konfigurationsdatei definiert ist. Der Edge-Agent überträgt Daten in die Cloud, wenn das Fahrzeug wieder eine Verbindung herstellt.
  - Mit Partitionen auf Festplatte gespeichert — Das Fahrzeug speichert Daten immer vorübergehend auf dem Edge in der von Ihnen angegebenen Datenpartition. Sie können wählen, wann Sie Ihre gespeicherten Daten an die Cloud weiterleiten möchten.
    1. (Optional) Geben Sie eine Partitions-ID ein, um einen bestimmten Datensatz zu kennzeichnen.
    2. Geben Sie einen Ordnernamen als Speicherort für die Daten ein. Der absolute Pfad des Speicherorts lautet`{persistency_path} / {vehicle_name} / {campaign_name} / {storage_location}`.
    3. Geben Sie die maximale Speichergröße der in der Partition gespeicherten Daten ein. Neuere Daten überschreiben ältere Daten, wenn die Partition die maximale Größe erreicht.
    4. Geben Sie die Mindestdauer ein, für die Daten in dieser Partition auf der Festplatte gespeichert werden.
    5. (Optional) Geben Sie die Upload-Bedingungen für die Partition ein.

Geben Sie Signale an

Sie können die Signale angeben, aus denen während der Kampagne Daten gesammelt werden sollen.

Um die Signale zu spezifizieren, von denen Daten gesammelt werden sollen

1. Wählen Sie den Signalnamen aus.

2. (Optional) Geben Sie für Max. Stichprobenanzahl die maximale Anzahl von Datenproben ein, die die Edge Agent-Software während der Kampagne sammelt und in die Cloud überträgt.
3. (Optional) Geben Sie für Min. Stichprobenintervall die Mindestdauer zwischen zwei Datenprobenerfassungsereignissen in Millisekunden ein. Wenn sich ein Signal häufig ändert, können Sie diesen Parameter verwenden, um Daten langsamer zu sammeln.
4. Um ein weiteres Signal hinzuzufügen, wählen Sie Weitere Signale hinzufügen. Sie können bis zu 999 Signale hinzufügen.
5. Wählen Sie Weiter aus.

Schritt 3: Datenziel konfigurieren

> **Note**
>
> Wenn die Kampagne Datensignale des Bildverarbeitungssystems enthält, können Sie die Fahrzeugdaten nur in Amazon S3 speichern. Sie können es nicht in Timestream speichern oder an ein MQTT-Thema senden.
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.
> Amazon Timestream ist in der Region Asien-Pazifik (Mumbai) nicht verfügbar.

Wählen Sie das Ziel aus, an das Sie die im Rahmen der Kampagne gesammelten Daten senden oder speichern möchten. Sie können Fahrzeugdaten an ein MQTT-Thema senden oder sie in Amazon S3 oder Amazon Timestream speichern.

Gehen Sie in den Zieleinstellungen wie folgt vor:

- Wählen Sie Amazon S3, Amazon Timestream oder MQTT-Thema aus der Drop-down-Liste aus.

Amazon S3

> **Important**
>
> Sie können Daten nur an S3 übertragen, wenn AWS IoT FleetWise über Schreibberechtigungen in den S3-Bucket verfügt. Weitere Informationen zur Zugriffsgewährung finden Sie unter Zugriffskontrolle mit AWS IoT FleetWise.

Um Fahrzeugdaten in einem S3-Bucket zu speichern, wählen Sie Amazon S3. S3 ist ein Objektspeicherdienst, der Daten als Objekte in Buckets speichert. Weitere Informationen finden Sie unter Erstellen, Konfigurieren und Arbeiten mit Amazon S3 S3-Buckets im Amazon Simple Storage Service-Benutzerhandbuch.

S3 optimiert die Kosten für die Datenspeicherung und bietet zusätzliche Mechanismen zur Nutzung von Fahrzeugdaten, z. B. Data Lakes, zentrale Datenspeicherung, Datenverarbeitungspipelines und Analysen. Sie können S3 verwenden, um Daten für die Stapelverarbeitung und Analyse zu speichern. Sie können beispielsweise Berichte über Ereignisse mit starker Bremsung für Ihr Modell für maschinelles Lernen (ML) erstellen. Eingehende Fahrzeugdaten werden vor der Auslieferung für 10 Minuten zwischengespeichert.

Gehen Sie in den S3-Zieleinstellungen wie folgt vor:

1. Wählen Sie für den S3-Bucket einen Bucket aus, der AWS IoT FleetWise über Berechtigungen für verfügt.
2. (Optional) Geben Sie ein benutzerdefiniertes Präfix ein, mit dem Sie die im S3-Bucket gespeicherten Daten organisieren können.
3. Wählen Sie das Ausgabeformat aus. Dabei handelt es sich um die Formatdateien, die wie im S3-Bucket gespeichert werden.
4. Wählen Sie aus, ob Sie die im S3-Bucket gespeicherten Daten als .gzip-Datei komprimieren möchten. Wir empfehlen die Komprimierung von Daten, da dadurch die Speicherkosten minimiert werden.
5. Die Optionen, die Sie in den S3-Zieleinstellungen auswählen, ändern den Beispiel-S3-Objekt-URI. Dies ist ein Beispiel dafür, wie Dateien in S3 gespeichert werden.

Amazon Timestream

> **Important**
>
> Sie können Daten nur in eine Tabelle übertragen, wenn AWS IoT berechtigt FleetWise ist, Daten in Timestream zu schreiben. Weitere Informationen zur Zugriffsgewährung finden Sie unter Zugriffskontrolle mit AWS IoT FleetWise.
> Amazon Timestream ist in der Region Asien-Pazifik (Mumbai) nicht verfügbar.

Um Fahrzeugdaten in einer Timestream-Tabelle zu speichern, wählen Sie Amazon Timestream. Sie können Timestream verwenden, um Fahrzeugdaten abzufragen, um Trends und Muster zu erkennen. Sie können Timestream beispielsweise verwenden, um einen Alarm für den Kraftstoffstand des Fahrzeugs zu erstellen. Eingehende Fahrzeugdaten werden nahezu in Echtzeit an Timestream übertragen. Weitere Informationen finden Sie unter Was ist Amazon Timestream? im Amazon Timestream Developer Guide.

Gehen Sie in den Timestream-Tabelleneinstellungen wie folgt vor:

1. Wählen Sie als Timestream-Datenbankname den Namen Ihrer Timestream-Datenbank aus der Dropdownliste aus.
2. Wählen Sie als Timestream-Tabellenname den Namen Ihrer Timestream-Tabelle aus der Drop-down-Liste aus.

Gehen Sie unter Servicezugriff für Timestream wie folgt vor:

- Wählen Sie in der Dropdown-Liste eine IAM-Rolle aus.

MQTT-Thema

> **Important**
>
> Sie können Daten nur an ein MQTT-Thema weiterleiten, wenn AWS IoT FleetWise über Berechtigungen für AWS IoT Themen verfügt. Weitere Informationen zur Zugriffsgewährung finden Sie unter Zugriffskontrolle mit AWS IoT FleetWise.

Um Fahrzeugdaten an ein MQTT-Thema zu senden, wählen Sie MQTT-Thema.

Fahrzeugdaten, die über MQTT-Messaging gesendet werden, werden nahezu in Echtzeit übermittelt und ermöglichen es Ihnen, Regeln zu verwenden, um Maßnahmen zu ergreifen oder Daten an andere Ziele weiterzuleiten. Weitere Informationen zur Verwendung von MQTT finden Sie unter Gerätekommunikationsprotokolle und Regeln für AWS IoT im AWS IoT Core Entwicklerhandbuch.

1. Geben Sie unter MQTT-Thema den Namen des Themas ein.
2. Wählen Sie unter dem Thema Dienstzugriff für MQTT aus, ob Sie AWS IoT eine neue Servicerolle für Sie FleetWise erstellen und verwenden lassen möchten. Wenn Sie eine

bestehende Servicerolle verwenden möchten, wählen Sie die Rolle in der Dropdownliste unter Rolle auswählen aus.

- Wählen Sie Weiter aus.

Schritt 4: Fahrzeuge hinzufügen

Um auszuwählen, für welche Fahrzeuge deine Kampagne eingesetzt werden soll, wähle sie in der Fahrzeugliste aus. Filtern Sie Fahrzeuge, indem Sie nach den Attributen und ihren Werten suchen, die Sie bei der Erstellung der Fahrzeuge hinzugefügt haben, oder nach Fahrzeugnamen.

Gehen Sie unter Fahrzeuge filtern wie folgt vor:

1. Suchen Sie im Suchfeld das Attribut oder den Fahrzeugnamen und wählen Sie es aus der Liste aus.

   > **Note**
   >
   > Jedes Attribut kann nur einmal verwendet werden.

2. Geben Sie den Wert des Attributs oder den Fahrzeugnamen ein, für das Sie die Kampagne bereitstellen möchten. Wenn der vollqualifizierte Name des Attributs beispielsweise lautet`fuelType`, geben Sie `gasoline` als Wert ein.
3. Um nach einem anderen Fahrzeugattribut zu suchen, wiederholen Sie die vorherigen Schritte. Sie können nach bis zu fünf Fahrzeugattributen und einer unbegrenzten Anzahl von Fahrzeugnamen suchen.
4. Fahrzeuge, die Ihrer Suche entsprechen, werden unter Fahrzeugname aufgeführt. Wählen Sie die Fahrzeuge aus, für die Sie die Kampagne einsetzen möchten.

   > **Note**
   >
   > In den Suchergebnissen werden bis zu 100 Fahrzeuge angezeigt. Wählen Sie Alle auswählen, um alle Fahrzeuge zur Kampagne hinzuzufügen.

5. Wählen Sie Weiter aus.

Schritt 5: Überprüfen und Erstellen

Überprüfen Sie die Konfigurationen für die Kampagne und wählen Sie dann Kampagne erstellen aus.

> **Note**
>
> Nachdem eine Kampagne erstellt wurde, müssen Sie oder Ihr Team die Kampagne für Fahrzeuge bereitstellen.

Schritt 6: Implementieren Sie eine Kampagne

Nachdem Sie eine Kampagne erstellt haben, müssen Sie oder Ihr Team die Kampagne in Fahrzeugen einsetzen.

Um eine Kampagne zu implementieren

1. Wählen Sie auf der Seite mit der Kampagnenübersicht die Option Bereitstellen aus.
2. Überprüfe und bestätige, dass du mit der Bereitstellung beginnen und mit der Erfassung von Daten von Fahrzeugen beginnen möchtest, die mit der Kampagne verbunden sind.
3. Wählen Sie Bereitstellen.

Wenn Sie die Erfassung von Daten von Fahrzeugen, die mit der Kampagne verbunden sind, unterbrechen möchten, wählen Sie auf der Seite mit der Kampagnenübersicht die Option Sperren aus. Wenn du die Erfassung von Daten von Fahrzeugen fortsetzen möchtest, die mit der Kampagne verbunden sind, wähle „Fortfahren“.

Erstelle eine Kampagne (AWS CLI)

Sie können den CreateCampaignAPI-Vorgang verwenden, um eine Kampagne zu erstellen. Das folgende Beispiel verwendet die AWS CLI.

Wenn Sie eine Kampagne erstellen, können von Fahrzeugen gesammelte Daten an ein MQTT-Thema gesendet oder entweder in Amazon S3 (S3) oder Amazon Timestream gespeichert werden. Wählen Sie Timestream für eine schnelle, skalierbare und serverlose Zeitreihendatenbank, z. B. zum Speichern von Daten, die nahezu in Echtzeit verarbeitet werden müssen. Wählen Sie S3 als Objektspeicher mit branchenführender Skalierbarkeit, Datenverfügbarkeit, Sicherheit und Leistung.

Wählen Sie MQTT, um Daten nahezu in Echtzeit bereitzustellen und Regeln zu verwenden AWS IoT, um von Ihnen definierte Aktionen auszuführen oder die Daten an andere Ziele weiterzuleiten.

> **Important**
>
> Sie können Fahrzeugdaten nur an ein MQTT-Thema, Amazon S3 oder Amazon Timestream übertragen, wenn AWS IoT berechtigt FleetWise ist, MQTT-Nachrichten in Ihrem Namen zu senden oder Daten in S3 oder Timestream zu schreiben. Weitere Informationen zur Zugriffsgewährung finden Sie unter Zugriffskontrolle mit AWS IoT FleetWise.
> Amazon Timestream ist in der Region Asien-Pazifik (Mumbai) nicht verfügbar.

Kampagne erstellen

> **Important**
>
> - Sie benötigen einen Signalkatalog und ein Fahrzeug oder eine Flotte, bevor Sie eine Kampagne erstellen können. Weitere Informationen finden Sie unter AWS FleetWise IoT-Signalkataloge verwalten, AWS FleetWise IoT-Fahrzeuge verwalten und Verwaltung von Flotten im Internet der Dinge AWS FleetWise.
> - Nachdem eine Kampagne erstellt wurde, müssen Sie den `UpdateCampaign` API-Vorgang verwenden, um die Kampagne zu genehmigen. Weitere Informationen finden Sie unter Aktualisieren Sie eine AWS FleetWise IoT-Kampagne.

Führen Sie den folgenden Befehl aus, um eine Kampagne zu erstellen.

*`file-name`*Ersetzen Sie ihn durch den Namen der JSON-Datei, die die Kampagnenkonfiguration enthält.

```
aws iotfleetwise create-campaign --cli-input-json file://file-name.json
```

- Ersetze es *`campaign-name`* durch den Namen der Kampagne, die du gerade erstellst.
- *`signal-catalog-arn`*Ersetzen Sie durch den Amazon-Ressourcennamen (ARN) des Signalkatalogs.
- *`target-arn`*Ersetzen Sie es durch den ARN einer Flotte oder eines Fahrzeugs, das Sie erstellt haben.

- *bucket-arn*Ersetzen Sie es durch den ARN des S3-Buckets.

```
{
    "name": "campaign-name",
    "targetArn": "target-arn",
    "signalCatalogArn": "signal-catalog-arn",
    "collectionScheme": {
        "conditionBasedCollectionScheme": {
            "conditionLanguageVersion": 1,
            "expression": "$variable.`Vehicle.DemoBrakePedalPressure` > 7000",
            "minimumTriggerIntervalMs": 1000,
            "triggerMode": "ALWAYS"
        }
    },
    "compression": "SNAPPY",
    "diagnosticsMode": "OFF",
    "postTriggerCollectionDuration": 1000,
    "priority": 0,
    "signalsToCollect": [
        {
         "maxSampleCount": 100,
         "minimumSamplingIntervalMs": 0,
         "name": "Vehicle.DemoEngineTorque"
        },
        {
         "maxSampleCount": 100,
         "minimumSamplingIntervalMs": 0,
         "name": "Vehicle.DemoBrakePedalPressure"
        }
    ],
    "spoolingMode": "TO_DISK",
    "dataDestinationConfigs": [
        {
         "s3Config": {
             "bucketArn": "bucket-arn",
             "dataFormat": "PARQUET",
             "prefix": "campaign-name",
              "storageCompressionFormat": "GZIP"
      }
    }
  ],
     "dataPartitions": [
      { ...  }
```

```
  ]
}
```

> **Note**
>
> Amazon Timestream ist in der Region Asien-Pazifik (Mumbai) nicht verfügbar.

- *campaign-name*Ersetzen Sie es durch den Namen der Kampagne, die Sie erstellen.
- Durch *signal-catalog-arn* den ARN des Signalkatalogs ersetzen.
- *target-arn*Ersetzen Sie es durch den ARN einer Flotte oder eines Fahrzeugs, das Sie erstellt haben.
- *role-arn*Ersetzen Sie es durch den ARN der Aufgabenausführungsrolle, die AWS FleetWise IoT-Berechtigungen zur Übermittlung von Daten an die Timestream-Tabelle erteilt.
- *table-arn*Ersetzen Sie durch den ARN der Timestream-Tabelle.

```
{
  "name": "campaign-name",
  "targetArn": "target-arn",
  "signalCatalogArn": "signal-catalog-arn",
  "collectionScheme": {
    "conditionBasedCollectionScheme": {
      "conditionLanguageVersion": 1,
      "expression": "$variable.`Vehicle.DemoBrakePedalPressure` > 7000",
      "minimumTriggerIntervalMs": 1000,
      "triggerMode": "ALWAYS"
    }
  },
  "compression": "SNAPPY",
  "diagnosticsMode": "OFF",
  "postTriggerCollectionDuration": 1000,
  "priority": 0,
  "signalsToCollect": [
    {
      "maxSampleCount": 100,
      "minimumSamplingIntervalMs": 0,
      "name": "Vehicle.DemoEngineTorque"
    },
    {
```

```
      "maxSampleCount": 100,
      "minimumSamplingIntervalMs": 0,
      "name": "Vehicle.DemoBrakePedalPressure"
    }
  ],
  "spoolingMode": "TO_DISK",
  "dataDestinationConfigs": [
    {
      "timestreamConfig": {
        "executionRoleArn": "role-arn",
        "timestreamTableArn": "table-arn"
      }
    }
  ],
   "dataPartitions": [
      { ...  }
  ]
}
```

- Ersetze es *campaign-name* durch den Namen der Kampagne, die du gerade erstellst.
- *signal-catalog-arn*Ersetzen Sie durch den Amazon-Ressourcennamen (ARN) des Signalkatalogs.
- *target-arn*Ersetzen Sie es durch den ARN einer Flotte oder eines Fahrzeugs, das Sie erstellt haben.
- *topic-arn*Ersetzen Sie es durch den ARN des MQTT-Themas, das Sie als Ziel für Nachrichten mit Fahrzeugdaten angegeben haben.
- *role-arn*Ersetzen Sie es durch den ARN der Aufgabenausführungsrolle, die AWS FleetWise IoT-Berechtigungen zum Senden, Empfangen und Ausführen von Aktionen für Nachrichten für das von Ihnen angegebene MQTT-Thema erteilt.

```
{
  "name": "campaign-name",
  "targetArn": "target-arn",
  "signalCatalogArn": "signal-catalog-arn",
  "collectionScheme": {
    "conditionBasedCollectionScheme": {
      "conditionLanguageVersion": 1,
      "expression": "$variable.`Vehicle.DemoBrakePedalPressure` > 7000",
      "minimumTriggerIntervalMs": 1000,
```

```
      "triggerMode": "ALWAYS"
    }
  },
  "compression": "SNAPPY",
  "diagnosticsMode": "OFF",
  "postTriggerCollectionDuration": 1000,
  "priority": 0,
  "signalsToCollect": [
    {
      "maxSampleCount": 100,
      "minimumSamplingIntervalMs": 0,
      "name": "Vehicle.DemoEngineTorque"
    },
    {
      "maxSampleCount": 100,
      "minimumSamplingIntervalMs": 0,
      "name": "Vehicle.DemoBrakePedalPressure"
    }
  ],
  "spoolingMode": "TO_DISK",
  "dataDestinationConfigs": [
      {
          "mqttTopicConfig": {
              "mqttTopicArn": "topic-arn",
              "executionRoleArn": "role-arn"
          }
      }
  ]
}
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `CreateCampaign` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Logische Ausdrücke für AWS FleetWise IoT-Kampagnen

AWS IoT FleetWise verwendet einen logischen Ausdruck, um zu erkennen, welche Daten im Rahmen einer Kampagne gesammelt werden sollen. Weitere Informationen zu Ausdrücken finden Sie unter Ausdrücke im AWS IoT Events Entwicklerhandbuch.

Die Ausdrucksvariable sollte so konstruiert sein, dass sie den Regeln für den Typ der gesammelten Daten entspricht. Bei Telemetriesystemdaten sollte die Ausdrucksvariable der vollständig qualifizierte Name des Signals sein. Bei Bildverarbeitungssystemdaten kombiniert der Ausdruck den vollständig qualifizierten Namen des Signals mit dem Pfad, der vom Datentyp des Signals zu einer seiner Eigenschaften führt.

Wenn der Signalkatalog beispielsweise die folgenden Knoten enthält:

```
{
    myVehicle.ADAS.Camera:
    type: sensor
    datatype: Vehicle.ADAS.CameraStruct
    description: "A camera sensor"

    myVehicle.ADAS.CameraStruct:
    type: struct
    description: "An obstacle detection camera output struct"
}
```

Wenn die Knoten der ROS 2-Definition folgen:

```
{
    Vehicle.ADAS.CameraStruct.msg:
    boolean obstaclesExists
```

```
    uint8[] image
    Obstacle[30] obstacles
}
{
    Vehicle.ADAS.Obstacle.msg:
    float32: probability
    uint8 o_type
    float32: distance
}
```

Im Folgenden sind alle möglichen Variablen für Ereignisausdrücke aufgeführt:

```
{
...
    $variable.`myVehicle.ADAS.Camera.obstaclesExists`
    $variable.`myVehicle.ADAS.Camera.Obstacle[0].probability`
    $variable.`myVehicle.ADAS.Camera.Obstacle[1].probability`
...
    $variable.`myVehicle.ADAS.Camera.Obstacle[29].probability`
    $variable.`myVehicle.ADAS.Camera.Obstacle[0].o_type`
    $variable.`myVehicle.ADAS.Camera.Obstacle[1].o_type`
...
    $variable.`myVehicle.ADAS.Camera.Obstacle[29].o_type`
    $variable.`myVehicle.ADAS.Camera.Obstacle[0].distance`
    $variable.`myVehicle.ADAS.Camera.Obstacle[1].distance`
...
    $variable.`myVehicle.ADAS.Camera.Obstacle[29].distance`
}
```

Aktualisieren Sie eine AWS FleetWise IoT-Kampagne

Sie können den UpdateCampaignAPI-Vorgang verwenden, um eine bestehende Kampagne zu aktualisieren. Der folgende Befehl verwendet AWS CLI.

- Ersetze es *`campaign-name`* durch den Namen der Kampagne, die du aktualisierst.
- Ersetzen Sie *`action`* durch einen der folgenden Werte:
  - `APPROVE`— Genehmigt die Kampagne, damit das AWS IoT der Dinge FleetWise in einem Fahrzeug oder einer Flotte eingesetzt werden kann.
  - `SUSPEND`— Setzt die Kampagne aus. Die Kampagne wird aus den Fahrzeugen gelöscht und alle Fahrzeuge der ausgesetzten Kampagne senden keine Daten mehr.

  - `RESUME`— Reaktiviert die `SUSPEND` Kampagne. Die Kampagne ist so eingestellt, dass sie beim nächsten Check-in auf alle Fahrzeuge übertragen wird und die Fahrzeuge wieder Daten senden.
  - `UPDATE`— Aktualisiert die Kampagne, indem Attribute definiert und mit der Kampagne verknüpft werden.
- *`description`*Durch eine neue Beschreibung ersetzen.

  Die Beschreibung kann bis zu 2.048 Zeichen enthalten.
- *`data-extra-dimensions`*Ersetzen Sie sie durch bestimmte Fahrzeugattribute, um die während der Kampagne gesammelten Daten zu erweitern. Sie können der Kampagne beispielsweise Fahrzeugmarke und -modell hinzufügen, und AWS IoT verknüpft FleetWise die Daten mit diesen Attributen als Dimensionen in Amazon Timestream. Anschließend können Sie die Daten anhand von Fahrzeugmarke und -modell abfragen.

```
aws iotfleetwise update-campaign \
            --name campaign-name \
            --action action \
            --description description \
            --data-extra-dimensions data-extra-dimensions
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `UpdateCampaign` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
```

```
    ]
}
```

Eine AWS FleetWise IoT-Kampagne löschen

Sie können die AWS FleetWise IoT-Konsole oder API verwenden, um Kampagnen zu löschen.

Löschen Sie eine Kampagne (Konsole)

Verwenden Sie die AWS FleetWise IoT-Konsole, um eine Kampagne zu löschen.

Um eine Kampagne zu löschen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Kampagnen aus.
3. Wählen Sie auf der Seite Kampagnen die Zielkampagne aus.
4. Wählen Sie Löschen aus.
5. Unter Löschen**`campaign-name`**? , geben Sie den Namen der Kampagne ein, die gelöscht werden soll, und wählen Sie dann Bestätigen.

Löschen Sie eine Kampagne (AWS CLI)

Sie können den DeleteCampaignAPI-Vorgang verwenden, um eine Kampagne zu löschen. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um eine Kampagne zu löschen.

*`campaign-name`*Ersetzen Sie es durch den Namen des Fahrzeugs, das Sie löschen möchten.

```
aws iotfleetwise delete-campaign --name campaign-name
```

> **Gelöschte Datenpartitionen können nicht wiederhergestellt werden**
>
> Durch das Löschen einer Kampagne werden alle Daten von Geräten entfernt und die Daten in einer Partition werden nicht in die Cloud hochgeladen.

Bestätigen Sie das Löschen der Kampagne

Sie können den ListCampaignsAPI-Vorgang verwenden, um zu überprüfen, ob eine Kampagne gelöscht wurde. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um eine paginierte Liste mit Zusammenfassungen für alle Kampagnen abzurufen.

```
aws iotfleetwise list-campaigns
```

Informationen zu AWS FleetWise IoT-Kampagnen abrufen

Sie können den GetCampaignAPI-Vorgang verwenden, um Fahrzeuginformationen abzurufen. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um die Metadaten einer Kampagne abzurufen.

*campaign-name*Ersetzen Sie es durch den Namen der Kampagne, die Sie abrufen möchten.

```
aws iotfleetwise get-campaign --name campaign-name
```

> **Note**
>
> Diese Operation ist letztlich konsistent. Mit anderen Worten, Änderungen an der Kampagne werden möglicherweise nicht sofort übernommen.

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle den `GetCampaign` API-Vorgang aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Kampagnendaten speichern und weiterleiten

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Verwenden Sie Datenpartitionen innerhalb von Kampagnen, um Signaldaten für Fahrzeuge und Flotten vorübergehend auf dem Edge zu speichern. Durch die Konfiguration von Upload- und Speicheroptionen für Datenpartitionen können Sie Ihre idealen Bedingungen für die Datenweiterleitung an Ihre angegebenen Datenziele (z. B. einen Amazon S3 S3-Bucket) optimieren. Sie können die Datenpartition beispielsweise so konfigurieren, dass Daten auf einem Fahrzeug gespeichert werden, bis es eine Wi-Fi-Verbindung herstellt. Sobald das Fahrzeug eine Verbindung hergestellt hat, löst die Kampagne aus, dass die Daten in dieser bestimmten Partition an die Cloud gesendet werden. Alternativ können Sie Daten mithilfe von AWS IoT Jobs sammeln.

Themen

- Datenpartitionen erstellen
- Laden Sie Kampagnendaten hoch
- Laden Sie Daten mithilfe von AWS IoT Jobs hoch

Datenpartitionen erstellen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Eine Datenpartition in einer Kampagne speichert vorübergehend Signaldaten. Sie konfigurieren, wann und wie die Daten an die Cloud weitergeleitet werden sollen.

Bei einer Datenpartition wird zunächst ein bestimmter Datensatz anhand der `dataPartitionId` für eine Kampagne verwendet. Anschließend können Sie weitere Speicheroptionen für Partitionen definieren, z. B. die maximale Größe, die Mindestdauer für die Aktualisierung der Datenpartition (auf der Festplatte) und den Speicherort der Daten auf dem Edge. Sie können den Speicherort im Fahrzeug mithilfe von ermittelnstorageLocation. Der Speicherort bestimmt den Ordnernamen für die Datenpartition unter dem Kampagnen-Speicherordner. Der Kampagnen-Speicherordner befindet sich in einem Ordner, der nach dem Fahrzeugnamen benannt ist, unter einem Persistenzpfad, der in der Edge-Konfigurationsdatei definiert ist. Dies ist der absolute Pfad des Speicherorts:. `{persistency_path} / {vehicle_name} / {campaign_name} / {storage_location}`

Der auf eingestellte Spooling-Modus `TO_DISK` gibt an, dass die partitionierten Daten auf einer Festplatte im Fahrzeug gespeichert werden sollen. Die Datenspeicherung für Datenpartitionen erfolgt auf FIFO-Basis (First In, First Out). Wenn Sie eine Kampagne löschen, löschen Sie auch die Daten in der zugehörigen Datenpartition. Wenn Sie keine Datenpartition für on/off Konnektivitätsanwendungsfälle angeben, speichert AWS IoT FleetWise dennoch Daten in einem Ringpuffer im Fahrzeug, wenn keine Konnektivität besteht. Wenn die Konnektivität wieder hergestellt wird, FleetWise lädt AWS IoT die Daten in die Cloud hoch. Dieses Verhalten ist in der Edge Agent for AWS FleetWise IoT-Software konfigurierbar.

> **Important**
>
> Wenn Ihre Datenpartition das festgelegte maximale Speicherlimit überschreitet, überschreiben neuere Daten ältere Daten, wenn die Partition die maximale Größe erreicht. Verlorene Daten auf dem Edge können nicht wiederhergestellt werden. Die Speichergröße wird durch Ihr Edge-Speicherlimit bestimmt.

Wenn Daten in die Cloud hochgeladen werden, können sie nach Ablauf der Mindestzeit entfernt werden. Stellen Sie die Mindestlebensdauer entsprechend ein, um ein unbeabsichtigtes Löschen zu vermeiden.

Die Upload-Optionen bestimmen die Variablenausdrücke und die Sprache der Bedingungen. Wenn Upload-Optionen angegeben sind, müssen Sie auch Speicheroptionen angeben. Sie können auch verlangen, dass Signale in Datenpartitionen in die Cloud hochgeladen werden. Weitere Informationen finden Sie unter Laden Sie Kampagnendaten hoch.

Nachdem die Bedingungen für die Datenpartition definiert wurden`signalsToCollect`, können Sie festlegen, welche Signale in der Datenpartition berücksichtigt werden sollen. Sie können entweder IDs Datenpartitionen angeben oder `dataPartitionId` auf `default` festlegen, dass eine etablierte Standarddatenpartition verwendet wird. Ein Signal ohne Angabe `dataPartitionId` wird der Standardeinstellung zugeordnet`dataPartition`.

Um eine Datenpartition zu erstellen

Erstellen Sie anhand des folgenden Beispiels eine Kampagne mit einer Speicherbedingung für Datenpartitionen. Diese Beispielkampagne ist so konfiguriert, dass Fahrzeugdaten in Amazon Timestream gespeichert werden.

1. *`campaign-name`*Ersetzen Sie es durch den Namen der Kampagne, die Sie erstellen.
2. (Optional) Geben Sie eine Beschreibung ein.
3. *`role-arn`*Ersetzen Sie es durch den Amazon-Ressourcennamen (ARN) der Aufgabenausführungsrolle, die AWS FleetWise IoT-Berechtigungen zur Übermittlung von Daten an die Timestream-Tabelle erteilt.
4. *`table-arn`*Ersetzen Sie durch den ARN der Timestream-Tabelle.
5. Durch *`signal-catalog-arn`* den ARN des Signalkatalogs ersetzen.
6. Ersetzen Sie *`data-partition-id`* sowohl die `dataPartitions` ID als auch die ID, der Sie zuordnen möchten`signalsToCollect`. Ersetzen Sie zunächst die ID der Datenpartition, die im Signal verwendet werden soll. Denn `signalsToCollect` die ID muss mit einer der unter IDs angegebenen übereinstimmen`dataPartitions`.

> **Note**
>
> Richten Sie eine Standarddatenpartition für eine Kampagne ein, indem Sie `default` sie als ID verwenden.

7. *target-arn*Ersetzen Sie es durch den ARN einer Flotte oder eines Fahrzeugs, das Sie erstellt haben.

```
{
    "name": "campaign-name",
    "description": "Measurement of SOC, SOH, thermal, and power optimization for Fleet
 2704",
    "targetArn": "target-arn",
    "collectionScheme": {
        "conditionBasedCollectionScheme": {
            "conditionLanguageVersion": 1,
            "expression": "$variable.`Vehicle.BMS` > 50",
            "minimumTriggerIntervalMs": 1000,
            "triggerMode": "ALWAYS"
        }
    },
    "compression": "SNAPPY",
    "dataDestinationConfigs": [{
        "timestreamConfig": {
            "executionRoleArn": "role-arn",
            "timestreamTableArn": "table-arn"
        }
    }],
    "dataPartitions": [{
        "id": "data-partition-id",
        "storageOptions": {
            "maximumSize": {
                "unit": "GB",
                "value": 1024
            },
            "minimumTimeToLive": {
                "unit": "WEEKS",
                "value": 6
            },
            "storageLocation": "string"
        },
```

```
        "uploadOptions": {
            "conditionLanguageVersion": 1,
            "expression": "$variable.`Vehicle.BMS.PowerOptimization` > 90"
        }
    }],
    "signalCatalogArn": "signal-catalog-arn",
    "signalsToCollect": [{
        "dataPartitionId": "data-partition-id",
        "maxSampleCount": 50000,
        "minimumSamplingIntervalMs": 100,
        "name": "Below-90-percent"
    }],
    "spoolingMode": "TO_DISK",
    "tags": [{
        "Key": "BMS",
        "Value": "Under-90"
    }]
}
```

Nachdem alle angegebenen Bedingungen erfüllt sind, werden die partitionierten Daten an die Cloud weitergeleitet, sodass neue partitionierte Signale gesammelt und gespeichert werden können.

Als Nächstes rufen Sie die `UpdateCampaign` API auf, um sie in der Edge Agent for AWS FleetWise IoT-Software bereitzustellen. Weitere Informationen finden Sie unter Laden Sie Kampagnendaten hoch.

Laden Sie Kampagnendaten hoch

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Es gibt zwei Möglichkeiten, Kampagnendaten auf den Edge hochzuladen:

- Kampagnen, die Ihre Upload-Bedingungen erfüllen, laden nach ihrer Genehmigung automatisch Daten in die Cloud hoch. Verwenden Sie den `updateCampaign` API-Vorgang, um eine Kampagne zu genehmigen.

- Mithilfe von AWS IoT Jobs können Sie das Hochladen von Daten erzwingen, auch wenn bestimmte Bedingungen nicht erfüllt sind. Weitere Informationen finden Sie unter Laden Sie Daten mithilfe von AWS IoT Jobs hoch.

Um Kampagnendaten mithilfe der **`UpdateCampaign`** API-Operation hochzuladen

Nachdem Sie die Kampagne erstellt haben, wird der Kampagnenstatus so `WAITING_FOR_APPROVAL` lange angezeigt, bis Sie das `action` in ändern`APPROVED`.

- Verwenden Sie das folgende Beispiel, um die Kampagne zu aktualisieren, `action` indem Sie den UpdateCampaignAPI-Vorgang aufrufen.

```
{
   "action": "APPROVED",
   "dataExtraDimensions": [ "string" ],
   "description": "string",
   "name": "string"
}
```

Laden Sie Daten mithilfe von AWS IoT Jobs hoch

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Mit AWS IoT Jobs können Sie Kampagnen so konfigurieren, dass gespeicherte Fahrzeugdaten bei Bedarf in die Cloud hochgeladen werden.

Um ein Jobdokument für Ihre Kampagne zu erstellen

- Verwenden Sie das folgende Beispiel, um ein Jobdokument für die Kampagne zu erstellen. Ein Auftragsdokument ist eine JSON-Datei, die Informationen über Fahrzeuge oder Flotten enthält, die für die Ausführung eines Auftrags erforderlich sind. Weitere Informationen zum Erstellen von Auftragsdokumenten finden Sie unter Erstellen und Verwalten von Aufträgen mithilfe von AWS CLI im AWS IoT Entwicklerhandbuch.

Um anzufordern, dass nur ein Fahrzeug Daten hochlädt, legen Sie als Jobziel das Objekt AWS IoT fest, das dem Fahrzeug zugeordnet ist. Um zu verlangen, dass mehrere Fahrzeuge (in derselben Kampagne) Daten hochladen, erstellen Sie eine Dinggruppe mit allen Dingen, die den Fahrzeugen entsprechen, und legen Sie dann das Jobziel auf die Dinggruppe fest.

```
{
  "version": "1.0",
  "parameters": {
     "campaignArn": ${aws:iot:parameter:campaignArn},
     "endTime": ${aws:iot:parameter:endTime}
  }
}
```

a. `CampaignArn`Ersetzen Sie es durch den Amazon-Ressourcennamen (ARN) einer Kampagne in derselben Region und demselben Konto. Der Kampagnen-ARN ist erforderlich.

b. (Optional) `endTime` Ersetzen Sie ihn durch den Zeitstempel der am Fahrzeug gesammelten Daten im UTC-Format ISO 8601 (ohne Millisekunden). Beispiel, `2024-03-05T23:00:00Z`. Der Zeitstempel ist exklusiv und bestimmt den letzten hochzuladenden Datenpunkt. Wenn Sie diesen Wert weglassen`endTime`, fährt die Edge Agent-Software mit dem Upload fort, bis alle gespeicherten Daten einer Kampagne hochgeladen sind. Nachdem alle Daten hochgeladen wurden, wird der Status der Auftragsausführung auf `SUCCEEDED` aktualisiert. Der Status des Jobs wird auf aktualisiert`COMPLETED`.

Um einen Job mit einer verwalteten Jobvorlage zu erstellen

1. Wählen Sie IoT-IO TFleet Wise- CollectCampaignData aus der Liste der verwalteten Vorlagen aus. Weitere Informationen finden Sie im AWS IoT Entwicklerhandbuch unter Erstellen eines Jobs aus AWS verwalteten Vorlagen.
2. Die verwaltete Vorlage hat die `endTime` Parameter `CampaignArn` und.

   a. `CampaignArn`Ersetzen Sie es durch den Amazon-Ressourcennamen (ARN) einer Kampagne in derselben Region und demselben Konto. Der Kampagnen-ARN ist erforderlich.

   b. (Optional) `endTime` Ersetzen Sie ihn durch den Zeitstempel der am Fahrzeug gesammelten Daten im UTC-Format ISO 8601 (ohne Millisekunden). Beispiel, `2024-03-05T23:00:00Z`.

Der Zeitstempel ist exklusiv und bestimmt den letzten hochzuladenden Datenpunkt. Wenn Sie diesen Wert weglassen`endTime`, fährt die Edge Agent-Software mit dem Upload fort, bis alle gespeicherten Daten einer Kampagne hochgeladen sind. Nachdem alle Daten hochgeladen wurden, wird der Status der Auftragsausführung auf `SUCCEEDED` aktualisiert. Der Status des Jobs wird auf aktualisiert`COMPLETED`.

Verwandte Themen zur Problembehandlung finden Sie unterProbleme speichern und weiterleiten.

Weitere Informationen zu AWS IoT Jobs finden Sie im AWS IoT Developer Guide unter Jobs.

Sammeln Sie diagnostische Fehlercode-Daten mithilfe von AWS IoT FleetWise

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Wenn ein Fahrzeug einen Fehler erkennt, generiert es einen Diagnose-Fehlercode (DTC) und zeichnet eine Momentaufnahme der betroffenen Sensoren oder Aktuatoren auf. DTCs hilft Ihnen, nahezu in Echtzeit über Fehler zu erfahren, zu verstehen, was sie verursacht, und Korrekturmaßnahmen zu ergreifen. AWS IoT FleetWise unterstützt die Erfassung von DTCs, einschließlich entsprechender DTC-Snapshots und erweiterter Daten, im Rahmen einer Datenerfassungskampagne. In diesem Thema werden die Konzepte, Workflows und Schlüsselwörter vorgestellt, die die DTC-Datenerfassung erleichtern, anhand von Beispielen veranschaulicht.

Im Folgenden werden die wichtigsten Konzepte für die Verwendung von DTC beschrieben.

Benutzerdefiniert definierte Funktionen

Eine benutzerdefinierte Funktion ist die Möglichkeit, Ihre eigenen, auf dem Edge Agent vordefinierten Funktionen aufzurufen und auszuführen, wodurch das benutzerdefinierte Dekodierungskonzept erweitert wird. Diese Funktionen werden in Abstimmung mit dem AWS IoT FleetWise Agent verwendet. Die Edge Agent for AWS FleetWise IoT-Software bietet integrierte Funktionen zur Berechnung von Signalstatistiken wie Minimal-, Maximal- und

Durchschnittswerten. Eine benutzerdefinierte Funktion erweitert diese Funktion, indem sie es Ihnen ermöglicht, maßgeschneiderte Logik für bestimmte Anwendungsfälle zu erstellen. Für die Erfassung von DTC-Daten (Diagnostic Trouble Code) können Entwickler benutzerdefinierte Funktionen nutzen, um erweiterte Datenabrufmechanismen zu implementieren, z. B. das Abrufen von DTC-Codes, Schnappschüssen und erweiterten Daten direkt vom Edge des Fahrzeugs über Unified Diagnostic Services (UDS) oder alternative Diagnoseschnittstellen.

Weitere Informationen finden Sie im Leitfaden für benutzerdefinierte Funktionen und in der Referenzimplementierung der DTC-Datenerfassung im Edge Agent Developer Guide.

Abrufen von Signalen

Bei Datenerfassungskampagnen werden Signale in der Regel kontinuierlich von einem Gerät erfasst und in der Edge Agent-Software zwischengespeichert. Signale werden dann in regelmäßigen Abständen in zeitbasierten Kampagnen hochgeladen oder gespeichert oder bei zustandsabhängigen Kampagnen durch bestimmte Bedingungen ausgelöst. Aufgrund von Bedenken hinsichtlich einer Überlastung des Geräteverkehrs können DTC-Signale jedoch nicht von Geräten erfasst und kontinuierlich zwischengespeichert werden. Um diesem Problem zu begegnen, FleetWise bietet AWS IoT einen Signalabruf, der sicherstellt, dass das Zielsignal diskontinuierlich von einem Gerät abgerufen wird.

Das Abrufen von Signalen unterstützt sowohl periodische als auch zustandsabhängige Aktionen. Sie können die vom Abruf gesteuerte Methode, die Bedingungen und die genauen Aktionen mithilfe benutzerdefinierter Funktionen für jedes Signal definieren, das nicht kontinuierlich von einem Gerät erfasst werden soll. Bei Signalen, die vom Signalabrufmechanismus verwaltet werden, werden der Triggertyp und die Bedingungen für die lokale Speicherung oder den `CollectionScheme` Cloud-Upload weiterhin von beiden bestimmt `timeBasedCollectionScheme` und `conditionBasedCollectionScheme` unterstützt, was den regulären Signalen entspricht.

In den folgenden Themen erfahren Sie, wie Sie etwas erstellen und verwenden DTCs können.

Themen

- Schlüsselwörter für den diagnostischen Fehlercode
- Erstellen Sie eine Kampagne zur Datenerfassung für diagnostische Fehlercodes
- Anwendungsfälle für diagnostische Fehlercodes

Schlüsselwörter für den diagnostischen Fehlercode

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

`signalsToFetch`Parameter für die Kampagnenerstellung

Verwenden Sie die signalsToFetchSyntax, um zu konfigurieren, wie die Signalinformationen auf dem Edge abgerufen werden können. Das Abrufen von Standardsignalen wird durch Modellierung anhand von Regeln gesteuert, die explizit in einem Decoder-Manifest definiert sind, oder durch Edge First Modeling benutzerdefiniert. Mithilfe von Signalen zum Abrufen können Sie definieren, wann und wie Daten während Kampagnen abgerufen werden.

Abzurufende Signale ermöglichen die Erfassung von DTC-Informationen. Sie können beispielsweise ein Signal vom Typ Zeichenfolge mit dem Namen erstellen`DTC_Info`, das DTC-Informationen für jedes Motorsteuergerät (ECU) enthalten kann. Sie können auch nach einem bestimmten Steuergerät filtern.

- `SignalFetchInformation`Struktur- und Parameterdefinitionen.

```
structure SignalFetchInformation {
    @required
    fullyQualifiedName: NodePath,
    @required
    signalFetchConfig: SignalFetchConfig,
    // Conditional language version for this config
    conditionLanguageVersion: languageVersion,
    @required
    actions: EventExpressionList,
}
```

  - `fullyQualifiedName`: der vollqualifizierte Name (FQDN) des Signals, für das Sie den benutzerdefinierten Abruf verwenden möchten.
  - `signalFetchConfig`: definiert Regeln, wie die oben definierten Signale abgerufen werden sollen. Es unterstützt zeit- und bedingungsbasiertes Abrufen.

- `conditionLanguageVersion`: Die bedingte Sprachversion, die zum Analysieren des Ausdrucks in der Konfiguration verwendet wird.
- `actions`: eine Liste aller Aktionsausdrücke, die auf dem Edge ausgewertet wurden. Der Edge erhält den Wert des definierten Signals.

> **Important**
>
> Aktionen können nur verwenden`custom_function`.

Schlüsselwörter für Kampagnenausdrücke

Der folgende Ausdruck verwendet den vollständig qualifizierten Namen eines Signals, der vom Fahrzeug unterstützt wird, und gibt „true“ zurück, wenn das Signal keine Daten in den Signalpuffern auf dem Edge enthält. Andernfalls wird False zurückgegeben.

```
isNull(signalFqdn:String): Boolean
```

Example Nutzung

```
isNull($variable.`Vehicle.ECU1.DTC_INFO`) == false

We want to make sure DTC_Info signal is being generated
on edge.
```

Dieser Ausdruck benötigt die folgende Eingabe:

Funktionsname: Zeichenfolge

Der Name der benutzerdefinierten Funktion, die vom Edge unterstützt wird

Parameter: varargs ***Expression***

Parameter für. `functionName` Dies kann eine beliebige Liste von Ausdrücken sein.

Parameter unterstützen Literaltypen: String, Int, Boolean oder Double.

```
custom_function(functionName:String, params: varargsExpression): Void
```

Example Nutzung

```
{
       "fullyQualifiedName":"Vehicle.ECU1.DTC_INFO",
       "signalFetchConfig":{
          "timeBased":{
             "executionFrequencyMs":2000
          }
       },
       "actions":"custom_function(“DTC_QUERY”, -1, 2, -1)"
    }
```

Erstellen Sie eine Kampagne zur Datenerfassung für diagnostische Fehlercodes

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

In diesem Thema wird beschrieben, wie Sie eine Kampagne zur Datenerfassung für Diagnose-Fehlercodes (DTC) erstellen.

1. Definieren Sie ein benutzerdefiniertes Signal auf dem Edge. Sie müssen die Dekodierungsregeln für das DTC-Signal auf dem Edge als benutzerdefiniertes dekodiertes Signal definieren. Weitere Informationen finden Sie unter Tutorial: Konfiguration der netzwerkunabhängigen Datenerfassung mithilfe einer benutzerdefinierten Dekodierungsschnittstelle.
2. Definieren Sie eine benutzerdefinierte Funktion auf dem Edge. Sie müssen eine benutzerdefinierte Funktion zum Sammeln von DTC-Signalen auf dem Edge zu einem kompilierten Zeitpunkt definieren.

   Weitere Informationen finden Sie im Leitfaden für benutzerdefinierte Funktionen und in der Referenzimplementierung der DTC-Datenerfassung im Edge Agent Developer Guide.

> **Note**
>
> Ein Beispiel für eine benutzerdefinierte Funktion ist `DTC_QUERY` im Demoskript dargestellt.

3. Erstellen Sie einen Signalkatalog, der ein DTC-Signal als Zeichenfolgentyp modelliert.

```
[
 {
    "branch": {
        "fullyQualifiedName": "Vehicle",
        "description": "Vehicle"
        }
      },
      {
    "branch": {
        "fullyQualifiedName": "Vehicle.ECU1",
        "description": "Vehicle.ECU1"
        }
      },
      {
    "sensor": {
        "fullyQualifiedName": "Vehicle.ECU1.DTC_INFO",
        "description": "Vehicle.ECU1.DTC_INFO",
        "dataType": "STRING"
      }
   }
 ]
```

4. Erstellen und aktivieren Sie ein Fahrzeugmodell mit dem hinzugefügten DTC-Signal.
5. Erstellen und aktivieren Sie ein Decoder-Manifest mit dem hinzugefügten DTC-Signal. Das DTC-Signal sollte ein `CUSTOM_DECODING_SIGNAL` Signaldecodertyp mit einem `CUSTOM_DECODING_INTERFACE` Netzwerkschnittstellentyp sein.

   Example Signaldecoder

```
[
  {
    "fullyQualifiedName": "Vehicle.ECU1.DTC_INFO",
    "interfaceId": "UDS_DTC",
    "type": "CUSTOM_DECODING_SIGNAL",
```

```
    "customDecodingSignal": {
      "id": "Vehicle.ECU1.DTC_INFO"
    }
  }
 ]
```

Example Netzwerkschnittstelle

```
[
  {
    "interfaceId": "UDS_DTC",
    "type": "CUSTOM_DECODING_INTERFACE",
    "customDecodingInterface": {
      "name": "NamedSignalInterface"
    }
  }
]
```

> **Note**
>
> CAN-Signale (Controller Area Network) unterstützen den String-Datentyp nicht.

6. Fahrzeuge bereitstellen und erstellen. Die Fahrzeuge müssen ein Fahrzeugmodell (Modellmanifest) und ein Decoder-Manifest verwenden, die in den vorherigen Schritten aktiviert wurden.
7. Erstellen und genehmigen Sie die Kampagne. Sie müssen eine Kampagne erstellen, indem Sie DTC-Signale (optional mit Telemetriesignalen) definieren und sie in Fahrzeugen einsetzen.
8. Greifen Sie auf die Daten im definierten Ziel zu. Zu den DTC-Daten gehören das`DTCCode`,`DTCSnapshot`, und `DTCExtendedDatastrings` als unformatierte Zeichenfolge in dem in der Kampagne definierten Datenziel.

Anwendungsfälle für diagnostische Fehlercodes

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

In den folgenden Anwendungsfällen wird davon ausgegangen, dass die `DTC_QUERY` Funktion im Demoskript definiert wurde.

Regelmäßiger Abruf

Ruft eine DTC-Sammlung in konfigurierten Intervallen ab.

Das folgende Beispiel zeigt eine Kampagne mit regelmäßigem Abrufen von Signalen `Vehicle.DTC_INFO` für alle DTCs mit einer Statusmaske für alle. ECUs Es gibt eine Bedingung für die Erfassung von Daten für`Vehicle.DTC_INFO`.

```
{
  "compression": "SNAPPY",
  "spoolingMode": "TO_DISK",
  "signalsToFetch": [
    {
      "fullyQualifiedName": "Vehicle.ECU1.DTC_INFO",
      "signalFetchConfig": {
        "timeBased": {
        // The FleetWise Edge Agent will query the UDS module for all DTCs every five
 seconds.
          "executionFrequencyMs": 5000
        }
      },
      "actions": [
      // Every five seconds, this action is called and its output is stored in the
      // signal history buffer of Vehicle.DTC_INFO
        "custom_function(\"DTC_QUERY\", -1, 2, -1)"
      ]
    }
  ],
  "signalsToCollect": [
```

```
    {
      "name": "Vehicle.ECU1.DTC_INFO"
    }
  ],
  "collectionScheme": {
    "conditionBasedCollectionScheme": {
      "conditionLanguageVersion": 1,
      // Whenever a new DTC is filled into the signal, the data is ingested.
      "expression": "!isNull($variable.`Vehicle.ECU1.DTC_INFO`)",
      "minimumTriggerIntervalMs": 1000,
      // Make sure that data is ingested only when there are new DTCs.
      "triggerMode": "RISING_EDGE"
    }
  },
  "dataDestinationConfigs": [
    {
      "s3Config":
        {
          "bucketArn": "bucket-arn",
          "dataFormat": "PARQUET",
          "prefix": "campaign-name",
          "storageCompressionFormat": "GZIP"
        }
    }
  ]
}
```

Bedingungsgesteuerter Abruf

Ruft eine DTC-Sammlung ab, wenn eine Bedingung erfüllt ist. Wenn beispielsweise das CAN-Signal anliegt`Vehicle.Ignition == 1`, rufen Sie die DTC-Daten ab und laden Sie sie hoch.

In der folgenden Beispielkampagne wird ein bedingungsgesteuerter Signalabruf aktiviert, `Vehicle.ECU1.DTC_INFO` um zu überprüfen, ob der DTC (“ AAA123 „) mit RecordNumber 1 für ECU-1 aussteht. Bei dieser Kampagne werden Daten zeitbasiert erfasst und hochgeladen.

```
{
  "compression": "SNAPPY",
  "spoolingMode": "TO_DISK",
  "signalsToFetch": [
    {
      "fullyQualifiedName": "Vehicle.ECU1.DTC_INFO",
      "signalFetchConfig": {
```

```
        "conditionBased": {
        // The action will only run when the ignition is on.
          "conditionExpression": "$variable.`Vehicle.Ignition` == 1",
          "triggerMode": "ALWAYS"
        }
      },
      // The UDS module is only requested for the specific ECU address and the specific
 DTC Number/Status.
      "actions": ["custom_function(\"DTC_QUERY\", 1, 2, 8, \"0xAAA123\")"]
    }
  ],
  "signalsToCollect": [
    {
      "name": "Vehicle.ECU1.DTC_INFO"
    },
    {
      "name": "Vehicle.Ignition"
    }
  ],
  "collectionScheme": {
    "timeBasedCollectionScheme": {
      "periodMs": 10000
    }
  },
  "dataDestinationConfigs": [
    {
      "s3Config":
        {
          "bucketArn": "bucket-arn",
          "dataFormat": "PARQUET",
          "prefix": "campaign-name",
          "storageCompressionFormat": "GZIP"
        }
    }
  ]
}
```

Abruf auf Abruf

Ruft einen bestimmten DTC für eine Flotte ab.

Für einen On-Demand-Anwendungsfall können Sie dieselbe Kampagne verwenden, die im regelmäßigen Abruf definiert ist. Der On-Demand-Effekt wird erreicht, indem die Kampagne kurz nach

der Bereitstellung der Kampagne über die AWS FleetWise IoT-Konsole oder durch Ausführen des folgenden CLI-Befehls unterbrochen wird.

- Ersetzen Sie es *command-name* durch den Befehlsnamen.

```
aws iotfleetwise update-campaign \
    --name campaign-name \
    --action APPROVE
```

Unterbrechen Sie dann die Kampagne, nachdem die DTC-Daten eingetroffen sind.

```
aws iotfleetwise update-campaign \
    --name campaign-name \
    --action SUSPEND
```

Sie können die Kampagne erneut fortsetzen, um DTC-Daten abzurufen.

```
aws iotfleetwise update-campaign \
    --name campaign-name \
    --action RESUME
```

Visualisieren Sie AWS FleetWise IoT-Fahrzeugdaten

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Die Edge Agent for AWS FleetWise IoT-Software sendet ausgewählte Fahrzeugdaten an ein MQTT-Thema oder überträgt sie an Amazon Timestream oder Amazon Simple Storage Service (Amazon S3). Nachdem Ihre Daten am Datenziel angekommen sind, können Sie andere AWS Dienste verwenden, um sie zu verarbeiten, umzuleiten, zu visualisieren und zu teilen.

> **Note**
>
> Amazon Timestream ist in der Region Asien-Pazifik (Mumbai) nicht verfügbar.

Verarbeitung von Fahrzeugdaten, die an ein MQTT-Thema gesendet wurden

Über MQTT-Messaging gesendete Fahrzeugdaten werden nahezu in Echtzeit übermittelt und ermöglichen es Ihnen, mithilfe von Regeln Maßnahmen zu ergreifen oder Daten an andere Ziele weiterzuleiten. Weitere Informationen zur Verwendung von MQTT finden Sie unter Gerätekommunikationsprotokolle und Regeln für AWS IoT im AWS IoT Core Entwicklerhandbuch.

Das Standardschema für Daten, die in einer MQTT-Nachricht gesendet werden, enthält die folgenden Felder.

Feldname	Datentyp	Description
`eventId`	varchar	Die ID des Datenerfassungsereignisses.
`vehicleName`	varchar	Die ID des Fahrzeugs, von dem die Daten gesammelt wurden.
`name`	varchar	Der Name der Kampagne, mit der die Edge Agent-Software Daten sammelt.
`time`	Zeitstempel	Der Zeitstempel des Datenpunkts.
`measure_name`	varchar	Der Name des Signals.
`measure_value::bigint`	bigint	Signalwerte vom Typ Integer.

Feldname	Datentyp	Description
`measure_value::double`	double	Signalwerte vom Typ Double.
`measure_value::boolean`	boolesch	Signalwerte des Typs Boolean.
`measure_value::varchar`	varchar	Signalwerte des Typs varchar.

Fahrzeugdaten in Timestream verarbeiten

Timestream ist eine vollständig verwaltete Zeitreihendatenbank, die Billionen von Zeitreihendatenpunkten pro Tag speichern und analysieren kann. Ihre Daten werden in einer vom Kunden verwalteten Timestream-Tabelle gespeichert. Sie können Timestream verwenden, um Fahrzeugdaten abzufragen, um Einblicke in Ihre Fahrzeuge zu gewinnen. Weitere Informationen finden Sie unter Was ist Amazon Timestream?

Das Standardschema für Daten, die an Timestream übertragen werden, enthält die folgenden Felder.

Feldname	Datentyp	Description
`eventId`	varchar	Die ID des Datenerfassungsereignisses.
`vehicleName`	varchar	Die ID des Fahrzeugs, von dem die Daten gesammelt wurden.
`name`	varchar	Der Name der Kampagne, mit der die Edge Agent-Software Daten sammelt.
`time`	Zeitstempel	Der Zeitstempel des Datenpunkts.

Feldname	Datentyp	Description
`measure_name`	varchar	Der Name des Signals.
`measure_value::bigint`	bigint	Signalwerte vom Typ Integer.
`measure_value::double`	double	Signalwerte vom Typ Double.
`measure_value::boolean`	boolesch	Signalwerte des Typs Boolean.
`measure_value::varchar`	varchar	Signalwerte des Typs varchar.

Visualisieren Sie die in Timestream gespeicherten Fahrzeugdaten

Nachdem Ihre Fahrzeugdaten an Timestream übertragen wurden, können Sie die folgenden AWS Dienste verwenden, um Ihre Daten zu visualisieren, zu überwachen, zu analysieren und zu teilen.

- Visualisieren und überwachen Sie Daten in Dashboards mithilfe von Grafana oder Amazon Managed Grafana. Sie können Daten aus mehreren AWS Quellen (wie Amazon CloudWatch und Timestream) und anderen Datenquellen mit einem einzigen Grafana-Dashboard visualisieren.
- Analysieren und visualisieren Sie Daten in Dashboards mithilfe von Quick.

Fahrzeugdaten in Amazon S3 verarbeiten

Amazon S3 ist ein Objektspeicherservice, der beliebige Datenmengen speichert und schützt. Sie können S3 für eine Vielzahl von Anwendungsfällen verwenden, z. B. für Datenseen, Sicherung und Wiederherstellung, Archivierung, Unternehmensanwendungen, AWS IoT Geräte und Big-Data-Analysen. Ihre Daten werden in S3 als Objekte in Buckets gespeichert. Weitere Informationen finden Sie unter Was ist Amazon S3?

Das Standardschema von Daten, die an Amazon S3 übertragen werden, enthält die folgenden Felder.

Feldname	Datentyp	Description
`eventId`	varchar	Die ID des Datenerfassungsereignisses.
`vehicleName`	varchar	Die ID des Fahrzeugs, von dem die Daten gesammelt wurden.
`name`	varchar	Der Name der Kampagne, mit der die Edge Agent-Software Daten sammelt.
`time`	Zeitstempel	Der Zeitstempel des Datenpunkts.
`measure_name`	varchar	Der Name des Signals.
`measure_value_BIGINT`	bigint	Signalwerte vom Typ Integer.
`measure_value_DOUBLE`	double	Signalwerte vom Typ Double.
`measure_value_BOOLEAN`	boolesch	Signalwerte des Typs Boolean.
`measure_value_STRUCT`	struct	Signalwerte des Typs Struct.
`measure_value_VARCHAR`	varchar	Signalwerte des Typs varchar.

Amazon S3 S3-Objektformat

AWS IoT FleetWise überträgt Fahrzeugdaten an S3, wo sie als Objekt gespeichert werden. Sie können den Objekt-URI verwenden, der die Daten eindeutig identifiziert, um Daten aus der Kampagne zu finden. Das S3-Objekt-URI-Format hängt davon ab, ob es sich bei den gesammelten Daten um unstrukturierte oder verarbeitete Daten handelt.

Unstrukturierte Daten

Unstrukturierte Daten werden in S3 auf nicht vordefinierte Weise gespeichert. Es kann in verschiedenen Formaten wie Bildern oder Videos vorliegen.

Fahrzeugnachrichten, die FleetWise mit Signaldaten aus Amazon Ion-Dateien an das AWS IoT übergeben werden, werden dekodiert und als Objekte an S3 übertragen. Die S3-Objekte repräsentieren jedes Signal und sind binär codiert.

Der S3-Objekt-URI für unstrukturierte Daten verwendet das folgende Format:

```
s3://bucket-name/prefix/unstructured-data/random-ID-yyyy-MM-dd-HH-mm-ss-SSS-
vehicleName-signalName-fieldName
```

Verarbeitete Daten

Verarbeitete Daten werden in S3 gespeichert und durchlaufen Verarbeitungsschritte, die Nachrichten validieren, anreichern und transformieren. Objektlisten und Geschwindigkeit sind Beispiele für verarbeitete Daten.

An S3 übertragene Daten werden als Objekte gespeichert, die Datensätze darstellen, die für einen Zeitraum von etwa 10 Minuten zwischengespeichert wurden. Standardmäßig FleetWise fügt AWS IoT dem Format ein UTC-Zeitpräfix hinzu, `year=YYYY/month=MM/date=DD/hour=HH` bevor Objekte in S3 geschrieben werden. Dieses Präfix erstellt eine logische Hierarchie im Bucket, in der jeder Schrägstrich (`/`) eine Ebene in der Hierarchie erzeugt. Die verarbeiteten Daten enthalten auch den S3-Objekt-URI für unstrukturierte Daten.

Der S3-Objekt-URI für verarbeitete Daten verwendet das folgende Format:

```
s3://bucket-name/prefix/processed-data/year=YYYY/month=MM/day=DD/hour=HH/
part-0000-random-ID.gz.parquet
```

Rohdaten

Rohdaten, auch Primärdaten genannt, sind Daten, die aus Amazon Ion-Dateien gesammelt wurden. Sie können Rohdaten verwenden, um Probleme zu beheben oder Fehler zu beheben.

Der S3-Objekt-URI für Rohdaten verwendet das folgende Format:

```
s3://bucket-name/prefix/raw-data/vehicle-name/eventID-timestamp.10n
```

Analysieren Sie die in Amazon S3 gespeicherten Fahrzeugdaten

Nachdem Ihre Fahrzeugdaten an S3 übertragen wurden, können Sie die folgenden AWS Dienste verwenden, um Ihre Daten zu überwachen, zu analysieren und zu teilen.

Extrahieren und analysieren Sie Daten mit Amazon SageMaker AI für nachgelagerte Labeling- und Machine Learning-Workflows (ML).

Weitere Informationen finden Sie in den folgenden Themen im Amazon SageMaker AI Developer Guide:

- Daten verarbeiten
- Trainieren Sie Modelle für maschinelles Lernen
- Bilder beschriften

Katalogisieren Sie Ihre Daten mithilfe AWS-Glue-Crawler und analysieren Sie sie in Amazon Athena. Standardmäßig verfügen in S3 geschriebene Objekte über Zeitpartitionen im Apache Hive-Stil mit Datenpfaden, die Schlüssel-Wert-Paare enthalten, die durch Gleichheitszeichen verbunden sind.

Weitere Informationen finden Sie in den folgenden Themen im Amazon Athena Athena-Benutzerhandbuch:

- Daten in Athena partitionieren
- Wird verwendet AWS Glue , um eine Verbindung zu Datenquellen in Amazon S3 herzustellen
- Bewährte Methoden bei der Verwendung von Athena mit AWS Glue

Visualisieren Sie Daten mit Quick, indem Sie entweder Ihre Athena-Tabelle oder Ihren S3-Bucket direkt lesen.

> **Tip**
>
> Wenn Sie direkt aus S3 lesen, vergewissern Sie sich, dass Ihre Fahrzeugdaten im JSON-Format vorliegen, da Quick das Apache Parquet-Format nicht unterstützt.

Weitere Informationen finden Sie in den folgenden Themen in der Amazon-Kurzanleitung:

- Unterstützte Datenquellen
- Eine Datenquelle erstellen

Befehle

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.
> In dieser Dokumentation wird beschrieben, wie Sie die Befehlsfunktion für AWS IoT verwenden FleetWise. Informationen zur Verwendung der Befehlsfunktion in AWS IoT Device Management finden Sie unter Befehle.
> Sie sind allein dafür verantwortlich, Befehle auf sichere und gesetzeskonforme Weise bereitzustellen. Weitere Informationen zu Ihren Pflichten finden Sie in den AWS Nutzungsbedingungen für AWS IoT Dienste.

Verwenden Sie die Befehlsfunktion, um Befehle an einem Fahrzeug aus der Cloud auszuführen. Befehle zielen jeweils auf ein Gerät ab und können für Anwendungen mit niedriger Latenz und hohem Durchsatz verwendet werden, z. B. zum Abrufen der geräteseitigen Protokolle oder zum Initiieren einer Änderung des Gerätestatus.

Der Befehl ist eine Ressource, die von verwaltet wird. AWS IoT Device Management Er enthält wiederverwendbare Konfigurationen, die angewendet werden, wenn eine Befehlsausführung an das Fahrzeug gesendet wird. Sie können eine Reihe von Befehlen für bestimmte Anwendungsfälle vordefinieren oder sie verwenden, um wiederverwendbare Konfigurationen für wiederkehrende Anwendungsfälle zu erstellen. Sie können beispielsweise Befehle konfigurieren, die von einer App verwendet werden können, um die Tür eines Fahrzeugs zu verriegeln oder die Temperatur aus der Ferne zu ändern.

Mit der AWS IoT Befehlsfunktion können Sie:

- Erstellen Sie eine Befehlsressource und verwenden Sie die Konfiguration erneut, um mehrere Befehle an Ihr Zielgerät zu senden und sie dann auf dem Gerät auszuführen.
- Steuern Sie die Granularität, mit der jeder Befehl auf dem Gerät ausgeführt werden soll. Sie können beispielsweise ein Fahrzeug als Objekt bereitstellen und dann AWS IoT einen Befehl zum Ver- oder Entriegeln der Fahrzeugtüren senden.
- Führen Sie mehrere Befehle gleichzeitig auf dem Zielgerät aus, ohne darauf zu warten, dass der vorherige Befehl abgeschlossen ist.

- Aktivieren Sie Benachrichtigungen für Befehlsereignisse und rufen Sie die Status- und Ergebnisinformationen vom Gerät ab, während es den Befehl ausführt und wenn er abgeschlossen ist.

In den folgenden Themen erfahren Sie, wie Sie Befehle erstellen, senden, empfangen und verwalten.

Themen

- Befehlskonzepte
- Fahrzeuge und Kommandos
- Erstellen und Verwalten von Befehlen
- Starten und Überwachen von Befehlsausführungen
- Beispiel: Verwendung von Befehlen zur Steuerung eines Fahrzeuglenkmodus ()AWS CLI
- Szenarien zur Verwendung von Befehlen

Befehlskonzepte

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Befehle sind Anweisungen, die von der Cloud an Ihr Zielgerät gesendet werden. Das Zielgerät kann ein Fahrzeug sein und es muss als AWS IoT Ding in der Ding-Registrierung registriert sein. Der Befehl kann Parameter enthalten, die eine Aktion definieren, die die Aktuatoren des Fahrzeugs ausführen müssen. Das Fahrzeug analysiert dann den Befehl und seine Parameter und verarbeitet sie, um die entsprechende Aktion auszuführen. Es antwortet dann auf die Cloud-Anwendung mit dem Status der Befehlsausführung.

Den detaillierten Arbeitsablauf finden Sie unterFahrzeuge und Kommandos.

Themen

- Befehle und wichtige Konzepte
- Status der Befehlsausführung

Befehle und wichtige Konzepte

Im Folgenden werden einige wichtige Konzepte für die Verwendung der Befehlsfunktion und ihre Funktionsweise mit Statusvorlagen für den letzten bekannten Status (LKS) beschrieben.

Befehl

Ein Befehl ist eine Entität, mit der Sie Anweisungen an ein physisches Fahrzeug senden können, damit es Aktionen wie das Einschalten des Motors oder das Ändern der Fensterposition ausführt. Sie können eine Reihe von Befehlen für bestimmte Anwendungsfälle vordefinieren oder sie verwenden, um wiederverwendbare Konfigurationen für wiederkehrende Anwendungsfälle zu erstellen. Sie können beispielsweise Befehle konfigurieren, die von einer App verwendet werden können, um die Tür eines Fahrzeugs zu verriegeln oder die Temperatur aus der Ferne zu ändern.

Namespace

Wenn Sie die Befehlsfunktion verwenden, müssen Sie den Namespace für den Befehl angeben. Wenn Sie einen Befehl in AWS IoT erstellen FleetWise, müssen Sie ihn `AWS-IoT-FleetWise` als Ihren Namespace auswählen. Wenn Sie diesen Namespace verwenden, müssen Sie die Parameter angeben, die zur Ausführung des Befehls auf dem Fahrzeug verwendet werden. Wenn Sie stattdessen einen Befehl in erstellen möchten, müssen Sie AWS IoT Device Management stattdessen den `AWS-IoT` Namespace verwenden. Weitere Informationen finden Sie unter Befehle im AWS IoT Device Management Entwicklerhandbuch.

Befehlsstatus

Die Befehle, die Sie erstellen, befinden sich im Status „Verfügbar“, was bedeutet, dass sie verwendet werden können, um eine Befehlsausführung auf dem Fahrzeug zu starten. Wenn ein Befehl veraltet ist, können Sie ihn als veraltet markieren. Bei einem Befehl im Status „Veraltet“ werden bestehende Befehlsausführungen bis zum Abschluss ausgeführt. Sie können den Befehl nicht aktualisieren oder neue Ausführungen ausführen. Um neue Ausführungen zu senden, müssen Sie den Befehl wiederherstellen, damit er verfügbar wird.

Sie können einen Befehl auch löschen, wenn er nicht mehr benötigt wird. Wenn Sie einen Befehl zum Löschen markieren und der Befehl für eine Dauer, die länger als das maximale Timeout von 24 Stunden ist, als veraltet gilt, wird der Befehl sofort gelöscht. Wenn der Befehl nicht veraltet ist oder für eine Dauer, die kürzer als das maximale Timeout ist, als veraltet gilt, befindet sich der Befehl im Status Ausstehende Löschung. Der Befehl wird nach 24 Stunden automatisch aus Ihrem Konto entfernt.

Parameter

Beim Erstellen eines Befehls können Sie optional die Parameter angeben, die das Zielfahrzeug bei der Ausführung des Befehls ausführen soll. Der Befehl, den Sie erstellen, ist eine wiederverwendbare Konfiguration und kann verwendet werden, um mehrere Befehlsausführungen an Ihr Fahrzeug zu senden und diese gleichzeitig auszuführen. Alternativ können Sie die Parameter auch nur zur Laufzeit angeben und einen einmaligen Vorgang ausführen, bei dem ein Befehl erstellt und an Ihr Fahrzeug gesendet wird.

Zielfahrzeug

Wenn Sie den Befehl ausführen möchten, müssen Sie ein Zielfahrzeug angeben, das den Befehl empfängt und bestimmte Aktionen ausführt. Das Zielfahrzeug muss bereits als Ding bei registriert worden sein AWS IoT. Nachdem Sie den Befehl an das Fahrzeug gesendet haben, beginnt es mit der Ausführung einer Befehlsinstanz, die auf den von Ihnen angegebenen Parametern und Werten basiert.

Aktuatoren

Wenn Sie den Befehl ausführen möchten, müssen Sie die Aktuatoren am Fahrzeug angeben, die den Befehl erhalten sollen, und deren Werte, die die auszuführenden Aktionen bestimmen. Sie können optional Standardwerte für die Aktuatoren konfigurieren, um das Senden ungenauer Befehle zu vermeiden. Sie können beispielsweise den Standardwert für einen Türverriegelungsaktuator verwenden, damit der Befehl die Türen nicht versehentlich entriegelt. `LockDoor` Allgemeine Informationen zu Aktuatoren finden Sie unterDie wichtigsten Konzepte.

Unterstützung von Datentypen

Die folgenden Datentypen werden für die Aktuatoren unterstützt, die für die Befehlsfunktion verwendet werden.

> **Note**
>
> Arrays werden für Telematikdaten, Befehle oder LKS (Last Known State) nicht unterstützt. Sie können den Array-Datentyp nur für Bildverarbeitungssystemdaten verwenden.

- Fließkommatypen. Die folgenden Typen werden unterstützt.
  - Float (32 Bit)
  - Doppelt (64 Bit)

- Ganzzahl (sowohl mit als auch ohne Vorzeichen). Die folgenden Integer-Typen werden unterstützt.
  - int8 und uint8
  - int16 und uint16
  - int32 und uint32
- Lang. Die folgenden langen Typen werden unterstützt.
  - Lang (int64)
  - Long ohne Vorzeichen (uint64)
- Zeichenfolge
- Boolesch

Ausführung von Befehlen

Eine Befehlsausführung ist eine Instanz eines Befehls, der auf einem Zielgerät ausgeführt wird. Das Fahrzeug führt den Befehl entweder mit den Parametern aus, die Sie bei der Erstellung des Befehls oder beim Start der Befehlsausführung angegeben haben. Das Fahrzeug führt dann die angegebenen Operationen aus und gibt den Status der Ausführung zurück.

> **Note**
>
> Für ein bestimmtes Fahrzeug können Sie mehrere Befehle gleichzeitig ausführen. Informationen zur maximalen Anzahl gleichzeitiger Ausführungen, die Sie für jedes Fahrzeug ausführen können, finden Sie unter AWS IoT Device Management Befehlsquoten.

Vorlagen für den Status des letzten bekannten Zustands (LKS)

Anhand von Vorlagen für Bundesstaaten können Fahrzeughalter den Zustand ihres Fahrzeugs nachverfolgen. Um den letzten bekannten Zustand (LKS) Ihrer Fahrzeuge nahezu in Echtzeit zu überwachen, können Sie Statusvorlagen erstellen und diese Ihren Fahrzeugen zuordnen.

Mithilfe der Befehlsfunktion können Sie „On Demand“ -Operationen ausführen, die für die Erfassung und Verarbeitung von Zustandsdaten verwendet werden können. Sie können beispielsweise einmalig den aktuellen Fahrzeugstatus anfordern (Abrufen) oder zuvor bereitgestellte LKS-Statusvorlagen aktivieren oder deaktivieren, um die Meldung von Fahrzeugdaten zu starten oder zu beenden. Beispiele, die zeigen, wie Befehle mit Statusvorlagen verwendet werden, finden Sie unter. Szenarien zur Verwendung von Befehlen

Status der Befehlsausführung

Nachdem Sie mit der Befehlsausführung begonnen haben, kann Ihr Fahrzeug den Status der Ausführung veröffentlichen und die Gründe für den Status als zusätzliche Informationen zur Ausführung angeben. In den folgenden Abschnitten werden die verschiedenen Status der Befehlsausführung und die Statuscodes beschrieben.

Themen

- Status der Befehlsausführung, Grund, Code und Beschreibung
- Status und Statuscodes der Befehlsausführung
- Status des Timeouts bei der Befehlsausführung

Status der Befehlsausführung, Grund, Code und Beschreibung

Um Aktualisierungen des Status der Befehlsausführung zu melden, können Ihre Fahrzeuge die `UpdateCommandExecution` API verwenden, um die aktualisierten Statusinformationen in der Cloud zu veröffentlichen. Verwenden Sie dazu die reservierten Themen für Befehle, die im AWS IoT Core Entwicklerhandbuch beschrieben sind. Wenn Sie die Statusinformationen melden, können Ihre Geräte zusätzlichen Kontext über den Status der einzelnen Befehlsausführungen mithilfe des `StatusReason` Objekts sowie über die Felder `reasonCode` bereitstellen`reasonDescription`, die im Objekt enthalten sind.

Status und Statuscodes der Befehlsausführung

Die folgende Tabelle zeigt die verschiedenen Statuscodes für die Befehlsausführung und die zulässigen Status, zu denen eine Befehlsausführung übergehen kann. Außerdem wird angezeigt, ob eine Befehlsausführung „terminal“ erfolgt (d. h., es stehen keine weiteren Statusaktualisierungen bevor), ob die Änderung vom Fahrzeug oder der Cloud initiiert wurde, und welche verschiedenen vordefinierten Statuscodes vorliegen und wie sie den von der Cloud gemeldeten Status zugeordnet sind.

- Informationen zur AWS IoT FleetWise Verwendung der vordefinierten Statuscodes und des `statusReason` Objekts finden Sie unter Befehlsstatus in der FleetWise Softwaredokumentation zu Edge Agent for AWS IoT.
- Weitere Informationen zu Ausführungen über Terminals und ohne Terminal sowie zu den Übergängen zwischen den Status finden Sie unter Status der Befehlsausführung im AWS IoT Core Entwicklerhandbuch.

Status und Quelle der Befehlsausführung

Status der Befehlsausführung	Description	Vom Gerät/der Cloud initiiert?	Ausführung im Terminal?	Zulässige Statusübergänge	Vordefinierte Statuscodes
`CREATED`	Wenn die API-Anfrage zum Starten der Ausführung des Befehls (`StartCommandExecution` API) erfolgreich ist, ändert sich der Status der Befehlsausführung auf`CREATED`.	Cloud	Nein	• `IN_PROGRESS` • `SUCCEEDED` • `FAILED` • `REJECTED` • `TIMED_OUT`	Keine
`IN_PROGRESS`	Wenn das Fahrzeug mit der Ausführung des Befehls beginnt, kann es im Antwortthema eine Nachricht veröffentlichen, in der der Status aktualisiert	Gerät	Nein	• IN_PROGRESS • SUCCEEDED • FEHLGESCHLAGEN • ABGELEHNT • TIMED_OUT	`COMMAND_STATUS_COMMAND_IN_PROGRESS`

Status der Befehlsausführung	Description	Vom Gerät/der Cloud initiiert?	Ausführung im Terminal?	Zulässige Statusübergänge	Vordefinierte Statuscodes
	wird`IN_PROGRESS` .				
`SUCCEEDED`	Wenn das Fahrzeug den Befehl erfolgreich verarbeitet und die Ausführung abgeschlossen hat, kann es eine Nachricht im Antwortthema veröffentlichen, um den Status zu aktualisieren. `SUCCEEDED`	Gerät	Ja	Nicht zutreffend	`COMMAND_STATUS_SUCCEEDED`

Status der Befehlsausführung	Description	Vom Gerät/der Cloud initiiert?	Ausführung im Terminal?	Zulässige Statusübergänge	Vordefinierte Statuscodes
`FAILED`	Wenn das Fahrzeug den Befehl nicht ausführen konnte, kann es eine Nachricht im Antwortthema veröffentlichen, um den Status zu aktualisieren `FAILED`.	Gerät	Ja	Nicht zutreffend	`COMMAND_STATUS_EXECUTION_FAILED`
`REJECTED`	Wenn das Fahrzeug den Befehl nicht akzeptiert, kann es eine Nachricht im Antwortthema veröffentlichen, in der der Status aktualisiert wird `REJECTED`.	Gerät	Ja	Nicht zutreffend	Keine

Status der Befehlsausführung	Description	Vom Gerät/der Cloud initiiert?	Ausführung im Terminal?	Zulässige Statusübergänge	Vordefinierte Statuscodes
`TIMED_OUT`	Der Status der Befehlsausführung kann sich `TIMED_OUT` aus einem der folgenden Gründe auf ändern. • Das Ergebnis der Befehlsausführung wurde nicht empfangen und die Cloud meldet automatisch einen `TIMED_OUT` Status. • Das Fahrzeug meldet, dass beim Versuch, den Befehl auszuführen, ein Timeout	Gerät und Cloud	Nein	• SUCCEEDED • FEHLGESCHLAGEN • ABGELEHNT • TIMED_OUT	`COMMAND_STATUS_EXECUTION_TIMEOUT`

Status der Befehlsausführung	Description	Vom Gerät/der Cloud initiiert?	Ausführung im Terminal?	Zulässige Statusübergänge	Vordefinierte Statuscodes
	aufgetreten ist. In diesem Fall wird die Befehlsausführung zum Terminal. Weitere Hinweise zu diesem Status finden Sie unterStatus des Timeouts bei der Befehlsausführung.				

Status des Timeouts bei der Befehlsausführung

Ein Timeout bei der Befehlsausführung kann sowohl von der Cloud als auch vom Gerät gemeldet werden. Nachdem der Befehl an das Gerät gesendet wurde, startet ein Timer. Wenn innerhalb der angegebenen Dauer keine Antwort vom Gerät eingegangen ist, meldet die Cloud einen `TIMED_OUT` Status. In diesem Fall erfolgt die Befehlsausführung im `TIMED_OUT` Status nicht über ein Terminal.

Das Gerät kann diesen Status durch einen Terminalstatus wie `SUCCEEDEDFAILED`, oder `REJECTED` ersetzen. Es kann auch melden, dass bei der Ausführung des Befehls ein Timeout aufgetreten ist. In diesem Fall bleibt der Status der Befehlsausführung unverändert, `TIMED_OUT` aber die Felder des `StatusReason` Objekts werden auf der Grundlage der vom Gerät gemeldeten Informationen aktualisiert. Die Befehlsausführung im `TIMED_OUT` Status wird jetzt zum Terminal.

Weitere Informationen finden Sie im AWS IoT Core Entwicklerhandbuch unter Überlegungen zum Timeout bei der Befehlsausführung.

Fahrzeuge und Kommandos

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.
> Sie sind allein dafür verantwortlich, Befehle auf sichere und gesetzeskonforme Weise bereitzustellen.

So verwenden Sie die Befehlsfunktion:

1. Erstellen Sie zunächst eine Befehlsressource. Geben Sie optional die Parameter an, die die Informationen enthalten, die zur Ausführung des Befehls erforderlich sind.
2. Geben Sie das Zielfahrzeug an, das den Befehl erhalten und die angegebenen Aktionen ausführen soll.
3. Jetzt können Sie den Befehl auf dem Zielgerät ausführen und die Details zur Befehlsausführung überprüfen, um den Status abzurufen und CloudWatch Protokolle zur weiteren Behebung von Problemen verwenden.

In den folgenden Abschnitten wird der Arbeitsablauf zwischen Fahrzeugen und Befehlen veranschaulicht.

Themen

- Workflow-Übersicht
- Arbeitsablauf bei Fahrzeugen
- Arbeitsablauf für Befehle
- (Optional) Befehle und Benachrichtigungen

Workflow-Übersicht

Die folgenden Schritte bieten einen Überblick über den Befehlsablauf zwischen Ihren Fahrzeugen und Befehlen. Wenn Sie einen der HTTP-API-Befehle verwenden, wird die Anfrage mit Sigv4-Anmeldeinformationen signiert.

> **Note**
>
> Mit Ausnahme der `StartCommandExecution` API-Operation verwenden alle Operationen, die über das HTTP-Protokoll ausgeführt werden, den Endpunkt der Steuerungsebene.

1. Stellen Sie eine MQTT-Verbindung her und abonnieren Sie die Themen zu Befehlen

   Um sich auf den Befehls-Workflow vorzubereiten, müssen die Geräte eine MQTT-Verbindung mit dem `iot:Data-ATS` Endpunkt aufbauen und das oben erwähnte Thema zur Befehlsanforderung abonnieren. Optional können Ihre Geräte auch die Antwortthemen „Akzeptierte Befehle“ und „Abgelehnte Befehle“ abonnieren.

2. Erstellen Sie ein Fahrzeugmodell und eine Befehlsressource

   Sie können jetzt mithilfe der API-Operationen und der `CreateCommand` Steuerungsebene ein Fahrzeug `CreateVehicle` und eine Befehlsressource erstellen. Die Befehlsressource enthält die Konfigurationen, die angewendet werden sollen, wenn der Befehl auf dem Fahrzeug ausgeführt wird.

3. Starten Sie die Befehlsausführung auf dem Zielgerät

   Starten Sie die Befehlsausführung auf dem Fahrzeug mithilfe der `StartCommandExecution` Datenebenen-API mit Ihrem kontospezifischen Endpunkt`iot:Jobs`. Die API veröffentlicht eine Protobuf-kodierte Nutzdatennachricht zum Thema der Befehlsanfrage.

4. Aktualisieren Sie das Ergebnis der Befehlsausführung

   Das Fahrzeug verarbeitet den Befehl und die empfangene Nutzlast und veröffentlicht dann das Ergebnis der Befehlsausführung mithilfe der `UpdateCommandExecution` API im Antwortthema. Wenn Ihr Fahrzeug die Antwortthemen akzeptierte und abgelehnte Befehle abonniert hat, erhält es eine Meldung, die angibt, ob die Antwort vom Cloud-Dienst akzeptiert oder abgelehnt wurde.

5. (Optional) Rufen Sie das Ergebnis der Befehlsausführung ab

   Um das Ergebnis der Befehlsausführung abzurufen, können Sie den API-Vorgang der `GetCommandExecution` Steuerungsebene verwenden. Nachdem Ihr Fahrzeug das Ergebnis der Befehlsausführung im Antwortthema veröffentlicht hat, gibt diese API die aktualisierten Informationen zurück.

6. (Optional) Abonnieren und verwalten Sie Befehlsereignisse

   Um Benachrichtigungen über den Status der Befehlsausführung zu erhalten, können Sie das Thema Befehlsereignisse abonnieren. Anschließend können Sie die API der `CreateTopicRule` Steuerungsebene verwenden, um die Daten der Befehlsereignisse an andere Anwendungen wie AWS Lambda Funktionen oder Amazon SQS weiterzuleiten und darauf aufbauend Anwendungen zu erstellen.

Arbeitsablauf bei Fahrzeugen

In den folgenden Schritten wird der Fahrzeug-Arbeitsablauf bei Verwendung der Befehlsfunktion detailliert beschrieben.

> **Note**
>
> Die in diesem Abschnitt beschriebenen Operationen verwenden das MQTT-Protokoll.

1. Stellen Sie eine MQTT-Verbindung her

   Um Ihre Fahrzeuge auf die Verwendung der Befehlsfunktion vorzubereiten, müssen sie sich zunächst mit dem AWS IoT Core Message Broker verbinden. Ihr Fahrzeug muss in der Lage sein, die `iot:Connect` Aktion auszuführen, um eine Verbindung zum Message Broker herzustellen AWS IoT Core und eine MQTT-Verbindung mit diesem herzustellen. Verwenden Sie die `DescribeEndpoint` API oder den `describe-endpoint` CLI-Befehl AWS-Konto, um den Datenebenen-Endpunkt für Ihren zu finden, wie unten gezeigt.

   ```
   aws iot describe-endpoint --endpoint-type iot:Data-ATS
   ```

   Wenn Sie diesen Befehl ausführen, wird der kontospezifische Endpunkt der Datenebene zurückgegeben, wie unten gezeigt.

```
account-specific-prefix.iot.region.amazonaws.com
```

2. Abonnieren Sie das Thema „Befehle anfordern“

   Nachdem eine Verbindung hergestellt wurde, können Ihre Geräte das Thema AWS IoT Commands MQTT Request abonnieren. Wenn Sie einen Befehl erstellen und die Befehlsausführung auf Ihrem Zielgerät starten, wird eine Protobuf-kodierte Nutzdatennachricht vom Message Broker unter dem Thema der Anfrage veröffentlicht. Ihr Gerät kann dann die Payload-Nachricht empfangen und den Befehl verarbeiten. In diesem Beispiel *<DeviceID>* ersetzen Sie es durch die eindeutige Kennung Ihres Zielfahrzeugs. Diese ID kann die eindeutige Kennung Ihres Fahrzeugs oder ein Objektname sein

   > **Note**
   >
   > Die Nutzdatennachricht, die an das Gerät gesendet wird, muss das Protobuf-Format verwenden.

   ```
   $aws/commands/things/<DeviceID>/executions/+/request/protobuf
   ```

3. (Optional) Abonnieren Sie Themen zur Befehlsantwort

   Optional können Sie diese Themen zur Befehlsantwort abonnieren, um eine Meldung zu erhalten, die angibt, ob der Cloud-Dienst die Antwort des Geräts akzeptiert oder abgelehnt hat.

   > **Note**
   >
   > Es ist optional, dass Ihre Fahrzeuge die Themen `/accepted` und `/rejected` Antworten abonnieren. Ihre Fahrzeuge erhalten diese Antwortnachrichten automatisch, auch wenn sie diese Themen nicht ausdrücklich abonniert haben.

   ```
   $aws/commands/things/<DeviceID>/executions/<ExecutionId>/response/protobuf/accepted
   $aws/commands/things/<DeviceID>/executions/<ExecutionId>/response/protobuf/rejected
   ```

4. Aktualisieren des Ergebnisses einer Befehlsausführung

   Das Zielfahrzeug verarbeitet dann den Befehl. Anschließend verwendet es die `UpdateCommandExecution` API, um das Ergebnis der Ausführung im folgenden MQTT-Antwortthema zu veröffentlichen.

   > **Note**
   >
   > Für ein bestimmtes Fahrzeug und die Ausführung eines bestimmten Befehls *<DeviceID>* müssen sie mit dem entsprechenden Feld im Anforderungsthema übereinstimmen, das das Gerät abonniert hat.

   ```
   $aws/commands/things/<DeviceID>/executions/<ExecutionId>/response/protobuf
   ```

   Die `UpdateCommandExecution` API ist ein API-Vorgang auf Datenebene über MQTT, der mit TLS authentifiziert ist.

   - Wenn der Cloud-Dienst das Ergebnis der Befehlsausführung erfolgreich verarbeitet hat, wird eine Nachricht im Thema MQTT Accepted veröffentlicht. Das akzeptierte Thema verwendet das folgende Format.

     ```
     $aws/commands/things/<DeviceID>/executions/<ExecutionId>/response/protobuf/
     accepted
     ```

   - Wenn der Cloud-Dienst das Ergebnis der Befehlsausführung nicht verarbeiten konnte, wird eine Antwort auf das abgelehnte MQTT-Thema veröffentlicht. Das abgelehnte Thema verwendet das folgende Format.

     ```
     $aws/commands/things/<DeviceID>/executions/<ExecutionId>/response/protobuf/
     rejected
     ```

   Weitere Informationen zu dieser API und ein Beispiel finden Sie unterDas Ergebnis der Befehlsausführung aktualisieren.

Arbeitsablauf für Befehle

In den folgenden Schritten wird der Befehlsablauf detailliert beschrieben.

> **Note**
>
> Die in diesem Abschnitt beschriebenen Operationen verwenden das HTTP-Protokoll.

1. Registrieren Sie Ihr Fahrzeug

   Nachdem Sie Ihr Fahrzeug für die Verwendung der Befehlsfunktion vorbereitet haben, können Sie Ihren Antrag vorbereiten, indem Sie Ihr Fahrzeug registrieren und dann einen Befehl erstellen, der an das Fahrzeug gesendet wird. Um das Fahrzeug zu registrieren, erstellen Sie mithilfe der API-Operation für die `CreateVehicle`Steuerungsebene eine Instanz eines Fahrzeugmodells (Modellmanifest). Weitere Informationen und Beispiele finden Sie unter Fahrzeug erstellen.

2. Erstellen Sie einen Befehl

   Verwenden Sie den API-Vorgang der `CreateCommand`HTTP-Steuerungsebene, um Befehle zu modellieren, die für das Fahrzeug gelten, auf das Sie abzielen. Geben Sie alle Parameter und Standardwerte an, die bei der Ausführung des Befehls verwendet werden sollen, und stellen Sie sicher, dass der `AWS-IoT-FleetWise` Namespace verwendet wird. Weitere Informationen und Beispiele für die Verwendung dieser API finden Sie unterErstellen Sie eine Befehlsressource.

3. Starten Sie die Befehlsausführung

   Sie können jetzt den Befehl, den Sie auf dem Fahrzeug erstellt haben, mithilfe der API-Operation für die `StartCommandExecution`Datenebene ausführen. AWS IoT Device Management ruft den Befehl und die Befehlsparameter ab und validiert die eingehende Anfrage. Anschließend ruft es die AWS IoT FleetWise API mit den erforderlichen Parametern auf, um die fahrzeugspezifische Nutzlast zu generieren. Die Nutzdaten werden dann AWS IoT Device Management über MQTT an das Befehlsanforderungsthema, das Ihr Gerät abonniert hat, an das Gerät gesendet. Weitere Informationen und Beispiele für die Verwendung dieser API finden Sie unter. Senden Sie einen Befehl ()AWS CLI

   ```
   $aws/commands/things/<DeviceID>/executions/+/request/protobuf
   ```

> **Note**
>
> Wenn das Gerät offline war, als der Befehl aus der Cloud gesendet wurde und persistente MQTT-Sitzungen verwendet werden, wartet der Befehl beim Message Broker. Wenn das Gerät vor Ablauf des Timeouts wieder online ist und das Thema für die Befehlsanfrage abonniert hat, kann das Gerät den Befehl verarbeiten und das Ergebnis im Antwortthema veröffentlichen. Wenn das Gerät vor Ablauf des Timeouts nicht wieder online ist, kommt es bei der Befehlsausführung zu einem Timeout und die Payload-Meldung läuft ab.

4. Rufen Sie die Befehlsausführung ab

   Nachdem Sie den Befehl auf dem Gerät ausgeführt haben, verwenden Sie den API-Vorgang der `GetCommandExecution`Steuerungsebene, um das Ergebnis der Befehlsausführung abzurufen und zu überwachen. Sie können die API auch verwenden, um zusätzliche Informationen zu den Ausführungsdaten abzurufen, z. B. wann sie zuletzt aktualisiert wurden, wann die Ausführung abgeschlossen wurde und welche Parameter angegeben wurden.

   > **Note**
   >
   > Um die aktuellsten Statusinformationen abzurufen, muss Ihr Gerät das Ergebnis der Befehlsausführung im Antwortthema veröffentlicht haben.

   Weitere Informationen und Beispiele für die Verwendung dieser API finden Sie unterHolen Sie sich die Befehlsausführung.

(Optional) Befehle und Benachrichtigungen

Sie können Befehlsereignisse abonnieren, um Benachrichtigungen zu erhalten, wenn sich der Status der Befehlsausführung ändert. In den folgenden Schritten erfahren Sie, wie Sie Befehlsereignisse abonnieren und anschließend verarbeiten.

1. Erstellen einer Themenregel

   Sie können das Thema Befehlsereignisse abonnieren und Benachrichtigungen erhalten, wenn sich der Status der Befehlsausführung ändert. Sie können auch eine Themenregel erstellen, um

die vom Fahrzeug verarbeiteten Daten an andere Anwendungen wie AWS Lambda Funktionen weiterzuleiten. Sie können eine Themenregel entweder mithilfe der AWS IoT Konsole oder mithilfe der API-Operation auf der `CreateTopicRule` AWS IoT Core Steuerungsebene erstellen. Weitere Informationen finden Sie unter AWS IoT Regel erstellen.

Ersetzen Sie in diesem Beispiel *<CommandID>* durch die ID des Befehls, für den Sie Benachrichtigungen erhalten möchten, und *<CommandExecutionStatus>* durch den Status der Befehlsausführung.

```
$aws/events/commandExecution/<CommandID>/<CommandExecutionStatus>
```

> **Note**
>
> Um Benachrichtigungen für alle Befehle und den Status der Befehlsausführung zu erhalten, können Sie Platzhalterzeichen verwenden und das folgende Thema abonnieren.

```
$aws/events/commandExecution/+/#
```

2. Befehlsereignisse empfangen und verarbeiten

   Wenn Sie im vorherigen Schritt eine Themenregel zum Abonnieren von Befehlsereignissen erstellt haben, können Sie die Push-Benachrichtigungen für Befehle verwalten, die Sie erhalten. Sie können optional auch darauf aufbauende Anwendungen erstellen, z. B. mit Amazon SQS AWS Lambda, Amazon SNS oder AWS Step Functions, indem Sie die Themenregel verwenden, die Sie erstellt haben.

Der folgende Code zeigt eine Beispiel-Payload für die Befehle, Ereignisse und Benachrichtigungen, die Sie erhalten werden.

```
{
    "executionId": "2bd65c51-4cfd-49e4-9310-d5cbfdbc8554",
    "status":"FAILED",
    "statusReason": {
         "reasonCode": "4",
         "reasonDescription": ""
    },
```

```
    "eventType": "COMMAND_EXECUTION",
    "commandArn":"arn:aws:iot:us-east-1:123456789012:command/0b9d9ddf-
e873-43a9-8e2c-9fe004a90086",
    "targetArn":"arn:aws:iot:us-east-1:123456789012:thing/5006c3fc-
de96-4def-8427-7eee36c6f2bd",
    "timestamp":1717708862107
}
```

Erstellen und Verwalten von Befehlen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können wiederverwendbare Fernaktionen konfigurieren oder einmalige, sofortige Anweisungen an Ihre Geräte senden. Wenn Sie diese Funktion verwenden, können Sie die Anweisungen angeben, die Ihre Geräte nahezu in Echtzeit ausführen können. Mit einem Befehl können Sie wiederverwendbare Fernaktionen für Ihr Zielfahrzeug konfigurieren. Nachdem Sie einen Befehl erstellt haben, können Sie eine Befehlsausführung starten, die auf ein bestimmtes Fahrzeug abzielt.

In diesem Thema wird gezeigt, wie Sie eine Befehlsressource mithilfe der AWS IoT Core API oder der erstellen und verwalten können AWS CLI. Es zeigt Ihnen, wie Sie die folgenden Aktionen auf einer Befehlsressource ausführen.

Themen

- Erstellen Sie eine Befehlsressource
- Ruft Informationen zu einem Befehl ab
- Befehle in Ihrem Konto auflisten
- Eine Befehlsressource aktualisieren oder als veraltet kennzeichnen
- Löschen Sie eine Befehlsressource

Erstellen Sie eine Befehlsressource

Sie können den API-Betrieb der `CreateCommand` AWS IoT Core Steuerungsebene oder die AWS FleetWise IoT-Konsole verwenden, um einen Befehl zu erstellen.

Erstellen Sie einen Befehl (Konsole)

Sie können die AWS FleetWise IoT-Konsole verwenden, um einen Befehl zu erstellen.

Um einen Befehl zu erstellen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Befehle aus.
3. Wählen Sie Befehl erstellen aus.
4. Geben Sie eine eindeutige Befehls-ID an, anhand derer Sie den Befehl identifizieren können, den Sie auf dem Fahrzeug ausführen möchten.
5. (Optional) Geben Sie optional einen Anzeigenamen und eine Beschreibung an.
6. (Optional) Wählen Sie den Aktuator und den Standardparameterwert aus. Parameter geben an, welche Aktionen das Zielfahrzeug nach Erhalt des Befehls ausführen kann. Wenn Sie keine Parameter hinzufügen, müssen Sie diese angeben, wenn Sie den Befehl ausführen.
7. Wählen Sie eine IAM-Rolle aus, die Berechtigungen zum Generieren der Nutzdaten für Befehle gewährt. Siehe Zugriff kontrollieren.
8. Wählen Sie den Befehl Erstellen.

Erstellen Sie einen Befehl (AWS CLI)

Das folgende Beispiel zeigt, wie ein Befehl mit einem Parameter erstellt wird.

Überlegungen beim Erstellen eines Befehls

Wenn Sie einen Befehl erstellen in AWS IoT FleetWise:

- Sie müssen den angeben`roleArn`, der die Erlaubnis erteilt, Befehle auf Ihrem Fahrzeug zu erstellen und auszuführen. Weitere Informationen und Beispielrichtlinien, z. B. zur Aktivierung von KMS-Schlüsseln, finden Sie unterErteilen Sie die AWS IoT Device Management Berechtigung zum Generieren der Payload für Befehle mit AWS IoT FleetWise.
- Sie müssen `AWS-IoT-FleetWise` als Namespace angeben.

- Sie können das `mandatory-parameters` Feld überspringen und sie stattdessen zur Laufzeit angeben. Alternativ können Sie einen Befehl mit Parametern erstellen und optional Standardwerte für diese angeben. Wenn Sie Standardwerte angegeben haben, können Sie diese Werte zur Laufzeit verwenden oder sie überschreiben, indem Sie Ihre eigenen Werte angeben. Diese zusätzlichen Beispiele finden Sie unterSzenarien zur Verwendung von Befehlen.
- Sie können bis zu drei Name-Wert-Paare für das `mandatory-parameters` Feld angeben. Bei der Ausführung des Befehls auf dem Fahrzeug wird jedoch nur ein Name-Wert-Paar akzeptiert, und das `name` Feld muss den vollqualifizierten Namen mit dem Präfix verwenden. `$actuatorPath.`

- *`command-id`*Ersetzen Sie den Befehl durch einen eindeutigen Bezeichner. Sie können UUID, alphanumerische Zeichen, „-“ und „_“ verwenden.
- *`role-arn`*Ersetzen Sie es durch die IAM-Rolle, die Ihnen die Berechtigung erteilt, Befehle zu erstellen und auszuführen, z. B. `"arn:aws:iam:`*`accountId`*`:role/`*`FwCommandExecutionRole`*`"`
- (Optional) *`display-name`* Ersetzen Sie den Befehl durch einen benutzerfreundlichen Namen und *`description`* eine aussagekräftige Beschreibung des Befehls.
- Ersetzen Sie *`name`* und *`value`* des `mandatory-parameters` Objekts durch die erforderlichen Informationen für den Befehl, der gerade erstellt wird. Das `name` Feld ist der vollqualifizierte Name, wie er im Signalkatalog definiert ist, mit `$actuatorPath.` dem Präfix. Kann *`$actuatorPath.Vehicle.Chassis.SteeringWheel.HandsOff.HandsOffSteeringMode`* und `name` `value` kann beispielsweise ein boolescher Wert sein, der einen Lenkmodusstatus wie anzeigt. *`{"B": false}`*

```
aws iot create-command --command-id command-id \
    --role-arn role-arn \
    --description description \
    --display-name display-name \
    --namespace "AWS-IoT-FleetWise" \
    --mandatory-parameters '[
        {
            "name": name,
            "value": value
        }
   ]'
```

Die `CreateCommand` API-Operation gibt eine Antwort zurück, die die ID und den ARN (Amazon Resource Name) des Befehls enthält.

```
{
    "commandId": "HandsOffSteeringMode",
    "commandArn": "arn:aws:iot:ap-south-1:123456789012:command/HandsOffSteeringMode"
}
```

Ruft Informationen zu einem Befehl ab

Sie können den API-Vorgang der `GetCommand` AWS IoT Core Steuerungsebene verwenden, um Informationen über eine Befehlsressource abzurufen.

Führen Sie den folgenden Befehl aus, um Informationen zu einer Befehlsressource abzurufen. *`command-id`*Ersetzen Sie ihn durch den Bezeichner, der bei der Erstellung des Befehls verwendet wurde.

```
aws iot get-command --command-id command-id
```

Die `GetCommand` API-Operation gibt eine Antwort zurück, die die folgenden Informationen enthält.

- Die ID und der ARN (Amazon Resource Name) des Befehls.
- Datum und Uhrzeit der Erstellung und letzten Aktualisierung des Befehls.
- Der Befehlsstatus, der angibt, ob er für die Ausführung auf dem Fahrzeug verfügbar ist.
- Alle Parameter, die Sie bei der Erstellung des Befehls angegeben haben.

```
{
    "commandId": "HandsOffSteeringMode",
    "commandArn": "arn:aws:iot:ap-south-1:123456789012:command/HandsOffSteeringMode"",
    "namespace": "AWS-IoT-FleetWise",
    "mandatoryParameters":[
        {
            "name":
 "$actuatorPath.Vehicle.Chassis.SteeringWheel.HandsOff.HandsOffSteeringMode",
            "value": {"B": false }
        }
    ],
    "createdAt": "2024-03-23T11:24:14.919000-07:00",
```

```
    "lastUpdatedAt": "2024-03-23T11:24:14.919000-07:00",
    "deprecated": false,
    "pendingDeletion": false
}
```

Befehle in Ihrem Konto auflisten

Sie können den API-Vorgang der `ListCommands` AWS IoT Core Steuerungsebene verwenden, um alle Befehle in Ihrem Konto aufzulisten, die Sie erstellt haben.

Führen Sie den folgenden Befehl aus, um Befehle in Ihrem Konto aufzulisten. Standardmäßig gibt die API Befehle zurück, die für beide Namespaces erstellt wurden. Um die Liste so zu filtern, dass nur Befehle angezeigt werden, für die erstellt wurden AWS IoT FleetWise, führen Sie den folgenden Befehl aus.

> **Note**
>
> Sie können die Liste auch in aufsteigender oder absteigender Reihenfolge sortieren oder die Liste so filtern, dass nur Befehle angezeigt werden, die einen bestimmten Befehlsparameternamen haben.

```
aws iot list-commands --namespace "AWS-IoT-FleetWise"
```

Der `ListCommands` API-Vorgang gibt eine Antwort zurück, die die folgenden Informationen enthält.

- Die ID und der ARN (Amazon Resource Name) der Befehle.
- Datum und Uhrzeit der Erstellung und letzten Aktualisierung des Befehls.
- Der Befehlsstatus, der angibt, ob die Befehle für die Ausführung auf dem Fahrzeug verfügbar sind.

Eine Befehlsressource aktualisieren oder als veraltet kennzeichnen

Sie können den API-Betrieb der `UpdateCommand` AWS IoT Core Steuerungsebene oder die AWS FleetWise IoT-Konsole verwenden, um eine Befehlsressource zu aktualisieren. Sie können den Anzeigenamen und die Beschreibung eines Befehls aktualisieren. Sie können einen Befehl auch als veraltet markieren, wenn er derzeit nicht verwendet wird.

> **Note**
>
> Sie können die Namespace-Informationen oder die Parameter, die bei der Ausführung des Befehls verwendet werden sollen, nicht ändern.

Aktualisieren Sie einen Befehl (Konsole)

Einen Befehl aktualisieren

Um einen Befehl von der Konsole aus zu aktualisieren, rufen Sie die Befehlsseite der AWS FleetWise IoT-Konsole auf und führen Sie die folgenden Schritte aus.

1. Wählen Sie den Befehl aus, den Sie aktualisieren möchten, und wählen Sie dann Bearbeiten.
2. Bearbeiten Sie die Befehlsdetails und wählen Sie dann Änderungen speichern.

Verwerfen Sie einen Befehl als veraltet

Um einen Befehl aus der Konsole abzulehnen, rufen Sie die Befehlsseite der AWS FleetWise IoT-Konsole auf und führen Sie die folgenden Schritte aus.

1. Wählen Sie den Befehl aus, den Sie verwerfen möchten, und wählen Sie dann Veraltet aus.
2. Bestätigen Sie die veraltete Version und wählen Sie dann Veraltet aus.

Aktualisieren Sie einen Befehl ()AWS CLI

Einen Befehl aktualisieren

Führen Sie den folgenden Befehl aus, um eine Befehlsressource zu aktualisieren. *command-id*Ersetzen Sie ihn durch den Bezeichner des Befehls, den Sie aktualisieren möchten, und geben Sie das aktualisierte *display-name* und ein*description*.

```
aws iot update-command \
    --command-id command-id \
    --display-name display-name \
    --description description
```

Die `UpdateCommand` API-Operation gibt die folgende Antwort zurück.

```
{
```

```
    "commandId": "HandsOffSteeringMode",
    "deprecated": false,
    "lastUpdatedAt": "2024-05-09T23:16:51.370000-07:00"
}
```

Verwerfen Sie einen Befehl

Sie missbilligen einen Befehl, wenn Sie ihn nicht mehr für Ihr Gerät verwenden möchten oder wenn er veraltet ist. Das folgende Beispiel zeigt, wie ein Befehl als veraltet markiert wird.

```
aws iot update-command \
    --command-id command-id \
    --deprecated
```

Die `UpdateCommand` API-Operation gibt eine Antwort zurück, die die ID und den ARN (Amazon Resource Name) des Befehls enthält.

```
{
    "commandId": "HandsOffSteeringMode",
    "deprecated": true,
    "lastUpdatedAt": "2024-05-09T23:16:51.370000-07:00"
}
```

Sobald ein Befehl veraltet ist, werden bestehende Befehlsausführungen auf dem Fahrzeug weiterlaufen, bis sie zum Terminal werden. Um neue Befehlsausführungen auszuführen, müssen Sie die `UpdateCommand` API verwenden, um den Befehl wiederherzustellen, sodass er verfügbar wird. Weitere Informationen zum Verwerfen und Wiederherstellen eines Befehls sowie zu diesbezüglichen Überlegungen finden Sie unter Eine Befehlsressource als veraltet markieren im Entwicklerhandbuch.AWS IoT Core

Löschen Sie eine Befehlsressource

Sie können den API-Betrieb der `DeleteCommand` AWS IoT Core Steuerungsebene oder die AWS FleetWise IoT-Konsole verwenden, um eine Befehlsressource zu löschen.

> **Note**
>
> Löschvorgänge sind dauerhaft und können nicht rückgängig gemacht werden. Der Befehl wird dauerhaft aus Ihrem Konto entfernt.

Lösche einen Befehl (Konsole)

Um einen Befehl aus der Konsole zu löschen, rufen Sie die Befehlsseite der AWS FleetWise IoT-Konsole auf und führen Sie die folgenden Schritte aus.

1. Wählen Sie den Befehl aus, den Sie löschen möchten, und wählen Sie dann Löschen.
2. Bestätigen Sie, dass Sie den Befehl löschen möchten, und wählen Sie dann Löschen.

Löschen Sie einen Befehl (AWS CLI)

Führen Sie den folgenden Befehl aus, um eine Befehlsressource zu löschen. *command-id*Ersetzen Sie ihn durch den Bezeichner des Befehls, den Sie löschen möchten. Das folgende Beispiel zeigt, wie eine Befehlsressource gelöscht wird.

```
aws iot delete-command --command-id command-id
```

Wenn die Löschanforderung erfolgreich ist:

- Wenn der Befehl für eine Dauer, die länger als das maximale Timeout von 24 Stunden ist, als veraltet gilt, wird der Befehl sofort gelöscht und es wird der `statusCode` HTTP-Wert 204 angezeigt.
- Wenn der Befehl nicht veraltet ist oder für eine Dauer, die kürzer als das maximale Timeout ist, als veraltet gilt, befindet sich der Befehl in einem `pending deletion` Status und es wird ein HTTP-Wert von 202 angezeigt. `statusCode` Der Befehl wird nach Ablauf des maximalen Timeouts von 24 Stunden automatisch aus Ihrem Konto entfernt.

Starten und Überwachen von Befehlsausführungen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Nachdem Sie eine Befehlsressource erstellt haben, können Sie eine Befehlsausführung auf dem Zielfahrzeug starten. Sobald das Fahrzeug mit der Ausführung des Befehls beginnt, kann es damit

beginnen, das Ergebnis der Befehlsausführung zu aktualisieren und Statusaktualisierungen und Ergebnisinformationen in den reservierten MQTT-Themen zu veröffentlichen. Sie können dann den Status der Befehlsausführung abrufen und den Status der Ausführungen in Ihrem Konto überwachen.

In diesem Thema wird gezeigt, wie Sie mithilfe der AWS CLI oder der AWS FleetWise IoT-Konsole einen Befehl an Ihr Fahrzeug senden können. Außerdem erfahren Sie, wie Sie den Status der Befehlsausführung überwachen und aktualisieren können.

Themen

- Das Ergebnis der Befehlsausführung aktualisieren
- Holen Sie sich die Befehlsausführung
- Listet die Befehlsausführungen in Ihrem Konto auf
- Löscht die Ausführung eines Befehls

Senden Sie einen Befehl (Konsole)

Um einen Befehl von der Konsole aus zu senden, rufen Sie die Seite Fahrzeuge der AWS FleetWise IoT-Konsole auf und führen Sie die folgenden Schritte aus.

1. Wählen Sie das Fahrzeug aus, an das Sie einen Befehl senden möchten.
2. Wählen Sie Befehl ausführen aus.
3. Wählen Sie die Befehls-ID aus.
4. Geben Sie das Zeitlimit für die Befehlsausführung an und wählen Sie dann Befehl ausführen aus.

Senden Sie einen Befehl ()AWS CLI

Sie können den API-Vorgang für die `StartCommandExecution` AWS IoT Datenebene verwenden, um einen Befehl an ein Fahrzeug zu senden. Das Fahrzeug leitet den Befehl dann an einen Middleware-Dienst für die Automobilindustrie weiter (z. B. SOME/IP (Scalable Service-Oriented Middleware over IP)) oder veröffentlicht ihn in einem Fahrzeugnetzwerk (z. B. eine CAN-Geräteschnittstelle (Controller Area Network)). Das folgende Beispiel verwendet die AWS CLI.

Themen

- Überlegungen beim Senden eines Befehls
- Rufen Sie den kontospezifischen Endpunkt der Datenebene ab
- Senden Sie ein Befehlsbeispiel

Überlegungen beim Senden eines Befehls

Wenn Sie die Ausführung eines Befehls starten in AWS IoT FleetWise:

- Sie müssen AWS IoT etwas für das Fahrzeug bereitstellen. Weitere Informationen finden Sie unter Bereitstellung von AWS FleetWise IoT-Fahrzeugen.
- Sie müssen bereits einen Befehl mit dem Namespace `AWS-IoT-FleetWise` als Namespace erstellt und einen angegeben haben, der Ihnen `role-Arn` die Erlaubnis erteilt, Befehle in AWS IoT FleetWise zu erstellen und auszuführen. Weitere Informationen finden Sie unter Erstellen Sie eine Befehlsressource.
- Sie können das `parameters` Feld überspringen, wenn Sie Standardwerte verwenden möchten, die bei der Erstellung des Befehls für die Parameter angegeben wurden. Wenn das bei der Erstellung `mandatory-parameters` nicht angegeben wurde oder wenn Sie Standardwerte überschreiben möchten, indem Sie Ihre eigenen Werte für die Parameter angeben, müssen Sie das `parameters` Feld angeben. Diese zusätzlichen Beispiele finden Sie unterSzenarien zur Verwendung von Befehlen.
- Sie können bis zu drei Name-Wert-Paare für das `mandatory-parameters` Feld angeben. Bei der Ausführung des Befehls auf dem Fahrzeug wird jedoch nur ein Name-Wert-Paar akzeptiert, und das `name` Feld muss den vollqualifizierten Namen mit dem Präfix verwenden. `$actuatorPath.`

Rufen Sie den kontospezifischen Endpunkt der Datenebene ab

Bevor Sie den API-Befehl ausführen, müssen Sie die kontospezifische Endpunkt-URL für den Endpunkt abrufen. `iot:Jobs` Wenn Sie z. B. den folgenden Befehl ausführen:

```
aws iot describe-endpoint --endpoint-type iot:Jobs
```

Es wird die kontospezifische Endpunkt-URL zurückgegeben, wie in der folgenden Beispielantwort gezeigt.

```
{
    "endpointAddress": "<account-specific-prefix>.jobs.iot.<region>.amazonaws.com"
}
```

Senden Sie ein Befehlsbeispiel

Um einen Befehl an ein Fahrzeug zu senden, führen Sie den folgenden Befehl aus.

- *command-arn*Ersetzen Sie durch den ARN für den Befehl, den Sie ausführen möchten. Sie können diese Informationen aus der Antwort des `create-command` CLI-Befehls abrufen.
- *target-arn*Ersetzen Sie es durch den ARN für das Zielgerät oder die AWS IoT Sache, für die Sie den Befehl ausführen möchten.

> **Note**
>
> Sie können den Ziel-ARN einer AWS IoT Sache (AWS FleetWise IoT-Fahrzeug) angeben. Dinggruppen und Flotten werden derzeit nicht unterstützt.

- *endpoint-url*Ersetzen Sie es durch den kontospezifischen Endpunkt, den Sie in abgerufen habenRufen Sie den kontospezifischen Endpunkt der Datenebene ab, mit dem Präfix z. `https://` B. `https://`*`123456789012abcd`*`.jobs.iot.`*`ap-south-1`*`.amazonaws.com`
- Ersetzen Sie *name* und *value* durch das `mandatory-parameters` Feld, das Sie bei der Erstellung des Befehls mit der `create-command` CLI angegeben haben.

  Das `name` Feld ist der vollqualifizierte Name, wie er im Signalkatalog definiert ist, mit `$actuatorPath.` dem Präfix. Kann *`$actuatorPath.Vehicle.Chassis.SteeringWheel.HandsOff.HandsOffSteeringMode`* und `name` `value` kann beispielsweise ein boolescher Wert sein, der einen Lenkmodusstatus wie anzeigt. *`{"B": false}`*
- (Optional) Sie können auch einen zusätzlichen Parameter angeben,. `executionTimeoutSeconds` Dieses optionale Feld gibt die Zeit in Sekunden an, innerhalb derer das Gerät mit dem Ausführungsergebnis antworten muss. Sie können das Timeout auf einen Höchstwert von 24 Stunden konfigurieren.

  Wenn die Befehlsausführung erstellt wurde, wird ein Timer gestartet. Wenn sich der Status der Befehlsausführung vor Ablauf des Timers nicht in einen Status ändert, der ihn zum Terminal macht, z. B. `SUCCEEDED` oder`FAILED`, ändert sich der Status automatisch zu`TIMED_OUT`.

> **Note**
>
> Das Gerät kann auch einen `TIMED_OUT` Status melden oder diesen Status in einen Status wie`SUCCEEDED`, `FAILED` oder überschreiben`REJECTED`, und die Befehlsausführung wird zum Terminal. Weitere Informationen finden Sie unter Status des Timeouts bei der Befehlsausführung.

```
aws iot-jobs-data start-command-execution \
    --command-arn command-arn \
    --target-arn target-arn \
    --execution-timeout-seconds 30 \
    --endpoint-url endpoint-url \
    --parameters '[
        {
            "name": name,
            "value": value
        }
   ]'
```

Die `StartCommandExecution` API-Operation gibt eine Befehlsausführungs-ID zurück. Sie können diese ID verwenden, um den Status der Befehlsausführung, die Details und den Verlauf der Befehlsausführung abzufragen.

```
{
    "executionId": "07e4b780-7eca-4ffd-b772-b76358da5542"
 }
```

Nachdem Sie den Befehl ausgeführt haben, erhalten Ihre Geräte eine Benachrichtigung mit den folgenden Informationen. Das `issued_timestamp_ms` Feld entspricht der Uhrzeit, zu der die `StartCommandExecution` API aufgerufen wurde. Das `timeout_ms` entspricht dem Timeout-Wert, der beim Aufrufen der API mithilfe des `executionTimeoutSeconds` Parameters konfiguriert wurde. `StartCommandExecution`

```
timeout_ms: 9000000
issued_timestamp_ms: 1723847831317
```

Das Ergebnis der Befehlsausführung aktualisieren

Um den Status der Befehlsausführung zu aktualisieren, muss Ihr Gerät eine MQTT-Verbindung hergestellt und das folgende Thema zur Befehlsanfrage abonniert haben.

Ersetzen Sie in diesem Beispiel durch die eindeutige Kennung Ihres Zielgeräts, *<device-id>* bei der es sich um den Namen `VehicleId` oder den Namen der Sache handeln kann, und *<execution-id>* durch die Kennung für die Befehlsausführung.

> **Note**
>
> - Die Payload muss das Protobuf-Format verwenden.
> - Es ist für Ihre Geräte optional, die Themen `/accepted` und `/rejected` Antworten zu abonnieren. Ihre Geräte erhalten diese Antwortnachrichten auch dann, wenn sie sie nicht ausdrücklich abonniert haben.

```
// Request topic
$aws/devices/<DeviceID>/command_executions/+/request/protobuf

// Response topics (Optional)
$aws/devices/<DeviceID>/command_executions/<ExecutionId>/response/accepted/protobuf
$aws/devices/<DeviceID>/command_executions/<ExecutionId>/response/rejected/protobuf
```

Ihr Gerät kann eine Nachricht zum Thema Befehlsantwort veröffentlichen. Nach der Verarbeitung des Befehls wird eine protobuf-kodierte Antwort zu diesem Thema gesendet. Das *<DeviceID>* Feld muss mit dem entsprechenden Feld im Anforderungsthema übereinstimmen.

```
$aws/devices/<DeviceID>/command_executions/<ExecutionId>/response/<PayloadFormat>
```

Nachdem Ihr Gerät eine Antwort zu diesem Thema veröffentlicht hat, können Sie die aktualisierten Statusinformationen mithilfe der `GetCommandExecution` API abrufen. Der Status der Befehlsausführung kann einer der hier aufgeführten sein.

- `IN_PROGRESS`
- `SUCCEEDED`
- `FAILED`
- `REJECTED`
- `TIMED_OUT`

Beachten Sie, dass es sich bei der Ausführung eines Befehls in einem der Status `SUCCEEDEDFAILED`, und um `REJECTED` ein Terminal handelt und der Status vom Gerät gemeldet wird. Wenn ein Befehl terminalförmig ausgeführt wird, bedeutet dies, dass der Status oder die zugehörigen Felder nicht weiter aktualisiert werden. Ein `TIMED_OUT` Status kann vom Gerät oder

der Cloud gemeldet werden. Wenn es von der Cloud gemeldet wird, kann das Gerät später eine Aktualisierung des Felds für den Statusgrund vornehmen.

Im Folgenden wird beispielsweise eine MQTT-Beispielnachricht gezeigt, die vom Gerät veröffentlicht wurde.

> **Note**
>
> Für den Status der Befehlsausführung müssen Sie Folgendes sicherstellen, wenn Ihre Geräte das `statusReason` Objekt zum Veröffentlichen der Statusinformationen verwenden:
>
> - Der `reasonCode` verwendet das Muster `[A-Z0-9_-]+` und es ist nicht länger als 64 Zeichen.
> - Das `reasonDescription` ist nicht länger als 1.024 Zeichen. Es können alle Zeichen außer Steuerzeichen wie Zeilenumbrüchen verwendet werden.

```
{
    "deviceId": "",
    "executionId": "",
    "status": "CREATED",
    "statusReason": {
        "reasonCode": "",
        "reasonDescription": ""
    }
}
```

Ein Beispiel, das zeigt, wie Sie den AWS IoT Core MQTT-Testclient verwenden können, um die Themen zu abonnieren und die Meldungen zur Befehlsausführung zu sehen, finden Sie unter Befehlsaktualisierungen mit dem MQTT-Testclient anzeigen im AWS IoT Core Entwicklerhandbuch.

Holen Sie sich die Befehlsausführung

Sie können den API-Vorgang der `GetCommandExecution` AWS IoT Steuerungsebene verwenden, um Informationen über die Ausführung eines Befehls abzurufen. Sie müssen diesen Befehl bereits mithilfe der `StartCommandExecution` API-Operation ausgeführt haben.

Führen Sie den folgenden Befehl aus, um die Metadaten eines ausgeführten Befehls abzurufen.

- *execution-id*Ersetzen Sie ihn durch die ID des Befehls. Sie können diese Informationen aus der Antwort des `start-command-execution` CLI-Befehls abrufen.
- *target-arn*Ersetzen Sie es durch den ARN für das Zielfahrzeug AWS IoT oder Objekt, für das Sie den Befehl ausführen möchten.

```
aws iot get-command-execution --execution-id execution-id \
    --target-arn target-arn
```

Die `GetCommandExecution` API-Operation gibt eine Antwort zurück, die Informationen über den ARN der Befehlsausführung, den Ausführungsstatus und den Zeitpunkt enthält, zu dem der Befehl ausgeführt wurde und wann er abgeschlossen wurde. Der folgende Code zeigt eine Beispielantwort aus der API-Anfrage.

Um zusätzlichen Kontext zum Status der einzelnen Befehlsausführungen bereitzustellen, stellt die Befehlsfunktion ein `statusReason` Objekt bereit. Das Objekt enthält zwei Felder `reasonCode` und`reasonDescription`. Mithilfe dieser Felder können Ihre Geräte zusätzliche Informationen über den Status der Befehlsausführung bereitstellen. Diese Informationen überschreiben alle `reasonCode` Standardwerte`reasonDescription`, die aus der Cloud gemeldet werden.

Um diese Informationen zu melden, können Ihre Geräte die aktualisierten Statusinformationen in der Cloud veröffentlichen. Wenn Sie dann den Status der Befehlsausführung mithilfe der `GetCommandExecution` API abrufen, werden Ihnen die neuesten Statuscodes angezeigt.

> **Note**
>
> Das `completedAt` Feld in der Ausführungsantwort entspricht dem Zeitpunkt, zu dem das Gerät einen Terminalstatus an die Cloud meldet. Im Fall eines `TIMED_OUT` Status wird dieses Feld nur gesetzt, wenn das Gerät einen Timeout meldet. Wenn der `TIMED_OUT` Status von der Cloud festgelegt wird, wird der `TIMED_OUT` Status nicht aktualisiert. Weitere Informationen zum Timeout-Verhalten finden Sie unterStatus des Timeouts bei der Befehlsausführung.

```
{
    "executionId": "07e4b780-7eca-4ffd-b772-b76358da5542",
    "commandArn": "arn:aws:iot:ap-south-1:123456789012:command/LockDoor",
    "targetArn": "arn:aws:iot:ap-south-1:123456789012:thing/myFrontDoor",
    "status": "SUCCEEDED",
```

```
    "statusReason": {
        "reasonCode": "65536",
        "reasonDescription": "SUCCESS"
    },
    "createdAt": "2024-03-23T00:50:10.095000-07:00",
    "completedAt": "2024-03-23T00:50:10.095000-07:00",
    "Parameters": '{
         "$actuatorPath.Vehicle.Chassis.SteeringWheel.HandsOff.HandsOffSteeringMode":

         { "B": true }
    }'
}
```

Listet die Befehlsausführungen in Ihrem Konto auf

Verwenden Sie den HTTP-API-Vorgang der `ListCommandExecutions` AWS IoT Core Steuerungsebene, um alle Befehlsausführungen in Ihrem Konto aufzulisten. Das Beispiel verwendet das AWS CLI.

Themen

- Überlegungen beim Auflisten von Befehlsausführungen
- Beispiel für Befehlsausführungen auflisten

Überlegungen beim Auflisten von Befehlsausführungen

Im Folgenden sind einige Überlegungen zur Verwendung der `ListCommandExecutions` API aufgeführt.

- Sie müssen mindestens das `targetArn` oder das angeben, `commandArn` je nachdem, ob Sie Ausführungen für einen bestimmten Befehl oder ein Zielfahrzeug auflisten möchten. Die API-Anforderung darf nicht leer sein und nicht beide Felder in derselben Anfrage enthalten.
- Sie müssen nur die `startedTimeFilter` oder die `completedTimeFilter` Informationen angeben. Die API-Anforderung darf nicht leer sein und nicht beide Felder in derselben Anfrage enthalten. Sie können die `after` Felder `before` und des Objekts verwenden, um Befehlsausführungen aufzulisten, die innerhalb eines bestimmten Zeitraums entweder erstellt oder abgeschlossen wurden.
- `before`Sowohl die `after` Felder als auch dürfen nicht größer als die aktuelle Uhrzeit sein. Wenn Sie keinen Wert angeben, entspricht das `before` Feld standardmäßig der aktuellen Uhrzeit und

dem `after` Feld der aktuellen Uhrzeit — 6 Monate. Das heißt, je nach verwendetem Filter listet die API alle Ausführungen auf, die in den letzten sechs Monaten entweder erstellt oder abgeschlossen wurden.

- Sie können den `sort-order` Parameter verwenden, um anzugeben, ob Sie die Ausführungen in aufsteigender Reihenfolge auflisten möchten. Standardmäßig werden die Ausführungen in absteigender Reihenfolge aufgeführt, wenn Sie dieses Feld nicht angeben.
- Sie können die Befehlsausführungen nicht nach ihrem Status filtern, wenn Sie Befehlsausführungen für einen Befehl-ARN auflisten.

Beispiel für Befehlsausführungen auflisten

Das folgende Beispiel zeigt Ihnen, wie Sie Befehlsausführungen in Ihrem auflisten. AWS-Konto

Bei der Ausführung des Befehls müssen Sie angeben, ob die Liste so gefiltert werden soll, dass nur Befehlsausführungen angezeigt werden, die mit dem für ein bestimmtes Gerät erstellt wurden`targetArn`, oder Ausführungen für einen bestimmten Befehl, der mit dem angegeben wurde. `commandArn`

Ersetzen Sie in diesem Beispiel:

- *`<target-arn>`*mit der Amazon-Ressourcennummer (ARN) des Geräts, für das Sie die Ausführung planen, z. `arn:aws:iot:`*`us-east-1`*`:`*`123456789012`*`:thing/`*`b8e4157c98f332cffb37627f`* B.
- *`<target-arn>`*mit der Amazon-Ressourcennummer (ARN) des Geräts, für das Sie die Ausführung planen, z. `arn:aws:iot:`*`us-east-1`*`:`*`123456789012`*`:thing/`*`b8e4157c98f332cffb37627f`* B.
- *`<after>`*mit der Zeit, nach der Sie die Ausführungen auflisten möchten, die erstellt wurden, zum Beispiel. `2024-11-01T03:00`

```
aws iot list-command-executions \
--target-arn <target-arn> \
--started-time-filter '{after=<after>}' \
--sort-order "ASCENDING"
```

Wenn Sie diesen Befehl ausführen, wird eine Antwort generiert, die eine Liste der von Ihnen erstellten Befehlsausführungen sowie die Uhrzeit enthält, zu der die Ausführung begonnen und abgeschlossen

wurde. Es enthält auch Statusinformationen und das `statusReason` Objekt, das zusätzliche Informationen über den Status enthält.

```
{
    "commandExecutions": [
        {
            "commandArn": "arn:aws:iot:us-east-1:123456789012:command/TestMe002",
            "executionId": "b2b654ca-1a71-427f-9669-e74ae9d92d24",
            "targetArn": "arn:aws:iot:us-east-1:123456789012:thing/
b8e4157c98f332cffb37627f",
            "status": "TIMED_OUT",
            "createdAt": "2024-11-24T14:39:25.791000-08:00",
            "startedAt": "2024-11-24T14:39:25.791000-08:00"
        },
        {
            "commandArn": "arn:aws:iot:us-east-1:123456789012:command/TestMe002",
            "executionId": "34bf015f-ef0f-4453-acd0-9cca2d42a48f",
            "targetArn": "arn:aws:iot:us-east-1:123456789012:thing/
b8e4157c98f332cffb37627f",
            "status": "IN_PROGRESS",
            "createdAt": "2024-11-24T14:05:36.021000-08:00",
            "startedAt": "2024-11-24T14:05:36.021000-08:00"
        }
    ]
}
```

Löscht die Ausführung eines Befehls

Wenn Sie eine Befehlsausführung nicht mehr verwenden möchten, können Sie sie dauerhaft aus Ihrem Konto entfernen.

> **Note**
>
> Eine Befehlsausführung kann nur gelöscht werden, wenn sie einen Terminalstatus wie `SUCCEEDED`FAILED, oder erreicht hat`REJECTED`.

Das folgende Beispiel zeigt Ihnen, wie Sie eine Befehlsausführung mithilfe des `delete-command-execution` AWS CLI Befehls löschen. *`<execution-id>`*Ersetzen Sie es durch den Bezeichner der Befehlsausführung, die Sie löschen möchten.

```
aws iot delete-command-execution --execution-id <execution-id>
```

Wenn die API-Anfrage erfolgreich ist, generiert die Befehlsausführung den Statuscode 200. Sie können die `GetCommandExecution` API verwenden, um zu überprüfen, ob die Befehlsausführung in Ihrem Konto nicht mehr vorhanden ist.

Beispiel: Verwendung von Befehlen zur Steuerung eines Fahrzeuglenkmodus ()AWS CLI

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Das folgende Beispiel zeigt Ihnen, wie Sie die Befehlsfunktion mithilfe von verwenden AWS CLI. In diesem Beispiel wird ein AWS IoT FleetWise Fahrzeug als Zielgerät verwendet, um zu zeigen, wie Sie einen Befehl zur Fernsteuerung des Lenkmodus senden können.

Themen

- Überblick über das Beispiel für den Fahrzeuglenkmodus
- Voraussetzungen
- IAM-Richtlinie für die Verwendung von Remote-Befehlen
- AWS IoT Befehle ausführen (AWS CLI)
- Bereinigen

Überblick über das Beispiel für den Fahrzeuglenkmodus

In diesem Beispiel werden Sie:

1. Erstellen Sie eine Befehlsressource für den Vorgang, mit `create-command` AWS CLI dem Sie den Lenkmodus des Fahrzeugs ändern können.
2. Rufen Sie Informationen über den Befehl ab, z. B. den Zeitpunkt, zu dem er erstellt oder zuletzt aktualisiert wurde, mit dem `get-command` AWS CLI.

3. Senden Sie den Befehl `start-command-execution` AWS CLI mit dem Lenkmodus als obligatorischen Parameter an das Fahrzeug. Dieser wird dann auf dem Gerät ausgeführt.
4. Rufen Sie das Ergebnis der Befehlsausführung mit dem ab `get-command-execution` AWS CLI. Sie können überprüfen, wann die Ausführung abgeschlossen ist, und zusätzliche Details abrufen, z. B. das Ausführungsergebnis und die Zeit, die bis zur Ausführung des Befehls benötigt wurde.
5. Führen Sie Bereinigungsaktivitäten durch, indem Sie alle Befehle und Befehlsausführungen entfernen, die Sie nicht mehr verwenden möchten.

Voraussetzungen

Bevor Sie dieses Beispiel ausführen:

- Stellen Sie Ihr AWS IoT FleetWise Fahrzeug als Objekt AWS IoT in der AWS IoT Registrierung bereit. Sie müssen Ihrem Ding auch ein Zertifikat hinzufügen und es aktivieren und Ihrem Ding eine Richtlinie beifügen. Ihr Gerät kann sich dann mit der Cloud verbinden und die Befehle ausführen. Weitere Informationen finden Sie unter Fahrzeuge bereitstellen.
- Erstellen Sie einen IAM-Benutzer und eine IAM-Richtlinie, die Ihnen die Erlaubnis erteilt, die API-Operationen für die Verwendung von Befehlen auszuführen, wie unter gezeigt. IAM-Richtlinie für die Verwendung von Remote-Befehlen

IAM-Richtlinie für die Verwendung von Remote-Befehlen

Die folgende Tabelle zeigt ein Beispiel für eine IAM-Richtlinie, die Zugriff auf alle API-Operationen der Steuerungsebene und der Datenebene für die Befehlsfunktion gewährt. Der Benutzer der Anwendung verfügt über die erforderlichen Berechtigungen, um alle API-Operationen für Remote-Befehle auszuführen, wie in der Tabelle dargestellt.

API-Operation

API-Aktion	Steuerung/ Datenebene	Protocol (Protokoll)	Description	Ressource
`CreateCommand`	Steuerebene	HTTP	Erzeugt eine Befehlsressource	• command

API-Aktion	Steuerung/ Datenebene	Protocol (Protokoll)	Description	Ressource
`GetCommand`	Steuerebene	HTTP	Ruft Informationen über einen Befehl ab	• command
`UpdateCommand`	Steuerebene	HTTP	Aktualisiert Informationen über einen Befehl oder stellt ihn als veraltet dar	• command
`ListCommands`	Steuerebene	HTTP	Führt Befehle in Ihrem Konto auf	• command
`DeleteCommand`	Steuerebene	HTTP	Löscht einen Befehl	• command
`StartCommandExecution`	Datenebene	HTTP	Startet die Ausführung eines Befehls	• command • Objekt
`UpdateCommandExecution`	Datenebene	MQTT	Aktualisieren Sie die Ausführung eines Befehls	• command • Objekt
`GetCommandExecution`	Steuerebene	HTTP	Ruft Informationen über die Ausführung eines Befehls ab	• command • Objekt
`ListCommandExecutions`	Steuerebene	HTTP	Führt Befehlsausführungen in Ihrem Konto auf	• command • Objekt
`DeleteCommandExecution`	Steuerebene	HTTP	Löscht eine Befehlsausführung	• command • Objekt

Ersetzen Sie in diesem Beispiel:

- *`us-east-1`*mit deinem AWS-Region, wie`ap-south-1`.
- *`111122223333`*mit deiner AWS-Konto Nummer, wie`57EXAMPLE833`.

- *command-idcommand-id1*, und *command-id2* mit Ihrer eindeutigen Befehlskennung, z. B. `LockDoor` oder`TurnOffAC`.
- *thing-name*mit deinem AWS IoT Dingnamen, wiemy_car.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iot:CreateCommand",
                "iot:GetCommand",
                "iot:ListCommands",
                "iot:UpdateCommand",
                "iot:DeleteCommand"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iot:us-east-1:111122223333:command/command-id1",
                "arn:aws:iot:us-east-1:111122223333:command/command-id2"
            ]
        },
        {
            "Action": [
                "iot:GetCommandExecution",
                "iot:ListCommandExecutions",
                "iot:DeleteCommandExecution"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iot:us-east-1:111122223333:command/command-id",
                "arn:aws:iot:us-east-1:111122223333:thing/thing-name"
            ]
        },
        {
            "Action": "iot:StartCommandExecution",
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iot:us-east-1:111122223333:command/command-id",
                "arn:aws:iot:us-east-1:111122223333:thing/thing-name"
```

```
            ]
        }
    ]
}
```

AWS IoT Befehle ausführen (AWS CLI)

Im Folgenden wird gezeigt, wie Sie Befehle ausführen und den Lenkmodus des Fahrzeugs ändern können. AWS CLI

1. Erstellen Sie eine Befehlsressource für den Betrieb im Lenkmodus

   Erstellen Sie den Befehl, den Sie mit der `create-command` CLI an Ihr Gerät senden möchten. Geben Sie in diesem Beispiel Folgendes an:

   - `command-id` wie *`TurnOffSteeringMode`*
   - `role-arn`as "`arn:aws:iam:`*`accountId`*`:role/`*`FwCommandExecutionRole`*" Die `role-arn` muss angegeben werden, da es die IAM-Rolle ist, die Berechtigungen zum Erstellen und Ausführen von Befehlen auf Ihrem Fahrzeug gewährt. Weitere Informationen finden Sie unter Erteilen Sie die AWS IoT Device Management Berechtigung zum Generieren der Payload für Befehle mit AWS IoT FleetWise.
   - `display-name`als "“ *`Turn off steering mode`*
   - `namespace`muss sein `AWS-IoT-FleetWise`
   - `mandatory-parameters`als Name-Wert-Paar, mit `name` als "*`$actuatorPath.Vehicle.Chassis.SteeringWheel.TurnOffSteeringMode`*" und defaultValue als *`{ "S": "true" }`*

   > **Note**
   >
   > Sie können einen Befehl auch ohne Angabe obligatorischer Parameter erstellen. Anschließend müssen Sie die Parameter angeben, die bei der Ausführung des Befehls über die `start-command-execution` CLI verwendet werden sollen. Ein Beispiel finden Sie unter Szenarien zur Verwendung von Befehlen.

> **Important**
>
> Wenn Sie den `AWS-IoT-FleetWise` Namespace verwenden, müssen Sie sicherstellen, dass das `Name` im Rahmen von angegebene Feld das `$actuatorPath.` Präfix `mandatory-parameters` verwendet und dass das `Value` Feld den Datentyp Zeichenfolge verwendet.

```
aws iot create-command \
    --command-id TurnOffSteeringMode \
    --role-arn "arn:aws:iam:accountId:role/FwCommandExecutionRole" \
    --display-name "Turn off steering mode" \
    --namespace AWS-IoT-FleetWise \
    --mandatory-parameters '[
      {
        "name": "$actuatorPath.Vehicle.Chassis.SteeringWheel.TurnOffSteeringMode",
        "defaultValue": { "S": "true" }
      }
    ]'
```

Die folgende Ausgabe zeigt eine Beispielantwort von der CLI, wobei `ap-south-1` und Beispiele für die AWS-Konto ID AWS-Region und `123456789012` sind.

```
{
    "commandId": "TurnOffSteeringMode",
    "commandArn": "arn:aws:iot:ap-south-1:123456789012:command/TurnOffSteeringMode"
}
```

Weitere Beispiele zur Verwendung dieses Befehls finden Sie unterErstellen Sie eine Befehlsressource.

2. Rufen Sie Informationen über den Befehl ab

   Führen Sie den folgenden Befehl aus, um Informationen über den Befehl abzurufen. Dabei `command-id` steht die Befehls-ID in der Ausgabe der `create-command` Operation von oben.

> **Note**
>
> Wenn Sie mehr als einen Befehl erstellen, können Sie die `ListCommands` API verwenden, um alle Befehle in Ihrem Konto aufzulisten, und dann die `GetCommand` API verwenden, um zusätzliche Informationen zu einem bestimmten Befehl abzurufen. Weitere Informationen finden Sie unter Befehle in Ihrem Konto auflisten.

```
aws iot get-command --command-id TurnOffSteeringMode
```

Die Ausführung dieses Befehls generiert die folgende Antwort. Sie sehen den Zeitpunkt, zu dem der Befehl erstellt und zuletzt aktualisiert wurde, alle von Ihnen angegebenen Parameter und ob der Befehl für die Ausführung auf dem Gerät verfügbar ist.

```
{
    "commandId": "TurnOffSteeringMode",
    "commandArn": "arn:aws:iot:ap-south-1:123456789012:command/
TurnOffSteeringMode",
    "namespace": "AWS-IoT-FleetWise",
    "mandatoryParameters":[
        {
            "name":
 "$actuatorPath.Vehicle.Chassis.SteeringWheel.TurnOffSteeringMode",
            "defaultValue": {"S": "true" }
        }
    ],
    "createdAt": "2024-03-23T00:50:10.095000-07:00",
    "lastUpdatedAt": "2024-03-23T00:50:10.095000-07:00",
    "deprecated": false
}
```

Weitere Beispiele zur Verwendung dieses Befehls finden Sie unterRuft Informationen zu einem Befehl ab.

3. Starten Sie die Befehlsausführung

   Führen Sie den folgenden Befehl aus, um die Ausführung des Befehls zu starten. Dabei `command-arn` befindet sich der Befehl ARN in der Ausgabe der `get-command` Operation von

oben. Das `target-arn` ist der ARN des Zielgeräts, für das Sie den Befehl ausführen, zum Beispiel*myVehicle*.

Da Sie in diesem Beispiel bei der Erstellung des Befehls Standardwerte für die Parameter angegeben haben, kann die `start-command-execution` CLI diese Werte bei der Ausführung des Befehls verwenden. Sie können den Standardwert auch überschreiben, indem Sie bei Verwendung der CLI einen anderen Wert für die Parameter angeben.

```
aws iot-data start-command-execution \
    --command-arn arn:aws:iot:ap-south-1:123456789012:command/TurnOffSteeringMode \
    --target-arn arn:aws:iot:ap-south-1:123456789012:thing/myVehicle
```

Wenn Sie diesen Befehl ausführen, wird eine Befehlsausführungs-ID zurückgegeben. Sie können diese ID verwenden, um den Status der Befehlsausführung, die Details und den Verlauf der Befehlsausführung abzufragen.

```
{
    "executionId": "07e4b780-7eca-4ffd-b772-b76358da5542"
}
```

Weitere Beispiele zur Verwendung der CLI finden Sie unterSenden Sie einen Befehl ()AWS CLI.

4. Rufen Sie Informationen über die Befehlsausführung ab

Führen Sie den folgenden Befehl aus, um Informationen zu dem Befehl abzurufen, den Sie auf dem Zielgerät ausgeführt haben. Geben Sie den an`execution-id`, den Sie als Ausgabe des `start-command-execution` Vorgangs von oben erhalten haben, und den`target-arn`, welcher der ARN des Geräts ist, auf das Sie abzielen.

> **Note**
>
> - Um die neuesten Statusinformationen zu erhalten, müssen Ihre Geräte die aktualisierten Statusinformationen im MQTT-Antwortthema veröffentlicht haben, das für Befehle reserviert ist, die die `UpdateCommandExecution` MQTT-API verwenden. Weitere Informationen finden Sie unter Das Ergebnis der Befehlsausführung aktualisieren.
> - Wenn Sie mehr als eine Befehlsausführung starten, können Sie die `ListCommandExecutions` API verwenden, um alle Befehlsausführungen in Ihrem

Konto aufzulisten, und dann die `GetCommandExecution` API verwenden, um zusätzliche Informationen über eine bestimmte Ausführung zu erhalten. Weitere Informationen finden Sie unter Listet die Befehlsausführungen in Ihrem Konto auf.

```
aws iot get-command-execution \
    --execution-id <"07e4b780-7eca-4ffd-b772-b76358da5542"> \
    --target-arn arn:aws:iot:us-east-1:<account>:thing/myVehicle
```

Wenn Sie diesen Befehl ausführen, werden Informationen über die Befehlsausführung, den Ausführungsstatus, den Zeitpunkt, zu dem die Ausführung gestartet wurde, und den Zeitpunkt, zu dem die Ausführung abgeschlossen wurde, zurückgegeben. Die folgende Antwort zeigt beispielsweise, dass die Befehlsausführung auf dem Zielgerät erfolgreich war und der Steuermodus ausgeschaltet wurde.

```
{
    "executionId": "07e4b780-7eca-4ffd-b772-b76358da5542",
    "commandArn": "arn:aws:iot:ap-south-1:123456789012:command/
TurnOffSteeringMode",
    "targetArn": "arn:aws:iot:ap-south-1:123456789012:thing/myVehicle",
    "result": "SUCCEEDED",
     "statusReason": {
        "reasonCode": "65536",
        "reasonDescription": "SUCCESS"
    },
    "result": {
        "KeyName": {
            "S": "",
            "B": true,
            "BIN": null
        }
    },
    "createdAt": "2024-03-23T00:50:10.095000-07:00",
    "completedAt": "2024-03-23T00:50:10.095000-07:00",
    "parameters": '{
         "$actuatorPath.Vehicle.Chassis.SteeringWheel.TurnOffSteeringMode":
         { "S": "true" }
    }'
}
```

Bereinigen

Nachdem Sie einen Befehl erstellt und auf Ihrem Gerät ausgeführt haben, können Sie ihn löschen, wenn Sie diesen Befehl nicht mehr verwenden möchten. Alle ausstehenden Befehlsausführungen, die gerade ausgeführt werden, werden weiterhin ausgeführt, ohne dass sich die Löschanforderung darauf auswirkt.

> **Note**
>
> Alternativ können Sie einen Befehl auch als veraltet markieren, wenn er veraltet ist und Sie ihn möglicherweise später verwenden müssen, um ihn auf dem Zielgerät auszuführen.

1. (Optional) Verwerfen Sie die Befehlsressource

   Führen Sie den folgenden Befehl aus, um den Befehl als veraltet zu kennzeichnen. Dabei `command-id` steht die Befehls-ID in der Ausgabe der Operation von oben. `get-command`

   ```
   aws iot update-command \
      --command-id TurnOffSteeringMode \
      --deprecated
   ```

   Die Ausführung dieses Befehls gibt eine Ausgabe zurück, die anzeigt, dass der Befehl veraltet ist. Sie können den Befehl auch mit der CLI wiederherstellen.

   > **Note**
   >
   > Sie können die `update-command` CLI auch verwenden, um den Anzeigenamen und die Beschreibung eines Befehls zu aktualisieren. Weitere Informationen finden Sie unter Eine Befehlsressource aktualisieren oder als veraltet kennzeichnen.

   ```
   {
       "commandId": "TurnOffSteeringMode",
       "deprecated": true,
       "lastUpdatedAt": "2024-05-09T23:16:51.370000-07:00"
   }
   ```

2. Löschen Sie den Befehl

   Führen Sie den folgenden Befehl aus, um den durch den angegebenen Befehl zu löschen`command-id`.

   > **Note**
   >
   > Die Löschaktion ist dauerhaft und kann nicht rückgängig gemacht werden.

   ```
   aws iot delete-command --command-id TurnOffSteeringMode
   ```

   Wenn die Löschanforderung erfolgreich ist, wird `statusCode` der HTTP-Wert 202 oder 204 angezeigt, je nachdem, ob Sie den Befehl als veraltet markiert haben und wann er als veraltet markiert wurde. Weitere Informationen sowie ein Beispiel finden Sie unter Löschen Sie eine Befehlsressource.

   Sie können die `get-command` CLI verwenden, um zu überprüfen, ob der Befehl aus Ihrem Konto entfernt wurde.

3. (Optional) Löschen Sie die Befehlsausführungen

   Standardmäßig werden alle Befehlsausführungen innerhalb von sechs Monaten ab dem Datum, an dem Sie sie erstellen, gelöscht. Sie können diese Informationen mithilfe des `timeToLive` Parameters in der `GetCommandExecution` API anzeigen.

   Alternativ können Sie die Befehlsausführung löschen, wenn Ihre Befehlsausführung zu einem Terminal geworden ist `SUCCEEDEDFAILED`, z. B. wenn Ihr Ausführungsstatus einer von`REJECTED`, oder ist. Führen Sie den folgenden Befehl aus, um die Ausführung zu löschen. Dabei `execution-id` steht die Ausführungs-ID in der Ausgabe der `get-command-execution` Operation von oben.

   ```
   aws iot delete-command-execution \
               --execution-id "07e4b780-7eca-4ffd-b772-b76358da5542"
   ```

   Sie können die `get-command-execution` CLI verwenden, um zu überprüfen, ob die Befehlsausführung aus Ihrem Konto entfernt wurde.

Szenarien zur Verwendung von Befehlen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Wenn Sie die Befehlsfunktion verwenden, können Sie Befehle in den folgenden Szenarien erstellen und ausführen:

- Sie können die Parameter bei der Erstellung weglassen und nur die Befehls-ID angeben. In diesem Fall müssen Sie die Parameter angeben, die bei der Ausführung des Befehls auf dem Zielgerät verwendet werden sollen.
- Sie können einen oder mehrere Parameter angeben und Standardwerte für diese Parameter konfigurieren, wenn Sie den Befehl erstellen. Die Angabe von Standardwerten schützt Sie davor, ungenaue Befehle zu senden.
- Sie können einen oder mehrere Parameter angeben und Werte für diese Parameter konfigurieren, wenn Sie den Befehl erstellen. Es können mehrere Parameter angegeben werden, aber nur einer davon wird ausgeführt, und das `Name` Feld dieses Parameters muss das `$actuatorPath` Präfix verwenden.

Dieser Abschnitt enthält einige Nutzungsszenarien für die `CreateCommand` und die `StartCommandExecution` API und die Verwendung der Parameter. Außerdem werden Ihnen einige Beispiele für die Verwendung von Befehlen mit Statusvorlagen gezeigt.

Themen

- Einen Befehl ohne Parameter erstellen
- Einen Befehl mit Standardwerten für Parameter erstellen
- Einen Befehl mit Parameterwerten erstellen
- Befehle mit Statusvorlagen verwenden

Einen Befehl ohne Parameter erstellen

Der folgende Anwendungsfall zeigt, wie Sie die `CreateCommand` API oder die `create-command` CLI verwenden können, um einen Befehl ohne Parameter zu erstellen. Wenn Sie einen Befehl erstellen, müssen Sie nur eine Befehls-ID und einen Rollen-ARN angeben.

Dieser Anwendungsfall ist besonders nützlich in wiederkehrenden Anwendungsfällen, z. B. wenn Sie denselben Befehl mehrmals an ein Fahrzeug senden möchten. In diesem Fall ist der Befehl nicht an einen bestimmten Aktuator gebunden und bietet Ihnen die Flexibilität, den Befehl auf einem beliebigen Aktuator auszuführen. Sie müssen die Parameter stattdessen zur Laufzeit angeben, wenn Sie den Befehl mithilfe der `StartCommandExecution` API oder der `start-command-execution` CLI ausführen, was die Aktuatoren und physikalischen Signalwerte umfasst.

Einen Befehl ohne **`mandatory-parameters`** Eingabe erstellen

Dieser Anwendungsfall zeigt, wie ein Befehl ohne obligatorische Parametereingabe erstellt wird.

```
aws iot create-command \
    --command-id "UserJourney1" \
    --role-arn "arn:aws:iam:accountId:role/FwCommandExecutionRole" \
    --description "UserJourney1 - No mandatory parameters" \
    --namespace "AWS-IoT-FleetWise"
```

Einen Befehl ausführen, der ohne **`mandatory-parameters`** Eingabe erstellt wurde

In diesem ersten Beispiel können Sie mit dem oben erstellten Befehl einen Befehl an einem beliebigen Aktuator ohne Einschränkungen ausführen. `actuator1`Um einen Wert von 10 festzulegen, führen Sie folgenden Befehl aus:

```
aws iot-jobs-data start-command-execution \
    --command-arn arn:aws:iot:region:111122223333:command/UserJourney1 \
    --target-arn arn:aws:iot:region:111122223333:thing/target-vehicle \
    --parameters '{
        "$actuatorPath.Vehicle.actuator1": {"S": "10"}
    }'
```

Auf ähnliche Weise können Sie einen Befehl ausführen, der `actuator3` auf einen Wert von festgelegt ist`true`.

```
aws iot-jobs-data start-command-execution \
```

```
    --command-arn arn:aws:iot:region:111122223333:command/UserJourney1 \
    --target-arn arn:aws:iot:region:111122223333:thing/target-vehicle \
    --parameters '{
        "$actuatorPath.Vehicle.actuator3": {"S": "true"}
    }'
```

Einen Befehl mit Standardwerten für Parameter erstellen

Mit diesem Befehl können Sie nur einen Befehl auf dem angegebenen Aktuator ausführen. Die Angabe von Standardwerten schützt Sie davor, ungenaue Befehle zu senden. Beispielsweise kann ein `LockDoor` Befehl zum Sperren und Entsperren von Türen mit einem Standardwert konfiguriert werden, um zu verhindern, dass der Befehl versehentlich Türen entriegelt.

Dieser Anwendungsfall ist besonders nützlich, wenn Sie denselben Befehl mehrmals senden und verschiedene Aktionen an demselben Aktuator ausführen möchten, z. B. die Türen eines Fahrzeugs ver- und entriegeln. Wenn Sie den Aktuator auf den Standardwert setzen möchten, müssen Sie keinen Wert an die `start-command-execution` CLI übergeben. `parameters` Wenn Sie `parameters` in der `start-command-execution` CLI einen anderen Wert für angeben, überschreibt dieser den Standardwert.

Erstellen eines Befehls mit Standardwerten für **`mandatory-parameters`**

Der folgende Befehl zeigt, wie ein Standardwert für actuator1 bereitgestellt wird.

```
aws iot create-command \
    --command-id "UserJourney2" \
    --namespace "AWS-IoT-FleetWise" \
    --role-arn "arn:aws:iam:accountId:role/FwCommandExecutionRole" \
    --mandatory-parameters '[
        {
            "name": "$actuatorPath.Vehicle.actuator1",
            "defaultValue": {"S": "0"}
        }
    ]'
```

Ausführen eines Befehls, der mit Standardwerten für erstellt wurde **`mandatory-parameters`**

Mit dem Befehl `UserJourney2` können Sie einen Befehl ausführen, ohne während der Laufzeit einen Eingabewert übergeben zu müssen. In diesem Fall verwendet die Ausführung zur Laufzeit die bei der Erstellung angegebenen Standardwerte.

```
aws iot-data start-command-execution \
    --command-arn arn:aws:iot:region:111122223333:command/UserJourney3 \
    --target-arn arn:aws:iot:region:111122223333:thing/target-vehicle
```

Sie können während der Laufzeit auch einen anderen Wert für denselben Aktuator, actuator1, übergeben, wodurch der Standardwert überschrieben wird.

```
aws iot-jobs-data start-command-execution \
    --command-arn arn:aws:iot:region:111122223333:command/UserJourney3 \
    --target-arn arn:aws:iot:region:111122223333:thing/target-vehicle \
    --parameters '{
        "$actuatorPath.Vehicle.actuator1": {"S": "139"}
    }'
```

Einen Befehl mit Parameterwerten erstellen

Mit diesem Befehl können Sie nur einen Befehl auf dem angegebenen Aktuator ausführen. Außerdem müssen Sie während der Laufzeit einen Wert für den Aktuator festlegen.

Dieser Anwendungsfall ist besonders nützlich, wenn Sie möchten, dass der Endbenutzer nur bestimmte, festgelegte Aktionen an einigen der Aktuatoren ausführt, wenn sie im Fahrzeug ausgeführt werden.

> **Note**
>
> Sie können mehr als Name-Wert-Paare für die `mandatory-parameters` Eingabe verwenden, wobei Standardwerte für einige oder alle von ihnen verwendet werden können. Zur Laufzeit können Sie dann den Parameter bestimmen, den Sie bei der Ausführung auf dem Aktuator verwenden möchten, vorausgesetzt, der Name des Aktuators verwendet den vollqualifizierten Namen mit dem Präfix. `$actuatorPath.`

Befehl ohne Standardwerte wird erstellt für **`mandatory-parameters`**

Mit diesem Befehl können Sie nur einen Befehl auf dem angegebenen Aktuator ausführen. Außerdem müssen Sie während der Laufzeit einen Wert für den Aktuator festlegen.

```
aws iot create-command \
    --command-id "UserJourney2" \
    --namespace "AWS-IoT-FleetWise" \
```

```
    --role-arn "arn:aws:iam:accountId:role/FwCommandExecutionRole" \
    --mandatory-parameters '[
        {
            "name": "$actuatorPath.Vehicle.actuator1"
        }
    ]'
```

Ausführen eines Befehls, der ohne Standardwerte für erstellt wurde **`mandatory-parameters`**

Wenn Sie den Befehl ausführen, müssen Sie in diesem Fall einen Wert für actuator1 angeben. Bei der unten gezeigten Befehlsausführung wird der Wert von `actuator1` auf erfolgreich gesetzt. `10`

```
aws iot-data start-command-execution \
    --command-arn arn:aws:iot:region:111122223333:command/UserJourney2 \
    --target-arn arn:aws:iot:region:111122223333:thing/target-vehicle \
    --parameters '{
        "$actuatorPath.Vehicle.actuator1": {"S": "10"}
    }'
```

Befehle mit Statusvorlagen verwenden

Sie können auch die API-Operationen der Befehle für die Erfassung und Verarbeitung von Statusdaten verwenden. Sie können beispielsweise einen einmaligen Status-Snapshot abrufen oder Statusvorlagen aktivieren oder deaktivieren, um die Erfassung von Fahrzeugzustandsdaten zu starten oder zu beenden. Die folgenden Beispiele zeigen, wie Sie die Befehlsfunktion mit Statusvorlagen verwenden können. Weitere Informationen finden Sie unter Vorlagenoperationen für die Datenerfassung und -verarbeitung angeben.

> **Note**
>
> Das als Teil der `mandatory-parameters` Eingabe angegebene Namensfeld muss das `$stateTemplate` Präfix verwenden.

Beispiel 1: Befehle für Statusvorlagen mit Standardwerten erstellen

Dieses Beispiel zeigt, wie Sie die `create-command` CLI verwenden, um Statusvorlagen zu aktivieren.

```
aws iot create-command \
```

```
    --command-id <COMMAND_ID> \
    --display-name "Activate State Template" \
    --namespace AWS-IoT-FleetWise \
    --mandatory-parameters '[
      {
          "name": "$stateTemplate.name"
      },
      {
          "name": "$stateTemplate.operation",
          "defaultValue": {"S": "activate"}
      }
    ]'
```

In ähnlicher Weise zeigt der folgende Befehl ein Beispiel dafür, wie Sie die `start-command-execution` CLI für Statusvorlagen verwenden können.

```
aws iot-data start-command-execution \
    --command-arn arn:aws:iot:region:111122223333:command/<COMMAND_ID> \
    --target-arn arn:aws:iot:region:111122223333:thing/<VEHICLE_NAME> \
    --parameters '{
       "$stateTemplate.name": {"S": "ST345"}
    }'
```

Beispiel 2: Befehle für Statusvorlagen ohne Standardwerte erstellen

Der folgende Befehl erstellt mehrere Statusvorlagen ohne Standardwerte für einen der Parameter. Er zwingt Sie, den Befehl mit diesen Parametern und den entsprechenden Werten auszuführen.

```
aws iot create-command \
    --command-id <COMMAND_ID> \
    --display-name "Activate State Template" \
    --namespace AWS-IoT-FleetWise \
    --mandatory-parameters '[
      {
          "name": "$stateTemplate.name",
          "defaultValue": {"S": "ST123"}
      },
      {
          "name": "$stateTemplate.operation",
          "defaultValue": {"S": "activate"}
      },
      {
```

```
          "name": "$stateTemplate.deactivateAfterSeconds",
          "defaultValue": {"L": "120"}
      }
    ]'
```

Der folgende Befehl zeigt, wie Sie die `start-command-execution` CLI für das obige Beispiel verwenden können.

```
aws iot-data start-command-execution \
    --command-arn arn:aws:iot:region:111122223333:command/<COMMAND_ID> \
    --target-arn arn:aws:iot:region:111122223333:thing/<VEHICLE_NAME> \
    --parameters '{
        "$stateTemplate.name": {"S": "ST345"},
        "$stateTemplate.operation": {"S": "activate"},
        "$stateTemplate.deactivateAfterSeconds" : {"L": "120"}
```

Überwachen Sie den letzten bekannten Zustand Ihrer Fahrzeuge

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können den letzten bekannten Status Ihrer Fahrzeuge nahezu in Echtzeit überwachen, indem Sie Statusvorlagen erstellen und diese Ihren Fahrzeugen zuordnen. Fahrzeuge, die mit Statusvorlagen verknüpft sind, streamen Telemetriedaten entweder mit einer Strategie oder mit einer AktualisierungsstrategieonChange. `periodic` Bei einer Strategie zur Aktualisierung bei Änderungen streamen die zugehörigen Fahrzeuge Telemetriedaten, wenn es zu einer Änderung kommt. Bei einer Strategie zur regelmäßigen Aktualisierung streamen die zugehörigen Fahrzeuge Telemetriedaten während eines bestimmten Zeitraums.

Bei On-Demand-Vorgängen können Sie den aktuellen Fahrzeugstatus auf einmal abfragen (abrufen). Sie können auch zuvor bereitgestellte Statusvorlagen aktivieren oder deaktivieren, um die Meldung von Fahrzeugzustandsdaten zu starten oder zu beenden. Die letzten bekannten Statusoperationen werden mit dem AWS IoT Befehl ausgeführt APIs.

Jede Statusvorlage enthält die folgenden Informationen.

`name`

Der eindeutige Alias der Statusvorlage.

`signalCatalogArn`

Der Amazon-Ressourcenname (ARN) des Signalkatalogs, der der Statusvorlage zugeordnet ist.

`stateTemplateProperties`

Eine Liste von Signalen, aus denen Daten gesammelt werden. Die Eigenschaften der Statusvorlage bestimmen, welche spezifischen Signalaktualisierungen das Fahrzeug an die Cloud sendet.

`dataExtraDimensions`

Eine Liste von Fahrzeugattributen, die in die in den Protokollpuffern (Protobuf) kodierten verarbeiteten Daten aufgenommen werden sollen.

`metadataExtraDimensions`

Eine Liste von Fahrzeugattributen, die zusammen mit den verarbeiteten Daten als MQTT 5-Benutzereigenschaft veröffentlicht werden sollen.

`id`

Eine eindeutige, vom Dienst generierte Kennung.

Methoden zum Sammeln von Daten, die von einem Fahrzeug gesendet werden, das die FleetWise Software Edge Agent for AWS IoT verwendet, finden Sie unterVerarbeiten Sie die letzten bekannten Statusfahrzeugdaten mithilfe von MQTT-Messaging. Weitere Informationen zum Zuordnen einer Statusvorlage zu einem Fahrzeug finden Sie unterErstellen Sie ein AWS FleetWise IoT-Fahrzeug.

Themen

- Erstellen Sie eine AWS FleetWise IoT-Statusvorlage
- Eine AWS FleetWise IoT-Statusvorlage aktualisieren
- Löschen Sie eine AWS FleetWise IoT-Statusvorlage
- Informationen zur AWS FleetWise IoT-Statusvorlage abrufen
- Vorlagenoperationen für die Datenerfassung und -verarbeitung angeben

Erstellen Sie eine AWS FleetWise IoT-Statusvorlage

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können die AWS FleetWise IoT-API oder die IoT-Konsole verwenden, um eine Statusvorlage zu erstellen. Statusvorlagen bieten einen Mechanismus, mit dem Sie den Status Ihrer Fahrzeuge

verfolgen können. Die Edge Agent for AWS FleetWise IoT-Software, die auf dem Fahrzeug ausgeführt wird, sammelt Signalaktualisierungen und sendet sie an die Cloud.

Erstellen Sie eine Statusvorlage (Konsole)

Sie können die AWS FleetWise IoT-Konsole verwenden, um eine Statusvorlage zu erstellen.

Um eine Statusvorlage zu erstellen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich State Templates aus.
3. Wählen Sie auf der Seite Statusvorlagen die Option Statusvorlage erstellen aus.
4. Geben Sie unter Details zur Statusvorlage einen Namen für die Statusvorlage und optional eine Beschreibung ein.
5. Fügen Sie im Feld Signale auswählen die Signale hinzu, von denen Sie Fahrzeugstatusinformationen abrufen möchten.
6. Wählen Sie Statusvorlage erstellen aus.

Nachdem Sie erfolgreich eine Statusvorlage erstellt haben, wird sie auf der Seite mit den Bundesstaatenvorlagen aufgeführt. Sie können sie jetzt einem Fahrzeug zuordnen.

Erstellen Sie eine Statusvorlage (AWS CLI)

Sie können den CreateStateTemplateAPI-Vorgang verwenden, um eine Statusvorlage zu erstellen. Das folgende Beispiel verwendet die AWS CLI.

Führen Sie den folgenden Befehl aus, um eine Statusvorlage zu erstellen.

*`create-state-template`*Ersetzen Sie es durch den Namen der JSON-Datei, die die Statusvorlagenkonfiguration enthält.

```
aws iotfleetwise create-state-template \
    --cli-input-json file://create-state-template.json
```

Example Konfiguration der Statusvorlage

`stateTemplateProperties`sollte die vollqualifizierten Namen der Signale enthalten.

`dataExtraDimensions`und `metadataExtraDimensions` sollte die vollqualifizierten Namen der Fahrzeugattribute enthalten. Die angegebenen Abmessungen ersetzen alle vorhandenen Bemaßungswerte in der Statusvorlage.

```
{
    "name": "state-template-name",
    "signalCatalogArn": "arn:aws:iotfleetwise:us-east-1:account:signal-catalog/catalog-name",
    "stateTemplateProperties": [
        "Vehicle.Signal.One",
        "Vehicle.Signal.Two"
    ],
    "dataExtraDimensions": [
        "Vehicle.Attribute.One",
        "Vehicle.Attribute.Two"
    ],
    "metadataExtraDimensions": [
        "Vehicle.Attribute.Three",
        "Vehicle.Attribute.Four"
    ]
}
```

Ordnen Sie einem Fahrzeug eine AWS FleetWise IoT-Statusvorlage zu

Ordnen Sie einem Fahrzeug (Konsole) eine Statusvorlage zu

Sie können die AWS FleetWise IoT-Konsole verwenden, um einem Fahrzeug zugehörige Statusvorlagen hinzuzufügen.

Um eine Statusvorlage zuzuordnen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeuge aus.
3. Wählen Sie ein Fahrzeug aus der Liste aus, um die zugehörige Detailseite zu öffnen.
4. Wählen Sie auf der Registerkarte Statusvorlagen die Option Statusvorlagen verwalten aus.
5. Wählen Sie Statusvorlage hinzufügen aus.
6. Wählen Sie eine Vorlage für einen Bundesstaat und wählen Sie die zugehörige Berichtsmethode aus.

    a. Bei Änderung — In der Statusvorlage werden Änderungen des Fahrzeugstatus gemeldet.

b. Periodisch — Die Statusvorlage meldet Aktualisierungen im angegebenen Zeitintervall.

7. Wählen Sie Änderungen speichern aus.

Ordnen Sie einem Fahrzeug eine AWS FleetWise IoT-Statusvorlage zu (AWS CLI)

Ordnen Sie die erstellte Statusvorlage einem Fahrzeug zu, um die Erfassung von Statusaktualisierungen vom Fahrzeug in die Cloud zu ermöglichen. Verwenden Sie dazu:

- Verwenden Sie beim Erstellen eines Fahrzeugs das `stateTemplates` `create-vehicle` Befehlsfeld. Weitere Informationen finden Sie unter Erstellen Sie ein AWS FleetWise IoT-Fahrzeug.
- Verwenden Sie beim Aktualisieren eines Fahrzeugs die `stateTemplatesToRemove` Felder `stateTemplatesToAdd` oder des `update-vehicle` Befehls. Weitere Informationen finden Sie unter Aktualisieren Sie ein AWS FleetWise IoT-Fahrzeug.

Eine AWS FleetWise IoT-Statusvorlage aktualisieren

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können den UpdateStateTemplateAPI-Vorgang oder die AWS FleetWise IoT-Konsole verwenden, um eine vorhandene Statusvorlage zu aktualisieren.

Aktualisieren Sie eine Statusvorlage (Konsole)

Um eine Statusvorlage von der Konsole aus zu aktualisieren, rufen Sie die Seite mit den Statusvorlagen der AWS FleetWise IoT-Konsole auf und führen Sie die folgenden Schritte aus.

1. Wählen Sie die Statusvorlage aus, die Sie aktualisieren möchten, und klicken Sie dann auf Bearbeiten.
2. Bearbeiten Sie die Details der Statusvorlage und wählen Sie dann Änderungen speichern aus.

Aktualisieren Sie eine Statusvorlage (AWS CLI)

Führen Sie den folgenden Befehl aus, um eine Statusvorlage zu aktualisieren.

*update-state-template*Ersetzen Sie es durch den Namen der JSON-Datei, die die Konfiguration der Statusvorlage enthält.

```
aws iotfleetwise update-state-template \
    --cli-input-json file://update-state-template.json
```

Example Konfiguration der Statusvorlage

Die `stateTemplateProperties` sollte die vollqualifizierten Namen der Signale enthalten.

Das `dataExtraDimensions` und `metadataExtraDimensions` sollte die vollqualifizierten Namen der Fahrzeugattribute enthalten.

```
{
    "identifier": "state-template-name",
    "stateTemplatePropertiesToAdd": [
        "Vehicle.Signal.Three"
    ],
    "stateTemplatePropertiesToRemove": [
        "Vehicle.Signal.One"
    ],
    "dataExtraDimensions": [
        "Vehicle.Attribute.One",
        "Vehicle.Attribute.Two"
    ],
    "metadataExtraDimensions": [
        "Vehicle.Attribute.Three",
        "Vehicle.Attribute.Four"
    ]
}
```

Löschen Sie eine AWS FleetWise IoT-Statusvorlage

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können den DeleteStateTemplateAPI-Vorgang oder die AWS FleetWise IoT-Konsole verwenden, um eine Statusvorlage zu löschen.

Löschen Sie eine Statusvorlage (Konsole)

Um eine Statusvorlage aus der Konsole zu löschen, rufen Sie die Seite mit den Statusvorlagen der AWS FleetWise IoT-Konsole auf und führen Sie die folgenden Schritte aus.

1. Wählen Sie die Statusvorlage aus, die Sie löschen möchten, und wählen Sie dann Löschen.
2. Bestätigen Sie, dass Sie die Statusvorlage löschen möchten, und wählen Sie dann Löschen.

Löschen Sie eine Statusvorlage (AWS CLI)

Führen Sie den folgenden Befehl aus, um eine Statusvorlage zu löschen.

*identifier*Ersetzen Sie durch den Namen oder die ID der Statusvorlage.

```
aws iotfleetwise delete-state-template \
    --identifier idenitfier
```

Informationen zur AWS FleetWise IoT-Statusvorlage abrufen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können den GetStateTemplateAPI-Vorgang verwenden, um Informationen zu einer Statusvorlage abzurufen. Das folgende Beispiel verwendet die AWS CLI.

*identifier*Ersetzen Sie es durch den Namen der Statusvorlage.

```
aws iotfleetwise get-state-template \
    --identifier idenitfier
```

Sie können den ListStateTemplatesAPI-Vorgang verwenden, um eine Liste Ihrer erstellten Statusvorlagen abzurufen. Das folgende Beispiel verwendet die AWS CLI.

```
aws iotfleetwise list-state-templates
```

Wenn Sie die Verschlüsselung mit einem vom Kunden verwalteten AWS KMS Schlüssel aktiviert haben, fügen Sie die folgende Richtlinienerklärung bei, damit Ihre Rolle die `GetStateTemplate` `ListStateTemplates` API-Operationen aufrufen kann.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/KMS_KEY_ID"
            ]
        }
    ]
}
```

Vorlagenoperationen für die Datenerfassung und -verarbeitung angeben

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

In den folgenden Abschnitten wird beschrieben, wie Sie Statusvorlagen verwenden, um die Datenerfassung zu aktivieren und zu deaktivieren, einen Abrufvorgang durchzuführen und Zustandsdaten von Ihren Fahrzeugen zu verarbeiten.

Themen

- Aktivieren und deaktivieren Sie die Erfassung von Zustandsdaten mithilfe von Statusvorlagen
- Rufen Sie mithilfe von Statusvorlagen einen Fahrzeugstatus-Snapshot ab
- Verarbeiten Sie die letzten bekannten Statusfahrzeugdaten mithilfe von MQTT-Messaging

Aktivieren und deaktivieren Sie die Erfassung von Zustandsdaten mithilfe von Statusvorlagen

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

In den folgenden Abschnitten wird beschrieben, wie Sie die Datenaufnahme mit Statusvorlagen mithilfe von aktivieren und deaktivieren. AWS CLI

> **Important**
>
> Bevor Sie beginnen, stellen Sie sicher, dass Sie bereits eine Statusvorlage erstellt und diese und ihre Aktualisierungsstrategie einem Fahrzeug zugeordnet haben.

Sie müssen eine Statusvorlage aktivieren, damit der Edge Agent Signalaktualisierungen an die Cloud senden kann.

Um diese Operationen mit Statusvorlagen durchzuführen, erstellen Sie zunächst eine Befehlsressource und starten Sie dann die Befehlsausführung auf dem Fahrzeug. Im folgenden Abschnitt wird beschrieben, wie Sie diese API verwenden und wie Sie die Datenaufnahme aktivieren und deaktivieren.

Themen

- Verwenden der CreateCommand-API
- Beispiel: Aktivieren Sie eine Statusvorlage
- Beispiel: Deaktivieren Sie eine Statusvorlage

Verwenden der **`CreateCommand`**-API

Erstellen Sie eine Befehlsressource im Namespace `AWS-IoTFleetwise` "" und verwenden Sie die folgenden Parameter, wenn Sie eine Befehlsressource für eine Statusvorlage erstellen oder senden:

- `$stateTemplate.name`— Der Name der Statusvorlage, für die der Vorgang ausgeführt werden soll. Die Statusvorlage muss auf das Fahrzeug aufgebracht werden, bevor Sie einen Vorgang durchführen können. Weitere Informationen finden Sie unter Ordnen Sie einem Fahrzeug eine AWS FleetWise IoT-Statusvorlage zu.
- `$stateTemplate.operation`— Der Vorgang, der an der Statusvorlage ausgeführt werden soll. Verwenden Sie einen der folgenden Werte für diesen Parameter:
  - `activate`— Der Edge-Agent beginnt mit dem Senden von Signalaktualisierungen an die Cloud, die `stateTemplateUpdateStrategy` Sie bei der Anwendung der Statusvorlage auf das Fahrzeug angegeben haben (bei Änderung oder in regelmäßigen Abständen). Weitere Informationen finden Sie unter Ordnen Sie einem Fahrzeug eine AWS FleetWise IoT-Statusvorlage zu.

Sie können auch eine automatische Deaktivierungszeit für die Statusvorlage definieren, um Aktualisierungen nach einem bestimmten Zeitraum zu beenden. Wenn keine automatische Deaktivierungszeit angegeben ist, senden die Statusvorlagen so lange Updates, bis ein Deaktivierungsaufruf erfolgt.

Sobald der `activate` Befehl empfangen wurde, sollte das Gerät die in der Statusvorlage angegebenen Signale gemäß der Aktualisierungsstrategie senden. AWS IoT FleetWise empfiehlt, dass, wenn ein Aktivierungsbefehl vom Gerät empfangen wird, die erste Nachricht, die es sendet, eine Momentaufnahme aller Signale in der Statusvorlage enthalten sollte. Nachfolgende Nachrichten sollten entsprechend der Aktualisierungsstrategie gesendet werden.

- `deactivate`— Der Edge-Agent sendet keine Signalaktualisierungen mehr an die Cloud.
- `fetchSnapshot`— Der Edge-Agent sendet einen einmaligen Snapshot der in der Statusvorlage definierten Signale, unabhängig davon, welche `stateTemplateUpdateStrategy` Sie bei der Anwendung der Statusvorlage auf das Fahrzeug angegeben haben.

- (Optional) `$stateTemplate.deactivateAfterSeconds` — Die Statusvorlage wird nach Ablauf der angegebenen Zeit automatisch deaktiviert. Dieser Parameter kann nur verwendet werden, wenn der Wert des `$stateTemplate.operation` Parameters „activate“ ist. Wenn dieser Parameter nicht angegeben ist oder wenn der Wert dieses Parameters 0 ist, sendet der Edge-Agent weiterhin Signalaktualisierungen an die Cloud, bis eine Deaktivierungsoperation für die Statusvorlage empfangen wird. Die Statusvorlage wird niemals automatisch deaktiviert.

  Mindestwert: 0, Höchstwert: 4294967295.

> **Note**
>
> - Die API gibt als Antwort auf eine Aktivierungsanfrage für eine Vorlage, die sich bereits im aktiven Status befindet, erfolgreich zurück.
> - Die API gibt als Antwort auf eine Deaktivierungsanfrage für eine Vorlage, die sich bereits im Deaktivierungsstatus befindet, erfolgreich zurück.
> - Die letzte Anfrage, die Sie für eine Statusvorlage stellen, ist diejenige, die wirksam wird. Wenn Sie beispielsweise die Deaktivierung einer Statusvorlage in einer Stunde beantragen und dann eine zweite Anfrage zur Deaktivierung derselben Vorlage innerhalb von vier Stunden stellen, wird die Deaktivierung innerhalb von vier Stunden wirksam, da es sich um die neueste Anforderung handelt.

> **Important**
>
> Eine Validierungsausnahme kann in jedem der folgenden Szenarien auftreten:
>
> - Es wird eine Vorlage für den Bundesstaat bereitgestellt, die nicht im `ASSOCIATED` Lieferumfang eines Fahrzeugs enthalten ist.
> - Es wurde eine Anfrage zur Aktivierung einer Statusvorlage gestellt, die jedoch noch nicht in `DEPLOYED` einem Fahrzeug enthalten war.
> - Es wird eine Anfrage an eine Statusvorlage gestellt, die sich jedoch `DELETED` auf ein Fahrzeug bezieht.

Beispiel: Aktivieren Sie eine Statusvorlage

Um eine Statusvorlage zu aktivieren, erstellen Sie zunächst eine Befehlsressource. Anschließend können Sie den folgenden Befehl an das Fahrzeug senden, für das Sie die Statusvorlage aktivieren möchten. Dieses Beispiel zeigt, wie Sie beim Erstellen eines Befehls Standardwerte für die Parameter angeben können. Diese Parameter und ihre Werte werden verwendet, wenn die Befehlsausführung gestartet wird, um die Statusvorlage zu aktivieren.

1. Erstellen Sie eine Befehlsressource

   Bevor Sie einen Befehl an das Fahrzeug senden können, müssen Sie eine Befehlsressource erstellen. Sie können alternative Werte für obligatorische Parameter angeben, wenn Sie den Befehl an das Fahrzeug senden. Weitere Informationen finden Sie unter Erstellen Sie eine Befehlsressource.

   > **Important**
   >
   > `$stateTemplate.name`und die `$stateTemplate.operation` Parameter müssen als String-Datentyp bereitgestellt werden. Wenn ein anderer Datentyp angegeben wird oder einer dieser beiden Parameter fehlt, schlägt die Befehlsausführung mit einer Validierungsausnahme fehl. Der `$stateTemplate.deactivateAfterSeconds` Parameter muss als `Long` Datentyp bereitgestellt werden.

```
aws iot create-command \
    --description "This command activates a state template on a vehicle"
```

```
    --command-id ActivateStateTemplate \
    --display-name "Activate State Template" \
    --namespace AWS-IoTFleetWise \
    --mandatory-parameters '[
    {
        "name": "$stateTemplate.name",
        "defaultValue": {"S": "ST123"}
    },
    {
        "name": "$stateTemplate.operation",
        "defaultValue": {"S": "activate"}
    },
    {
        "name": "$stateTemplate.deactivateAfterSeconds",
        "defaultValue": {"L": "120"}
    }
]'
```

2. Starten Sie die Befehlsausführung auf dem Fahrzeug

   Nachdem der Befehl erstellt wurde, senden Sie den Befehl an das Fahrzeug. Wenn Sie bei der Erstellung der Befehlsressource keine Werte für die obligatorischen Parameter angegeben haben, müssen Sie sie jetzt angeben. Weitere Informationen finden Sie unter Senden Sie einen Befehl ()AWS CLI.

   > **Important**
   >
   > Stellen Sie sicher, dass Sie den API-Endpunkt der kontospezifischen AWS IoT Auftragsdatenebene für den API-Vorgang verwenden.

```
aws iot-jobs-data start-command-execution \
    --endpoint-url <endpoint-url> \
    --command-arn arn:aws:iot:region:111122223333:command/ActivateStateTemplate \
    --target-arn arn:aws:iot:region:111122223333:thing/<VEHICLE_NAME>
```

3. Rufen Sie den Status des Vorlagenvorgangs für den Status ab

   Nachdem Sie die Befehlsausführung gestartet haben, können Sie die `GetCommandExecution` API verwenden, um die Statusvorlage abzurufen.

```
aws iot get-command-execution --execution-id <EXECUTION_ID>
```

Beispiel: Deaktivieren Sie eine Statusvorlage

Um eine Statusvorlage zu deaktivieren, erstellen Sie zunächst eine Befehlsressource. Anschließend können Sie den folgenden Befehl an das Fahrzeug senden, für das Sie die Statusvorlage deaktivieren möchten. Dieses Beispiel zeigt, wie Sie beim Erstellen eines Befehls Standardwerte für die Parameter angeben können. Diese Parameter und ihre Werte werden verwendet, wenn die Befehlsausführung gestartet wird, um die Statusvorlage zu deaktivieren.

1. Erstellen Sie eine Befehlsressource

   Bevor Sie einen Befehl an das Fahrzeug senden können, müssen Sie eine Befehlsressource erstellen. Sie können alternative Werte für obligatorische Parameter angeben, wenn Sie den Befehl an das Fahrzeug senden. Weitere Informationen finden Sie unter Erstellen Sie eine Befehlsressource.

   ```
   aws iot create-command \
       --description "This command deactivates a state template on a vehicle"
       --command-id DeactivateStateTemplate \
       --display-name "Deactivate State Template" \
       --namespace AWS-IoTFleetWise \
       --mandatory-parameters '[
       {
           "name": "$stateTemplate.name",
           "defaultValue": {"S": "ST123"}
       },
       {
           "name": "$stateTemplate.operation",
           "defaultValue": {"S": "deactivate"}
       }
   ]'
   ```

2. Starten Sie die Befehlsausführung am Fahrzeug

   Nachdem der Befehl erstellt wurde, senden Sie den Befehl an das Fahrzeug. Wenn Sie bei der Erstellung der Befehlsressource keine Werte für die obligatorischen Parameter angegeben haben, müssen Sie sie jetzt angeben. Weitere Informationen finden Sie unter Senden Sie einen Befehl ()AWS CLI.

```
aws iot-jobs-data start-command-execution \
    --endpoint-url <endpoint-url> \
    --command-arn arn:aws:iot:region:111122223333:command/DeactivateStateTemplate \
    --target-arn arn:aws:iot:region:111122223333:thing/<VEHICLE_NAME>
```

3. Rufen Sie den Status des Vorgangs mit der Statusvorlage ab

   Nachdem Sie die Befehlsausführung gestartet haben, können Sie die `GetCommandExecution` API verwenden, um die Statusvorlage abzurufen.

   ```
   aws iot get-command-execution  --execution-id <EXECUTION_ID>
   ```

Rufen Sie mithilfe von Statusvorlagen einen Fahrzeugstatus-Snapshot ab

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Sie können den letzten bekannten Status eines Fahrzeugs mithilfe der API-Operation der `CreateCommand` AWS IoT Core Steuerungsebene oder der AWS FleetWise IoT-Konsole abrufen.

> **Important**
>
> Eine Validierungsausnahme kann in jedem der folgenden Szenarien auftreten:
>
> - Es wird eine Vorlage für den Bundesstaat bereitgestellt, die nicht im `ASSOCIATED` Lieferumfang eines Fahrzeugs enthalten ist.
> - Es wurde eine Anfrage zur Aktivierung einer Statusvorlage gestellt, die jedoch noch nicht in `DEPLOYED` einem Fahrzeug enthalten war.
> - Es wird eine Anfrage an eine Statusvorlage gestellt, die sich jedoch `DELETED` auf ein Fahrzeug bezieht.

Ruft einen Fahrzeugstatus-Snapshot ab (Konsole)

Sie können die AWS FleetWise IoT-Konsole verwenden, um den letzten bekannten Status eines Fahrzeugs abzurufen. AWS IoT erstellt FleetWise einen Befehl, mit dem Sie Daten abrufen können.

Um den Status eines Fahrzeugs abzurufen

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie im Navigationsbereich Fahrzeuge aus.
3. Wählen Sie ein Fahrzeug aus der Liste aus, um die zugehörige Detailseite zu öffnen.
4. Wählen Sie auf der Registerkarte Vorlagen für Bundesstaaten die Option Daten abrufen aus.
5. Wählen Sie die IAM-Rolle aus, die AWS FleetWise IoT-Berechtigungen zum Senden eines Befehls und zum Abrufen von Daten gewährt. Weitere Informationen finden Sie unter Zugriffskontrolle.
6. Wählen Sie Status abrufen aus.

Ruft einen Fahrzeugstatus-Snapshot ab ()AWS CLI

Um einen Status-Snapshot abzurufen, erstellen Sie zunächst eine Befehlsressource. Anschließend können Sie den folgenden Befehl an das Fahrzeug senden, für das Sie den Status-Snapshot abrufen möchten. Weitere Hinweise zur Verwendung der `CreateCommand` API und ihrer Parameter finden Sie unterVerwenden der `CreateCommand`-API.

1. Erstellen Sie eine Befehlsressource

   Das folgende Beispiel zeigt, wie Sie die Befehlsressource erstellen, um den Abrufvorgang auszuführen. Sie können alternative Werte für obligatorische Parameter angeben, wenn Sie den Befehl an das Fahrzeug senden. Weitere Informationen finden Sie unter Erstellen Sie eine Befehlsressource.

   ```
   aws iot create-command \
       --command-id <COMMAND_ID> \
       --display-name "FetchSnapshot State Template" \
       --namespace AWS-IoTFleetWise \
       --mandatory-parameters '[
         {
             "name": "$stateTemplate.name",
             "defaultValue": {"S": "ST123"}
         },
   ```

```
      {
          "name": "$stateTemplate.operation",
          "defaultValue": {"S": "fetchSnapshot"}
      }
    ]'
```

Antwort:

```
{
    "commandId": "<COMMAND_ID>",
    "commandArn": "arn:aws:iot:<REGION>:111122223333:command/<COMMAND_ID>"
}
```

2. Starten Sie die Befehlsausführung, um den Status-Snapshot abzurufen

   Nachdem der Befehl erstellt wurde, senden Sie den Befehl an das Fahrzeug. Wenn Sie bei der Erstellung der Befehlsressource keine Werte für die obligatorischen Parameter angegeben haben, müssen Sie sie jetzt angeben. Weitere Informationen finden Sie unter Senden Sie einen Befehl ()AWS CLI.

   ```
   aws iot-jobs-data start-command-execution \
       --command-arn arn:aws:iot:region:111122223333:command/<COMMAND_ID> \
       --target-arn arn:aws:iot:region:111122223333:thing/<VEHICLE_NAME>
   ```

   Antwort:

   ```
   {
       "executionId": "<UNIQUE_UUID>"
   }
   ```

3. Rufen Sie den Status des Vorgangs mit der Statusvorlage ab

   Nachdem Sie die Befehlsausführung gestartet haben, können Sie die `GetCommandExecution` API verwenden, um die Statusvorlage abzurufen.

   ```
   aws iot get-command-execution --execution-id <EXECUTION_ID>
   ```

Verarbeiten Sie die letzten bekannten Statusfahrzeugdaten mithilfe von MQTT-Messaging

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Abonnieren Sie das folgende MQTT-Thema, um Updates von Ihrem Fahrzeug zu erhalten und dessen Daten zu verarbeiten. Weitere Informationen finden Sie unter MQTT-Themen im AWS IoT Core Developer Guide.

```
$aws/iotfleetwise/vehicles/$vehicle_name/last_known_state/$state_template_name/data
```

Die letzten bekannten Statussignal-Aktualisierungsnachrichten werden möglicherweise nicht in der richtigen Reihenfolge empfangen, da MQTT die Reihenfolge nicht garantiert. Alle Kunden, die MQTT zum Empfangen und Verarbeiten von Fahrzeugdaten verwenden, müssen damit umgehen. Die letzten bekannten Statussignal-Aktualisierungsnachrichten folgen dem MQTT 5-Messaging-Protokoll.

Der Nachrichtenkopf für jede MQTT-Nachricht hat die folgenden Benutzereigenschaften:

- VehicleName — Eine eindeutige Kennung der Fahrzeuge.
- stateTemplateName— Eine eindeutige Kennung der letzten bekannten Statusvorlage.

Darüber hinaus können Sie Fahrzeugattribute angeben, die in den MQTT-Nachrichtenkopf aufgenommen werden sollen, indem Sie den `metadataExtraDimensions` Anforderungsparameter bei der Aktualisierung oder Erstellung einer Statusvorlage angeben. (Siehe Vorlagen für Bundesstaaten.)

Die Benutzereigenschaften im MQTT-Nachrichtenkopf sind nützlich, um Nachrichten an verschiedene Ziele weiterzuleiten, ohne die Nutzlast zu überprüfen.

Die Nutzdaten der MQTT-Nachricht enthalten Daten, die von den Fahrzeugen gesammelt wurden. Sie können Fahrzeugattribute angeben, die in die MQTT-Nachrichtennutzlast aufgenommen werden sollen, indem Sie bei der Erstellung oder Aktualisierung einer Statusvorlage den `extraDimensions`

Anforderungsparameter angeben (siehe). Erstellen Sie eine AWS FleetWise IoT-Statusvorlage Die zusätzlichen Dimensionen bereichern die von den Fahrzeugen gesammelten Daten, indem sie ihnen zusätzliche Dimensionen zuordnen.

Die Nutzdaten der MQTT-Nachricht sind in Protokollpuffern (Protobuf) kodiert, und der MQTT-Nachrichtenkopf enthält einen Inhaltstypindikator, der als application/octet-stream definiert ist. Das Protobuf-Kodierungsschema lautet wie folgt:

```
// Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
// SPDX-License-Identifier: Apache-2.0

syntax = "proto3";

option java_package = "com.amazonaws.iot.autobahn.schemas.lastknownstate";
package Aws.IoTFleetWise.Schemas.CustomerMessage;

message LastKnownState {

  /*
   * The absolute timestamp in milliseconds since Unix Epoch of when the event was
 triggered in vehicle.
   */
  uint64 time_ms = 1;

  /*
   * This field is deprecated, use signals instead
   */
  repeated Signal signal = 2 [ deprecated = true ];

  repeated Signal signals = 3;

  repeated ExtraDimension extra_dimensions = 4;
}

message Signal {

  /*
   * The Fully Qualified Name of the signal is the path to the signal plus the signal's
 name.
   * For example, Vehicle.Chassis.SteeringWheel.HandsOff.HandsOffSteeringState
   * The fully qualified name can have up to 150 characters. Valid characters: a-z, A-
Z, 0-9, : (colon), and _ (underscore).
   */
```

```
  string name = 1;

  /*
   * The FWE reported signal value can be one of the following data types.
   */
  oneof SignalValue {
    double double_value = 2;

    bool boolean_value = 3;

    sint32 int8_value = 4;

    uint32 uint8_value = 5;

    sint32 int16_value = 6;

    uint32 uint16_value = 7;

    sint32 int32_value = 8;

    uint32 uint32_value = 9;

    sint64 int64_value = 10;

    uint64 uint64_value = 11;

    float float_value = 12;
    /*
     * An UTF-8 encoded or 7-bit ASCII string
     */
    string string_value = 13;
  }
}

message ExtraDimension {
  /*
   * The Fully Qualified Name of the attribute is the path to the attribute plus the
 attribute's name.
   * For example, Vehicle.Model.Color
   * The fully qualified name can have up to 150 characters. Valid characters: a-z, A-
Z, 0-9, : (colon), and _ (underscore).
   */
  string name = 1;
```

```
  oneof ExtraDimensionValue {
    /*
     * An UTF-8 encoded or 7-bit ASCII string
     */
    string string_value = 2;
  }
}
```

Wobei Folgendes gilt:

- `time_ms`:

  Der absolute Zeitstempel (in Millisekunden seit der Unix-Epoche), zu dem das Ereignis im Fahrzeug ausgelöst wurde. Die Edge Agent-Software verwendet für diesen Zeitstempel die Uhr des Fahrzeugs.
- `signal`:

  Ein Array von `Signal` s, das die Signalinformationen `name` (Zeichenfolge) enthält und `signalValue` die folgenden Datentypen unterstützt:`double`,`bool`,`int8`,`uint8`,`int16`,`uint16`,`int32`,`uint32`,`int64`, `uint64float`,`string`.
- `extra_dimensions`:

  Ein Array `ExtraDimensions` davon enthält Fahrzeugattributinformationen: `name` (Zeichenfolge) und `extraDimensionValue` unterstützt derzeit nur den `string` Datentyp.

Tutorial: Konfiguration der netzwerkunabhängigen Datenerfassung mithilfe einer benutzerdefinierten Dekodierungsschnittstelle

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Einführung

In diesem Tutorial wird beschrieben, wie AWS IoT so konfiguriert wird FleetWise , dass es Daten sammelt und Befehle mithilfe einer netzwerkunabhängigen Datenerfassung ausführt, die eine benutzerdefinierte Dekodierungsschnittstelle verwendet. Mit der netzwerkunabhängigen Datenerfassung können Sie Ihre eigenen Methoden verwenden, um Signale zu dekodieren, bevor Sie sie an das angegebene Datenziel senden. Das spart Zeit, da Sie keine Signaldecoder speziell für AWS IoT FleetWise erstellen müssen. Sie können eine Teilmenge von Signalen mithilfe Ihrer eigenen Implementierung dekodieren lassen oder diese `defaultForUnmappedSignals` bei der Erstellung oder Aktualisierung eines Decoder-Manifests verwenden. Dies bietet auch Flexibilität beim Sammeln von Signalen und Triggern aus einer Vielzahl von Quellen im Fahrzeug.

Dieses Tutorial ist für Fahrzeugsignale vorgesehen, die sich nicht auf einer standardmäßigen CAN-Bus-Schnittstelle (Controller Area Network) befinden. Zum Beispiel Daten, die in einem benutzerdefinierten Format oder Schema im Fahrzeug codiert sind.

Einrichtung der Umgebung

In diesem Tutorial wird davon ausgegangen, dass Sie die Schritte zum Einrichten Ihrer Umgebungen für den Zugriff auf die AWS FleetWise IoT-Cloud sowie die Edge-Implementierung APIs und die Codebasis durchgeführt haben.

Datenmodelle

Im nächsten Abschnitt wird veranschaulicht, wie Fahrzeugeigenschaften mithilfe einer benutzerdefinierten Dekodierungsschnittstelle modelliert werden. Dies gilt sowohl für die Datenerfassung als auch für Anwendungsfälle für Befehle. Dies gilt auch für alle zugrunde liegenden Datenquellenmodelle, die im Fahrzeug verwendet werden, beispielsweise IDLs.

In diesem Beispiel gibt es zwei Fahrzeugeigenschaften: einen Fahrzeugsensor (aktuelle Fahrzeugposition) zur Erfassung und einen Fahrzeugaktuator (Klimaanlage) zur Fernsteuerung. Beide sind in diesem Schema definiert:

```
// Vehicle WGS84 Coordinates
double Latitude;
double Longitude;

// Vehicle AC
Boolean ActivateAC;
```

Der nächste Schritt besteht darin, diese Definitionen FleetWise mithilfe der benutzerdefinierten Dekodierungsschnittstelle APIs in AWS IoT zu importieren.

Aktualisierungen des Signalkatalogs

Importieren Sie diese Definitionen in Ihren Signalkatalog. Wenn Sie FleetWise bereits einen Signalkatalog in AWS IoT haben, verwenden Sie die Update-API direkt. Wenn Sie noch keinen haben, erstellen Sie zuerst einen Signalkatalog und rufen Sie dann die Update-API auf.

Zunächst müssen Sie die VSS-Darstellung dieser Fahrzeugsignale erstellen. VSS wird als Taxonomie zur Darstellung von Fahrzeugdaten im IoT verwendet. AWS FleetWise Erstellen Sie eine JSON-Datei mit dem Namen „vehicle-signals.json“ mit diesem Inhalt:

```
// vehicle-signals.json
// Verify that branches and nodes are unique in terms of fully qualified name
// in the signal catalog.
[
 {
    "branch": {
      "fullyQualifiedName": "Vehicle",
      "description": "Vehicle Branch"
    }
```

```
  },
  {
    "branch": {
      "fullyQualifiedName": "Vehicle.CurrentLocation",
      "description": "CurrentLocation"
    }
  },
  {
    "sensor": {
      "dataType": "DOUBLE",
      "fullyQualifiedName": "Vehicle.CurrentLocation.Latitude",
      "description": "Latitude"
    }
  },
  {
    "sensor": {
      "dataType": "DOUBLE",
      "fullyQualifiedName": "Vehicle.CurrentLocation.Longitude",
      "description": "Longitude"
    }
  },
  {
    "actuator": {
      "fullyQualifiedName": "Vehicle.ActivateAC",
      "description": "AC Controller",
      "dataType": "BOOLEAN"
    }
  }
]
```

Wenn Sie keinen Signalkatalog haben, müssen Sie Folgendes aufrufen: `create-signal-catalog`

```
VEHICLE_NODES=`cat vehicle-signals.json`
aws iotfleetwise create-signal-catalog \
        --name my-signal-catalog \
        --nodes "${VEHICLE_NODES}"
```

Wenn Sie bereits einen Signalkatalog haben, können Sie diese Signale mithilfe der `update-signal-catalog` API hinzufügen:

```
VEHICLE_NODES=`cat vehicle-signals.json`
aws iotfleetwise update-signal-catalog \
        --name my-signal-catalog \
```

```
        --nodes-to-add "${VEHICLE_NODES}"
```

Fahrzeugmodell und Decoder

Nachdem Sie die Signale in den Signalkatalog eingefügt haben, besteht der nächste Schritt darin, ein Fahrzeugmodell zu erstellen und diese Signale zu instanziieren. Dafür verwenden Sie das `create-model-manifest` und. `create-decoder-manifest` APIs

Formatieren Sie zunächst die Signalnamen, die Sie in das Fahrzeugmodell einfügen möchten:

```
# Prepare the signals for insertion into the vehicle model.
VEHICLE_NODES=`cat vehicle-signals.json`
VEHICLE_NODES=`echo ${VEHICLE_NODES} | jq -r ".[] | .actuator,.sensor
 | .fullyQualifiedName" | grep Vehicle\\.`
VEHICLE_NODES=`echo "${VEHICLE_NODES}" | jq -Rn [inputs]`
# This is how the vehicle model input looks.
echo $VEHICLE_NODES
# [ "Vehicle.CurrentLocation.Latitude",
#   "Vehicle.CurrentLocation.Longitude",
#   "Vehicle.ActivateAC" ]
# Create the vehicle model with those signals.
aws iotfleetwise create-model-manifest \
    --name my-model-manifest \
    --signal-catalog-arn arn:xxxx:signal-catalog/my-signal-catalog \
    --nodes "${VEHICLE_NODES}"

# Activate the vehicle model.
 aws iotfleetwise update-model-manifest \
    --name my-model-manifest --status ACTIVE
```

Verwenden Sie jetzt die benutzerdefinierte Dekodierungsschnittstelle, um ein Decoder-Manifest zu erstellen.

> **Note**
>
> Sie müssen nur Netzwerkschnittstellen und Signale erstellen, wenn Sie benutzerdefinierte Werte angeben möchten IDs, was in diesem Beispiel nicht enthalten ist.
> Informationen zur Zuordnung von Dekodierungsinformationen, wenn sich der vollqualifizierte Name (FQN) von der benutzerdefinierten Dekodierungssignal-ID unterscheidet, finden Sie im Edge Agent Developer Guide.

```
// Create a network interface that is of type : CUSTOM_DECODING_INTERFACE
// custom-interface.json
[
  {
    "interfaceId": "NAMED_SIGNAL",
    "type": "CUSTOM_DECODING_INTERFACE",
    "customDecodingInterface": {
      "name": "NamedSignalInterface"
    }
  },
  {
    "interfaceId": "AC_ACTUATORS",
    "type": "CUSTOM_DECODING_INTERFACE",
    "customDecodingInterface": {
      "name": "NamedSignalInterface"
    }
  }
]
// custom-decoders.json
// Refer to the fully qualified names of the signals, make them of
// type CUSTOM_DECODING_SIGNAL, and specify them as part of the same interface ID
// that was defined above.
[
    {
      "fullyQualifiedName": "Vehicle.CurrentLocation.Longitude",
      "interfaceId": "NAMED_SIGNAL",
      "type": "CUSTOM_DECODING_SIGNAL",
      "customDecodingSignal": {
        "id": "Vehicle.CurrentLocation.Longitude"
      }
    },
    {
      "fullyQualifiedName": "Vehicle.CurrentLocation.Latitude",
      "interfaceId": "NAMED_SIGNAL",
      "type": "CUSTOM_DECODING_SIGNAL",
      "customDecodingSignal": {
        "id": "Vehicle.CurrentLocation.Latitude"
      }
    },
    {
        "fullyQualifiedName": "Vehicle.ActivateAC",
        "interfaceId": "AC_ACTUATORS",
        "type": "CUSTOM_DECODING_SIGNAL",
```

```
        "customDecodingSignal": {
          "id": "Vehicle.ActivateAC"
        }
    }
]
# Create the decoder manifest.
 CUSTOM_INTERFACE=`cat custom-interface.json`
 CUSTOM_DECODERS=`cat custom-decoders.json`

aws iotfleetwise create-decoder-manifest \
   --name my-decoder-manifest \
   --model-manifest-arn arn:xxx:model-manifest/my-model-manifest \
   --network-interfaces "${CUSTOM_INTERFACE}" \
   --signal-decoders "${CUSTOM_DECODERS}"

 # Activate the decoder manifest.
 aws iotfleetwise update-decoder-manifest \
    --name my-decoder-manifest \
    --status ACTIVE
```

Zu diesem Zeitpunkt haben Sie diese Signale im AWS IoT FleetWise vollständig modelliert. Als Nächstes erstellen Sie das Fahrzeug und verknüpfen es mit dem von Ihnen erstellten Modell. Sie verwenden dafür die `create-vehicle` API:

```
aws iotfleetwise create-vehicle \
    --decoder-manifest-arn arn:xxx:decoder-manifest/my-decoder-manifest \
    --association-behavior ValidateIotThingExists \
    --model-manifest-arn arn:xxx:model-manifest/my-model-manifest \
    --vehicle-name "my-vehicle"
```

Der nächste Schritt besteht darin, sich auf die AWS IoT FleetWise Edge-Codebasis zu konzentrieren und die erforderliche Codeerweiterung zu schreiben.

> **Note**
>
> Informationen zur Edge-Implementierung finden Sie im Edge Agent Developer Guide.

Befehl senden

Kompilieren Sie nun die Software (stellen Sie sicher, dass Sie Ihre Header und C++-Dateien zur CMake Datei hinzufügen) und kehren Sie dann zur Cloud zurück, APIs um einen Befehl auf diesem Aktuator zu testen:

```
// Create a command targeting your vehicle.
aws iot create-command --command-id activateAC \
    --namespace "AWS-IoT-Fleetwise" \
    --endpoint-url endpoint-url \
    --role-arn ${SERVICE_ROLE_ARN} \
    --mandatory-parameters '[ { "name": "$actuatorPath.Vehicle.ActivateAC",
 "defaultValue": {"B": "false"} } ]' \
// You will receive the command ARN.

{
    "commandId": "activateAC",
    "commandArn": "arn:aws:iot:xxx:command/activateAC"
}

// You can send the command to activate the AC targeting your vehicle.

JOBS_ENDPOINT_URL=`aws iot describe-endpoint --endpoint-type iot:Jobs | jq -
j .endpointAddress`
aws iot-jobs-data start-command-execution \
    --command-arn arn:aws:iot:xxx:command/activateAC \
    --target-arn arn:xxx:vehicle/my-vehicle \
    --parameters '{ "$actuatorPath.Vehicle.ActivateAC" : {"B": "true"}}' \
    --endpoint-url https://${JOBS_ENDPOINT_URL}
// You will receive the corresponding execution ID.
{
    "executionId": "01HSK4ZH6ME7D43RB2BV8JC51D"
}

// If you have the AWS IoT FleetWise Edge Agent running, you can see the logs.
[AcCommandDispatcher.cpp:26] [setActuatorValue()]:
[Actuator Vehicle.ActivateAC executed successfully for command ID
 01HSK4ZH6ME7D43RB2BV8JC51D]
```

Verwenden Sie AWS CLI und AWS SDKs mit AWS IoT FleetWise

Dieser Abschnitt enthält Informationen zum Stellen von AWS FleetWise IoT-API-Anfragen. Weitere Informationen zu AWS FleetWise IoT-Vorgängen und Datentypen finden Sie in der AWS FleetWise IoT-API-Referenz.

Um AWS IoT FleetWise mit einer Vielzahl von Programmiersprachen zu verwenden, verwenden Sie die AWS SDKs, die die folgenden automatischen Funktionen enthalten:

- Kryptographisches Signieren Ihrer Serviceanfragen
- Wiederholen von Anfragen
- Umgang mit Fehlerreaktionen

Verwenden Sie für den Befehlszeilenzugriff AWS IoT FleetWise mit dem AWS CLI. Sie können AWS IoT FleetWise und Ihre anderen Dienste von der Befehlszeile aus steuern und sie mithilfe von Skripten automatisieren.

AWS IoT-Problembehebung FleetWise

Verwenden Sie die Informationen und Lösungen zur Fehlerbehebung in diesem Abschnitt, um Probleme mit AWS IoT zu lösen FleetWise.

Die folgenden Informationen können Ihnen helfen, häufig auftretende Probleme mit AWS IoT zu beheben FleetWise.

Topics

- AWS Probleme mit dem FleetWise IoT-Decoder-Manifest
- FleetWise Softwareprobleme mit dem Edge Agent für AWS IoT
- Probleme speichern und weiterleiten

AWS Probleme mit dem FleetWise IoT-Decoder-Manifest

Beheben Sie Probleme mit dem Decoder-Manifest.

Diagnose von API-Aufrufen des Decoder-Manifests

Fehler	Richtlinien für die Fehlerbehebung
`UpdateOperationFailure.ConflictingDecoderUpdate`	Das gleiche Decoder-Manifest enthält mehrere Aktualisierungsanforderungen. Warten Sie und versuchen Sie es erneut.
`UpdateOperationFailure.InternalFailure`	InternalFailure wird als gekapselte Ausnahme gestartet. Das Problem selbst hängt von der gekapselten Ausnahme ab.
`UpdateOperationFailure.ActiveDecoderUpdate`	Das Decoder-Manifest befindet sich in einem `Active` Zustand und kann nicht aktualisiert werden. Ändern Sie den Status des Decoder-Manifests auf`DRAFT`, und versuchen Sie es erneut.
`UpdateOperationFailure.ConflictingModelUpdate`	AWS IoT FleetWise versucht, anhand eines Fahrzeugmodells (Modellmanifest) zu validiere

Fehler	Richtlinien für die Fehlerbehebung
	n, das von einer anderen Person geändert wird. Warten Sie und versuchen Sie es erneut.
`UpdateOperationFailure.ModelManifestValidationResponse : FailureReason.MODEL_DATA_ENTRIES_NOT_FOUND`	Dem Fahrzeugmodell sind keine Signale zugeordnet. Fügen Sie dem Fahrzeugmodell Signale hinzu und überprüfen Sie, ob die Signale im zugehörigen Signalkatalog zu finden sind.
`UpdateOperationFailure.ModelManifestValidationResponse : FailureReason.MODEL_NOT_ACTIVE`	Aktualisieren Sie das Fahrzeugmodell, sodass es den aktuellen `ACTIVE` Status aufweist, und versuchen Sie es erneut.
`UpdateOperationFailure.ModelManifestValidationResponse : FailureReason.MODEL_NOT_FOUND`	AWS IoT FleetWise kann das mit dem Decoder-Manifest verknüpfte Fahrzeugmodell nicht finden. Überprüfen Sie den Amazon Resource Name (ARN) des Fahrzeugmodells und versuchen Sie es erneut.
`UpdateOperationFailure.ModelManifestValidationResponse (FailureReason.MODEL_DATA_ENTRIES_READ_FAILURE`	Die Validierung des Fahrzeugmodells ist fehlgeschlagen, weil die Signalnamen des Fahrzeugmodells nicht im Signalkatalog gefunden wurden. Stellen Sie sicher, dass die Signale im Fahrzeugmodell alle im zugehörigen Signalkatalog enthalten sind.
`UpdateOperationFailure.ValidationFailure`	In der Anforderung zur Aktualisierung des Decoder-Manifests wurden ungültige Signale oder Netzwerkschnittstellen gefunden. Stellen Sie sicher, dass alle von der Ausnahme zurückgegebenen Signale und Netzwerks chnittstellen vorhanden sind, dass alle verwendeten Signale einer verfügbaren Schnittstelle zugeordnet sind und dass Sie keine Schnittstelle entfernen, der Signale zugeordnet sind.

Fehler	Richtlinien für die Fehlerbehebung
`UpdateOperationFailure.KmsKeyAccessDenied`	Bei der für den Vorgang verwendeten Taste AWS Key Management Service (AWS KMS) ist ein Berechtigungsproblem aufgetreten. Stellen Sie sicher, dass Sie eine Rolle verwenden, die Zugriff auf den Schlüssel hat, und versuchen Sie es erneut.
`UpdateOperationFailure.DecoderDoesNotExist`	Das Decoder-Manifest ist nicht vorhanden. Überprüfen Sie den Namen des Decoder-Manifests und versuchen Sie es erneut.

Fehlermeldungen in den Daten des Vision-Systems mit Angabe des `SIGNAL_DECODER_INCOMPATIBLE_WITH_SIGNAL_CATALOG` Grundes enthalten in der Antwort einen Hinweis, der Aufschluss darüber gibt, warum die Anfrage fehlgeschlagen ist. Anhand des Hinweises können Sie festlegen, welche Richtlinien zur Fehlerbehebung befolgt werden müssen.

> **Note**
>
> Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Diagnose, Decoder, Validierung der Daten des Bildverarbeitungssystems

Fehler	Richtlinien für die Fehlerbehebung
`InvalidSignalDecoder.withReason(SignalDecoderFailureReason.NO_SIGNAL_IN_CATALOG_FOR_DECODER_SIGNAL)`	AWS IoT FleetWise hat die im Signaldecoder verwendete Wurzelsignalstruktur mithilfe des Signalkatalogs nicht gefunden. Stellen Sie sicher, dass das Wurzelsignal der Struktur im Signalkatalog korrekt definiert ist.
`InvalidSignalDecoder.withReason(SignalDecoderFailureReason.SIGNAL_DECODER_TYPE_IN`	Eine primitive Nachricht im Signalkatalog wurde nicht mit demselben Datentyp in der Aktualisierungsanforderung des Decoder-Manifests definiert. Stellen Sie sicher, dass die in der

Fehler	Richtlinien für die Fehlerbehebung
`COMPATIBLE_WITH_MESSAGE_SIG NAL_TYPE)`	Anfrage definierten primitiven Nachrichten mit der entsprechenden Signalkatalogdefinition übereinstimmen.
`InvalidSignalDecoder.withRe ason(SignalDecoderFailureRe ason.STRUCT_SIZE_MISMATCH)`	Die Anzahl der in einer Struktur im Signalkat alog definierten Eigenschaften entspricht nicht der Anzahl der Eigenschaften, die Sie im Decoder-Manifest zu dekodieren versuchen. Stellen Sie sicher, dass Sie die richtige Anzahl von Signalen zum Dekodieren haben, indem Sie diese mit den im Signalkatalog definierten Signalen vergleichen.
`InvalidSignalDecoder.withRe ason(SignalDecoderFailureRe ason.SIGNAL_DECODER_INCOMPA TIBLE_WITH_SIGNAL_CATALOG)`	AWS IoT FleetWise hat im Signalkatalog ein als STRUCT definiertes Signal ohne eine im Decoder structuredMessageDefinition definierte Manifest-Anfrage gefunden. Stellen Sie sicher, dass jede Struktur als Aktualisierungsanf orderung structuredMessageDefinition im Decoder-Manifest definiert ist.
`InvalidSignalDecoder.withRe ason(SignalDecoderFailureRe ason.SIGNAL_DECODER_INCOMPA TIBLE_WITH_SIGNAL_CATALOG)`	Das Wurzelsignal der im Decoder-Manifest verwendeten Struktur ist nicht korrekt als Struktur im Signalkatalog definiert. Für die im Decoder-Manifest verwendete Wurzelsig nalstruktur muss das Feld structFullyQualified Name definiert sein. Dazu benötigt sie auch einen STRUCT-Knoten. fullyQualifiedName
`InvalidSignalDecoder.withRe ason(SignalDecoderFailureRe ason.SIGNAL_DECODER_INCOMPA TIBLE_WITH_SIGNAL_CATALOG)`	Eine der in der Decoder-Manifest-Anfrage verwendeten Blattnachrichten ist nicht als primitive Nachricht definiert. Stellen Sie sicher, dass alle Blattobjekte in der Anfrage als primitive Nachrichten definiert sind.

Fehler	Richtlinien für die Fehlerbehebung
`InvalidSignalDecoder.withReason(SignalDecoderFailureReason.SIGNAL_DECODER_INCOMPATIBLE_WITH_SIGNAL_CATALOG)`	Ein Array-Objekt im Signalkatalog wurde in der Aktualisierungsanforderung des Decoder-Manifests nicht als structuredMessageList Definition definiert. Stellen Sie sicher, dass alle Array-Eigenschaften in der Aktualisierungsanforderung für das Decoder-Manifest als structuredMessageList Definition definiert sind.

FleetWise Softwareprobleme mit dem Edge Agent für AWS IoT

Beheben Sie Probleme mit der Edge Agent-Software.

Problembereiche

- Problem: Die Edge Agent-Software startet nicht.
- Problem: [FEHLER] [Io TFleetWiseEngine: :connect]: [Persistenzbibliothek konnte nicht initialisiert werden]
- Problem: Die Edge Agent-Software erfasst keine On-Board-Diagnosen (OBD) II PIDs und Diagnose-Fehlercodes (). DTCs
- Problem: Die Edge Agent for AWS FleetWise IoT-Software sammelt keine Daten aus dem Netzwerk oder kann keine Dateninspektionsregeln anwenden.
- Problem: [FEHLER] [AwsIotConnectivityModule: :connect]: [Verbindung mit Fehler fehlgeschlagen] oder [WARN] [AwsIotChannel: :send]: [Keine aktive MQTT-Verbindung.]

Problem: Die Edge Agent-Software startet nicht.

Möglicherweise werden die folgenden Fehler angezeigt, wenn die Edge Agent-Software nicht gestartet wird.

- ```
  Error from reader: * Line 1, Column 1
  Syntax error: value, object or array expected.
  ```
  ```

Lösung: Stellen Sie sicher, dass die FleetWise Softwarekonfigurationsdatei für den Edge-Agenten für AWS IoT ein gültiges JSON-Format verwendet. Stellen Sie z. B. sicher, dass Kommas korrekt verwendet werden. Gehen Sie wie folgt vor, um weitere Informationen zur Konfigurationsdatei zu erhalten, um das Edge Agent for AWS FleetWise IoT-Softwareentwicklerhandbuch herunterzuladen.

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Wählen Sie auf der Service-Startseite im FleetWise Abschnitt Erste Schritte mit AWS IoT die Option Explore Edge Agent aus.

- ```
  [ERROR] [SocketCANBusChannel::connect]: [ SocketCan with name xxx is not accessible]
  [ERROR] [IoTFleetWiseEngine::connect]: [ Failed to Bind Consumers to Producers ]
  ```

  Lösung: Dieser Fehler wird möglicherweise angezeigt, wenn die Edge Agent-Software keine Socket-Kommunikation mit den in der Konfigurationsdatei definierten Netzwerkschnittstellen herstellen kann.

  Führen Sie den folgenden Befehl aus, um zu überprüfen, ob jede in der Konfiguration definierte Netzwerkschnittstelle verfügbar ist.

  ```
  ip link show
  ```

  Führen Sie den folgenden Befehl aus, um eine Netzwerkschnittstelle online zu schalten. *network-interface-id*Ersetzen Sie es durch die ID der Netzwerkschnittstelle.

  ```
  sudo ip link set network-interface-id up
  ```

- ```
  [ERROR] [AwsIotConnectivityModule::connect]: [Connection failed with error]
  [WARN] [AwsIotChannel::send]: [No alive MQTT Connection.]
  # or
  [WARN] [AwsIotChannel::send]: [aws-c-common: AWS_ERROR_FILE_INVALID_PATH]
  ```

  Lösung: Dieser Fehler wird möglicherweise angezeigt, wenn die Edge Agent-Software keine MQTT-Verbindung zu AWS IoT Core herstellen kann. Überprüfen Sie, ob die folgenden Komponenten korrekt konfiguriert sind, und starten Sie die Edge Agent-Software neu.

  - `mqttConnection::endpointUrl`— Endpunkt des IoT-Geräts des AWS Kontos.

- `mqttConnection::clientID`— Die ID des Fahrzeugs, in dem die Edge Agent-Software ausgeführt wird.
- `mqttConnection::certificateFilename`— Der Pfad zur Fahrzeugzertifikatsdatei.
- `mqttConnection::privateKeyFilename`— Der Pfad zur Datei mit dem privaten Fahrzeugschlüssel.
- Sie haben es AWS IoT Core zur Bereitstellung des Fahrzeugs verwendet. Weitere Informationen finden Sie unter Bereitstellung von AWS FleetWise IoT-Fahrzeugen.

Weitere Informationen zur Fehlerbehebung finden Sie unter AWS IoT Device SDK for C++ Häufig gestellte Fragen.

Problem: [FEHLER] [Io TFleetWiseEngine: :connect]: [Persistenzbibliothek konnte nicht initialisiert werden]

Lösung: Dieser Fehler wird möglicherweise angezeigt, wenn die Edge Agent-Software den Persistenzspeicher nicht finden kann. Vergewissern Sie sich, dass Folgendes korrekt konfiguriert ist, und starten Sie die Edge Agent-Software neu.

`persistency:persistencyPath`— Ein lokaler Pfad, der zur Beibehaltung von Sammlungsschemas, Decodermanifesten und Datenschnappschüssen verwendet wird.

Problem: Die Edge Agent-Software erfasst keine On-Board-Diagnosen (OBD) II PIDs und Diagnose-Fehlercodes (). DTCs

Lösung: Dieser Fehler wird möglicherweise angezeigt, wenn `obdInterface:pidRequestIntervalSeconds` oder auf `obdInterface:dtcRequestIntervalSeconds` 0 konfiguriert ist.

Wenn die Edge Agent-Software in einem Fahrzeug mit Automatikgetriebe ausgeführt wird, stellen Sie sicher, dass sie auf konfiguriert `obdInterface:hasTransmissionEcu` ist`true`.

Wenn Ihr Fahrzeug die erweiterte Controller Area Network (CAN-Bus) -Arbitrierung unterstützt IDs, stellen Sie sicher, dass dies konfiguriert `obdInterface:useExtendedIds` ist`true`.

Problem: Die Edge Agent for AWS FleetWise IoT-Software sammelt keine Daten aus dem Netzwerk oder kann keine Dateninspektionsregeln anwenden.

Lösung: Dieser Fehler wird möglicherweise angezeigt, wenn die Standardkontingente überschritten werden.

Ressource	Kontingent	Einstellbar	Hinweis
Wert der Signal-ID	Die Signal-ID muss kleiner oder gleich 50.000 sein	Ja	Die Edge Agent-Software sammelt keine Daten von Signalen, deren ID größer als 50.000 ist. Wir empfehlen Ihnen, zu überprüfen, wie viele Signale der Signalkatalog enthält, bevor Sie dieses Kontingent ändern.
Anzahl der aktiven Datenerfassungssysteme pro Fahrzeug	256	Ja	Wir empfehlen Ihnen, zu überprüfen, wie viele Kampagnen Sie in der Cloud erstellt haben und wie viele Schemas jede Kampagne enthält, bevor Sie dieses Kontingent ändern.
Größe des Puffers für die Signalhistorie	20 MB	Ja	Wenn das Kontingent überschritten wird, hört die Edge Agent-Software auf, neue Daten zu sammeln.

Problem: [FEHLER] [AwsIotConnectivityModule: :connect]: [Verbindung mit Fehler fehlgeschlagen] oder [WARN] [AwsIotChannel: :send]: [Keine aktive MQTT-Verbindung.]

Lösung: Dieser Fehler wird möglicherweise angezeigt, wenn die Edge Agent-Software nicht mit der Cloud verbunden ist. Standardmäßig sendet die Edge Agent-Software AWS IoT Core jede Minute eine Ping-Anfrage und wartet drei Minuten. Wenn keine Antwort erfolgt, stellt die Edge Agent-Software die Verbindung zur Cloud automatisch wieder her.

Probleme speichern und weiterleiten

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Problem: Empfangen eines **`AccessDeniedException`** mit allen erforderlichen IAM-Berechtigungen

Lösung: Die Funktion „Speichern und Weiterleiten“ für die Datenpartitionierung in Kampagnen erfordert einen eingeschränkten Zugriff über eine Zulassungsliste. Wenden Sie sich an das Serviceteam, um sicherzustellen, dass Ihre Ressourcen über die erforderlichen Berechtigungen verfügen, indem Sie die Option „Listing zulassen“ aktivieren.

Problem: Die in AWS IoT Jobs hochgeladenen Daten ignorieren die **`endTime`**

Lösung: Sie haben im Jobdokument einen `endtime` ungültigen Wert angegeben. Zum Beispiel entspricht das `endtime` nicht dem ISO 8601 UTC-Format). In den AWS IoT FleetWise Agent-Protokollen könnte es eine Warnung geben, die besagt,. `Malformed IoT Job endTime:` **`customer configured endTime.`** `Not setting endTime`

Problem: Der Datenupload in AWS IoT Jobs hat den Status „Ausführung“. `REJECTED`

Lösung: Sie haben `campaignArn` im Jobdokument einen ungültigen Wert angegeben. Wenn Sie beispielsweise einen ARN für eine Kampagne angeben, die nicht auf einem Fahrzeug läuft, könnte es `CampaignArn value in the received job document does not match the ARN of a Store and Forward campaign` in den AWS IoT FleetWise Agent-Protokollen eine Anweisung auf Fehlerebene geben, die besagt, dass

Sicherheit im AWS IoT FleetWise

Cloud-Sicherheit AWS hat höchste Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung von Ihnen AWS und Ihnen. Das Modell der übergreifenden Verantwortlichkeit beschreibt dies als Sicherheit der Cloud und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der AWS . Weitere Informationen zu den Compliance-Programmen, die für das Internet der AWS IoT gelten FleetWise, finden Sie unter AWS-Services in Umfang nach Compliance-Programm .
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von AWS IoT anwenden können FleetWise. Es zeigt Ihnen, wie Sie AWS IoT konfigurieren FleetWise , um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie lernen auch, wie Sie andere AWS Dienste nutzen können, die Ihnen helfen, Ihre AWS FleetWise IoT-Ressourcen zu überwachen und zu sichern.

Inhalt

- Datenschutz im AWS Internet der Dinge FleetWise
- Steuerung des Zugriffs mit AWS IoT FleetWise
- Identity and Access Management für AWS IoT FleetWise
- AWS verwaltete Richtlinienaktualisierungen für AWS IoT FleetWise
- Konformitätsvalidierung für das AWS IoT FleetWise
- Resilienz im AWS Internet der Dinge FleetWise
- Infrastruktursicherheit im AWS Internet der Dinge FleetWise
- Konfiguration und Schwachstellenanalyse im AWS IoT FleetWise

- Bewährte Sicherheitsmethoden für das AWS IoT FleetWise

Datenschutz im AWS Internet der Dinge FleetWise

Das Modell der AWS gemeinsamen Verantwortung, der , gilt für den Datenschutz im AWS Internet der Dinge FleetWise. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der alle Systeme laufen AWS Cloud. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Sie sind auch für die Sicherheitskonfiguration und die Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services verantwortlich. Weitere Informationen zum Datenschutz finden Sie unter Häufig gestellte Fragen zum Datenschutz. Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag AWS -Modell der geteilten Verantwortung und in der DSGVO im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie AWS-Konto Anmeldeinformationen schützen und einzelne Benutzer mit AWS IAM Identity Center oder AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem empfehlen wir, die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Wird verwendet SSL/TLS , um mit AWS Ressourcen zu kommunizieren. Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein AWS CloudTrail. Informationen zur Verwendung von CloudTrail Pfaden zur Erfassung von AWS Aktivitäten finden Sie unter Arbeiten mit CloudTrail Pfaden im AWS CloudTrail Benutzerhandbuch.
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen darin enthaltenen Standardsicherheitskontrollen AWS-Services.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu schützen.
- Wenn Sie für den Zugriff AWS über eine Befehlszeilenschnittstelle oder eine API FIPS 140-3-validierte kryptografische Module benötigen, verwenden Sie einen FIPS-Endpunkt. Weitere Informationen über verfügbare FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-3.

Wir empfehlen dringend, in Freitextfeldern, z. B. im Feld Name, keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit AWS IoT FleetWise oder anderen Geräten arbeiten und die Konsole, die API oder AWS-Services

verwenden AWS SDKs. AWS CLI Alle Daten, die Sie in Tags oder Freitextfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

AWS IoT FleetWise ist für die Verwendung mit einem Edge-Agenten vorgesehen, den Sie entwickeln und auf unterstützter Fahrzeughardware installieren, um Fahrzeugdaten in die AWS Cloud zu übertragen. Das Extrahieren von Daten aus Fahrzeugen kann in bestimmten Ländern den Datenschutzbestimmungen unterliegen. Bevor Sie AWS IoT verwenden FleetWise und Ihren Edge Agent installieren, empfehlen wir Ihnen dringend, Ihre Compliance-Verpflichtungen nach geltendem Recht zu überprüfen. Dazu gehören alle geltenden gesetzlichen Anforderungen zur Bereitstellung rechtlich angemessener Datenschutzhinweise und zur Einholung aller erforderlichen Einwilligungen für die Extraktion von Fahrzeugdaten.

Verschlüsselung im Ruhezustand im AWS Internet der Dinge FleetWise

Die von einem Fahrzeug gesammelten Daten werden über eine AWS IoT Core Nachricht mit dem MQTT-Nachrichtenprotokoll in die Cloud übertragen. AWS IoT FleetWise liefert die Daten an Ihre Amazon Timestream Timestream-Datenbank. In Timestream sind Ihre Daten verschlüsselt. Alle AWS-Services verschlüsseln standardmäßig Daten im Ruhezustand. Weitere Informationen finden Sie unter Schützen von Daten durch Verschlüsselung im Amazon S3 S3-Benutzerhandbuch und Datenschutz in Timestream für LiveAnalytics.

Encryption at Rest ist in AWS Key Management Service (AWS KMS) integriert, um den Verschlüsselungsschlüssel zu verwalten, der zur Verschlüsselung Ihrer Daten verwendet wird. Sie können wählen, ob Sie einen vom Kunden verwalteten Schlüssel verwenden möchten, um die vom AWS IoT FleetWise gesammelten Daten zu verschlüsseln. Sie können Ihren Verschlüsselungsschlüssel über AWS KMS erstellen, verwalten und einsehen. Weitere Informationen finden Sie unter Was ist AWS Key Management Service? im AWS Key Management Service Entwicklerhandbuch.

Verschlüsselung während der Übertragung

Alle mit AWS IoT Diensten ausgetauschten Daten werden bei der Übertragung mithilfe von Transport Layer Security (TLS) verschlüsselt. Weitere Informationen finden Sie unter Transportsicherheit im AWS IoT -Entwicklerhandbuch.

AWS IoT Core Unterstützt außerdem Authentifizierung und Autorisierung, um den Zugriff auf AWS FleetWise IoT-Ressourcen sicher zu kontrollieren. Fahrzeuge können X.509-Zertifikate verwenden, um sich für die Nutzung des AWS IoT zu authentifizieren (anzumelden), FleetWise und AWS IoT Core Richtlinien verwenden, um sich autorisieren zu lassen (über Berechtigungen zu verfügen), um bestimmte Aktionen auszuführen. Weitere Informationen finden Sie unter the section called “Fahrzeuge bereitstellen”.

Datenverschlüsselung im AWS IoT FleetWise

Datenverschlüsselung bezieht sich auf den Schutz von Daten während der Übertragung (auf dem Weg zum und vom AWS IoT FleetWise sowie zwischen Gateways und Servern) und im Ruhezustand (während sie auf lokalen Geräten oder in AWS-Services) gespeichert werden. Sie können Daten im Ruhezustand mithilfe einer clientseitigen Verschlüsselung schützen.

> **Note**
>
> AWS FleetWise IoT-Edge-Processing-Exposés APIs , die in AWS FleetWise IoT-Gateways gehostet werden und über das lokale Netzwerk zugänglich sind. Diese APIs werden über eine TLS-Verbindung bereitgestellt, die durch ein Serverzertifikat unterstützt wird, das dem AWS IoT FleetWise Edge-Connector gehört. Für die Client-Authentifizierung APIs verwenden diese ein Passwort für die Zugriffskontrolle. Der private Schlüssel des Serverzertifikats und das Passwort für die Zugriffskontrolle werden beide auf der Festplatte gespeichert. AWS Die FleetWise IoT-Edge-Verarbeitung stützt sich auf die Dateisystemverschlüsselung, um die Sicherheit dieser gespeicherten Anmeldeinformationen zu gewährleisten.

Weitere Informationen zur serverseitigen Verschlüsselung und zur clientseitigen Verschlüsselung finden Sie in den unten aufgeführten Themen.

Inhalt

- Verschlüsselung im Ruhezustand im AWS Internet der Dinge FleetWise
- Schlüsselmanagement im AWS Internet der Dinge FleetWise

Verschlüsselung im Ruhezustand im AWS Internet der Dinge FleetWise

AWS IoT FleetWise speichert Ihre Daten in der AWS Cloud und auf Gateways.

Daten im Ruhezustand in der Cloud AWS

AWS IoT FleetWise speichert Daten in anderen AWS-Services , die Daten im Ruhezustand standardmäßig verschlüsseln. Encryption at Rest ist in AWS Key Management Service (AWS KMS) integriert, um den Verschlüsselungsschlüssel zu verwalten, der zur Verschlüsselung Ihrer Immobilienwerte und aggregierten Werte im AWS IoT FleetWise verwendet wird. Sie können sich dafür entscheiden, einen vom Kunden verwalteten Schlüssel zu verwenden, um Immobilienwerte und aggregierte Werte im AWS IoT FleetWise zu verschlüsseln. Sie können Ihren Verschlüsselungsschlüssel über AWS KMS erstellen, verwalten und einsehen.

Sie können einen AWS-eigener Schlüssel oder einen vom Kunden verwalteten Schlüssel wählen, um Ihre Daten zu verschlüsseln.

Funktionsweise

Encryption at Rest ist in die Verwaltung des Verschlüsselungsschlüssels integriert, der zur Verschlüsselung Ihrer Daten verwendet wird. AWS KMS

- AWS-eigener Schlüssel — Standard-Verschlüsselungsschlüssel. AWS IoT FleetWise besitzt diesen Schlüssel. Sie können diesen Schlüssel nicht in Ihrem anzeigen, verwalten oder verwenden AWS-Konto. Sie können auch keine Operationen mit dem Schlüssel in AWS CloudTrail Protokollen sehen. Sie können diesen Schlüssel ohne zusätzliche Kosten verwenden.
- Vom Kunden verwalteter Schlüssel — Der Schlüssel wird in Ihrem Konto gespeichert, das Sie erstellen, besitzen und verwalten. Sie haben die volle Kontrolle über den KMS-Schlüssel. Es AWS KMS fallen zusätzliche Gebühren an.

AWS-eigene Schlüssel

AWS-eigene Schlüssel sind nicht in Ihrem Konto gespeichert. Sie sind Teil einer Sammlung von KMS-Schlüsseln, die mehrere AWS besitzen und verwalten, sodass sie in mehreren Fällen verwendet AWS-Konten werden können. AWS-Services kann AWS-eigene Schlüssel zum Schutz Ihrer Daten verwendet werden.

Sie können ihre Verwendung nicht einsehen, verwalten AWS-eigene Schlüssel, verwenden oder überprüfen. Sie müssen jedoch keine Maßnahmen ergreifen oder Programme ändern, um Schlüssel zu schützen, die Ihre Daten verschlüsseln.

Für die Nutzung fallen keine Gebühren an AWS-eigene Schlüssel, und sie werden nicht auf die AWS KMS Kontingente für Ihr Konto angerechnet.

Kundenseitig verwaltete Schlüssel

Kundenverwaltete Schlüssel sind KMS-Schlüssel in Ihrem , die Sie erstellen, besitzen und verwalten. Sie haben die volle Kontrolle über diese KMS-Schlüssel, z. B. über die folgenden:

- Festlegung und Pflege ihrer wichtigsten Richtlinien, IAM-Richtlinien und Zuschüsse
- Sie aktivieren und deaktivieren
- Rotation ihres kryptografischen Materials
- Hinzufügen von -Tags
- Aliase erstellen, die auf sie verweisen
- Sie für das Löschen planen

Sie können auch Amazon CloudWatch Logs verwenden CloudTrail , um die Anfragen zu verfolgen, an die AWS IoT in AWS KMS Ihrem Namen FleetWise sendet.

Wenn Sie vom Kunden verwaltete Schlüssel verwenden, müssen Sie AWS FleetWise IoT-Zugriff auf den in Ihrem Konto gespeicherten KMS-Schlüssel gewähren. AWS IoT FleetWise verwendet Umschlagverschlüsselung und Schlüsselhierarchie, um Daten zu verschlüsseln. Ihr AWS KMS Verschlüsselungsschlüssel wird verwendet, um den Stammschlüssel dieser Schlüsselhierarchie zu verschlüsseln. Weitere Informationen zur Envelope-Verschlüsselung finden Sie im AWS Key Management Service -Entwicklerhandbuch.

Die folgende Beispielrichtlinie gewährt AWS FleetWise IoT-Berechtigungen zur Verwendung Ihres AWS KMS Schlüssels.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "Allow use of the key",
      "Effect": "Allow",
      "Principal": {
        "Service": "iotfleetwise.amazonaws.com"
      },
      "Action": [
```

```
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

Important

Wenn Sie die neuen Abschnitte zu Ihrer KMS-Schlüsselrichtlinie hinzufügen, ändern Sie keine vorhandenen Abschnitte in der Richtlinie. AWS IoT FleetWise kann keine Operationen mit Ihren Daten durchführen, wenn die Verschlüsselung für AWS IoT aktiviert ist FleetWise und eine der folgenden Bedingungen zutrifft:

- Der KMS-Schlüssel ist deaktiviert oder gelöscht.
- Die KMS-Schlüsselrichtlinie ist für den Dienst nicht richtig konfiguriert.

Verwendung von Bildverarbeitungssystemdaten mit Verschlüsselung im Ruhezustand

Note

Die Daten des Bildverarbeitungssystems befinden sich in der Vorschauversion und können sich ändern.

Wenn Sie in Ihrem AWS FleetWise IoT-Konto eine vom Kunden verwaltete Verschlüsselung mit aktivierten AWS KMS Schlüsseln haben und Bildverarbeitungssystemdaten verwenden möchten, setzen Sie Ihre Verschlüsselungseinstellungen zurück, damit sie mit komplexen Datentypen kompatibel sind. Auf diese Weise kann FleetWise das AWS IoT zusätzliche Berechtigungen einrichten, die für Bildverarbeitungssystemdaten erforderlich sind.

> **Note**
>
> Ihr Decoder-Manifest befindet sich möglicherweise in einem Validierungsstatus, wenn Sie Ihre Verschlüsselungseinstellungen für Bildverarbeitungssystemdaten nicht zurückgesetzt haben.

1. Verwenden Sie den GetEncryptionConfigurationAPI-Vorgang, um zu überprüfen, ob die AWS KMS Verschlüsselung aktiviert ist. Wenn der Verschlüsselungstyp aktiviert ist, sind keine weiteren Maßnahmen erforderlich`FLEETWISE_DEFAULT_ENCRYPTION`.
2. Wenn der Verschlüsselungstyp ist`KMS_BASED_ENCRYPTION`, verwenden Sie den PutEncryptionConfigurationAPI-Vorgang, um den Verschlüsselungstyp auf zurückzusetzen`FLEETWISE_DEFAULT_ENCRYPTION`.

   ```
   aws iotfleetwise put-encryption-configuration \
         --encryption-type FLEETWISE_DEFAULT_ENCRYPTION
   ```

3. Verwenden Sie den PutEncryptionConfigurationAPI-Vorgang, um den Verschlüsselungstyp wieder zu `KMS_BASED_ENCRYPTION` aktivieren.

   ```
   aws iotfleetwise put-encryption-configuration \
           --encryption-type KMS_BASED_ENCRYPTION \
           --kms-key-id kms_key_id
   ```

Weitere Informationen zur Aktivierung der Verschlüsselung finden Sie unterSchlüsselmanagement im AWS Internet der Dinge FleetWise.

Schlüsselmanagement im AWS Internet der Dinge FleetWise

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

AWS FleetWise IoT-Cloud-Schlüsselverwaltung

Standardmäßig FleetWise verwendet AWS IoT Von AWS verwaltete Schlüssel zum Schutz Ihrer Daten in der AWS Cloud. Sie können Ihre Einstellungen aktualisieren, um einen vom Kunden verwalteten Schlüssel zur Verschlüsselung von Daten im AWS IoT FleetWise zu verwenden. Sie können Ihren Verschlüsselungsschlüssel über AWS Key Management Service (AWS KMS) erstellen, verwalten und einsehen.

AWS IoT FleetWise unterstützt serverseitige Verschlüsselung mit vom Kunden verwalteten Schlüsseln AWS KMS , um Daten für die folgenden Ressourcen zu verschlüsseln.

AWS FleetWise IoT-Ressource	Datentyp	Felder, die im Ruhezustand mit vom Kunden verwalteten Schlüsseln verschlüsselt sind
Signalkatalog		description
	Attribut	Beschreibung, allowedValues, defaultValue, min, max
	Aktuator	Beschreibung, allowedValues, min, max
	Sensor	Beschreibung, allowedValues, min, max
Fahrzeugmodell (Modellmanifest)		description
Decoder-Manifest		description
	CanInterface	Protokollname, Protokollversion
	ObdInterface	requestMessageId, dtcRequestInterval Sekunden,, OBDStandard, Sekunden hasTransmissionEcu, pidRequestInterval useExtendedIds
	CanSignal	Faktor, isBigEndian IsSigned, Länge, messageId, Offset, StartBit

AWS FleetWise IoT-Ressource	Datentyp	Felder, die im Ruhezustand mit vom Kunden verwalteten Schlüsseln verschlüsselt sind
	ObdSignal	ByteLength, Offset, PID, Skalierung, ServiceMode pidResponseLength, StartByte, bitMaskLength bitRightShift
Fahrzeug		Attribute
Kampagne		description
	conditionBasedCollectionSchema	Ausdruck, minimumTriggerIntervalMs conditionLanguageVersion, TriggerMode
	TimeBasedCollectionScheme	Perioden MS
Vorlage für den Bundesstaat		description

> **Note**
>
> Andere Daten und Ressourcen werden mit der Standardverschlüsselung mit vom AWS IoT verwalteten Schlüsseln verschlüsselt FleetWise. Dieser Schlüssel wird erstellt und im AWS FleetWise IoT-Konto gespeichert.

Weitere Informationen finden Sie unter Was ist AWS Key Management Service? im AWS Key Management Service Entwicklerhandbuch.

Aktivieren Sie die Verschlüsselung mit KMS-Schlüsseln (Konsole)

Um vom Kunden verwaltete Schlüssel mit AWS IoT zu verwenden FleetWise, müssen Sie Ihre AWS FleetWise IoT-Einstellungen aktualisieren.

Um die Verschlüsselung mit KMS-Schlüsseln zu aktivieren (Konsole)

1. Öffnen Sie die AWS FleetWise IoT-Konsole.
2. Navigieren Sie zu Einstellungen.
3. Wählen Sie unter Verschlüsselung die Option Bearbeiten aus, um die Seite Verschlüsselung bearbeiten zu öffnen.
4. Wählen Sie als Verschlüsselungsschlüsseltyp die Option Anderen AWS KMS Schlüssel auswählen aus. Dies ermöglicht die Verschlüsselung mit vom Kunden verwalteten Schlüsseln, die in gespeichert sind AWS KMS.

   > **Note**
   >
   > Sie können die vom Kunden verwaltete Schlüsselverschlüsselung nur für AWS FleetWise IoT-Ressourcen verwenden. Dazu gehören der Signalkatalog, das Fahrzeugmodell (Modellmanifest), das Decoder-Manifest, das Fahrzeug, die Flotte und die Kampagne.

5. Wählen Sie Ihren KMS-Schlüssel mit einer der folgenden Optionen:

   - Um einen vorhandenen KMS-Schlüssel zu verwenden — Wählen Sie Ihren KMS-Schlüsselalias aus der Liste aus.
   - Um einen neuen KMS-Schlüssel zu erstellen — Wählen Sie Create an AWS KMS key.

   > **Note**
   >
   > Dadurch wird die AWS KMS Konsole geöffnet. Weitere Informationen zum Erstellen eines KMS-Schlüssels finden Sie unter Creating Keys im AWS Key Management Service Developer Guide.

6. Wählen Sie Speichern, um Ihre Einstellungen zu aktualisieren.

Aktivieren Sie die Verschlüsselung mit KMS-Schlüsseln (AWS CLI)

Sie können den PutEncryptionConfigurationAPI-Vorgang verwenden, um die Verschlüsselung für Ihr AWS FleetWise IoT-Konto zu aktivieren. Das folgende Beispiel verwendet AWS CLI.

Führen Sie den folgenden Befehl aus, um die Verschlüsselung zu aktivieren.

- *kms_key_id*Ersetzen Sie ihn durch die ID des KMS-Schlüssels.

```
aws iotfleetwise put-encryption-configuration \
      --encryption-type KMS_BASED_ENCRYPTION \
      --kms-key-id kms_key_id
```

Example response

```
{
 "kmsKeyId": "customer_kms_key_id",
 "encryptionStatus": "PENDING",
 "encryptionType": "KMS_BASED_ENCRYPTION"
}
```

KMS-Schlüsselrichtlinie

Nachdem Sie einen KMS-Schlüssel erstellt haben, müssen Sie Ihrer KMS-Schlüsselrichtlinie mindestens die folgende Anweisung hinzufügen, damit er mit AWS IoT funktioniert FleetWise. Der AWS FleetWise IoT-Dienstprinzipal `iotfleetwise.amazonaws.com` in der KMS-Schlüsselrichtlinienerklärung ermöglicht AWS IoT den FleetWise Zugriff auf den KMS-Schlüssel.

```
{
  "Sid": "Allow FleetWise to encrypt and decrypt data when customer managed KMS key
 based encryption is enabled",
  "Effect": "Allow",
  "Principal": {
    "Service": "iotfleetwise.amazonaws.com"
  },
  "Action": [
    "kms:GenerateDataKey*",
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:DescribeKey",
    "kms:CreateGrant",
    "kms:RetireGrant",
    "kms:RevokeGrant"
  ],
  "Resource": "*"
}
```

Aus Sicherheitsgründen empfiehlt es sich, Schlüssel zur KMS-Schlüsselrichtlinie hinzuzufügen `aws:SourceArn` und zu `aws:SourceAccount` konditionieren. Der globale IAM-

Bedingungsschlüssel `aws:SourceArn` trägt dazu bei, dass AWS IoT den KMS-Schlüssel nur für servicespezifische Ressourcen FleetWise verwendet — Amazon Resource Names ()ARNs.

Wenn Sie den Wert von festlegen`aws:SourceArn`, muss er immer so sein. `arn:aws:iotfleetwise:us-east-1:account_id:*` Dadurch kann der KMS-Schlüssel auf alle AWS FleetWise IoT-Ressourcen zugreifen AWS-Konto. AWS IoT FleetWise unterstützt einen KMS-Schlüssel pro Konto für alle darin enthaltenen Ressourcen AWS-Region. Wenn Sie einen anderen Wert für das `SourceArn` ARN-Ressourcenfeld verwenden oder den Platzhalter (*) nicht verwenden, kann AWS IoT nicht auf FleetWise den KMS-Schlüssel zugreifen.

Der Wert von `aws:SourceAccount` ist Ihre Konto-ID, die verwendet wird, um den KMS-Schlüssel weiter einzuschränken, sodass er nur für Ihr spezielles Konto verwendet werden kann. Wenn Sie dem KMS-Schlüssel Schlüssel hinzufügen `aws:SourceAccount` und diese mit `aws:SourceArn` Bedingungen versehen, stellen Sie sicher, dass der Schlüssel nicht von einem anderen Dienst oder Konto verwendet wird. Dies hilft, Ausfälle zu vermeiden.

Die folgende Richtlinie umfasst einen Dienstprinzipal (eine Kennung für einen Dienst) sowie `aws:SourceAccount` die `aws:SourceArn` Einrichtung für die Verwendung auf der Grundlage der AWS-Region und Ihrer Konto-ID.

```
{
  "Sid": "Allow use of the key",
  "Effect": "Allow",
  "Principal": {
    "Service": "iotfleetwise.amazonaws.com"
  },
  "Action": [
    "kms:Encrypt",
    "kms:Decrypt",
    "kms:ReEncrypt*",
    "kms:GenerateDataKey*",
    "kms:DescribeKey"
  ],
  "Resource": "*",
  "Condition": {
    "StringLike": {
      "aws:SourceAccount": "AWS-account-ID"
    },
    "ArnLike": {
      "aws:SourceArn": "arn:aws:iotfleetwise:region:AWS-account-ID:*"
    }
  }
```

```
}
```

Weitere Informationen zum Bearbeiten einer KMS-Schlüsselrichtlinie für die Verwendung mit AWS IoT FleetWise finden Sie unter Ändern einer Schlüsselrichtlinie im AWS Key Management Service Entwicklerhandbuch.

> **Important**
>
> Wenn Sie die neuen Abschnitte zu Ihrer KMS-Schlüsselrichtlinie hinzufügen, ändern Sie keine vorhandenen Abschnitte in der Richtlinie. AWS IoT FleetWise kann keine Operationen mit Ihren Daten durchführen, wenn die Verschlüsselung für AWS IoT aktiviert ist FleetWise und eine der folgenden Bedingungen zutrifft:
>
> - Der KMS-Schlüssel ist deaktiviert oder gelöscht.
> - Die KMS-Schlüsselrichtlinie ist für den Dienst nicht richtig konfiguriert.

Berechtigungen für die AWS KMS Verschlüsselung

Wenn Sie die AWS KMS Verschlüsselung aktiviert haben, müssen Sie in der Rollenrichtlinie Berechtigungen angeben, damit Sie AWS IoT aufrufen können FleetWise APIs. Die folgende Richtlinie ermöglicht den Zugriff auf alle AWS FleetWise IoT-Aktionen sowie auf AWS KMS bestimmte Berechtigungen.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iotfleetwise:*",
        "kms:GenerateDataKey*",
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:DescribeKey"
      ],
      "Resource": [
```

```
        "*"
      ]
    }
  ]
}
```

Die folgende Richtlinienerklärung ist erforderlich, damit Ihre Rolle die Verschlüsselung APIs aufrufen kann. Diese Grundsatzerklärung erlaubt `PutEncryptionConfiguration` und `GetEncryptionConfiguration` ergreift Maßnahmen aus dem AWS IoT FleetWise.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "iotfleetwise:GetEncryptionConfiguration",
        "iotfleetwise:PutEncryptionConfiguration",
        "kms:GenerateDataKey*",
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:DescribeKey"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

Wiederherstellung nach dem Löschen des AWS KMS Schlüssels

Wenn Sie einen AWS KMS Schlüssel löschen, nachdem Sie die Verschlüsselung mit AWS IoT aktiviert haben FleetWise, müssen Sie Ihr Konto zurücksetzen, indem Sie alle Daten löschen, bevor Sie AWS IoT FleetWise erneut verwenden können. Sie können die Liste verwenden und API-Operationen löschen, um Ressourcen in Ihrem Konto zu bereinigen.

Um Ressourcen in Ihrem Konto zu bereinigen

1. Verwenden Sie die Liste, APIs bei der der `listResponseScope` Parameter auf gesetzt ist`METADATA_ONLY`. Dies stellt eine Liste von Ressourcen bereit, einschließlich Ressourcennamen und anderer Metadaten wie ARNs Zeitstempel.
2. Verwenden Sie Löschen APIs , um einzelne Ressourcen zu entfernen.

Sie müssen Ressourcen in der folgenden Reihenfolge bereinigen.

1. Kampagnen
   a. Listet alle Kampagnen auf, bei denen der `listResponseScope` Parameter auf gesetzt ist`METADATA_ONLY`.
   b. Löschen Sie die Kampagnen.
2. Flotten und Fahrzeuge
   a. Listet alle Flotten auf, bei denen der `listResponseScope` Parameter auf gesetzt ist. `METADATA_ONLY`
   b. Listet alle Fahrzeuge für jede Flotte auf, bei der der `listResponseScope` Parameter auf `METADATA_ONLY` gesetzt ist.
   c. Trennen Sie alle Fahrzeuge von jeder Flotte.
   d. Löschen Sie die Flotten.
   e. Löschen Sie die Fahrzeuge.
3. Decoder-Manifestationen
   a. Listet alle Decoder-Manifeste auf, deren `listResponseScope` Parameter auf gesetzt ist. `METADATA_ONLY`
   b. Löscht alle Decoder-Manifeste.
4. Fahrzeugmodelle (Modellmanifeste)
   a. Listet alle Fahrzeugmodelle auf, bei denen der `listResponseScope` Parameter auf gesetzt ist`METADATA_ONLY`.
   b. Löscht alle Fahrzeugmodelle.
5. Vorlagen für Bundesstaaten

   a. Listet alle Statusvorlagen auf, bei denen der `listResponseScope` Parameter auf gesetzt ist`METADATA_ONLY`.
   b. Löscht alle Statusvorlagen.
6. Signalkataloge
   a. Listet alle Signalkataloge auf.
   b. Löscht alle Signalkataloge.

Steuerung des Zugriffs mit AWS IoT FleetWise

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

In den folgenden Abschnitten wird beschrieben, wie Sie den Zugriff auf und von Ihren AWS IoT FleetWise Ressourcen steuern können. Zu den Informationen, die sie behandeln, gehört, wie Sie Ihrer Anwendung Zugriff gewähren FleetWise können, damit AWS IoT Fahrzeugdaten während Kampagnen übertragen kann. Sie beschreiben auch, wie Sie AWS IoT FleetWise Zugriff auf Ihren Amazon S3 (S3) -Bucket oder Ihre Amazon Timestream Timestream-Datenbank und -Tabelle gewähren können, um Daten zu speichern, oder auf MQTT-Nachrichten, die zum Senden von Daten von Fahrzeugen verwendet werden.

Die Technologie zur Verwaltung all dieser Zugriffsformen ist AWS Identity and Access Management (IAM). Weitere Informationen zu IAM finden Sie unter Was ist IAM?.

Inhalt

- Erteilen Sie die AWS IoT FleetWise Erlaubnis zum Senden und Empfangen von Daten zu einem MQTT-Thema
- AWS IoT FleetWise Zugriff auf ein Amazon S3 S3-Ziel gewähren
- AWS IoT FleetWise Zugriff auf ein Amazon Timestream Timestream-Ziel gewähren
- Erteilen Sie die AWS IoT Device Management Berechtigung zum Generieren der Payload für Befehle mit AWS IoT FleetWise

Erteilen Sie die AWS IoT FleetWise Erlaubnis zum Senden und Empfangen von Daten zu einem MQTT-Thema

Wenn Sie ein MQTT-Thema verwenden, senden Ihre Fahrzeuge Daten über den AWS IoT MQTT-Nachrichtenbroker. Sie müssen die AWS IoT FleetWise Erlaubnis erteilen, das von Ihnen angegebene MQTT-Thema zu abonnieren. Wenn Sie AWS IoT Regeln auch verwenden, um Maßnahmen zu ergreifen oder Daten an andere Ziele weiterzuleiten, müssen Sie Richtlinien an eine IAM-Rolle anhängen, damit AWS IoT FleetWise Daten an IoT-Regeln weitergeleitet werden können.

Darüber hinaus können Ihre anderen Apps oder Geräte das von Ihnen angegebene Thema abonnieren, um Fahrzeugdaten nahezu in Echtzeit zu erhalten, und diesen Apps oder Geräten müssen bei Bedarf Berechtigungen und Zugriff gewährt werden.

Weitere Informationen zur Verwendung von MQTT und zu den erforderlichen Rollen und Berechtigungen finden Sie unter:

- Kommunikationsprotokolle für Geräte
- Regeln für AWS IoT
- Einer AWS IoT Regel den Zugriff gewähren, den sie benötigt
- Rollenberechtigungen weitergeben

Bevor Sie beginnen, überprüfen Sie Folgendes:

> **Important**
>
> - Sie müssen dieselbe AWS Region verwenden, wenn Sie Ressourcen für Fahrzeugkampagnen für das Internet der AWS IoT FleetWise. Wenn Sie die AWS Region wechseln, haben Sie möglicherweise Probleme beim Zugriff auf die Ressourcen.
> - AWS IoT FleetWise ist in den USA Ost (Nord-Virginia) und Europa (Frankfurt) verfügbar.

Sie können die verwenden AWS CLI , um eine IAM-Rolle mit einer Vertrauensrichtlinie für MQTT-Messaging zu erstellen. Führen Sie den folgenden Befehl aus, um eine IAM-Rolle zu erstellen.

Um eine IAM-Rolle mit einer Vertrauensrichtlinie zu erstellen

- *`IotTopicExecutionRole`*Ersetzen Sie es durch den Namen der Rolle, die Sie erstellen.

- *trust-policy*Ersetzen Sie es durch die JSON-Datei, die die Vertrauensrichtlinie enthält.

```
aws iam create-role --role-name IotTopicExecutionRole --assume-role-policy-document
 file://trust-policy.json
```

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "mqttTopicTrustPolicy",
      "Effect": "Allow",
      "Principal": {
        "Service": "iotfleetwise.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
          "aws:SourceArn": [
            "arn:aws:iotfleetwise:region:123456789012:campaign/campaign-name"
          ],
          "aws:SourceAccount": [
            "123456789012"
          ]
        }
      }
    }
  ]
}
```

Erstellen Sie eine Berechtigungsrichtlinie, um AWS FleetWise IoT-Berechtigungen zum Veröffentlichen von Nachrichten zu dem von Ihnen angegebenen MQTT-Thema zu erteilen. Führen Sie den folgenden Befehl aus, um eine Berechtigungsrichtlinie zu erstellen.

Um eine Berechtigungsrichtlinie zu erstellen

- Ersetze sie *AWSIoTFleetwiseAccessIotTopicPermissionsPolicy* durch den Namen der Richtlinie, die du gerade erstellst.

- *permissions-policy*Ersetzen Sie es durch den Namen der JSON-Datei, die die Berechtigungsrichtlinie enthält.

```
aws iam create-policy --policy-name AWSIoTFleetwiseAccessIotTopicPermissionsPolicy --
policy-document file://permissions-policy.json
```

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
          "iot:Publish"
        ],
        "Resource": [
          "topic-arn"
        ]
      }
    ]
}
```

Um die Berechtigungsrichtlinie an Ihre IAM-Rolle anzuhängen

1. Kopieren Sie aus der Ausgabe den Amazon-Ressourcennamen (ARN) der Berechtigungsrichtlinie.
2. Führen Sie den folgenden Befehl aus, um die IAM-Berechtigungsrichtlinie an Ihre IAM-Rolle anzuhängen.

   - *permissions-policy-arn*Ersetzen Sie es durch den ARN, den Sie im vorherigen Schritt kopiert haben.
   - *IotTopicExecutionRole*Ersetzen Sie durch den Namen der IAM-Rolle, die Sie erstellt haben.

```
aws iam attach-role-policy --policy-arn permissions-policy-arn --role-
name IotTopicExecutionRole
```

Weitere Informationen finden Sie unter Zugriffsverwaltung für AWS Ressourcen im IAM-Benutzerhandbuch.

AWS IoT FleetWise Zugriff auf ein Amazon S3 S3-Ziel gewähren

Wenn Sie ein Amazon S3 S3-Ziel verwenden, AWS IoT FleetWise übermittelt Fahrzeugdaten an Ihren S3-Bucket und kann optional einen AWS KMS Schlüssel, den Sie besitzen, für die Datenverschlüsselung verwenden. Wenn die Fehlerprotokollierung aktiviert ist, werden AWS IoT FleetWise auch Fehler bei der Datenübermittlung an Ihre CloudWatch Protokollgruppe und Ihre Streams gesendet. Sie benötigen eine IAM-Rolle, wenn Sie einen Lieferstream erstellen.

AWS IoT FleetWise verwendet eine Bucket-Richtlinie mit dem Service Principal für das S3-Ziel. Weitere Informationen zum Hinzufügen von Bucket-Richtlinien finden Sie unter Hinzufügen einer Bucket-Richtlinie mithilfe der Amazon S3 S3-Konsole im Amazon Simple Storage Service-Benutzerhandbuch.

Verwenden Sie die folgende Zugriffsrichtlinie, um den Zugriff auf Ihren S3-Bucket AWS IoT FleetWise zu aktivieren. Wenn Sie nicht Eigentümer des S3-Buckets sind, fügen Sie `s3:PutObjectAcl` der Liste der Amazon-S3-Aktionen hinzu. Dadurch wird dem Bucket-Besitzer vollen Zugriff auf die Objekte gewährt, die von bereitgestellt wurden AWS IoT FleetWise. Weitere Informationen darüber, wie Sie den Zugriff auf Objekte in Ihren Buckets sichern können, finden Sie unter Beispiele für Bucket-Richtlinien im Amazon Simple Storage Service-Benutzerhandbuch.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "iotfleetwise.amazonaws.com"
        ]
```

```
      },
      "Action": [
        "s3:ListBucket"
      ],
      "Resource": "arn:aws:s3:::bucket-name"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "iotfleetwise.amazonaws.com"
        ]
      },
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::bucket-name/*",
      "Condition": {
        "StringEquals": {
          "aws:SourceArn": "campaign-arn",
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

Die folgende Bucket-Richtlinie gilt für alle Kampagnen in einem Konto in einer AWS Region.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "iotfleetwise.amazonaws.com"
        ]
```

```
      },
      "Action": [
        "s3:ListBucket"
      ],
      "Resource": "arn:aws:s3:::bucket-name"
    },
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "iotfleetwise.amazonaws.com"
        ]
      },
      "Action": [
        "s3:GetObject",
        "s3:PutObject"
      ],
      "Resource": "arn:aws:s3:::bucket-name/*",
      "Condition": {
        "StringLike": {
          "aws:SourceArn": "arn:aws:iotfleetwise:region:123456789012:campaign/*",
          "aws:SourceAccount": "123456789012"
        }
      }
    }
  ]
}
```

Wenn Sie einen KMS-Schlüssel an Ihren S3-Bucket angehängt haben, benötigt der Schlüssel die folgende Richtlinie. Informationen zur Schlüsselverwaltung finden Sie unter Schützen von Daten mithilfe serverseitiger Verschlüsselung mit AWS Key Management Service Schlüsseln (SSE-KMS) im Amazon Simple Storage Service-Benutzerhandbuch.

```
{
  "Version": "2012-10-17",
  "Effect": "Allow",
  "Principal": {
    "Service": "iotfleetwise.amazonaws.com"
  },
  "Action": [
    "kms:GenerateDataKey",
    "kms:Decrypt"
```

```
    ],
  "Resource": "key-arn"
}
```

> **Important**
>
> Wenn Sie einen Bucket erstellen, erstellt S3 eine Standard-Zugriffskontrollliste (ACL), die dem Eigentümer der Ressource die volle Kontrolle über die Ressource gewährt. Wenn AWS IoT keine Daten an S3 liefern FleetWise kann, stellen Sie sicher, dass Sie die ACL auf dem S3-Bucket deaktivieren. Weitere Informationen finden Sie unter Deaktivierung ACLs für alle neuen Buckets und Durchsetzung des Objektbesitzes im Amazon Simple Storage Service-Benutzerhandbuch.

AWS IoT FleetWise Zugriff auf ein Amazon Timestream Timestream-Ziel gewähren

Wenn Sie ein Timestream-Ziel verwenden, AWS IoT FleetWise überträgt Fahrzeugdaten an eine Timestream-Tabelle. Sie müssen die Richtlinien an die IAM-Rolle anhängen, damit Daten an Timestream AWS IoT FleetWise gesendet werden können.

Wenn Sie die Konsole verwenden, um eine Kampagne zu erstellen, fügt AWS IoT der Rolle FleetWise automatisch die erforderliche Richtlinie hinzu.

> **Note**
>
> Amazon Timestream ist in der Region Asien-Pazifik (Mumbai) nicht verfügbar.

Bevor Sie beginnen, überprüfen Sie Folgendes:

> **Important**
>
> - Sie müssen dieselbe AWS Region verwenden, wenn Sie Timestream-Ressourcen für AWS IoT FleetWise erstellen. Wenn Sie die AWS Region wechseln, haben Sie möglicherweise Probleme beim Zugriff auf die Timestream-Ressourcen.
> - AWS IoT FleetWise ist in den USA Ost (Nord-Virginia), Europa (Frankfurt) und im asiatisch-pazifischen Raum (Mumbai) verfügbar.

- Eine Liste der unterstützten Regionen finden Sie unter Timestream-Endpunkte und Kontingente in der. Allgemeine AWS-Referenz

- Sie müssen über eine Timestream-Datenbank verfügen. Ein Tutorial finden Sie unter Create a database im Amazon Timestream Developer Guide.
- Sie müssen eine Tabelle in der angegebenen Timestream-Datenbank erstellt haben. Ein Tutorial finden Sie unter Erstellen einer Tabelle im Amazon Timestream Developer Guide.

Sie können das verwenden AWS CLI , um eine IAM-Rolle mit einer Vertrauensrichtlinie für Timestream zu erstellen. Führen Sie den folgenden Befehl aus, um eine IAM-Rolle zu erstellen.

Um eine IAM-Rolle mit einer Vertrauensrichtlinie zu erstellen

- *TimestreamExecutionRole*Ersetzen Sie es durch den Namen der Rolle, die Sie erstellen.
- *trust-policy*Ersetzen Sie es durch die JSON-Datei, die die Vertrauensrichtlinie enthält.

```
aws iam create-role --role-name TimestreamExecutionRole --assume-role-policy-document
 file://trust-policy.json
```

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "timestreamTrustPolicy",
      "Effect": "Allow",
      "Principal": {
        "Service": "iotfleetwise.amazonaws.com"
      },
      "Action": "sts:AssumeRole",
      "Condition": {
        "StringEquals": {
           "aws:SourceArn": [
            "arn:aws:iotfleetwise:region:123456789012:campaign/campaign-name"
           ],
```

```
            "aws:SourceAccount": [
              "123456789012"
            ]
          }
        }
      }
    ]
}
```

Erstellen Sie eine Berechtigungsrichtlinie, um AWS FleetWise IoT-Berechtigungen zum Schreiben von Daten in Timestream zu erteilen. Führen Sie den folgenden Befehl aus, um eine Berechtigungsrichtlinie zu erstellen.

Um eine Berechtigungsrichtlinie zu erstellen

- Ersetze sie *AWSIoTFleetwiseAccessTimestreamPermissionsPolicy* durch den Namen der Richtlinie, die du gerade erstellst.
- *permissions-policy*Ersetzen Sie es durch den Namen der JSON-Datei, die die Berechtigungsrichtlinie enthält.

```
aws iam create-policy --policy-name AWSIoTFleetwiseAccessTimestreamPermissionsPolicy --
policy-document file://permissions-policy.json
```

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "timestreamIngestion",
      "Effect": "Allow",
      "Action": [
        "timestream:WriteRecords",
        "timestream:Select",
        "timestream:DescribeTable"
      ],
      "Resource": "table-arn"
    },
```

```
    {
      "Sid": "timestreamDescribeEndpoint",
      "Effect": "Allow",
      "Action": [
        "timestream:DescribeEndpoints"
      ],
      "Resource": "*"
    }
  ]
}
```

Um die Berechtigungsrichtlinie an Ihre IAM-Rolle anzuhängen

1. Kopieren Sie aus der Ausgabe den Amazon-Ressourcennamen (ARN) der Berechtigungsrichtlinie.
2. Führen Sie den folgenden Befehl aus, um die IAM-Berechtigungsrichtlinie an Ihre IAM-Rolle anzuhängen.

   - *permissions-policy-arn*Ersetzen Sie es durch den ARN, den Sie im vorherigen Schritt kopiert haben.
   - *TimestreamExecutionRole*Ersetzen Sie durch den Namen der IAM-Rolle, die Sie erstellt haben.

```
aws iam attach-role-policy --policy-arn permissions-policy-arn --role-
name TimestreamExecutionRole
```

Weitere Informationen finden Sie unter Zugriffsverwaltung für AWS Ressourcen im IAM-Benutzerhandbuch.

Erteilen Sie die AWS IoT Device Management Berechtigung zum Generieren der Payload für Befehle mit AWS IoT FleetWise

Wenn Sie die Befehlsfunktion verwenden, um eine Befehlsausführung zu starten, AWS IoT Device Management werden der Befehl und die Befehlsparameter aus der eingehenden Anforderung abgerufen. Anschließend sind Berechtigungen für den Zugriff auf AWS FleetWise IoT-Ressourcen erforderlich, um die Anfrage zu validieren und die Nutzlast zu generieren. Die Nutzlast wird dann

AWS IoT Device Management über MQTT an das Befehlsanforderungsthema, das Ihr Fahrzeug abonniert hat, an das Fahrzeug gesendet.

Sie müssen zunächst eine IAM-Rolle erstellen, die die erforderlichen Berechtigungen für AWS IoT Device Management die Generierung der Nutzlast gewährt. Geben Sie dann mithilfe des `roleArn` Felds den ARN dieser Rolle für die `CreateCommand`API ein. Im Folgenden werden einige Richtlinienbeispiele gezeigt.

> **Important**
>
> Für die IAM-Rolle müssen Sie dieselbe Rolle verwenden AWS-Region wie die, in der Sie die Fahrzeug- und Befehlsressourcen erstellt haben. Wenn Sie wechseln AWS-Region, haben Sie möglicherweise Probleme beim Zugriff auf die Ressourcen.

Für die IAM-Rolle muss die folgende Vertrauensrichtlinie gelten.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "RemoteCommandsTrustPolicy",
      "Effect": "Allow",
      "Principal": {
        "Service": "iot.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Erteilen Sie Berechtigungen für alle Fahrzeuge (IoT-Dinge)

Das folgende Beispiel zeigt, wie Berechtigungen zur Generierung der Nutzlast für alle Fahrzeuge erteilt werden, die als AWS IoT Dinge registriert sind.

> **Note**
>
> - Diese Richtlinie kann zu freizügig sein. Verwenden Sie das Prinzip der geringsten Rechte, um sicherzustellen, dass Sie nur die erforderlichen Berechtigungen gewähren.
> - Um stattdessen Berechtigungen zu verweigern, ändern Sie `"Effect": "Deny"` in der IAM-Richtlinie `"Effect": "Allow"` zu.

Ersetzen Sie in diesem Beispiel:

- *region*mit Ihrem Standort AWS-Region , an dem Sie die AWS FleetWise IoT-Ressourcen verwenden.
- *111122223333*mit deiner AWS-Konto Nummer.

JSON

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

Erteilen Sie die Erlaubnis für ein bestimmtes Fahrzeug (IoT-Sache)

Das folgende Beispiel zeigt, wie Berechtigungen zur Generierung der Nutzlast für ein bestimmtes Fahrzeug erteilt werden, AWS IoT das als Objekt registriert ist.

Ersetzen Sie in diesem Beispiel:

- *region*mit Ihrem Standort AWS-Region , an dem Sie die AWS FleetWise IoT-Ressourcen verwenden.
- *111122223333*mit deiner AWS-Konto Nummer.

- *<VEHICLE_NAME>*mit dem IoT-Dingnamen für Ihr Fahrzeug.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iotfleetwise:GenerateCommandPayload",
            "Resource": "arn:aws:iot:us-east-1:111122223333:thing/<VEHICLE_NAME>"
        }
    ]
}
```

Erteilen Sie Genehmigungen für bestimmte Fahrzeuge und Signale

Das folgende Beispiel zeigt, wie Berechtigungen zur Generierung der Nutzlast für den Aktuator für ein bestimmtes Fahrzeug erteilt werden.

Ersetzen Sie in diesem Beispiel:

- *region*mit Ihrem Standort AWS-Region , an dem Sie die AWS FleetWise IoT-Ressourcen verwenden.
- *111122223333*mit deiner AWS-Konto Nummer.
- *<VEHICLE_NAME>*mit dem IoT-Dingnamen für Ihr Fahrzeug.
- *<SIGNAL_FQN>*mit dem Namen des Signals, wie *<Vehicle.actuator2>* z.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": "iotfleetwise:GenerateCommandPayload",
```

```
            "Resource": "arn:aws:iot:us-
east-1:111122223333:thing/<VEHICLE_NAME>",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "iotfleetwise:Signals": [
                        "<SIGNAL_FQN>"
                    ]
                }
            }
        }
    ]
}
```

Erteilen Sie Genehmigungen für bestimmte Fahrzeuge und Statusvorlagen

Das folgende Beispiel zeigt, wie Berechtigungen zum Generieren der Nutzdaten für ein bestimmtes Fahrzeug und eine bestimmte Statusvorlage erteilt werden.

Ersetzen Sie in diesem Beispiel:

- *region*ist Ihr Standort AWS-Region , an dem Sie die AWS FleetWise IoT-Ressourcen verwenden.
- *111122223333*ist deine AWS-Konto Nummer.
- *<VEHICLE_NAME>*ist der IoT-Dingname für Ihr Fahrzeug.
- *<STATE_TEMPLATE_ID>*mit der Kennung Ihrer Statusvorlage.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "Statement1",
            "Effect": "Allow",
            "Action": "iotfleetwise:GenerateCommandPayload",
            "Resource": [
                "arn:aws:iot:us-east-1:111122223333:thing/<VEHICLE_NAME>",
                "arn:aws:iotfleetwise:us-east-1:111122223333:state-
template/<STATE_TEMPLATE_ID>"
            ]
        }
```

```
    ]
}
```

Erteilen Sie Berechtigungen zur Verwendung von kundenverwalteten KMS-Schlüsseln

Wenn Sie vom Kunden verwaltete KMS-Schlüssel für aktiviert haben AWS IoT FleetWise, zeigt das folgende Beispiel, wie Sie Berechtigungen zum Generieren der Payload gewähren.

Ersetzen Sie in diesem Beispiel:

- *region*mit Ihrem Standort AWS-Region , an dem Sie die AWS FleetWise IoT-Ressourcen verwenden.
- *111122223333*mit deiner AWS-Konto Nummer.
- *<KMS_KEY_ID>*mit der ID Ihres KMS-Schlüssels.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iotfleetwise:GenerateCommandPayload",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "arn:aws:kms:us-east-1:111122223333:key/<KMS_KEY_ID>"
        }
    ]
}
```

Identity and Access Management für AWS IoT FleetWise

AWS Identity and Access Management (IAM) hilft einem Administrator AWS-Service , den Zugriff auf AWS Ressourcen sicher zu kontrollieren. IAM-Administratoren kontrollieren, wer authentifiziert

(angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um AWS FleetWise IoT-Ressourcen zu verwenden. IAM ist ein Programm AWS-Service , das Sie ohne zusätzliche Kosten nutzen können.

Themen

- Zielgruppe
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- So FleetWise funktioniert AWS IoT mit IAM
- Beispiele für identitätsbasierte Richtlinien für das IoT AWS FleetWise
- Fehlerbehebung bei AWS FleetWise IoT-Identität und -Zugriff
- AWS Referenz zu FleetWise IoT-API-Aktionen und Ressourcen

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von Ihrer Rolle ab:

- Servicebenutzer – Fordern Sie von Ihrem Administrator Berechtigungen an, wenn Sie nicht auf Features zugreifen können (siehe Fehlerbehebung bei AWS FleetWise IoT-Identität und -Zugriff).
- Serviceadministrator – Bestimmen Sie den Benutzerzugriff und stellen Sie Berechtigungsanfragen (siehe So FleetWise funktioniert AWS IoT mit IAM).
- IAM-Administrator – Schreiben Sie Richtlinien zur Zugriffsverwaltung (siehe Beispiele für identitätsbasierte Richtlinien für das IoT AWS FleetWise).

Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich AWS mit Ihren Identitätsdaten anmelden. Sie müssen sich als IAM-Benutzer authentifizieren oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich als föderierte Identität anmelden, indem Sie Anmeldeinformationen aus einer Identitätsquelle wie AWS IAM Identity Center (IAM Identity Center), Single Sign-On-Authentifizierung oder Anmeldeinformationen verwenden. Google/Facebook Weitere Informationen zum Anmelden finden Sie unter So melden Sie sich bei Ihrem AWS-Konto an im Benutzerhandbuch für AWS-Anmeldung .

AWS Bietet für den programmatischen Zugriff ein SDK und eine CLI zum kryptografischen Signieren von Anfragen. Weitere Informationen finden Sie unter AWS Signature Version 4 for API requests im IAM-Benutzerhandbuch.

AWS-Konto Root-Benutzer

Wenn Sie einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, dem sogenannten AWS-Konto Root-Benutzer, der vollständigen Zugriff auf alle AWS-Services Ressourcen hat. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Eine Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Tasks that require root user credentials im IAM-Benutzerhandbuch.

Verbundidentität

Es hat sich bewährt, dass menschliche Benutzer für den Zugriff AWS-Services mithilfe temporärer Anmeldeinformationen einen Verbund mit einem Identitätsanbieter verwenden müssen.

Eine föderierte Identität ist ein Benutzer aus Ihrem Unternehmensverzeichnis, Ihrem Directory Service Web-Identitätsanbieter oder der AWS-Services mithilfe von Anmeldeinformationen aus einer Identitätsquelle zugreift. Verbundene Identitäten übernehmen Rollen, die temporäre Anmeldeinformationen bereitstellen.

Für die zentrale Zugriffsverwaltung empfehlen wir AWS IAM Identity Center. Weitere Informationen finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center -Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität mit bestimmten Berechtigungen für eine einzelne Person oder Anwendung. Wir empfehlen die Verwendung temporärer Anmeldeinformationen anstelle von IAM-Benutzern mit langfristigen Anmeldeinformationen. Weitere Informationen finden Sie im IAM-Benutzerhandbuch unter Erfordern, dass menschliche Benutzer den Verbund mit einem Identitätsanbieter verwenden müssen, um AWS mithilfe temporärer Anmeldeinformationen darauf zugreifen zu können.

Eine IAM-Gruppe spezifiziert eine Sammlung von IAM-Benutzern und erleichtert die Verwaltung von Berechtigungen für große Gruppen von Benutzern. Weitere Informationen finden Sie unter Anwendungsfälle für IAM-Benutzer im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität mit spezifischen Berechtigungen, die temporäre Anmeldeinformationen bereitstellt. Sie können eine Rolle übernehmen, indem Sie von einer Benutzer- zu einer IAM-Rolle (Konsole) wechseln AWS CLI oder einen AWS API-Vorgang aufrufen. Weitere Informationen finden Sie unter Methoden, um eine Rolle zu übernehmen im IAM-Benutzerhandbuch.

IAM-Rollen sind nützlich für den Verbundbenutzer-Zugriff, temporäre IAM-Benutzerberechtigungen, kontoübergreifenden Zugriff, serviceübergreifenden Zugriff und Anwendungen, die auf Amazon EC2 laufen. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie definiert Berechtigungen, wenn sie mit einer Identität oder Ressource verknüpft sind. AWS bewertet diese Richtlinien, wenn ein Principal eine Anfrage stellt. Die meisten Richtlinien werden AWS als JSON-Dokumente gespeichert. Weitere Informationen zu JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Mit Hilfe von Richtlinien legen Administratoren fest, wer Zugriff auf was hat, indem sie definieren, welches Prinzipal welche Aktionen auf welchen Ressourcenund unter welchen Bedingungendurchführen darf.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator erstellt IAM-Richtlinien und fügt sie zu Rollen hinzu, die die Benutzer dann übernehmen können. IAM-Richtlinien definieren Berechtigungen unabhängig von der Methode, die zur Ausführung der Operation verwendet wird.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität (Benutzer, Gruppe oder Rolle) anfügen können. Diese Richtlinien steuern, welche Aktionen Identitäten für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können Inline-Richtlinien (direkt in eine einzelne Identität eingebettet) oder verwaltete Richtlinien (eigenständige Richtlinien, die mit mehreren Identitäten verbunden sind)

sein. Informationen dazu, wie Sie zwischen verwalteten und Inline-Richtlinien wählen, finden Sie unter Choose between managed policies and inline policies im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele hierfür sind Vertrauensrichtlinien für IAM-Rollen und Amazon S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben.

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Weitere Richtlinientypen

AWS unterstützt zusätzliche Richtlinientypen, mit denen die maximalen Berechtigungen festgelegt werden können, die durch gängigere Richtlinientypen gewährt werden:

- Berechtigungsgrenzen – Eine Berechtigungsgrenze legt die maximalen Berechtigungen fest, die eine identitätsbasierte Richtlinie einer IAM-Entität erteilen kann. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im -IAM-Benutzerhandbuch.
- Richtlinien zur Dienstkontrolle (SCPs) — Geben Sie die maximalen Berechtigungen für eine Organisation oder Organisationseinheit in an AWS Organizations. Weitere Informationen finden Sie unter Service-Kontrollrichtlinien im AWS Organizations -Benutzerhandbuch.
- Richtlinien zur Ressourcenkontrolle (RCPs) — Legen Sie die maximal verfügbaren Berechtigungen für Ressourcen in Ihren Konten fest. Weitere Informationen finden Sie im AWS Organizations Benutzerhandbuch unter Richtlinien zur Ressourcenkontrolle (RCPs).
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die als Parameter übergeben werden, wenn Sie eine temporäre Sitzung für eine Rolle oder einen Verbundbenutzer erstellen. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn für eine Anfrage mehrere Arten von Richtlinien gelten, sind die daraus resultierenden Berechtigungen schwieriger zu verstehen. Informationen darüber, wie AWS bestimmt wird, ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie unter Bewertungslogik für Richtlinien im IAM-Benutzerhandbuch.

So FleetWise funktioniert AWS IoT mit IAM

Bevor Sie IAM zur Verwaltung des Zugriffs auf AWS IoT verwenden, sollten Sie sich darüber informieren FleetWise, welche IAM-Funktionen für die Verwendung mit AWS IoT verfügbar sind. FleetWise

IAM-Funktionen, die Sie mit AWS IoT verwenden können FleetWise

IAM-Feature	AWS FleetWise IoT-Unterstützung
Identitätsbasierte Richtlinien	Ja
Ressourcenbasierte Richtlinien	Nein
Richtlinienaktionen	Ja
Richtlinienressourcen	Ja
Bedingungsschlüssel für die Richtlinie	Ja
ACLs	Nein
ABAC (Tags in Richtlinien)	Teilweise
Temporäre Anmeldeinformationen	Ja
Prinzipalberechtigungen	Ja
Servicerollen	Nein
Serviceverknüpfte Rollen	Nein

Einen allgemeinen Überblick darüber, wie AWS IoT FleetWise und andere AWS Dienste mit den meisten IAM-Funktionen funktionieren, finden Sie im IAM-Benutzerhandbuch unter AWS Dienste, die mit IAM funktionieren.

Identitätsbasierte Richtlinien für das Internet der IoT AWS FleetWise

Unterstützt Richtlinien auf Identitätsbasis: Ja

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Definieren benutzerdefinierter IAM-Berechtigungen mit vom Kunden verwalteten Richtlinien im IAM-Benutzerhandbuch.

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der IAM-Referenz für JSON-Richtlinienelemente im IAM-Benutzerhandbuch.

Beispiele für identitätsbasierte Richtlinien für das IoT AWS FleetWise

Beispiele für FleetWise identitätsbasierte AWS IoT-Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für das IoT AWS FleetWise

Ressourcenbasierte Richtlinien im Internet der Dinge AWS FleetWise

Unterstützt ressourcenbasierte Richtlinien: Nein

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Um kontoübergreifenden Zugriff zu ermöglichen, können Sie ein gesamtes Konto oder IAM-Entitäten in einem anderen Konto als Prinzipal in einer ressourcenbasierten Richtlinie angeben. Weitere Informationen finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

Politische Maßnahmen für das AWS IoT FleetWise

Unterstützt Richtlinienaktionen: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Nehmen Sie Aktionen in eine Richtlinie auf, um Berechtigungen zur Ausführung des zugehörigen Vorgangs zu erteilen.

Eine Liste der AWS FleetWise IoT-Aktionen finden Sie unter Von AWS IoT definierte Aktionen FleetWise in der Serviceautorisierungsreferenz.

Richtlinienaktionen in AWS IoT FleetWise verwenden das folgende Präfix vor der Aktion:

```
iotfleetwise
```

Um mehrere Aktionen in einer einzigen Anweisung anzugeben, trennen Sie sie mit Kommata:

```
"Action": [
      "iotfleetwise:action1",
      "iotfleetwise:action2"
]
```

Sie können auch Platzhalter verwenden, um mehrere Aktionen anzugeben. Beispielsweise können Sie alle Aktionen festlegen, die mit dem Wort `List` beginnen, einschließlich der folgenden Aktion:

```
"Action": "iotfleetwise:List*"
```

Beispiele für FleetWise identitätsbasierte AWS IoT-Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für das IoT AWS FleetWise

Politische Ressourcen für das AWS IoT FleetWise

Unterstützt Richtlinienressourcen: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das JSON-Richtlinienelement `Resource` gibt die Objekte an, auf welche die Aktion angewendet wird. Als Best Practice geben Sie eine Ressource mit dem zugehörigen Amazon-Ressourcennamen (ARN) an. Verwenden Sie für Aktionen, die keine Berechtigungen auf Ressourcenebene unterstützen, einen Platzhalter (*), um anzugeben, dass die Anweisung für alle Ressourcen gilt.

```
"Resource": "*"
```

Eine Liste der AWS FleetWise IoT-Ressourcentypen und ihrer ARNs Eigenschaften finden Sie unter Von AWS IoT definierte Ressourcen FleetWise in der Service Authorization Reference. Informationen zu den Aktionen, mit denen Sie den ARN jeder Ressource angeben können, finden Sie unter Von AWS IoT definierte Aktionen FleetWise .

Beispiele für FleetWise identitätsbasierte AWS IoT-Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für das IoT AWS FleetWise

Schlüssel zur Richtlinienbedingung für das AWS IoT FleetWise

Unterstützt servicespezifische Richtlinienbedingungsschlüssel: Ja

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen kann.

Das Element `Condition` gibt an, wann Anweisungen auf der Grundlage definierter Kriterien ausgeführt werden. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt. Eine Übersicht aller AWS globalen Bedingungsschlüssel finden Sie unter Kontextschlüssel für AWS globale Bedingungen im IAM-Benutzerhandbuch.

Eine Liste der AWS FleetWise IoT-Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für AWS IoT FleetWise in der Service Authorization Reference. Informationen zu den Aktionen und Ressourcen, mit denen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von AWS IoT definierte Aktionen FleetWise .

Beispiele für FleetWise identitätsbasierte AWS IoT-Richtlinien finden Sie unter. Beispiele für identitätsbasierte Richtlinien für das IoT AWS FleetWise

Zugriffskontrolllisten (ACLs) im AWS IoT FleetWise

Unterstützt ACLs: Nein

Zugriffskontrolllisten (ACLs) steuern, welche Principals (Kontomitglieder, Benutzer oder Rollen) über Zugriffsberechtigungen für eine Ressource verfügen. ACLs ähneln ressourcenbasierten Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Attributbasierte Zugriffskontrolle (ABAC) mit IoT AWS FleetWise

Unterstützt ABAC (Tags in Richtlinien): Teilweise

Die attributbasierte Zugriffskontrolle (ABAC) ist eine Autorisierungsstrategie, bei der Berechtigungen basierend auf Attributen, auch als Tags bezeichnet, definiert werden. Sie können Tags an IAM-Entitäten und AWS -Ressourcen anhängen und dann ABAC-Richtlinien entwerfen, um Operationen zu ermöglichen, wenn das Tag des Prinzipals mit dem Tag auf der Ressource übereinstimmt.

Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder Bedingung `aws:TagKeys` verwenden.

Wenn ein Service alle drei Bedingungsschlüssel für jeden Ressourcentyp unterstützt, lautet der Wert für den Service Ja. Wenn ein Service alle drei Bedingungsschlüssel für nur einige Ressourcentypen unterstützt, lautet der Wert Teilweise.

Weitere Informationen zu ABAC finden Sie unter Definieren von Berechtigungen mit ABAC-Autorisierung im IAM-Benutzerhandbuch. Um ein Tutorial mit Schritten zur Einstellung von ABAC anzuzeigen, siehe Attributbasierte Zugriffskontrolle (ABAC) verwenden im IAM-Benutzerhandbuch.

> **Note**
>
> AWS IoT unterstützt FleetWise nur`iam:PassRole`, was für den `CreateCampaign` API-Betrieb erforderlich ist.

Temporäre Anmeldeinformationen mit AWS IoT verwenden FleetWise

Unterstützt temporäre Anmeldeinformationen: Ja

Temporäre Anmeldeinformationen ermöglichen kurzfristigen Zugriff auf AWS Ressourcen und werden automatisch erstellt, wenn Sie einen Verbund verwenden oder die Rollen wechseln. AWS empfiehlt, temporäre Anmeldeinformationen dynamisch zu generieren, anstatt langfristige Zugriffsschlüssel zu verwenden. Weitere Informationen finden Sie unter Temporäre Anmeldeinformationen in IAM und AWS-Services , die mit IAM funktionieren im IAM-Benutzerhandbuch.

Serviceübergreifende Prinzipalberechtigungen für IoT AWS FleetWise

Unterstützt Forward Access Sessions (FAS): Ja

Forward Access Sessions (FAS) verwenden die Berechtigungen des Prinzipals, der einen aufruft AWS-Service, in Kombination mit der Anforderung, Anfragen an nachgelagerte Dienste AWS-Service zu stellen. Einzelheiten zu den Richtlinien für FAS-Anforderungen finden Sie unter Zugriffssitzungen weiterleiten.

Servicerollen für AWS IoT FleetWise

Unterstützt Servicerollen: Nein

Eine Servicerolle ist eine IAM-Rolle, die ein Service annimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

> **Warning**
>
> Das Ändern der Berechtigungen für eine Servicerolle kann die AWS FleetWise IoT-Funktionalität beeinträchtigen. Bearbeiten Sie Servicerollen nur, FleetWise wenn AWS IoT Sie dazu anleitet.

Serviceverknüpfte Rollen für das IoT AWS FleetWise

Unterstützt serviceverknüpfte Rollen: Ja

Eine serviceverknüpfte Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Dienstbezogene Rollen werden in Ihrem Dienst angezeigt AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.

Details zum Erstellen oder Verwalten von serviceverknüpften Rollen finden Sie unter AWS -Services, die mit IAM funktionieren. Suchen Sie in der Tabelle nach einem Service mit einem `Yes` in der Spalte Service-linked role (Serviceverknüpfte Rolle). Wählen Sie den Link Yes (Ja) aus, um die Dokumentation für die serviceverknüpfte Rolle für diesen Service anzuzeigen.

Verwenden von serviceverknüpften Rollen für AWS IoT FleetWise

AWS IoT FleetWise verwendet AWS Identity and Access Management (IAM) serviceverknüpfte Rollen. Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit AWS IoT verknüpft ist. FleetWise Servicebezogene Rollen sind von AWS IoT vordefiniert FleetWise und beinhalten die Berechtigungen, die AWS IoT FleetWise benötigt, um Metriken an Amazon CloudWatch zu senden. Weitere Informationen finden Sie unter Überwachen Sie AWS IoT FleetWise mit Amazon CloudWatch.

Eine serviceverknüpfte Rolle FleetWise beschleunigt die Einrichtung von AWS IoT, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. AWS IoT FleetWise definiert die Berechtigungen seiner serviceverknüpften Rollen, und sofern nicht anders definiert, FleetWise kann nur AWS IoT seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keiner anderen IAM-Entität zugeordnet werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dies schützt Ihre FleetWise AWS-IoT-Ressourcen, da Sie die Zugriffsberechtigung für die Ressourcen nicht versehentlich entziehen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter AWS Services, die mit IAM funktionieren. Suchen Sie in der Spalte Service-verknüpfte Rollen nach den Services, für die Ja steht. Um die serviceverknüpfte Rollendokumentation für diesen Service anzuzeigen, wählen Sie ein Ja mit einem Link.

Servicebezogene Rollenberechtigungen für AWS IoT FleetWise

AWS IoT FleetWise verwendet die serviceverknüpfte Rolle namens AWSServiceRoleForIoTFleetWise — eine von AWS verwaltete Richtlinie, die für alle out-of-the-box Berechtigungen für AWS IoT FleetWise verwendet wird.

Die servicebezogene AWSService RoleForIo TFleet Wise-Rolle vertraut darauf, dass die folgenden Services die Rolle übernehmen:

- `IoTFleetWise`

Die genannte Rollenberechtigungsrichtlinie AWSIoTFleetwiseServiceRolePolicyermöglicht es AWS IoT FleetWise , die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

- Aktion: `cloudwatch:PutMetricData` auf Ressource: `*`

Informationen zu Änderungen an dieser Richtlinie finden Sie unter AWSIoTFleetwiseServiceRolePolicy Richtlinienaktualisierungen.

Die mit dem Dienst verknüpfte Rolle ist berechtigt, Metriken in den folgenden CloudWatch Namespaces zu veröffentlichen:

- `AWS/IoTFleetWise`— Für dienstspezifische Metriken
- `AWS/Usage`— Für Nutzungsmetriken

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter Serviceverknüpfte Rollenberechtigung im IAM-Benutzerhandbuch.

Eine serviceverknüpfte Rolle für AWS IoT erstellen FleetWise

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie ein Konto in der AWS FleetWise IoT-Konsole, der oder der AWS CLI AWS API registrieren, FleetWise erstellt AWS IoT die serviceverknüpfte Rolle für Sie. Weitere Informationen finden Sie unter Konfigurieren Sie Ihre AWS FleetWise IoT-Einstellungen.

Erstellen einer serviceverknüpften Rolle in AWS IoT FleetWise (Konsole)

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie ein Konto in der AWS FleetWise IoT-Konsole, der AWS CLI oder der AWS API registrieren, FleetWise erstellt AWS IoT die serviceverknüpfte Rolle für Sie.

Bearbeiten einer serviceverknüpften Rolle für AWS IoT FleetWise

Sie können die serviceverknüpfte AWSService RoleForIo TFleet Wise-Rolle in AWS IoT FleetWise nicht bearbeiten. Da verschiedene Entitäten auf jede von Ihnen erstellte serviceverknüpfte Rolle verweisen können, können Sie den Namen der Rolle nicht ändern. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter Bearbeiten einer serviceverknüpften Rolle im IAM-Benutzerhandbuch.

Bereinigen einer serviceverknüpften Rolle

Bevor mit IAM eine serviceverknüpfte Rolle löschen können, müssen Sie zunächst alle von der Rolle verwendeten Ressourcen löschen.

> **Note**
>
> Wenn AWS IoT FleetWise die Rolle verwendet, wenn Sie versuchen, die Ressourcen zu löschen, schlägt das Löschen möglicherweise fehl. Wenn dies passiert, warten Sie einige Minuten und versuchen Sie es erneut. Informationen zum Löschen von Rollen service-linked-role über die Konsole, AWS CLI oder AWS API finden Sie unter Verwenden von serviceverknüpften Rollen im IAM-Benutzerhandbuch.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie ein Konto bei AWS IoT FleetWise registrieren. AWS IoT erstellt FleetWise dann erneut die serviceverknüpfte Rolle für Sie.

Beispiele für identitätsbasierte Richtlinien für das IoT AWS FleetWise

Standardmäßig sind Benutzer und Rollen nicht berechtigt, AWS FleetWise IoT-Ressourcen zu erstellen oder zu ändern. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen.

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von IAM-Richtlinien (Konsole) im IAM-Benutzerhandbuch.

Einzelheiten zu den von AWS IoT definierten Aktionen und Ressourcentypen FleetWise, einschließlich des Formats ARNs für die einzelnen Ressourcentypen, finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für AWS IoT FleetWise in der Service Authorization Reference.

Themen

- Best Practices für Richtlinien
- Verwenden der AWS FleetWise IoT-Konsole
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- Greifen Sie auf Ressourcen in Amazon Timestream zu

Best Practices für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand AWS FleetWise IoT-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr

verursachen AWS-Konto. Beachten Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Richtlinien und Empfehlungen:

- Erste Schritte mit AWS verwalteten Richtlinien und Umstellung auf Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um damit zu beginnen, Ihren Benutzern und Workloads Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien oder Von AWS verwaltete Richtlinien für Auftragsfunktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. CloudFormation B. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung mit IAM Access Analyzer im IAM-Benutzerhandbuch.
- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren

Richtlinien MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Sicherer API-Zugriff mit MFA im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Best Practices für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Verwenden der AWS FleetWise IoT-Konsole

Um auf die AWS FleetWise IoT-Konsole zugreifen zu können, benötigen Sie ein Minimum an Berechtigungen. Diese Berechtigungen müssen es Ihnen ermöglichen, Details zu den AWS FleetWise IoT-Ressourcen in Ihrem aufzulisten und anzuzeigen AWS-Konto. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (Benutzer oder Rollen) mit dieser Richtlinie.

Sie müssen Benutzern, die nur die API AWS CLI oder die AWS API aufrufen, keine Mindestberechtigungen für die Konsole gewähren. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die der API-Operation entsprechen, die die Benutzer ausführen möchten.

Um sicherzustellen, dass Benutzer und Rollen die AWS FleetWise IoT-Konsole weiterhin verwenden können, fügen Sie den Entitäten auch die AWS IoT FleetWise `ConsoleAccess` - oder `ReadOnly` AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API AWS CLI oder AWS .

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
```

```
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Greifen Sie auf Ressourcen in Amazon Timestream zu

Bevor Sie AWS IoT nutzen können FleetWise, müssen Sie Ihr AWS Konto, IAM und Amazon Timestream Timestream-Ressourcen registrieren, um dem AWS IoT die FleetWise Erlaubnis zu erteilen, Fahrzeugdaten in AWS Cloud Ihrem Namen zu senden. Um sich zu registrieren, benötigen Sie:

- Eine Amazon Timestream Timestream-Datenbank.
- Eine Tabelle, die in der angegebenen Amazon Timestream Timestream-Datenbank erstellt wurde.
- Eine IAM-Rolle, die es AWS IoT ermöglicht, Daten FleetWise an Amazon Timestream zu senden.

Weitere Informationen, einschließlich Verfahren und Beispielrichtlinien, finden Sie unter. [Konfigurieren Sie Ihre AWS FleetWise IoT-Einstellungen]

Fehlerbehebung bei AWS FleetWise IoT-Identität und -Zugriff

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit AWS IoT FleetWise und IAM auftreten können.

Themen

- Ich bin nicht berechtigt, eine Aktion im AWS IoT durchzuführen FleetWise
- Ich bin nicht berechtigt, IAM auszuführen: PassRole
- Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine AWS FleetWise IoT-Ressourcen ermöglichen

Ich bin nicht berechtigt, eine Aktion im AWS IoT durchzuführen FleetWise

Wenn Ihnen AWS-Managementkonsole mitgeteilt wird, dass Sie nicht berechtigt sind, eine Aktion durchzuführen, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Der folgende Beispielfehler tritt auf, wenn der `mateojackson` IAM-Benutzer versucht, die Konsole zu verwenden, um Details zu einer fiktiven *`myVehicle`* Ressource anzuzeigen, aber nicht über die `iotfleetwise:`*`GetVehicleStatus`* entsprechenden Berechtigungen verfügt.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 iotfleetwise:GetVehicleStatus on resource: myVehicle
```

In diesem Fall bittet Mateo seinen Administrator um die Aktualisierung seiner Richtlinien, um unter Verwendung der Aktion *`myVehicle`* auf die Ressource `iotfleetwise:`*`GetVehicleStatus`* zugreifen zu können.

Ich bin nicht berechtigt, IAM auszuführen: PassRole

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion auszuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an AWS IoT übergeben können FleetWise.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Dienst zu übergeben, anstatt eine neue Servicerolle oder eine dienstverknüpfte Rolle zu erstellen. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in AWS IoT FleetWise auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren AWS Administrator. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Ich möchte Personen außerhalb von mir den Zugriff AWS-Konto auf meine AWS FleetWise IoT-Ressourcen ermöglichen

Sie können eine Rolle erstellen, mit der Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation auf Ihre Ressourcen zugreifen können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Für Dienste, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (ACLs) unterstützen, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob AWS IoT diese Funktionen FleetWise unterstützt, finden Sie unterSo FleetWise funktioniert AWS IoT mit IAM.
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen.
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte.
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter Gewähren von Zugriff für extern authentifizierte Benutzer (Identitätsverbund) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter Kontoübergreifender Ressourcenzugriff in IAM im IAM-Benutzerhandbuch.

AWS Referenz zu FleetWise IoT-API-Aktionen und Ressourcen

Wenn Sie den Zugriff mithilfe von Richtlinien für eine IAM-Identität verwalten, können Sie die folgende Tabelle als Referenz verwenden. In der Tabelle sind jede AWS FleetWise IoT-API, die entsprechenden Aktionen aufgeführt, für die Sie Berechtigungen zur Ausführung der Aktion erteilen können, und die AWS Ressource, für die Sie die Berechtigungen erteilen können.

Die Aktionen geben Sie im Feld `Action` und den Wert für die Ressource im Feld `Resource` der Richtlinie an. Um eine Aktion anzugeben, verwenden Sie das `iotfleetwise:` Präfix, gefolgt vom Namen der Aktion. Beispiel, `iotfleetwise:CreateSignalCatalog`.

Derzeit FleetWise unterstützt AWS IoT die folgenden Aktionen, Ressourcen und Bedingungsschlüssel.

Verwenden Sie die Bildlaufleiste, um den Rest der Tabelle zu sehen.

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
AssociateVehicleFleet	iotflotweise: Associate VehicleFleet	hat ::iotfleetwise: ::fleet/ ausgeführt *partition region account_id fleet-id* ran ::iotfleetwise*partition* : ::vehicle/ *region account_id vehicle-id*
CreateCampaign	iotfleetwise: CreateCampaign	hat ::iotfleetwise: :campaign/ ausgeführ t *partition region account_id campaign-name* arn ::iotfleetwise*partition* : ::signal-catalog/ *region account_id name* arn ::iotfleetwise*partition* : ::fahrzeug/ *region account_id vehicle-id* hat ::iotfleetwise*partition* : ::fleet/ ausgeführ t *region account_id fleet-id*

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
CreateDecoderManifest	iotfleetwise: CreateDecoderManifest	hat ::iotfleetwise: :decoder-manifest/ ausgeführt *partition region account_id name* arn ::iotfleetwise*partition* : ::model-manifest/ *region account_id name*
CreateFleet	iotfleetwise: CreateFleet	hat ::iotfleetwise: ::fleet/ ausgeführt *partition region account_id fleet-id* arn ::iotfleetwise*partition* : ::signal-catalog/ *region account_id name*
CreateModelManifest	iotfleetwise: CreateModelManifest	hat ::iotfleetwise: :model-manifest/ ausgeführt *partition region account_id name* arn ::iotfleetwise*partition* : ::signal-catalog/ *region account_id name*
CreateSignalCatalog	iotfleetwise: CreateSignalCatalog	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name*
CreateStateTemplate	iotfleetwise: CreateStateTemplate	hat ::iotfleetwise: :state-template/ ausgeführt *partition region account_id state-template-id* arn ::iotfleetwise*partition* : ::signal-catalog/ *region account_id name*

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
CreateVehicle	iotfleetwise: CreateVehicle	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id* rn::iotfleetwise*partition* : ::model-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::decoder-manifest/ *region account_id name*
BatchCreateVehicle	iotfleetwise: CreateVehicle	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id* rn::iotfleetwise*partition* : ::model-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::decoder-manifest/ *region account_id name*
DeleteCampaign	iotfleetwise: DeleteCampaign	hat ::iotfleetwise: :campaign/ ausgeführ t *partition region account_id campaign-name*
DeleteDecoderManif est	iotfleetwise: DeleteDecoderManif est	hat ::iotfleetwise: :decoder-manifest/ ausgeführt *partition region account_id name*
DeleteFleet	iotfleetwise: DeleteFle et	hat ::iotfleetwise: ::fleet/ ausgeführt *partition region account_id fleet-id*
DeleteModelManifest	iotfleetwise: DeleteModelManifest	hat ::iotfleetwise: :decoder-manifest/ ausgeführt *partition region account_id name*
DeleteSignalCatalog	iotfleetwise: DeleteSig nalCatalog	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name*

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
DeleteStateTemplate	iotfleetwise: DeleteStateTemplate	hat ::iotfleetwise: :state-template/ ausgeführt *partition region account_id state-template-id*
DeleteVehicle	iotfleetwise: DeleteVehicle	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id*
DisassociateVehicleFleet	iotfleetwise: DisassociateVehicleFleet	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id* hat ::iotfleetwise*partition* : ::fleet/ ausgeführt *region account_id fleet-id*
GetCampaign	iotfleetwise: GetCampaign	hat ::iotfleetwise: :campaign/ ausgeführt *partition region account_id campaign-name*
GetDecoderManifest	iotfleetwise: GetDecoderManifest	hat ::iotfleetwise: :decoder-manifest/ ausgeführt *partition region account_id name*
GetEncryptionConfiguration	iotfleetwise: GetEncryptionConfiguration	
GetFleet	iotfleetwise: GetFleet	hat ::iotfleetwise: ::fleet/ ausgeführt *partition region account_id fleet-id*
GetLoggingOptions	iotfleetwise: GetLoggingOptions	
GetModelManifest	iotfleetwise: GetModelManifest	hat ::iotfleetwise: :model-manifest/ ausgeführt *partition region account_id name*

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
GetRegisterAccountStatus	iotfleetwise: GetRegisterAccountStatus	
GetSignalCatalog	iotfleetwise: GetSignalCatalog	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name*
GetStateTemplate	iotfleetwise: GetStateTemplate	hat ::iotfleetwise: :state-template/ ausgeführt *partition region account_id state-template-id*
GetVehicle	iotfleetwise: GetVehicle	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id*
GetVehicleStatus	iotfleetwise: GetVehicleStatus	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id*
ImportDecoderManifest	iotfleetwise: ImportDecoderManifest	hat ::iotfleetwise: :decoder-manifest/ ausgeführt *partition region account_id name*
ImportSignalCatalog	iotfleetwise: ImportSignalCatalog	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name*
ListCampaigns	iotfleetwise: ListCampaigns	
ListDecoderManifestNetworkInterfaces	iotfleetwise: ListDecoderManifestNetworkInterfaces	hat ::iotfleetwise: :decoder-manifest/ ausgeführt *partition region account_id name*
ListDecoderManifests	iotfleetwise: ListDecoderManifests	

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
ListDecoderManifestSignals	iotfleetwise: ListDecoderManifestSignals	hat ::iotfleetwise: :decoder-manifest/ ausgeführt *partition region account_id name*
ListFleets	iotfleetwise: ListFleets	
ListFleetsForVehicle	iotfleetwise: ListFleetsForVehicle	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id*
ListModelManifestNodes	iotfleetwise: ListModelManifestNodes	hat ::iotfleetwise: :model-manifest/ ausgeführt *partition region account_id name*
ListModelManifests	iotfleetwise: ListModelManifests	
ListSignalCatalogNodes	iotfleetwise: ListSignalCatalogNodes	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name*
ListSignalCatalogs	iotfleetwise: ListSignalCatalogs	
ListStateTemplates	iotfleetwise: ListStateTemplates	
ListVehicles	iotfleetwise: ListVehicles	
ListVehiclesInFleet	iotfleetwise: ListVehiclesInFleet	hat ::iotfleetwise: ::fleet/ ausgeführt *partition region account_id fleet-id*

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
ListTagsForResource	iotfleetwise: ListTagsForResource	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name* arn ::iotfleetwise*partition* : ::model-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::decoder-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::fahrzeug/ *region account_id vehicle-id* hat ::iotfleetwise*partition* : ::fleet/ ausgeführt *region account_id fleet-id* ran ::iotfleetwise*partition* : ::campaign/ *region account_id campaign-name* rn::iotfleetwise*partition* : ::state-template/ *region account_id state-template-id*
PutEncryptionConfiguration	iotfleetwise: PutEncryptionConfiguration	
PutLoggingOptions	iotfleetwise: PutLoggingOptions	
RegisterAccount	iotfleetwise: RegisterAccount	

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
TagResource	iotfleetwise: TagResource	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name* arn ::iotfleetwise*partition* : ::model-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::decoder-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::fahrzeug/ *region account_id vehicle-id* hat ::iotfleetwise*partition* : ::fleet/ ausgeführt *region account_id fleet-id* ran ::iotfleetwise*partition* : ::campaign/ *region account_id campaign-name* rn::iotfleetwise*partition* : ::state-template/ *region account_id state-template-id*

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
UntagResource	iotfleetwise: UntagResource	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name* arn ::iotfleetwise*partition* : ::model-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::decoder-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::fahrzeug/ *region account_id vehicle-id* hat ::iotfleetwise*partition* : ::fleet/ ausgeführt *region account_id fleet-id* ran ::iotfleetwise*partition* : ::campaign/ *region account_id campaign-name* rn::iotfleetwise*partition* : ::state-template/ *region account_id state-template-id*
UpdateCampaign	iotfleetwise: UpdateCampaign	hat ::iotfleetwise: :campaign/ ausgeführt *partition region account_id campaign-name*
UpdateDecoderManifest	iotfleetwise: UpdateDecoderManifest	hat ::iotfleetwise: :decoder-manifest/ ausgeführt *partition region account_id name*
UpdateFleet	iotfleetwise: UpdateFleet	hat ::iotfleetwise: ::fleet/ ausgeführt *partition region account_id fleet-id*
UpdateModelManifest	iotfleetwise: UpdateModelManifest	hat ::iotfleetwise: :model-manifest/ ausgeführt *partition region account_id name*

AWS FleetWise IoT-Aktionen	Erforderliche Berechtigungen	Ressourcen
UpdateSignalCatalog	iotfleetwise: UpdateSignalCatalog	hat ::iotfleetwise: :signal-catalog/ ausgeführt *partition region account_id name*
UpdateStateTemplate	iotfleetwise: UpdateStateTemplate	hat ::iotfleetwise: :state-template/ ausgeführt *partition region account_id state-template-id*
UpdateVehicle	iotfleetwise: UpdateVehicle	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id* rn::iotfleetwise*partition* : ::model-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::decoder-manifest/ *region account_id name*
BatchUpdateVehicle	iotfleetwise: UpdateVehicle	hat ::iotfleetwise: :vehicle/ ausgeführt *partition region account_id vehicle-id* rn::iotfleetwise*partition* : ::model-manifest/ *region account_id name* arn ::iotfleetwise*partition* : ::decoder-manifest/ *region account_id name*

AWS verwaltete Richtlinienaktualisierungen für AWS IoT FleetWise

Sehen Sie sich Details zu Aktualisierungen der AWS verwalteten Richtlinien für AWS IoT an, FleetWise seit dieser Service begonnen hat, diese Änderungen zu verfolgen. Um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhalten, abonnieren Sie den RSS-Feed auf der Seite mit dem Verlauf der FleetWise AWS-IoT-Dokumente.

AWSIoTFleetwiseServiceRolePolicy

Diese Richtlinie ermöglicht es AWS IoT FleetWise , in Ihrem Namen Metriken CloudWatch auf Amazon zu veröffentlichen.

Änderungen	Beschreibung	Datum
`AWSIoTFleetwiseServiceRolePolicy` – Aktualisierung auf eine bestehende Richtlinie	Zusätzlich zu den bestehenden AWS/Usage Namespace-Berechtigungen wurden Berechtigungen zur Veröffentlichung von Nutzungsmetriken im `AWS/IoTFleetWise` Namespace hinzugefügt.	13. Juni 2025
`AWSIoTFleetwiseServiceRolePolicy` — AWS IoT FleetWise hat begonnen, Änderungen zu verfolgen	AWS IoT FleetWise hat damit begonnen, Änderungen für diese Richtlinie nachzuverfolgen.	27. September 2022

Konformitätsvalidierung für das AWS IoT FleetWise

> **Note**
>
> AWS IoT FleetWise fällt nicht in den Geltungsbereich von AWS Compliance-Programmen.

Informationen darüber, ob AWS-Service ein AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm AWS-Services unter . Wählen Sie dort das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter AWS Compliance-Programme AWS .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter Berichte herunterladen unter .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. Weitere Informationen zu Ihrer Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services finden Sie in der AWS Sicherheitsdokumentation.

Resilienz im AWS Internet der Dinge FleetWise

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones. Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die über hoch redundante Netzwerke mit niedriger Latenz und hohen Durchsätzen verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter AWS Globale Infrastruktur.

> **Note**
>
> Durch AWS IoT FleetWise verarbeitete Daten werden in einer Amazon Timestream Timestream-Datenbank gespeichert. Timestream unterstützt Backups in anderen AWS Availability Zones oder Regionen. Sie können jedoch mit dem Timestream SDK Ihre eigene Anwendung schreiben, um Daten abzufragen und sie an einem Ziel Ihrer Wahl zu speichern. Weitere Informationen zu Amazon Timestream finden Sie im Amazon Timestream Developer Guide.
> Amazon Timestream ist in der Region Asien-Pazifik (Mumbai) nicht verfügbar.

Infrastruktursicherheit im AWS Internet der Dinge FleetWise

Als verwalteter Dienst FleetWise ist AWS IoT durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter AWS Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um FleetWise über das Netzwerk auf AWS IoT zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Sie können diese API-Operationen von jedem Netzwerkstandort aus aufrufen, aber AWS IoT FleetWise unterstützt ressourcenbasierte Zugriffsrichtlinien, die Einschränkungen auf der Grundlage der Quell-IP-Adresse beinhalten können. Sie können auch AWS FleetWise IoT-Richtlinien verwenden, um den Zugriff von bestimmten Amazon Virtual Private Cloud (Amazon VPC) - Endpunkten oder bestimmten zu kontrollieren. VPCs Dadurch wird der Netzwerkzugriff auf eine bestimmte AWS FleetWise IoT-Ressource effektiv nur von der spezifischen VPC innerhalb des AWS Netzwerks isoliert.

Themen

- Verbindung zum AWS IoT FleetWise über eine Schnittstelle (VPC-Endpunkt)

Verbindung zum AWS IoT FleetWise über eine Schnittstelle (VPC-Endpunkt)

Sie können eine direkte Verbindung zum AWS IoT herstellen, FleetWise indem Sie einen VPC-Endpunkt (AWS PrivateLink) in Ihrer Virtual Private Cloud (VPC) verwenden, anstatt eine Verbindung über das Internet herzustellen. Wenn Sie einen VPC-Endpunkt mit Schnittstelle verwenden, FleetWise erfolgt die Kommunikation zwischen Ihrer VPC und dem AWS IoT vollständig innerhalb des AWS Netzwerks. Jeder VPC-Endpunkt wird durch eine oder mehrere Elastic Network-Schnittstellen (ENIs) mit privaten IP-Adressen in Ihren VPC-Subnetzen repräsentiert.

Der VPC-Schnittstellen-Endpunkt verbindet Ihre VPC FleetWise ohne Internet-Gateway, NAT-Gerät, VPN-Verbindung oder Direct Connect Verbindung direkt mit dem AWS IoT. Die Instances in Ihrer VPC benötigen keine öffentlichen IP-Adressen, um mit der AWS FleetWise IoT-API zu kommunizieren.

Um AWS IoT FleetWise über Ihre VPC nutzen zu können, müssen Sie eine Verbindung von einer Instance innerhalb der VPC herstellen oder Ihr privates Netzwerk mithilfe eines AWS Virtual Private Network (VPN) oder mit Ihrer VPC verbinden. Direct Connect Informationen zu Amazon VPN finden

Sie unter VPN-Verbindungen im Benutzerhandbuch für Amazon Virtual Private Cloud. Weitere Informationen dazu AWS Direct Connect finden Sie unter Verbindung erstellen im Direct Connect Benutzerhandbuch.

Sie können einen VPC-Schnittstellen-Endpunkt erstellen, um eine Verbindung zum AWS IoT herzustellen, FleetWise indem Sie die Befehle AWS Konsole oder AWS Command Line Interface (AWS CLI) verwenden. Weitere Informationen finden Sie unter Erstellen eines Schnittstellenendpunkts.

Wenn Sie nach der Erstellung eines VPC-Schnittstellen-Endpunkts private DNS-Hostnamen für den Endpunkt aktivieren, wird der AWS FleetWise Standard-IoT-Endpunkt zu Ihrem VPC-Endpunkt aufgelöst. Der Standardendpunkt für Servicenamen für AWS IoT FleetWise hat das folgende Format.

```
iotfleetwise.Region.amazonaws.com
```

Wenn Sie private DNS-Hostnamen nicht aktivieren, stellt Amazon VPC einen DNS-Endpunktnamen bereit, den Sie im folgenden Format verwenden können.

```
VPCE_ID.iotfleetwise.Region.vpce.amazonaws.com
```

Weitere Informationen finden Sie unter Schnittstellen-VPC-Endpunkte (AWS PrivateLink) im Amazon-VPC-Benutzerhandbuch.

AWS IoT FleetWise unterstützt Aufrufe all seiner API-Aktionen in Ihrer VPC.

Sie können VPC-Endpunktrichtlinien an einen VPC-Endpunkt anfügen, um den Zugriff für IAM-Prinzipale zu steuern. Sie können einem VPC-Endpunkt auch Sicherheitsgruppen zuordnen, um den eingehenden und ausgehenden Zugriff basierend auf Ursprung und Ziel des Netzwerkdatenverkehrs zu steuern, z. B. mit einem IP-Adressbereich. Weitere Informationen finden Sie unter Steuern des Zugriffs auf Services mit VPC-Endpunkten.

> **Note**
>
> AWS IoT FleetWise unterstützt alle VPC-Endpunkte im Dual-Stack-Modus. Informationen zu Dienstendpunkten finden Sie unter AWS FleetWise IoT-Endpunkte und Kontingente.

Erstellen einer VPC-Endpunktrichtlinie für IoT AWS FleetWise

Sie können eine Richtlinie für Amazon VPC-Endpunkte für AWS IoT erstellen FleetWise , um Folgendes anzugeben:

- Prinzipal, der Aktionen ausführen bzw. nicht ausführen kann
- Die Aktionen, die ausgeführt werden können oder nicht

Weitere Informationen finden Sie unter Steuerung des Zugriffs auf Services mit VPC-Endpunkten im Amazon-VPC-Benutzerhandbuch.

Example— VPC-Endpunktrichtlinie, um jeglichen Zugriff von einem bestimmten AWS Konto aus zu verweigern

Die folgende VPC-Endpunktrichtlinie verweigert dem AWS Konto *123456789012* alle API-Aufrufe, die den Endpunkt verwenden.

```
{
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "*",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": {
                "AWS": [
                    "123456789012"
                ]
            }
        }
    ]
}
```

Example– VPC-Endpunktrichtlinie zum Gewähren des VPC-Zugriffs auf einen angegebenen IAM-Prinzipal (Benutzer)

Die folgende VPC-Endpunktrichtlinie ermöglicht vollen Zugriff nur für das AWS Konto *123456789012* eines Benutzers*lijuan*. Sie verweigert allen anderen IAM-Prinzipalen den Zugriff auf den Endpunkt.

```
{
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": {
                "AWS": [
                    "arn:aws:iam::123456789012:user/lijuan"
                ]
            }
        }]
}
```

Example— VPC-Endpunktrichtlinie für AWS IoT-Aktionen FleetWise

Das Folgende ist ein Beispiel für eine Endpunktrichtlinie für AWS IoT FleetWise. Wenn diese Richtlinie an einen Endpunkt angehängt ist, gewährt sie dem IAM-Benutzer Zugriff auf die aufgelisteten AWS FleetWise IoT-Aktionen *fleetWise* in. AWS-Konto *123456789012*

```
{
    "Statement": [
        {
             "Principal": {
                "AWS": [
                    "arn:aws:iam::123456789012:user/fleetWise"
                },
            "Resource": "*",
            "Effect": "Allow",
            "Action": [
                "iotfleetwise:ListFleets",
                "iotfleetwise:ListCampaigns",
                "iotfleetwise:CreateVehicle",
            ]
           }
```

```
    ]
}
```

Konfiguration und Schwachstellenanalyse im AWS IoT FleetWise

IoT-Umgebungen können aus einer großen Anzahl von Geräten mit unterschiedlichsten Funktionen bestehen und sind langlebig und geografisch verteilt. Aufgrund dieser Merkmale ist die Geräteeinrichtung komplex und fehleranfällig. Da Geräte zudem häufig in Bezug auf Rechenleistung, Arbeitsspeicher und Speicherkapazitäten eingeschränkt sind, ist der Einsatz von Verschlüsselung und anderen Sicherheitsformen auf den Geräten begrenzt. Geräte verwenden häufig Software mit bekannten Schwachstellen. Diese Faktoren machen IoT-Geräte, einschließlich Fahrzeuge, die Daten für das AWS IoT der Dinge sammeln FleetWise, zu einem attraktiven Ziel für Hacker und erschweren es, sie dauerhaft zu sichern.

Konfiguration und IT-Steuerung liegen in der gemeinsamen Verantwortung AWS von Ihnen, unserem Kunden. Weitere Informationen finden Sie im Modell der AWS gemeinsamen Verantwortung.

Bewährte Sicherheitsmethoden für das AWS IoT FleetWise

AWS IoT FleetWise bietet eine Reihe von Sicherheitsfunktionen, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden stellen allgemeine Richtlinien und keine vollständige Sicherheitslösung dar. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Überlegungen und nicht als bindend ansehen.

Weitere Informationen zur Sicherheit AWS IoT finden Sie unter Bewährte Sicherheitsmethoden AWS IoT Core im AWS IoT Entwicklerhandbuch

Erteilen von Mindestberechtigungen

Folgen Sie dem Prinzip der geringsten Rechte, indem Sie die Mindestanzahl an Berechtigungen in IAM-Rollen verwenden. Beschränken Sie die Verwendung des * Platzhalters für die `Resource` Eigenschaften `Action` und in Ihren IAM-Richtlinien. Deklarieren Sie stattdessen, wenn möglich, eine endliche Menge von Aktionen und Ressourcen. Weitere Informationen zu den geringsten Berechtigungen und anderen bewährten Methoden für Richtlinien finden Sie unter the section called “Best Practices für Richtlinien”.

Keine Protokollierung sensibler Informationen

Sie sollten die Protokollierung von Anmeldeinformationen und anderen persönlich identifizierbaren Informationen (PII) verhindern. Wir empfehlen Ihnen, die folgenden Sicherheitsvorkehrungen zu implementieren:

- Verwenden Sie keine vertraulichen Informationen in Gerätenamen.
- Verwenden Sie keine vertraulichen Informationen in den Namen und in den Namen IDs von AWS FleetWise IoT-Ressourcen, z. B. in den Namen von Kampagnen, Decoder-Manifesten, Fahrzeugmodellen und Signalkatalogen oder in den Namen IDs von Fahrzeugen und Flotten.

Wird verwendet AWS CloudTrail , um den API-Aufrufverlauf anzuzeigen

Sie können einen Verlauf der AWS FleetWise IoT-API-Aufrufe einsehen, die auf Ihrem Konto getätigt wurden, um Sicherheitsanalysen durchzuführen und betriebliche Probleme zu beheben. Um einen Verlauf der AWS FleetWise IoT-API-Aufrufe zu erhalten, die auf Ihrem Konto getätigt wurden, schalten Sie einfach die CloudTrail Option ein AWS-Managementkonsole. Weitere Informationen finden Sie unter the section called “CloudTrail protokolliert”.

Synchronisieren der internen Uhr Ihres Geräts

Es ist wichtig, dass Sie eine genaue Uhrzeit auf Ihrem Gerät haben. X.509-Zertifikate haben ein Ablaufdatum und eine Ablaufzeit. Die Uhr auf Ihrem Gerät wird verwendet, um sicherzustellen, dass ein Serverzertifikat noch gültig ist. Geräteuhren können im Laufe der Zeit unpräzise werden, oder die Batterien werden entladen.

Weitere Informationen finden Sie in der bewährten Methode Synchronisieren der internen Uhr Ihres Geräts im AWS IoT Core -Entwicklerhandbuch.

Überwachen Sie das AWS IoT FleetWise

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von AWS IoT FleetWise und Ihren anderen AWS Lösungen. AWS bietet die folgenden Überwachungstools, um das AWS IoT zu beobachten FleetWise, zu melden, wenn etwas nicht stimmt, und gegebenenfalls automatische Maßnahmen zu ergreifen:

- Amazon CloudWatch überwacht Ihre AWS Ressourcen und die Anwendungen, auf denen Sie laufen, AWS in Echtzeit. Sie können Metriken erfassen und verfolgen, benutzerdefinierte Dashboards erstellen und Alarme festlegen, die Sie benachrichtigen oder Maßnahmen ergreifen, wenn eine Metrik einen von Ihnen festgelegten Schwellenwert erreicht. Sie können beispielsweise die CPU-Auslastung oder andere Kennzahlen Ihrer Amazon EC2 EC2-Instances CloudWatch verfolgen und bei Bedarf automatisch neue Instances starten. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.
- Amazon CloudWatch Logs kann verwendet werden, um Ihre Protokolldateien von Amazon EC2 EC2-Instances und anderen Quellen zu überwachen CloudTrail, zu speichern und darauf zuzugreifen. CloudWatch Logs können Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im Amazon CloudWatch Logs-Benutzerhandbuch.
- AWS CloudTrail erfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihrer AWS-Konto. Anschließend werden die Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket übermittelt. Sie können feststellen, welche Benutzer und Konten angerufen wurden AWS, von welcher Quell-IP-Adresse aus die Anrufe getätigt wurden und wann die Anrufe erfolgten. Weitere Informationen finden Sie im AWS CloudTrail -Benutzerhandbuch.

Überwachen Sie AWS IoT FleetWise mit Amazon CloudWatch

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Amazon CloudWatch Amazon-Metriken können Sie Ihre AWS Ressourcen und deren Leistung überwachen. AWS IoT FleetWise sendet Metriken an CloudWatch. Sie können die AWS-Managementkonsole, oder eine API verwenden AWS CLI, um die Metriken aufzulisten, an die AWS IoT FleetWise sendet CloudWatch. Weitere Informationen finden Sie im CloudWatch Amazon-Benutzerhandbuch.

> **Important**
>
> Sie müssen die Einstellungen so konfigurieren, dass AWS IoT Metriken an senden FleetWise kann CloudWatch. Weitere Informationen finden Sie unter Konfigurieren Sie Ihre AWS FleetWise IoT-Einstellungen.

Der `AWS/IoTFleetWise`-Namespace enthält die folgenden Metriken.

Metriken signalisieren

Metrik	Description
IllegalMessageFromEdge	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht entsprach FleetWise nicht dem erforderlichen Format. Einheiten: Anzahl Dimensionen: Keine Gültige Statistiken: Summe
MessageThrottled	Eine vom Fahrzeug an das AWS IoT gesendete Nachricht FleetWise wurde gedrosselt. Dies liegt daran, dass Sie die Dienstlimits für dieses Konto in der aktuellen Region überschritten haben. Einheiten: Anzahl Dimensionen: Keine Gültige Statistiken: Summe

Metrik	Description
ModelingError	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise enthält Signale, die nicht mit dem Fahrzeugmodell verglichen werden können. Einheiten: Anzahl Abmessungen: ModelName, StateTemplateName (optional), SignalCatalogName (optional)
DecodingError	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise enthält Signale, die nicht anhand des Decoder-Manifests des Fahrzeugs dekodiert werden können. Einheiten: Anzahl Abmessungen: DecoderName Gültige Statistiken: Summe
MessageSizeLimitExceeded	Eine vom Fahrzeug an das AWS IoT gesendete Nachricht FleetWise wurde gelöscht. Dies liegt daran, dass Sie die maximale Größe eines Nachrichtendienst-Limits für dieses Konto in der aktuellen Region überschritten haben. Einheiten: Anzahl Dimensionen: Keine Gültige Statistiken: Summe

Metrik	Description
CallCount	Die Anzahl der Nachrichten, die im angegeben en Zeitraum aufgenommen wurden. Einheiten: Anzahl Abmessungen: AccountID
CheckInThrottled	Ein Check-in, der vom Fahrzeug an das AWS IoT gesendet wurde, FleetWise wurde gedrosselt. Dies liegt daran, dass Sie das Servicelimit für dieses Konto in der aktuellen Region überschritten haben. Einheiten: Anzahl Abmessungen: VehicleName Gültige Statistiken: Summe
VehicleAttributeNotFound	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise konnte nicht mit den angegebenen Fahrzeugattributen angereichert werden. Einheiten: Anzahl Abmessungen: Kampagnenname (optional), stateTemplateName (optional), Fahrzeugname Gültige Statistiken: Summe

Fahrzeugkennzahlen

Metrik	Description
VehicleNotFound	Eine vom AWS IoT empfangene Nachricht FleetWise, bei der das Fahrzeug unbekannt ist.

Metrik	Description
	Einheiten: Anzahl Dimensionen: Keine Gültige Statistiken: Summe

Bereitstellungsmetriken

Metrik	Description
PayloadSize	Größe der Nachricht, die vom AWS IoT FleetWise an das Fahrzeug gesendet wird. Einheiten: Anzahl Abmessungen: VehicleName, ResourceTypes (StateTemplates, Kampagnen, DecoderMa nifest)
PayloadSizeLimitExceeded	Eine vom AWS IoT FleetWise an das Fahrzeug gesendete Nachricht hat die maximale Größe eines Nutzlastdienstlimits für dieses Konto in der aktuellen Region überschritten. Einheiten: Anzahl Abmessungen: VehicleName Gültige Statistiken: Summe

Kampagnenmetriken

Metrik	Description
CampaignInvalid	Eine Nachricht, die vom Fahrzeug gesendet und vom AWS IoT empfangen wurde FleetWise , wenn die Kampagne nicht gültig ist.

Metrik	Description
	Einheiten: Anzahl Abmessungen: CampaignName Gültige Statistiken: Summe
CampaignNotFound	Eine Nachricht, die vom Fahrzeug gesendet und vom AWS IoT empfangen wurde FleetWise , wobei die Kampagne nicht bekannt ist. Einheiten: Anzahl Abmessungen: CampaignName Gültige Statistiken: Summe

Metriken der Vorlage für Bundesstaaten

Metrik	Description
NoStateTemplatesAssociated	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise, wobei dem Fahrzeug keine Statusvorlagen zugeordne t sind. Einheiten: Anzahl Gültige Statistiken: Summe

Kampagnendaten, Zielkennzahlen

Metrik	Description
TimestreamWriteError	AWS IoT FleetWise konnte keine Nachricht vom Fahrzeug in die Amazon Timestream-Tabelle schreiben. Einheiten: Anzahl

Metrik	Description
	Abmessungen: DatabaseName, TableName Gültige Statistiken: Summe
S3 WriteError	AWS IoT FleetWise konnte keine Nachricht vom Fahrzeug in den Amazon Simple Storage Service (Amazon S3) -Bucket schreiben. Einheiten: Anzahl Abmessungen: BucketName Gültige Statistiken: Summe
S3 ReadError	AWS IoT FleetWise konnte keinen Objektsch lüssel aus dem Fahrzeug im Amazon Simple Storage Service (Amazon S3) -Bucket lesen. Einheiten: Anzahl Abmessungen: BucketName Gültige Statistiken: Summe

Vom Kunden verwaltete AWS KMS Schlüsselkennzahlen

Metrik	Description
KMSKeyAccessDenied	AWS IoT FleetWise konnte aufgrund eines Fehlers „ AWS KMS Schlüsselzugriff verweiger t“ keine Nachricht vom Fahrzeug in die Timestream-Tabelle oder den Amazon S3 S3-Bucket schreiben. Einheiten: Anzahl Abmessungen: Id KMSKey

Metrik	Description
	Gültige Statistiken: Summe

Überwachen Sie AWS IoT FleetWise mit Amazon CloudWatch Logs

> **Important**
>
> Der Zugriff auf bestimmte AWS FleetWise IoT-Funktionen ist derzeit gesperrt. Weitere Informationen finden Sie unter AWS Verfügbarkeit von Regionen und Funktionen im AWS Internet der Dinge FleetWise.

Amazon CloudWatch Logs überwacht die Ereignisse, die in Ihren Ressourcen auftreten, und benachrichtigt Sie, wenn Probleme auftreten. Wenn Sie eine Warnung erhalten, können Sie auf die Protokolldateien zugreifen, um Informationen über das jeweilige Ereignis zu erhalten. Weitere Informationen finden Sie im Amazon CloudWatch Logs-Benutzerhandbuch.

AWS FleetWise IoT-Protokolle in der CloudWatch Konsole anzeigen

> **Important**
>
> Bevor Sie die AWS FleetWise IoT-Protokollgruppe in der CloudWatch Konsole sehen können, stellen Sie sicher, dass Folgendes zutrifft:
>
> - Sie haben die Protokollierung im AWS IoT aktiviert FleetWise. Weitere Informationen zur Protokollierung finden Sie unter AWS FleetWise IoT-Protokollierung konfigurieren.
> - Es gibt bereits Protokolleinträge, die von AWS IoT Vorgängen geschrieben wurden.

So zeigen Sie Ihre AWS FleetWise IoT-Logs in der CloudWatch Konsole an

1. Öffnen Sie die CloudWatch -Konsole.
2. Wählen Sie im Navigationsbereich Logs, Log-Gruppen aus.
3. Wählen Sie die Protokollgruppe aus.

4. Wählen Sie Protokollgruppe suchen aus. Sie sehen eine vollständige Liste der Protokollereignisse, die für Ihr Konto generiert wurden.
5. Wählen Sie das Erweiterungssymbol, um sich einen einzelnen Stream anzusehen und alle Logs mit einem Log-Level von zu findenERROR.

   Sie können auch eine Abfrage in das Suchfeld Ereignisse filtern eingeben. Sie können beispielsweise die folgende Abfrage ausprobieren:

   ```
   { $.logLevel = "ERROR" }
   ```

Weitere Informationen zum Erstellen von Filterausdrücken finden Sie unter Filter- und Mustersyntax im Amazon CloudWatch Logs-Benutzerhandbuch.

Example Protokolleintrag

```
{
  "accountId": "123456789012",
  "vehicleName": "test-vehicle",
  "message": "Unrecognized signal ID",
  "eventType": "MODELING_ERROR",
  "logLevel": "ERROR",
  "timestamp": 1685743214239,
  "campaignName": "test-campaign",
  "signalCatalogName": "test-catalog",
  "signalId": 10242
}
```

Arten von Signalereignissen

Ereignistyp	Description
MODELING_ERROR	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise enthält Signale, die nicht mit dem Fahrzeugmodell verglichen werden können. Attribute: VehicleName, CampaignName (optional) signalCatalogName, SignalID (optional), SignalValue (optional), signalVal ueRange Min (optional), signalValueRange

Ereignistyp	Description
	Max (optional), (optional), SignalIDs, modelManifestName stateTemplateName
ILLEGAL_MESSAGE_FROM_EDGE	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht entsprach FleetWise nicht dem erforderlichen Format. Attribute: VehicleName, CampaignName, signalCatalogName
DECODIERUNGSFEHLER	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise enthält Signale, die nicht anhand des Decoder-M anifests des Fahrzeugs dekodiert werden können. Attribute: CampaignName,, (optional) SignalName signalCatalogName decoderMa nifestName, (optional) S3URI
MESSAGE_THROTTLED	Eine vom Fahrzeug an das AWS IoT gesendete Nachricht FleetWise wurde gedrosselt. Dies liegt daran, dass Sie die Dienstlimits für dieses Konto in der aktuellen Region überschritten haben. Attribute: accountId, Fahrzeugname, Nachricht, EventType, LogLevel, Zeitstempel
MESSAGE_SIZE_LIMIT_EXCEEDED	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise überschreitet die maximale Größe eines Nachrichtendienstlimits. Attribute: accountId, Fahrzeugname

Ereignistyp	Description
CHECKIN_THROTTLED	Ein Check-in, der vom Fahrzeug an das AWS IoT gesendet wurde, FleetWise wurde gedrosselt. Dies liegt daran, dass Sie das Servicelimit für dieses Konto in der aktuellen Region überschritten haben. Attribute: Fahrzeugname
FAHRZEUGATTRIBUTE_NICHT_GEFUNDEN	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise konnte nicht mit den angegebenen Fahrzeugattributen angereichert werden. Attribute: CampaignName (optional), stateTemplateName (optional), Fahrzeugname, vehicleAttributeNames

Arten von Fahrzeugereignissen

Ereignistyp	Description
VEHICLE_NOT_FOUND	Eine vom AWS IoT empfangene Nachricht FleetWise, bei der das Fahrzeug unbekannt war. Attribute: VehicleName, CampaignName (optional), (optional) stateTemplateName

Arten von Bereitstellungsereignissen

Ereignistyp	Description
PAYLOAD_SIZE_LIMIT_EXCEEDED	Eine vom AWS IoT FleetWise an das Fahrzeug gesendete Nachricht hat die maximale Servicegröße überschritten.

Ereignistyp	Description
	Attribute: VehicleName, CampaignName (optional), (optional) stateTemplateName

Ereignistypen der Kampagne

Ereignistyp	Description
CAMPAIGN_NOT_FOUND	Eine Nachricht, die vom Fahrzeug gesendet und von AWS IoT empfangen wurde FleetWise, wobei die Kampagne unbekannt war. Attribute: VehicleName (optional), CampaignName
CAMPAIGN_INVALID	Eine vom Fahrzeug gesendete und vom AWS IoT empfangene Nachricht FleetWise, bei der die Kampagne nicht gültig war. Attribute: VehicleName (optional), CampaignName

Kampagnendaten, Zielereignistypen

Ereignistyp	Description
TIMESTREAM_WRITE_ERROR	AWS IoT FleetWise konnte keine Nachricht vom Fahrzeug in die Amazon Timestream-Tabelle schreiben. Attribute: Fahrzeugname, Kampagnenname, timestreamDatabaseName timestreamTableName
S3_WRITE_ERROR	AWS IoT FleetWise konnte keine Nachricht vom Fahrzeug in den Amazon Simple Storage Service (Amazon S3) -Bucket schreiben.

Ereignistyp	Description
	Attribute: CampaignName, DestinationName
S3_READ_ERROR	AWS IoT FleetWise konnte keinen Objektsch lüssel aus dem Fahrzeug im Amazon Simple Storage Service (Amazon S3) -Bucket lesen. Attribute: CampaignName, DestinationName

Ereignistypen der Statusvorlage

Ereignistyp	Description
STATE_TEMPLATE_NOT_FOUND	Eine Nachricht, die vom Fahrzeug gesendet und vom AWS IoT empfangen wurde FleetWise , wobei die Statusvorlage unbekannt war. Attribute: Fahrzeugname (optional), stateTemp lateName

Vom Kunden verwaltete AWS KMS wichtige Ereignistypen

Ereignistyp	Description
KMS_KEY_ACCESS_DENIED	AWS IoT FleetWise konnte aufgrund eines Fehlers „ AWS KMS Schlüsselzugriff verweiger t“ keine Nachricht vom Fahrzeug in die Timestream-Tabelle oder den Amazon S3 S3-Bucket schreiben. Attribute: kmsKeyId (optional), resourceARN (optional)

Attribute

Alle CloudWatch Log-Einträge enthalten die folgenden Attribute:

accountId

Deine AWS-Konto ID.

eventType

Der Ereignistyp, für den das Protokoll generiert wurde. Der Wert des Ereignistyps hängt vom Ereignis ab, das den Protokolleintrag generiert hat. Jede Beschreibung des Protokolleintrags enthält den Wert von `eventType` für diesen Protokolleintrag.

logLevel

Die Protokollebene, die verwendet wird. Weitere Informationen finden Sie unter Protokollebenen im AWS IoT Core Entwicklerhandbuch.

Nachricht

Enthält spezifische Details zum Protokoll.

Zeitstempel

Der Epochen-Millisekunden-Zeitstempel, zu dem das IoT das Protokoll verarbeitet hat. AWS FleetWise

Optionale Attribute

CloudWatch Protokolleinträge enthalten optional diese Attribute, abhängig von: `eventType`

decoderManifestName

Der Name des Decoder-Manifests, das das Signal enthält.

Name des Ziels

Der Name des Ziels für Fahrzeugdaten. Zum Beispiel der Amazon S3 S3-Bucket-Name.

campaignName

Der Name der Kampagne.

signalCatalogName

Der Name des Signalkatalogs, der das Signal enthält.

SignalID

Die ID des Fehlersignals.

Signal-IDs

Eine Liste von Fehlersignalen. IDs

Signalname

Der Name des Signals.

signalTimestampEpochFrau

Der Zeitstempel des Fehlersignals.

SignalWert

Der Wert des Fehlersignals.

signalValueRangeMax

Die maximale Reichweite des Fehlersignals.

signalValueRangeMin.

Der Mindestbereich des Fehlersignals.

S3URI

Die eindeutige Amazon S3 S3-ID einer Amazon Ion-Datei aus einer Fahrzeugnachricht.

timestreamDatabaseName

Der Name der Timestream-Datenbank.

timestreamTableName

Der Name der Timestream-Tabelle.

Name des Fahrzeugs

Der Name des Fahrzeugs.

vehicleAttributeNames

Eine Liste von Fahrzeugattributen, die nicht gefunden werden konnten.

AWS FleetWise IoT-Protokollierung konfigurieren

Sie können Ihre AWS FleetWise IoT-Protokolldaten an eine CloudWatch Protokollgruppe senden. CloudWatch Protokolle bieten Transparenz für den Fall, dass FleetWise das AWS IoT Nachrichten

von Fahrzeugen nicht verarbeiten kann. Dies kann beispielsweise aufgrund einer fehlerhaften Konfiguration oder anderer Client-Fehler passieren. Sie werden über alle Fehler informiert, sodass Sie Probleme identifizieren und beheben können.

Bevor Sie Protokolle an senden können CloudWatch, müssen Sie eine CloudWatch Protokollgruppe erstellen. Konfigurieren Sie die Protokollgruppe mit demselben Konto und in derselben Region, die Sie mit AWS IoT verwendet haben FleetWise. Wenn Sie die Protokollierung in AWS IoT aktivieren FleetWise, geben Sie den Namen der Protokollgruppe an. Nachdem die Protokollierung aktiviert wurde, FleetWise übermittelt AWS IoT Protokolle in CloudWatch Protokollstreams an die Protokollgruppe.

Sie können die vom AWS IoT gesendeten Protokolldaten FleetWise in der CloudWatch Konsole anzeigen. Weitere Informationen zur Konfiguration einer CloudWatch Protokollgruppe und zum Anzeigen von Protokolldaten finden Sie unter Arbeiten mit Protokollgruppen. Weitere Informationen zur Einrichtung der Protokollierung für AWS Services finden Sie unter AWS Services, die Logs in CloudWatch Logs veröffentlichen, im Amazon CloudWatch Logs-Benutzerhandbuch.

Berechtigungen zum Veröffentlichen von Protokollen in CloudWatch

Für die Konfiguration der Protokollierung für eine CloudWatch Protokollgruppe sind die in diesem Abschnitt beschriebenen Berechtigungseinstellungen erforderlich. Informationen zur Verwaltung von Berechtigungen finden Sie unter Zugriffsverwaltung für AWS Ressourcen im IAM-Benutzerhandbuch.

Mit diesen Berechtigungen können Sie die Protokollierungskonfiguration ändern, die Protokollzustellung für CloudWatch Ihre Protokollgruppe konfigurieren und Informationen zu dieser Gruppe abrufen.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "iotfleetwise:PutLoggingOptions",
                "iotfleetwise:GetLoggingOptions"
            ],
            "Resource": [
                "*"
            ],
```

```
            "Effect": "Allow",
            "Sid": "IoTFleetwiseLoggingOptionsAPI"
        },
        {
            "Sid": "IoTFleetwiseLoggingCWL",
            "Action": [
                "logs:CreateLogDelivery",
                "logs:GetLogDelivery",
                "logs:UpdateLogDelivery",
                "logs:DeleteLogDelivery",
                "logs:ListLogDeliveries",
                "logs:PutResourcePolicy",
                "logs:DescribeResourcePolicies",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

Wenn Aktionen für alle AWS Ressourcen zulässig sind, wird dies in der Richtlinie mit der `"Resource"` Einstellung von angegeben`"*"`. Das bedeutet, dass die Aktionen für alle AWS Ressourcen zulässig sind, die jede Aktion unterstützt.

Konfigurieren Sie die Protokollierung mithilfe der Konsole

In diesem Abschnitt wird beschrieben, wie die AWS FleetWise IoT-Konsole zur Konfiguration der Protokollierung verwendet wird.

So konfigurieren Sie die Protokollierung mit der AWS FleetWise IoT-Konsole

1. Öffnen Sie die AWS FleetWiseIoT-Konsole.
2. Wählen Sie im linken Bereich Settings (Einstellungen) aus.
3. Wählen Sie auf der Einstellungsseite im Abschnitt Protokollierung die Option Bearbeiten aus.
4. Geben Sie im Abschnitt CloudWatch Protokollierung die Protokollgruppe ein.
5. Um Ihre Änderungen zu speichern, wählen Sie Submit.

Nachdem Sie die Protokollierung aktiviert haben, können Sie Ihre Protokolldaten in der CloudWatch Konsole anzeigen.

Konfigurieren Sie die Protokollierung mit der CLI

In diesem Abschnitt wird beschrieben, wie die Protokollierung für AWS IoT FleetWise mithilfe der CLI konfiguriert wird.

Sie können dieses Verfahren auch mit der API durchführen, indem Sie die Methoden in der AWS API verwenden, die den hier gezeigten CLI-Befehlen entsprechen. Sie können den GetLoggingOptionsAPI-Vorgang verwenden, um die aktuelle Konfiguration abzurufen, und den PutLoggingOptionsAPI-Vorgang, um die Konfiguration zu ändern.

So konfigurieren Sie die Protokollierung für AWS IoT mit der CLI FleetWise

1. Verwenden Sie den get-logging-options Befehl, um die Protokollierungsoptionen für Ihr Konto abzurufen.

   ```
   aws iotfleetwise get-logging-options
   ```

2. Verwenden Sie den put-logging-options Befehl, um die Protokollierung zu aktivieren.

   ```
   aws iotfleetwise put-logging-options --cloud-watch-log-delivery
    logType=ERROR,logGroupName=MyLogGroup
   ```

   Wobei:

   logType

   Der Protokolltyp, um Daten an CloudWatch Logs zu senden. Um die Protokollierung zu deaktivieren, ändern Sie den Wert auf`OFF`.

   logGroupName

   Die Gruppe CloudWatch Logs, an die der Vorgang Daten sendet. Stellen Sie sicher, dass Sie den Namen der Protokollgruppe erstellen, bevor Sie die Protokollierung für AWS IoT aktivieren FleetWise.

Nachdem Sie die Protokollierung aktiviert haben, finden Sie weitere Informationen unter Suchen nach Protokolleinträgen mit der AWS CLI.

AWS FleetWise IoT-API-Aufrufe protokollieren mit AWS CloudTrail

AWS IoT FleetWise ist in einen Dienst integriert AWS CloudTrail, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Dienst im AWS IoT ausgeführt wurden FleetWise. CloudTrail erfasst alle API-Aufrufe für AWS IoT FleetWise als Ereignisse. Zu den erfassten Aufrufen gehören Aufrufe von der AWS FleetWise IoT-Konsole und Codeaufrufen für die AWS FleetWise IoT-API-Operationen. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3 S3-Bucket aktivieren, einschließlich Ereignissen für AWS IoT FleetWise. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse trotzdem in der CloudTrail Konsole im Ereignisverlauf anzeigen. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an AWS IoT gestellt wurde FleetWise, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln.

Weitere Informationen CloudTrail dazu finden Sie im AWS CloudTrail Benutzerhandbuch.

AWS FleetWise IoT-Informationen in CloudTrail

CloudTrail ist in Ihrem AWS Konto aktiviert, wenn Sie das Konto erstellen. Wenn eine Aktivität im AWS IoT auftritt FleetWise, wird diese Aktivität zusammen mit anderen CloudTrail AWS Dienstereignissen im Ereignisverlauf in einem Ereignis aufgezeichnet. Sie können die neusten Ereignisse in Ihr AWS -Konto herunterladen und dort suchen und anzeigen. Weitere Informationen finden Sie unter Ereignisse mit CloudTrail Ereignisverlauf anzeigen.

Für eine fortlaufende Aufzeichnung von Ereignissen in Ihrem AWS Konto, einschließlich Ereignissen für das Internet der AWS IoT FleetWise, erstellen Sie einen Trail. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie einen Pfad in der Konsole anlegen, gilt dieser für alle AWS-Regionen. Der Trail protokolliert Ereignisse aus allen Regionen der AWS Partition und übermittelt die Protokolldateien an den von Ihnen angegebenen Amazon S3 S3-Bucket. Darüber hinaus können Sie andere AWS Dienste konfigurieren, um die in den CloudTrail Protokollen gesammelten Ereignisdaten weiter zu analysieren und darauf zu reagieren. Weitere Informationen finden Sie hier:

- Übersicht zum Erstellen eines Trails
- CloudTrail unterstützte Dienste und Integrationen
- Konfiguration von Amazon SNS SNS-Benachrichtigungen für CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen

- Empfangen von CloudTrail Protokolldateien von mehreren Konten

Alle AWS FleetWise IoT-Aktionen werden von der AWS FleetWise IoT-API-Referenz protokolliert CloudTrail und sind in dieser dokumentiert. Beispielsweise generieren Aufrufe der `GetModelManifest` Aktionen `CreateCampaignAssociateVehicleFleet`, und Einträge in den CloudTrail Protokolldateien.

Jeder Ereignis- oder Protokolleintrag enthält Informationen zu dem Benutzer, der die Anforderung generiert hat. Die Identitätsinformationen unterstützen Sie bei der Ermittlung der folgenden Punkte:

- Gibt an, ob die Anforderung mit Root- oder IAM-Benutzer-Anmeldeinformationen ausgeführt wurde.
- Gibt an, ob die Anforderung mit temporären Sicherheitsanmeldeinformationen für eine Rolle oder einen Verbundbenutzer gesendet wurde.
- Ob die Anfrage von einem anderen AWS Dienst gestellt wurde.

Weitere Informationen finden Sie unter CloudTrail -Element userIdentity.

AWS FleetWise IoT-Protokolldateieinträge verstehen

Ein Trail ist eine Konfiguration, die die Übertragung von Ereignissen als Protokolldateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket ermöglicht. CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, Datum und Uhrzeit der Aktion, Anforderungsparameter usw. CloudTrail Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe, sodass sie nicht in einer bestimmten Reihenfolge angezeigt werden.

Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der den *`AssociateVehicleFleet`* Vorgang demonstriert.


```
{
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE",
        "arn": "arn:aws:iam::111122223333:assumed-role/NikkiWolf",
        "accountId": "111122223333",
        "accessKeyId": "access-key-id",
        "userName": "NikkiWolf"
      },
```

```
    "eventTime": "2021-11-30T09:56:35Z",
    "eventSource": "iotfleetwise.amazonaws.com",
    "eventName": "AssociateVehicleFleet",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "192.0.2.21",
    "userAgent": "aws-cli/2.3.2 Python/3.8.8 Darwin/18.7.0 botocore/2.0.0",
    "requestParameters": {
        "fleetId": "f1234567890",
        "vehicleId": "v0213456789"
     },
    "responseElements": {
    },
    "requestID": "9f861429-11e3-11e8-9eea-0781b5c0ac21",
    "eventID": "17385819-4927-41ee-a6a5-29ml0br812v4",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

Dokumentenverlauf für den AWS IoT FleetWise Developer Guide

In der folgenden Tabelle werden die Dokumentationsversionen für AWS IoT beschrieben FleetWise.

Änderung	Beschreibung	Datum
Aktualisierung überwachen	Die AWS FleetWise IoT-Überwachung wurde aktualisiert und umfasst nun mehr Metriken und Protokollierungsoptionen. Weitere Informationen finden Sie unter AWS IoT überwachen FleetWise mit Amazon CloudWatch und AWS IoT FleetWise mit Amazon CloudWatch Logs überwachen.	23. Dezember 2025
Aktualisierung der Richtlinien für dienstbezogene Rollen	Das `AWSIoTFleetwiseServiceRolePolicy` wurde aktualisiert und umfasst nun auch Berechtigungen für die Veröffentlichung von Nutzungsmetriken im `AWS/Usage` Namespace. Weitere Informationen finden Sie unter AWSIoTFleetwiseServiceRolePolicy Richtlinienaktualisierungen.	13. Juni 2025
Regionale Erweiterung	AWS IoT FleetWise ist jetzt in der Region Asien-Pazifik (Mumbai) verfügbar (nur Gated Access).	21. November 2024

Eingeschränkte allgemeine Verfügbarkeit neuer Funktionen	AWS IoT unterstützt FleetWise jetzt Gated Access für Kampagnen, um Daten zu speichern und weiterzuleiten, ein MQTT-Thema als Datenziel zu konfigurieren und diagnostische Fehlercodedaten zu sammeln. Es unterstützt jetzt auch Gated Access für die netzwerkunabhängige Datenerfassung mithilfe benutzerdefinierter Dekodierungsschnittstellen, die Konfiguration von Befehlen und die Überwachung des letzten bekannten Zustands von Fahrzeugen.	21. November 2024
Senden Sie Kampagnendaten an ein MQTT-Thema	AWS IoT unterstützt FleetWise jetzt das Senden von während Kampagnen gesammelten Daten an ein von Ihnen festgelegtes MQTT-Thema sowie die Möglichkeit, die Daten in Amazon S3 oder Amazon Timestream zu speichern.	1. Mai 2024

Vorschau der Daten des Bildverarbeitungssystems	Sie können die Vorschau von Bildverarbeitungss ystemdaten aus dem AWS IoT verwenden FleetWise , um Daten aus Fahrzeugs ichtsystemen, einschließlich Kameras, Radaren und Lidars, zu sammeln und zu organisieren. Dabei werden sowohl strukturierte als auch unstrukturierte Bildsyste mdaten, Metadaten (Event-ID, Kampagne, Fahrzeug) und Standardsensordaten (Telemetriedaten) automatisch in der Cloud synchronisiert.	26. November 2023
AWS KMS vom Kunden verwaltete Schlüssel	AWS IoT unterstützt FleetWise jetzt vom AWS KMS Kunden verwaltete Schlüssel. Sie können den KMS-Schlüssel verwenden, um serverseitige Daten zu AWS FleetWise IoT-Ressourcen (Signalka talog, Fahrzeugmodell, Decodermanifest, Fahrzeuge und Konfigurationen für Datenerfassungskampagnen) zu verschlüsseln, die in gespeichert sind. AWS Cloud	16. Oktober 2023

Objektspeicher in Amazon S3	AWS IoT unterstützt FleetWise jetzt das Speichern von Daten mit Amazon Simple Storage Service (Amazon S3). Sie können Daten, die während Kampagnen gesammelt wurden, zusätzlich zu Amazon Timestream in Amazon S3 speichern.	01. Juni 2023
Allgemeine Verfügbarkeit	Dies ist die öffentliche Version von AWS IoT FleetWise.	27. September 2022
Erstversion	Dies ist die Vorabversion des AWS IoT FleetWise Developer Guide.	30. November 2021

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.