



Panduan Developer

AWS IoT FleetWise



AWS IoT FleetWise: Panduan Developer

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan hak milik masing-masing pemiliknya, yang mungkin atau tidak terafiliasi, terkait dengan, atau disponsori oleh Amazon.

Table of Contents

Apa itu AWS IoT? FleetWise	1
Keuntungan	2
Kasus penggunaan	2
Apakah Anda baru mengenal AWS IoT? FleetWise	3
Mengakses IoT AWS FleetWise	3
Harga untuk AWS IoT FleetWise	3
Bagaimana AWS IoT bekerja FleetWise	3
Konsep kunci	4
Fitur AWS IoT FleetWise	8
Layanan-layanan terkait	8
Pengaturan AWS IoT FleetWise	10
Siapkan Akun AWS	10
Mendaftar untuk Akun AWS	10
Buat pengguna dengan akses administratif	11
Memulai di konsol	12
Mengkonfigurasi pengaturan	12
Konfigurasikan pengaturan (konsol)	13
Konfigurasikan pengaturan (AWS CLI)	14
Memulai	16
Persyaratan	16
Menggunakan demo perangkat lunak Edge Agent	16
Memulai (konsol)	17
Prasyarat	18
Langkah 1: Siapkan perangkat lunak Edge Agent untuk AWS IoT FleetWise	19
Langkah 2: Buat model kendaraan	20
Langkah 3: Buat manifes decoder	22
Langkah 4: Konfigurasikan manifes decoder	23
Langkah 5: Buat kendaraan	24
Langkah 6: Buat kampanye	25
Langkah 7: Bersihkan	27
Langkah selanjutnya	27
Menelan data ke cloud	28
Kendaraan pemodelan	31
Katalog sinyal	34

Konfigurasi sinyal	36
Buat katalog sinyal (AWS CLI)	43
Impor katalog sinyal	47
Perbarui katalog sinyal (AWS CLI)	57
Hapus katalog sinyal (AWS CLI)	59
Dapatkan informasi katalog sinyal (AWS CLI)	59
Model kendaraan	60
Buat model kendaraan	61
Perbarui model kendaraan (AWS CLI)	67
Hapus model kendaraan	68
Dapatkan informasi model kendaraan (AWS CLI)	69
Manifestasi dekoder	70
Konfigurasi antarmuka jaringan dan sinyal decoder	72
Buat manifes decoder	75
Perbarui manifes decoder (AWS CLI)	83
Hapus manifes decoder	83
Dapatkan informasi manifes decoder (AWS CLI)	84
Kendaraan	86
Kendaraan penyediaan	87
Otentikasi kendaraan	88
Otorisasi kendaraan	90
Topik yang dipesan	91
Buat kendaraan	93
Buat kendaraan (konsol)	93
Buat kendaraan (AWS CLI)	95
Buat beberapa kendaraan (AWS CLI)	97
Perbarui kendaraan (AWS CLI)	98
Perbarui beberapa kendaraan (AWS CLI)	100
Hapus kendaraan	101
Hapus kendaraan (konsol)	101
Hapus kendaraan (AWS CLI)	101
Dapatkan informasi kendaraan (AWS CLI)	102
Armada	103
Buat armada (AWS CLI)	104
Mengaitkan kendaraan dengan armada (AWS CLI)	105
Putus kendaraan dari armada (AWS CLI)	105

Memperbarui armada (AWS CLI)	106
Menghapus armada (AWS CLI)	106
Dapatkan informasi armada (AWS CLI)	106
Kampanye	108
Buat kampanye	113
Buat kampanye (konsol)	114
Buat kampanye (AWS CLI)	121
Ekspresi logis untuk kampanye	125
Memperbarui kampanye (AWS CLI)	126
Menghapus kampanye	127
Menghapus kampanye (konsol)	127
Menghapus kampanye (AWS CLI)	127
Dapatkan informasi kampanye (AWS CLI)	127
Memproses dan memvisualisasikan data kendaraan	129
Memproses data kendaraan di Timestream	129
Memvisualisasikan data kendaraan yang disimpan di Timestream	130
Memproses data kendaraan di S3	130
Format objek S3	131
Menganalisis data kendaraan yang disimpan di S3	131
AWS CLI dan AWSSDK	134
Pemecahan Masalah	135
Masalah manifes decoder	135
Edge Agent untuk AWS masalah perangkat lunak IoT FleetWise	139
Masalah: Perangkat lunak Edge Agent tidak dimulai.	139
Masalah: [ERROR] [IoT FleetWise Engine: :connect]: [Gagal memuat perpustakaan persistensi]	140
Masalah: Perangkat lunak Edge Agent tidak mengumpulkan PID diagnostik on-board (OBD) II dan kode masalah diagnostik (DTC).	141
Masalah: Agen Edge untuk FleetWise perangkat lunak AWS IoT tidak mengumpulkan data dari jaringan atau tidak dapat menerapkan aturan pemeriksaan data.	141
Masalah: [ERROR] [AwsSlotConnectivityModule: :connect]: [Koneksi gagal dengan kesalahan] atau [WARN] [AwsSlotChannel: :send]: [Tidak ada Koneksi MQTT yang hidup.] ..	142
Keamanan	143
Perlindungan data	144
Enkripsi diam	145
Enkripsi bergerak	145

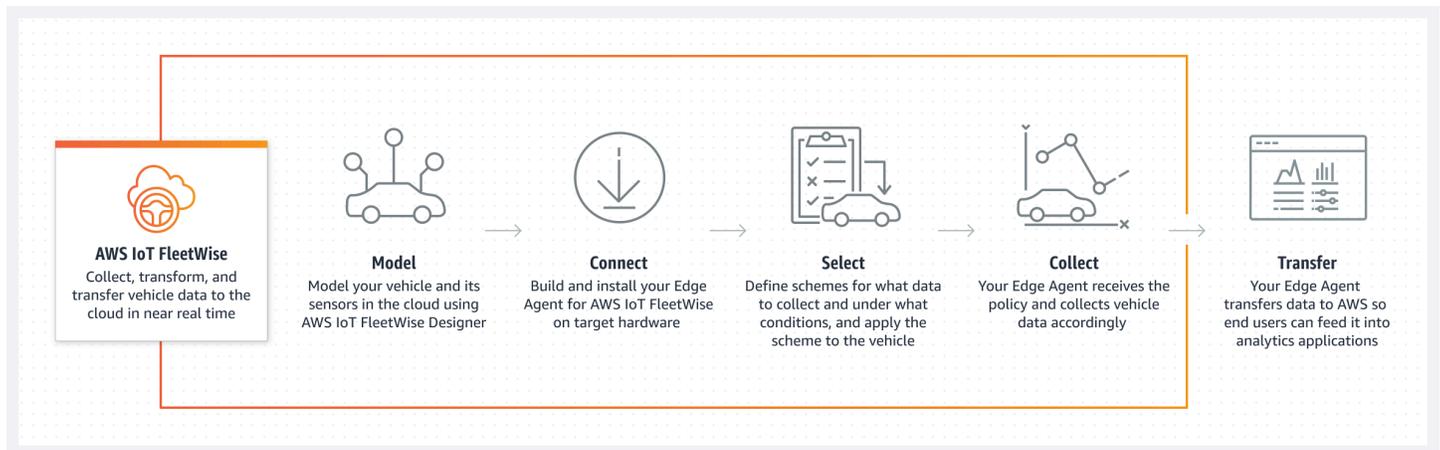
Enkripsi data	146
Mengendalikan akses	154
Pemberian Izin AWS IoT FleetWise akses ke tujuan Amazon S3	154
Pemberian Izin AWS IoT FleetWise akses ke tujuan Amazon Timestream	157
Identity and Access Management	160
Audiens	160
Mengautentikasi dengan identitas	161
Mengelola akses menggunakan kebijakan	165
Bagaimana AWS IoT bekerja FleetWise dengan IAM	167
Contoh kebijakan berbasis identitas	177
Pemecahan Masalah	180
Validasi Kepatuhan	182
Ketangguhan	184
Keamanan infrastruktur	184
Menghubungkan ke AWS IoT FleetWise melalui titik akhir antarmuka VPC	185
Konfigurasi dan analisis kerentanan	188
Praktik terbaik keamanan	189
Berikan izin minimum yang memungkinkan	189
Jangan log informasi sensitif	189
Gunakan AWS CloudTrail untuk melihat riwayat API panggilan	189
Sinkronkan jam perangkat Anda	190
Pemantauan	191
Pemantauan CloudWatch dengan	191
Pemantauan dengan CloudWatch Log	195
Melihat FleetWise log AWS IoT di konsol CloudWatch	195
Mengkonfigurasi logging	201
Log CloudTrail	203
AWSIoT FleetWise informasi dalam CloudTrail	204
Memahami AWSIoT FleetWise entri berkas log	205
Riwayat dokumen	207
.....	ccix

Apa itu AWS IoT? FleetWise

AWS IoT FleetWise adalah layanan terkelola yang dapat Anda gunakan untuk mengumpulkan data kendaraan dan mengaturnya di cloud. Anda dapat menggunakan data yang dikumpulkan untuk meningkatkan kualitas, kinerja, dan otonomi kendaraan. Dengan AWS IoT FleetWise, Anda dapat mengumpulkan dan mengatur data dari kendaraan yang menggunakan protokol dan format data yang berbeda. AWS IoT FleetWise membantu mengubah pesan tingkat rendah menjadi nilai yang dapat dibaca manusia dan menstandarisasi format data di cloud untuk analisis data. Anda juga dapat menentukan kampanye pengumpulan data untuk mengontrol data kendaraan apa yang akan dikumpulkan dan kapan harus mentransfer data tersebut ke cloud.

Saat data kendaraan berada di cloud, Anda dapat menggunakannya untuk aplikasi yang menganalisis kesehatan armada kendaraan. Data ini dapat membantu Anda mengidentifikasi potensi masalah pemeliharaan, membuat sistem infotainment di dalam kendaraan lebih cerdas, dan meningkatkan teknologi canggih seperti mengemudi otonom dan sistem bantuan pengemudi dengan analitik dan pembelajaran mesin (ML).

Diagram berikut menunjukkan arsitektur dasar AWS IoT FleetWise.



Topik

- [Keuntungan](#)
- [Kasus penggunaan](#)
- [Apakah Anda baru mengenal AWS IoT? FleetWise](#)
- [Mengakses IoT AWS FleetWise](#)
- [Harga untuk AWS IoT FleetWise](#)

- [Bagaimana AWS IoT bekerja FleetWise](#)
- [Layanan-layanan terkait](#)

Keuntungan

Manfaat utama AWS IoT FleetWise adalah:

Kumpulkan data kendaraan dengan lebih cerdas

Tingkatkan relevansi data dengan pengumpulan data cerdas yang hanya mengirimkan data yang Anda butuhkan ke cloud untuk dianalisis.

Menganalisis data standar dan luas armada dengan mudah

Menganalisis data standar dari armada kendaraan tanpa perlu mengembangkan pengumpulan data kustom atau sistem logging.

Sinkronisasi data otomatis di cloud

Dapatkan tampilan terpadu data yang dikumpulkan dari sensor standar (data telemetri) dan sistem penglihatan (data dari kamera, radar, dan lidar), dan jaga agar tetap disinkronkan secara otomatis di cloud. AWS IoT FleetWise menyimpan data sistem penglihatan, metadata, dan data sensor standar yang terstruktur dan tidak terstruktur secara otomatis disinkronkan di cloud. Ini merampingkan proses untuk mengumpulkan tampilan gambar lengkap peristiwa dan mendapatkan wawasan.

Note

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Kasus penggunaan

Skenario di mana Anda dapat menggunakan AWS IoT FleetWise meliputi yang berikut:

Latih model AI/ML

Terus meningkatkan model pembelajaran mesin yang digunakan untuk sistem bantuan pengemudi otonom dan canggih dengan mengumpulkan data dari kendaraan produksi.

Meningkatkan pengalaman pelanggan digital

Gunakan data dari sistem infotainment untuk membuat konten audiovisual dalam kendaraan dan wawasan dalam aplikasi menjadi lebih relevan.

Menjaga kesehatan armada kendaraan

Gunakan wawasan dari data armada untuk memantau kesehatan baterai EV dan tingkat pengisian daya, mengelola jadwal pemeliharaan, menganalisis konsumsi bahan bakar, dan banyak lagi.

Apakah Anda baru mengenal AWS IoT? FleetWise

Jika Anda baru mengenal AWS IoT FleetWise, kami sarankan Anda mulai dengan membaca bagian berikut:

- [Bagaimana AWS IoT bekerja FleetWise](#)
- [Mengatur AWS IoT FleetWise](#)
- [Demo perangkat lunak Edge Agent](#)
- [Menelan data ke cloud](#)

Mengakses IoT AWS FleetWise

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk mengakses AWS IoT. FleetWise

Harga untuk AWS IoT FleetWise

Kendaraan mengirim data ke cloud melalui pesan MQTT. Anda membayar pada akhir setiap bulan untuk kendaraan yang Anda buat di AWS IoT FleetWise. Anda juga membayar pesan yang Anda kumpulkan dari kendaraan. Untuk informasi terkini tentang harga, lihat halaman [FleetWise Harga AWS IoT](#). Untuk mempelajari lebih lanjut tentang protokol pesan MQTT, lihat [MQTT](#) di Panduan Pengembang. AWS IoT Core

Bagaimana AWS IoT bekerja FleetWise

Bagian berikut memberikan gambaran umum tentang komponen FleetWise layanan AWS IoT dan bagaimana mereka berinteraksi.

Setelah Anda membaca pendahuluan ini, lihat [Mengatur AWS IoT FleetWise](#) bagian untuk mempelajari cara mengatur AWS IoT FleetWise.

Topik

- [Konsep kunci](#)
- [Fitur AWS IoT FleetWise](#)

Konsep kunci

AWS IoT FleetWise menyediakan kerangka pemodelan kendaraan bagi Anda untuk memodelkan kendaraan Anda dan sensor serta aktuatornya di cloud. Untuk mengaktifkan komunikasi yang aman antara kendaraan Anda dan cloud, AWS IoT FleetWise juga menyediakan implementasi referensi untuk membantu Anda mengembangkan perangkat lunak Edge Agent yang dapat Anda instal di kendaraan Anda. Anda dapat menentukan skema pengumpulan data di cloud dan menyebarkannya ke kendaraan Anda. Perangkat lunak Edge Agent yang berjalan di kendaraan Anda menggunakan skema pengumpulan data untuk mengontrol data apa yang akan dikumpulkan dan kapan harus mentransfernya ke cloud.

Berikut ini adalah konsep inti dari AWS IoT FleetWise.

Sinyal

Sinyal adalah struktur fundamental yang Anda tentukan untuk berisi data kendaraan dan metadatanya. Sinyal dapat berupa atribut, cabang, sensor, atau aktuator. Misalnya, Anda dapat membuat sensor untuk menerima nilai suhu di dalam kendaraan, dan menyimpan metadatanya, termasuk nama sensor, tipe data, dan unit. Untuk informasi selengkapnya, lihat [Membuat dan mengelola katalog sinyal](#).

Atribut

Atribut mewakili informasi statis yang umumnya tidak berubah, seperti tanggal pabrikan dan pembuatan.

Cabang

Cabang mewakili sinyal dalam struktur bersarang. Cabang menunjukkan hierarki sinyal. Misalnya, `Vehicle` cabang memiliki cabang anak, `Powertrain`. `Powertrain` cabang memiliki cabang anak, `combustionEngine`. Untuk menemukan `combustionEngine` cabang, gunakan `Vehicle.Powertrain.combustionEngine` ekspresi.

Sensor

Data sensor melaporkan keadaan kendaraan saat ini dan berubah seiring waktu, karena keadaan kendaraan berubah, seperti level cairan, suhu, getaran, atau tegangan.

Aktuator

Data aktuator melaporkan keadaan perangkat kendaraan, seperti motor, pemanas, dan kunci pintu. Mengubah keadaan perangkat kendaraan dapat memperbarui data aktuator. Misalnya, Anda dapat menentukan aktuator untuk mewakili pemanas. Aktuator menerima data baru saat Anda menghidupkan atau mematikan pemanas.

Struktur kustom

Struktur kustom (juga dikenal sebagai struct) mewakili struktur data yang kompleks atau tingkat tinggi. Ini memfasilitasi pengikatan logis atau pengelompokan data yang berasal dari sumber yang sama. Struct digunakan ketika data dibaca atau ditulis dalam operasi atom, seperti untuk mewakili tipe data yang kompleks atau bentuk tingkat tinggi.

Sinyal tipe struct didefinisikan dalam katalog sinyal menggunakan referensi ke tipe data struct alih-alih tipe data primitif. Structs dapat digunakan untuk semua jenis sinyal termasuk sensor, atribut, aktuator, dan tipe data sistem visi. Jika sinyal tipe struct dikirim atau diterima, AWS FleetWise IoT mengharapkan semua item yang disertakan memiliki nilai yang valid, jadi semua item wajib. Misalnya, jika struct berisi item `Vehicle.camera.Image.Height`, `Vehicle.Camera.Image.Width`, dan `Vehicle.Camera.Image.Data` — diharapkan sinyal yang dikirim berisi nilai untuk semua item ini.

Note

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Properti kustom

Properti kustom mewakili anggota struktur data yang kompleks. Tipe data properti dapat berupa primitif atau struct lain.

Saat merepresentasikan bentuk tingkat tinggi menggunakan struct dan properti kustom, bentuk tingkat tinggi yang dimaksudkan selalu didefinisikan dan dilihat sebagai struktur pohon. Properti kustom digunakan untuk mendefinisikan semua node daun sementara struct digunakan untuk mendefinisikan semua node non-daun.

Katalog sinyal

Katalog sinyal berisi kumpulan sinyal. Sinyal dalam katalog sinyal dapat digunakan untuk memodelkan kendaraan yang menggunakan protokol dan format data yang berbeda. Misalnya, ada dua mobil yang dibuat oleh pembuat mobil yang berbeda: satu menggunakan protokol Control Area Network (CAN bus); yang lain menggunakan protokol On-board Diagnostics (OBD). Anda dapat menentukan sensor dalam katalog sinyal untuk menerima nilai suhu di dalam kendaraan. Sensor ini dapat digunakan untuk mewakili termokopel di kedua mobil. Untuk informasi selengkapnya, lihat [Membuat dan mengelola katalog sinyal](#).

Model kendaraan (manifes model)

Model kendaraan adalah struktur deklaratif yang dapat Anda gunakan untuk membakukan format kendaraan Anda dan untuk menentukan hubungan antara sinyal di kendaraan. Model kendaraan menegakkan informasi yang konsisten di beberapa kendaraan dari jenis yang sama. Anda menambahkan sinyal untuk membuat model kendaraan. Untuk informasi selengkapnya, lihat [Membuat dan mengelola model kendaraan](#).

Manifes dekoder

Manifestasi decoder berisi informasi decoding untuk setiap sinyal dalam model kendaraan. Sensor dan aktuator dalam kendaraan mengirimkan pesan tingkat rendah (data biner). Dengan manifes decoder, AWS FleetWise IoT mampu mengubah data biner menjadi nilai yang dapat dibaca manusia. Setiap manifes decoder dikaitkan dengan model kendaraan. Untuk informasi selengkapnya, lihat [Buat dan kelola manifes decoder](#).

Antarmuka jaringan

Berisi informasi tentang protokol yang digunakan jaringan dalam kendaraan. AWS IoT FleetWise mendukung protokol berikut.

Jaringan Area Pengontrol (CAN bus)

Protokol yang mendefinisikan bagaimana data dikomunikasikan antara unit kontrol elektronik (ECU). ECU dapat berupa unit kontrol mesin, airbag, atau sistem audio.

Diagnostik on-board (OBD) II

Protokol yang dikembangkan lebih lanjut yang mendefinisikan bagaimana data diagnostik mandiri dikomunikasikan antara ECU. Ini menyediakan sejumlah kode masalah diagnostik standar (DTC) yang membantu mengidentifikasi apa yang salah dengan kendaraan Anda.

Middleware kendaraan

Middleware kendaraan didefinisikan sebagai jenis antarmuka jaringan. Contoh middleware kendaraan termasuk Robot Operating System (ROS 2) dan Scalable Service-oriented Middleware over IP (SOME/IP).

Note

AWS IoT FleetWise mendukung middleware ROS 2 untuk data sistem visi.

Sinyal decoder

Memberikan informasi decoding terperinci untuk sinyal tertentu. Setiap sinyal yang ditentukan dalam model kendaraan harus dipasangkan dengan sinyal decoder. Jika manifes decoder berisi antarmuka jaringan CAN, itu harus berisi sinyal decoder CAN. Jika manifes decoder berisi antarmuka jaringan OBD, itu harus berisi sinyal decoder OBD.

Manifes decoder harus berisi sinyal decoder pesan jika juga berisi antarmuka middleware kendaraan.

Kendaraan

Representasi virtual kendaraan fisik Anda, seperti mobil atau truk. Kendaraan adalah contoh model kendaraan. Kendaraan yang dibuat dari model kendaraan yang sama mewarisi kelompok sinyal yang sama. Setiap kendaraan sesuai dengan suatu AWS IoT hal.

Armada

Armada mewakili sekelompok kendaraan. Sebelum Anda dapat dengan mudah mengelola armada kendaraan, Anda harus mengaitkan kendaraan individu dengan armada.

Kampanye

Berisi skema pengumpulan data. Anda menentukan kampanye di cloud dan menerapkannya ke kendaraan atau armada. Kampanye memberikan instruksi perangkat lunak Edge Agent tentang cara memilih, mengumpulkan, dan mentransfer data ke cloud.

Skema pengumpulan data

Skema pengumpulan data memberikan instruksi perangkat lunak Edge Agent tentang cara mengumpulkan data. Saat ini, AWS IoT FleetWise mendukung skema pengumpulan berbasis kondisi dan skema pengumpulan berbasis waktu.

Skema pengumpulan berbasis kondisi

Gunakan ekspresi logis untuk mengenali data apa yang akan dikumpulkan. Perangkat lunak Edge Agent mengumpulkan data ketika kondisi terpenuhi. Misalnya, jika ekspresinya `$variable.myVehicle.InVehicleTemperature >35.0`, perangkat lunak Edge Agent mengumpulkan nilai suhu yang lebih besar dari 35,0.

Skema pengumpulan berbasis waktu

Tentukan periode waktu dalam milidetik untuk menentukan seberapa sering mengumpulkan data. Misalnya, jika periode waktunya 10.000 milidetik, perangkat lunak Edge Agent mengumpulkan data setiap 10 detik sekali.

Fitur AWS IoT FleetWise

Berikut ini adalah fitur utama AWS IoT FleetWise.

Pemodelan kendaraan

Bangun representasi virtual kendaraan Anda dan terapkan format umum untuk mengatur sinyal kendaraan. AWS IoT FleetWise mendukung [Spesifikasi Sinyal Kendaraan \(VSS\)](#) yang dapat Anda gunakan untuk menstandarisasi sinyal kendaraan.

Pengumpulan data berbasis skema

Tentukan skema untuk mentransfer hanya data kendaraan bernilai tinggi ke cloud. Anda dapat menentukan skema berbasis kondisi untuk mengontrol data apa yang akan dikumpulkan, seperti nilai suhu data dalam kendaraan yang lebih besar dari 40 derajat. Anda juga dapat menentukan skema berbasis waktu untuk mengontrol seberapa sering mengumpulkan data.

Edge Agent untuk perangkat AWS lunak IoT FleetWise

Perangkat lunak Edge Agent yang berjalan di kendaraan memfasilitasi komunikasi antara kendaraan dan cloud. Sementara kendaraan terhubung ke cloud, perangkat lunak Edge Agent terus menerima skema pengumpulan data dan mengumpulkan data yang sesuai.

Layanan-layanan terkait

AWS IoT FleetWise terintegrasi dengan AWS layanan berikut untuk meningkatkan ketersediaan dan skalabilitas solusi cloud Anda.

- **AWS IoT Core**— Daftarkan dan kontrol AWS IoT perangkat yang mengunggah data kendaraan ke AWS IoT FleetWise. Untuk informasi selengkapnya, lihat [Apa yang ada AWS IoT](#) di Panduan AWS IoT Pengembang.
- **Amazon Timestream** — Gunakan database deret waktu untuk menyimpan dan menganalisis data kendaraan Anda. Untuk informasi selengkapnya, lihat [Apa itu Amazon Timestream di Panduan Pengembang Amazon Timestream](#).
- **Amazon S3** — Gunakan layanan penyimpanan objek untuk menyimpan dan mengelola data kendaraan Anda. Untuk informasi selengkapnya, lihat [Apa itu Amazon S3](#) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Mengatur AWS IoT FleetWise

Sebelum Anda menggunakan AWS IoT FleetWise untuk pertama kalinya, selesaikan langkah-langkah di bagian berikut.

Topik

- [Siapkan Akun AWS](#)
- [Memulai di konsol](#)
- [Mengkonfigurasi pengaturan](#)

Siapkan Akun AWS

Selesaikan tugas-tugas berikut untuk mendaftar AWS dan membuat pengguna administratif.

Mendaftar untuk Akun AWS

Jika Anda tidak memiliki Akun AWS, selesaikan langkah-langkah berikut untuk membuatnya.

Untuk mendaftar untuk Akun AWS

1. Buka <https://portal.aws.amazon.com/billing/pendaftaran>.
2. Ikuti petunjuk online.

Bagian dari prosedur pendaftaran melibatkan tindakan menerima panggilan telepon dan memasukkan kode verifikasi di keypad telepon.

Ketika Anda mendaftar untuk Akun AWS, sebuah Pengguna root akun AWS diciptakan. Pengguna root memiliki akses ke semua Layanan AWS dan sumber daya di akun. Sebagai praktik keamanan terbaik, tetapkan akses administratif ke pengguna, dan gunakan hanya pengguna root untuk melakukan [tugas yang memerlukan akses pengguna root](#).

AWS mengirim Anda email konfirmasi setelah proses pendaftaran selesai. Kapan saja, Anda dapat melihat aktivitas akun Anda saat ini dan mengelola akun Anda dengan masuk <https://aws.amazon.com/ke/> dan memilih Akun Saya.

Buat pengguna dengan akses administratif

Setelah Anda mendaftar untuk Akun AWS, amankan Pengguna root akun AWS, aktifkan AWS IAM Identity Center, dan buat pengguna administratif sehingga Anda tidak menggunakan pengguna root untuk tugas sehari-hari.

Amankan Pengguna root akun AWS

1. Masuk ke [AWS Management Console](#) sebagai pemilik akun dengan memilih pengguna Root dan memasukkan Akun AWS alamat email. Di laman berikutnya, masukkan kata sandi.

Untuk bantuan masuk menggunakan pengguna root, lihat [Masuk sebagai pengguna root](#) di AWS Sign-In Panduan Pengguna.

2. Aktifkan otentikasi multi-faktor (MFA) untuk pengguna root Anda.

Untuk petunjuk, lihat [Mengaktifkan MFA perangkat virtual untuk Akun AWS root user \(konsol\)](#) di Panduan IAM Pengguna.

Buat pengguna dengan akses administratif

1. Aktifkan Pusat IAM Identitas.

Untuk petunjuk, lihat [Mengaktifkan AWS IAM Identity Center](#) di AWS IAM Identity Center Panduan Pengguna.

2. Di Pusat IAM Identitas, berikan akses administratif ke pengguna.

Untuk tutorial tentang menggunakan Direktori Pusat Identitas IAM sebagai sumber identitas Anda, lihat [Mengkonfigurasi akses pengguna dengan default Direktori Pusat Identitas IAM](#) di AWS IAM Identity Center Panduan Pengguna.

Masuk sebagai pengguna dengan akses administratif

- Untuk masuk dengan pengguna Pusat IAM Identitas, gunakan login URL yang dikirim ke alamat email saat Anda membuat pengguna Pusat IAM Identitas.

Untuk bantuan masuk menggunakan pengguna Pusat IAM Identitas, lihat [Masuk ke AWS akses portal](#) di AWS Sign-In Panduan Pengguna.

Tetapkan akses ke pengguna tambahan

1. Di Pusat IAM Identitas, buat set izin yang mengikuti praktik terbaik menerapkan izin hak istimewa paling sedikit.

Untuk petunjuknya, lihat [Membuat set izin](#) di AWS IAM Identity Center Panduan Pengguna.

2. Tetapkan pengguna ke grup, lalu tetapkan akses masuk tunggal ke grup.

Untuk petunjuk, lihat [Menambahkan grup](#) di AWS IAM Identity Center Panduan Pengguna.

Note

Anda dapat menggunakan peran terkait layanan dengan AWS IoT FleetWise. Peran terkait layanan telah ditentukan sebelumnya oleh AWS IoT FleetWise dan sertakan izin yang AWS IoT FleetWise perlu mengirim metrik ke Amazon. CloudWatch Untuk informasi selengkapnya, lihat [Menggunakan peran terkait layanan untuk IoT AWS FleetWise](#).

Memulai di konsol

Jika Anda belum masuk ke Akun AWS, masuk, lalu buka [AWS Konsol IoT FleetWise](#) . Untuk memulai dengan AWS IoT FleetWise, buat model kendaraan. Model kendaraan menstandarisasi format kendaraan Anda.

1. Arahkan ke [AWS Konsol IoT FleetWise](#).
2. Di Memulai dengan AWS IoT FleetWise, pilih Memulai.

Untuk informasi selengkapnya tentang membuat model kendaraan, lihat [Buat model kendaraan \(konsol\)](#).

Mengkonfigurasi pengaturan

Anda dapat menggunakan AWS FleetWise Konsol IoT atau API untuk mengonfigurasi pengaturan untuk metrik Amazon CloudWatch Logs, Amazon CloudWatch Logs, dan mengenkripsi data dengan Kunci yang dikelola AWS.

Dengan CloudWatch metrik, Anda dapat memantau AWS IoT FleetWise dan lainnya AWS sumber daya. Anda dapat menggunakan CloudWatch metrik untuk mengumpulkan dan melacak metrik, seperti untuk menentukan apakah ada batas layanan yang terlampaui. Untuk informasi selengkapnya tentang CloudWatch metrik, lihat [Memantau AWS IoT FleetWise dengan Amazon CloudWatch](#).

Dengan CloudWatch Log, AWS IoT FleetWise mengirimkan data log ke grup CloudWatch log, tempat Anda dapat menggunakannya untuk mengidentifikasi dan mengurangi masalah apa pun. Untuk informasi selengkapnya tentang CloudWatch Log, lihat [Konfigurasi AWS pencatatan IoT FleetWise](#).

Dengan enkripsi data, AWS Penggunaan IoT FleetWise Kunci yang dikelola AWS untuk mengenkripsi data. Anda juga dapat memilih untuk membuat dan mengelola kunci dengan AWS KMS. Untuk informasi selengkapnya tentang enkripsi, lihat [Enkripsi data](#).

Konfigurasi pengaturan (konsol)

Jika Anda belum masuk ke Akun AWS, masuk, lalu buka [AWS Konsol IoT FleetWise](#).

1. Arahkan ke [AWS Konsol IoT FleetWise](#).
2. Di panel kiri, pilih Pengaturan.
3. Di Metrik, pilih Aktifkan. AWS IoT FleetWise secara otomatis melampirkan kebijakan CloudWatch terkelola ke peran terkait layanan dan mengaktifkan metrik. CloudWatch
4. Di Logging, pilih Edit.
 - a. Di bagian CloudWatch logging, masukkan grup Log.
 - b. Untuk menyimpan perubahan, pilih Kirim.
5. Di bagian Enkripsi, pilih Edit.
 - a. Pilih jenis kunci yang ingin Anda gunakan. Untuk informasi selengkapnya, lihat [Manajemen kunci](#).
 - i. Gunakan AWS kunci — AWS IoT FleetWise memiliki dan mengelola kuncinya.
 - ii. Pilih yang berbeda AWS Key Management Service kunci — Anda mengelola AWS KMS keys yang ada di akun Anda.
 - b. Untuk menyimpan perubahan, pilih Kirim.

Konfigurasi pengaturan (AWS CLI)

Dalam AWS CLI, daftarkan akun untuk mengkonfigurasi pengaturan.

1. Untuk mengkonfigurasi pengaturan, jalankan perintah berikut.

```
aws iotfleetwise register-account
```

2. Untuk memverifikasi pengaturan Anda, jalankan perintah berikut untuk mengambil status pendaftaran.

Note

Peran terkait layanan hanya digunakan untuk mempublikasikan AWS FleetWise Metrik IoT ke CloudWatch. Untuk informasi selengkapnya, lihat [Menggunakan peran terkait layanan untuk IoT AWS 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/AWSIoT FleetwiseServiceRole"
  },
  "lastModificationTime": "2022-07-28T11:31:22.854000-07:00",
}
```

Status pendaftaran dapat berupa salah satu dari yang berikut:

- **REGISTRATION_SUCCESS**— AWS sumber daya berhasil didaftarkan.

- `REGISTRATION_PENDING` – AWS IoT FleetWise sedang memproses permintaan pendaftaran. Proses ini memakan waktu sekitar lima menit untuk menyelesaikannya.
- `REGISTRATION_FAILURE` – AWS IoT FleetWise tidak dapat mendaftarkan AWS sumber daya. Coba lagi nanti.

Memulai dengan AWS IoT FleetWise

Dengan AWS IoT FleetWise, Anda dapat mengumpulkan, mengubah, dan mentransfer data kendaraan Anda. Gunakan tutorial di bagian ini untuk memulai dengan AWS IoT FleetWise.

Lihat topik berikut untuk mempelajari lebih lanjut tentang AWS IoT FleetWise:

- [Menelan data ke cloud](#)
- [Kendaraan pemodelan](#)
- [Membuat, menyediakan, dan mengelola kendaraan](#)
- [Membuat dan mengelola armada](#)
- [Mengumpulkan dan mentransfer data dengan kampanye](#)

Persyaratan

Anda harus memiliki Akun AWS untuk memulai dengan AWS IoT FleetWise. Jika Anda tidak memilikinya, lihat [Mengatur AWS IoT FleetWise](#).

Gunakan Wilayah tempat AWS IoT FleetWise tersedia. Untuk informasi selengkapnya, lihat [FleetWise titik akhir dan AWS kuota IoT](#). Anda dapat menggunakan pemilih Wilayah di AWS Management Console untuk beralih ke salah satu Wilayah ini.

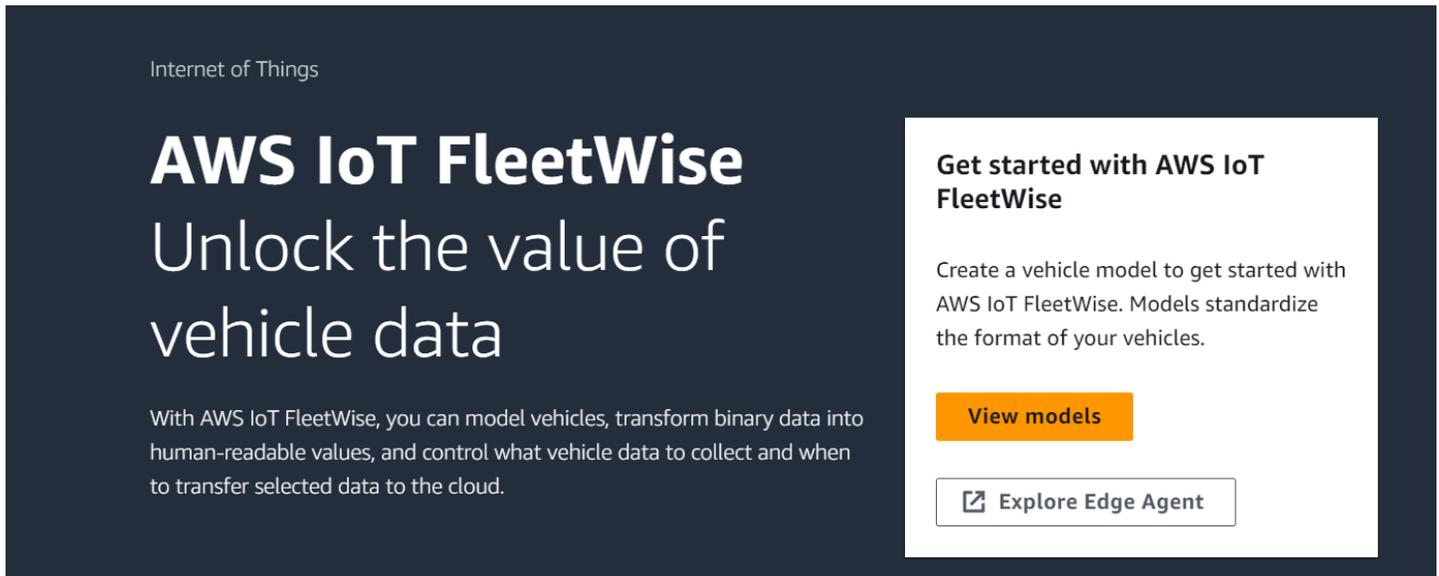
Demo perangkat lunak Edge Agent

Anda dapat menggunakan demo mulai cepat Explore Edge Agent untuk menjelajahi AWS IoT FleetWise dan mempelajari cara mengembangkan perangkat lunak Edge Agent untuk AWS IoT. FleetWise Demo ini menggunakan AWS CloudFormation template. Ini memandu Anda melalui peninjauan implementasi referensi Agen Edge, mengembangkan Agen Edge Anda, dan kemudian menerapkan perangkat lunak Edge Agent Anda di Amazon EC2 Graviton dan menghasilkan data kendaraan sampel. Demo ini juga menyediakan skrip yang dapat Anda gunakan untuk membuat katalog sinyal, model kendaraan, manifes decoder, kendaraan, armada, dan kampanye — semuanya ada di cloud. Untuk informasi lebih lanjut tentang demo mulai cepat, lakukan hal berikut untuk mengunduh Panduan Pengembang perangkat lunak Edge Agent.

Untuk mengunduh demo mulai cepat

1. Arahkan ke konsol [AWS IoT FleetWise](#).

2. Di halaman beranda layanan, di FleetWise bagian Memulai dengan AWS IoT, pilih Explore Edge Agent.



Internet of Things

AWS IoT FleetWise

Unlock the value of vehicle data

With AWS IoT FleetWise, you can model vehicles, transform binary data into human-readable values, and control what vehicle data to collect and when to transfer selected data to the cloud.

Get started with AWS IoT FleetWise

Create a vehicle model to get started with AWS IoT FleetWise. Models standardize the format of your vehicles.

[View models](#)

[Explore Edge Agent](#)

Tutorial: Memulai dengan AWS IoT FleetWise (konsol)

Gunakan AWS IoT FleetWise untuk mengumpulkan, mengubah, dan mentransfer format data unik dari kendaraan otomatis ke cloud dalam waktu dekat. Anda memiliki akses ke wawasan luas armada. Ini dapat membantu Anda mendeteksi dan mengurangi masalah kesehatan kendaraan secara efisien, mentransfer sinyal data bernilai tinggi, dan mendiagnosis masalah dari jarak jauh, sekaligus mengurangi biaya.

Tutorial ini menunjukkan kepada Anda bagaimana memulai dengan AWS IoT FleetWise. Anda akan belajar cara membuat model kendaraan (manifes model), manifes decoder, kendaraan, dan kampanye.

Untuk informasi selengkapnya tentang komponen dan konsep utama AWS IoT FleetWise, lihat [Bagaimana AWS IoT bekerja FleetWise](#)

Perkiraan waktu: Sekitar 45 menit.

Important

Anda akan dikenakan biaya untuk FleetWise sumber daya AWS IoT yang dibuat dan dikonsumsi oleh demo ini. Untuk informasi selengkapnya, lihat [AWS IoT FleetWise](#) di halaman Harga AWS FleetWise IoT.

Topik

- [Prasyarat](#)
- [Langkah 1: Siapkan perangkat lunak Edge Agent untuk AWS IoT FleetWise](#)
- [Langkah 2: Buat model kendaraan](#)
- [Langkah 3: Buat manifes decoder](#)
- [Langkah 4: Konfigurasi manifes decoder](#)
- [Langkah 5: Buat kendaraan](#)
- [Langkah 6: Buat kampanye](#)
- [Langkah 7: Bersihkan](#)
- [Langkah selanjutnya](#)

Prasyarat

Untuk menyelesaikan tutorial memulai ini, pertama-tama Anda perlu yang berikut ini:

- Sebuah Akun AWS. Jika Anda tidak memiliki Akun AWS, lihat [Membuat Akun AWS](#) di Panduan AWS Account Management Referensi.
- Akses ke Wilayah AWS yang mendukung AWS IoT FleetWise. Saat ini, AWS IoT FleetWise didukung di AS Timur (Virginia N.) dan Eropa (Frankfurt).
- Sumber daya Amazon Timestream:
 - Database Amazon Timestream. Untuk informasi selengkapnya, lihat [Membuat database](#) di Panduan Pengembang Amazon Timestream.
 - Tabel Amazon Timestream yang dibuat di Amazon Timestream yang akan menyimpan data Anda. Untuk informasi selengkapnya, lihat [Membuat tabel](#) di Panduan Pengembang Amazon Timestream.

Langkah 1: Siapkan perangkat lunak Edge Agent untuk AWS IoT FleetWise

Note

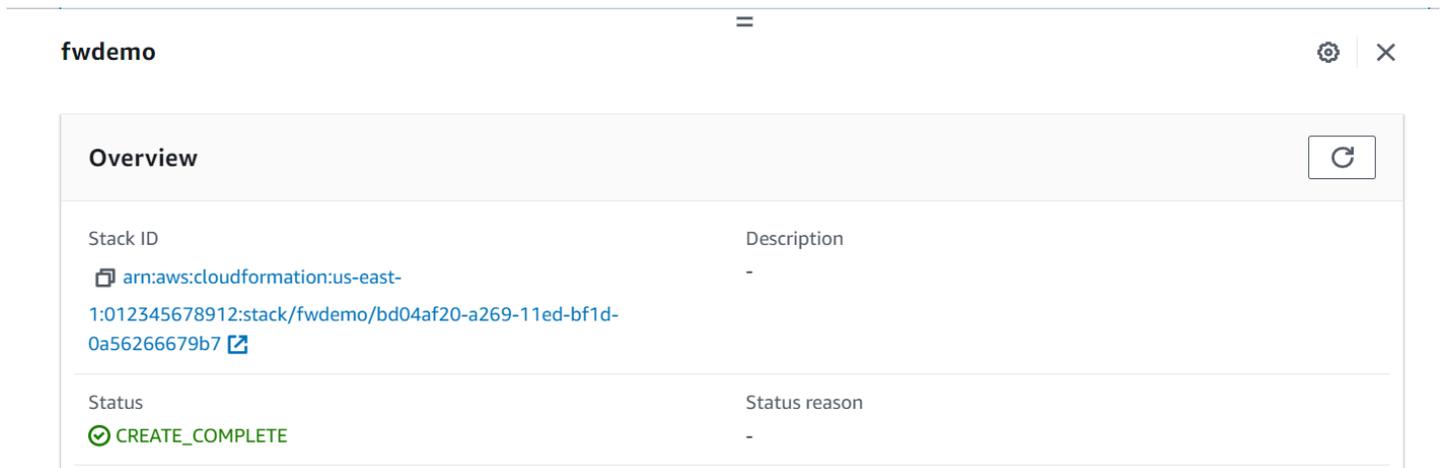
CloudFormation Tumpukan pada langkah ini menggunakan data telemetri. Anda juga dapat membuat CloudFormation tumpukan menggunakan data sistem visi. Untuk informasi selengkapnya, lihat [Panduan Pengembang Data Sistem Visi](#).

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Perangkat lunak Edge Agent Anda untuk AWS IoT FleetWise memfasilitasi komunikasi antara kendaraan dan cloud. Ini menerima instruksi dari skema pengumpulan data tentang cara mengumpulkan data dari kendaraan yang terhubung dengan cloud.

Untuk mengatur perangkat lunak Edge Agent Anda, dalam informasi Umum, lakukan hal berikut:

1. Buka [CloudFormation Template Peluncuran](#).
2. Pada halaman Quick create stack, untuk nama Stack, masukkan nama tumpukan sumber daya AWS IoT FleetWise Anda. Tumpukan adalah nama ramah yang muncul sebagai awalan pada nama sumber daya yang dibuat AWS CloudFormation template ini.
3. Di bawah Parameter, masukkan nilai kustom Anda untuk parameter yang terkait dengan tumpukan Anda.
 - a. Fleetsize - Anda dapat menambah jumlah kendaraan di armada Anda dengan memperbarui parameter Fleetsize.
 - b. IoT CoreRegion - Anda dapat menentukan Wilayah tempat AWS IoT benda itu dibuat dengan memperbarui parameter IoT. CoreRegion Anda harus menggunakan Wilayah yang sama dengan yang Anda gunakan untuk membuat kendaraan AWS IoT FleetWise Anda. Untuk informasi selengkapnya Wilayah AWS, lihat [Wilayah dan Zona - Amazon Elastic Compute Cloud](#).
4. Di bagian Kemampuan, pilih kotak untuk mengetahui bahwa AWS CloudFormation menciptakan sumber daya IAM.
5. Pilih Buat tumpukan, lalu tunggu sekitar 15 menit agar status tumpukan ditampilkan CREATE_COMPLETE.
6. Untuk mengonfirmasi tumpukan telah dibuat, pilih tab Stack info, segarkan tampilan, dan cari CREATE_COMPLETE.



The screenshot shows the AWS IoT FleetWise console interface. At the top, the title 'fwdemo' is displayed. Below it, there is a table with two columns: 'Stack ID' and 'Description'. The 'Stack ID' row contains the value 'arn:aws:cloudformation:us-east-1:012345678912:stack/fwdemo/bd04af20-a269-11ed-bf1d-0a56266679b7' with a copy icon and a link icon. The 'Description' row contains a hyphen '-'. Below this, there is another table with two columns: 'Status' and 'Status reason'. The 'Status' row contains 'CREATE_COMPLETE' with a green checkmark icon. The 'Status reason' row contains a hyphen '-'. There is a refresh icon in the top right corner of the table area.

Important

Anda akan dikenakan biaya untuk FleetWise sumber daya AWS IoT yang dibuat dan dikonsumsi oleh demo ini. Untuk informasi selengkapnya, lihat [AWS IoT FleetWise](#) di halaman Harga AWS FleetWise IoT.

Langkah 2: Buat model kendaraan

Important

Anda tidak dapat membuat model kendaraan dengan sinyal data sistem visi di konsol AWS IoT FleetWise . Sebagai gantinya, gunakan AWS CLI.

Anda menggunakan model kendaraan untuk menstandarisasi format kendaraan Anda, dan untuk membantu menentukan hubungan antara sinyal di kendaraan yang Anda buat. Katalog sinyal juga dibuat saat Anda membuat model kendaraan. Katalog sinyal adalah kumpulan sinyal standar yang dapat digunakan kembali untuk membuat model kendaraan. Sinyal adalah struktur fundamental yang Anda tentukan untuk berisi data kendaraan dan metadatanya. Saat ini, FleetWise layanan AWS IoT hanya mendukung satu katalog sinyal Wilayah AWS per akun. Ini membantu memverifikasi bahwa data yang diproses dari armada kendaraan konsisten.

Untuk membuat model kendaraan

1. Buka konsol AWS IoT FleetWise .
2. Pada panel navigasi, pilih Model kendaraan.

3. Pada halaman Model kendaraan, pilih Buat model kendaraan.
4. Di bagian Informasi umum, masukkan nama model kendaraan Anda, seperti Kendaraan1, dan deskripsi opsional. Lalu pilih Selanjutnya.
5. Pilih satu atau lebih sinyal dari katalog sinyal. Anda dapat memfilter sinyal berdasarkan nama di katalog pencarian, atau memilihnya dari daftar. Misalnya, Anda dapat memilih sinyal untuk tekanan ban dan tekanan rem sehingga Anda dapat mengumpulkan data yang terkait dengan sinyal ini. Pilih Berikutnya.
6. Pilih file.dbc Anda dan unggah dari perangkat lokal Anda. Pilih Berikutnya.

 Note

Untuk tutorial ini, Anda dapat mengunduh [contoh file.dbc](#) untuk diunggah untuk langkah ini.

7. Tambahkan atribut ke model kendaraan Anda dan kemudian pilih Berikutnya.
 - a. Nama - Masukkan nama atribut kendaraan, seperti nama pabrikan atau tanggal pembuatan.
 - b. Tipe Data - Pada menu Tipe data, pilih tipe data.
 - c. Unit - (Opsional) Masukkan nilai satuan, seperti kilometer atau Celcius.
 - d. Jalur - (Opsional) Masukkan nama untuk jalur ke sinyal, seperti `Vehicle.Engine.Light`. Titik (.) menunjukkan bahwa itu adalah sinyal anak.
 - e. Nilai default - (Opsional) Masukkan nilai default.
 - f. Deskripsi - (Opsional) Masukkan deskripsi atribut.
8. Tinjau konfigurasi Anda. Saat Anda siap, pilih Buat. Notifikasi muncul yang mengatakan model kendaraan Anda berhasil dibuat.

✔ **Vehicle model created** ✕
You successfully created the vehicle model: demo.

AWS IoT FleetWise > Vehicle models > Demo

demo

Duplicate Create vehicle Create decoder manifest

When a decoder manifest is associated with a vehicle model, you can create a vehicle. To use the API to create vehicles with this vehicle model, follow the instructions in the AWS IoT FleetWise Developer Guide. After you create vehicles, you can create campaigns for them.

Summary [Info](#)

Vehicle model ARN arn:aws:iotfleetwise:us-east-1:012345678912:model-manifest/demo	Status ✔ ACTIVE	Date created February 01, 2023 at 14:40 (UTC-05)
Signal catalog ARN arn:aws:iotfleetwise:us-east-1:012345678912:signal-catalog/DefaultSignalCatalog	Description -	Last modified February 01, 2023 at 14:40 (UTC-05)

Langkah 3: Buat manifes decoder

Manifestasi decoder dikaitkan dengan model kendaraan yang Anda buat. Mereka berisi informasi yang membantu AWS IoT FleetWise memecahkan kode dan mengubah data kendaraan dari format biner menjadi nilai yang dapat dibaca manusia yang dapat dianalisis. Antarmuka jaringan dan sinyal decoder adalah komponen yang membantu mengkonfigurasi manifes decoder. Antarmuka jaringan berisi informasi tentang protokol CAN atau OBD yang digunakan jaringan kendaraan Anda. Sinyal decoder menyediakan informasi decoding untuk sinyal tertentu.

Untuk membuat manifes decoder

1. Buka konsol AWS IoT FleetWise .
2. Pada panel navigasi, pilih Model kendaraan.
3. Di bagian Model kendaraan, pilih model kendaraan yang ingin Anda gunakan untuk membuat manifes dekoder.
4. Pilih Buat manifes dekoder.

Langkah 4: Konfigurasi manifes decoder

Untuk mengkonfigurasi manifes decoder

Important

Anda tidak dapat mengonfigurasi sinyal data sistem penglihatan dalam manifes decoder menggunakan konsol IoT AWS . FleetWise Sebagai gantinya, gunakan AWS CLI. Untuk informasi selengkapnya, lihat [Buat manifes decoder \(\)AWS CLI](#).

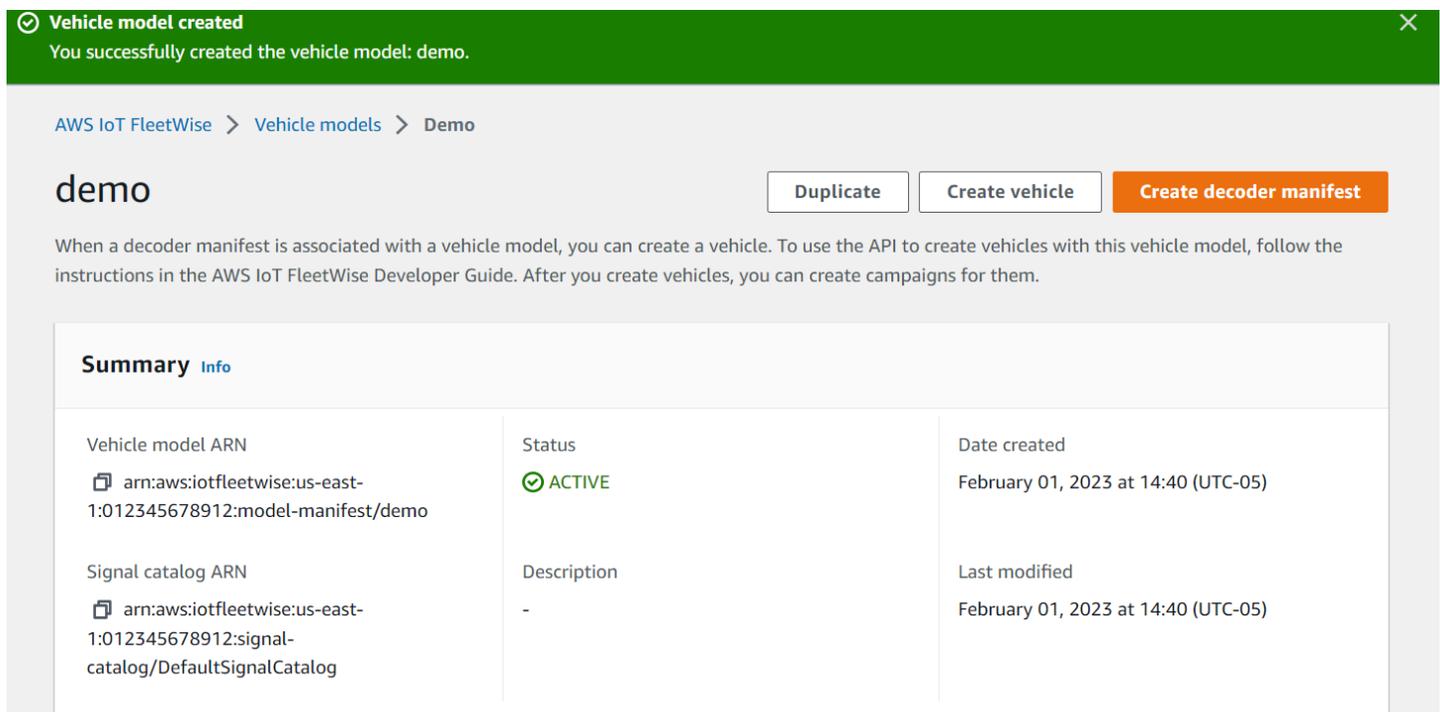
1. Untuk membantu Anda mengidentifikasi manifes decoder Anda, masukkan nama dan deskripsi opsional untuk itu. Lalu, pilih Selanjutnya.
2. Untuk menambahkan satu atau beberapa antarmuka jaringan, pilih jenis CAN_INTERFACE atau OBD_INTERFACE.
 - Antarmuka diagnostik on-board (OBD) - Pilih jenis antarmuka ini jika Anda menginginkan protokol yang mendefinisikan bagaimana data diagnostik mandiri dikomunikasikan antara unit kontrol elektronik (ECU). Protokol ini menyediakan sejumlah kode masalah diagnostik standar (DTC) yang dapat membantu Anda memecahkan masalah dengan kendaraan Anda.
 - Antarmuka Controller Area Network (CAN bus) - Pilih jenis antarmuka ini jika Anda menginginkan protokol yang mendefinisikan bagaimana data dikomunikasikan antara ECU. ECU dapat berupa unit kontrol mesin, airbag, atau sistem audio.
3. Masukkan nama antarmuka jaringan.
4. Untuk menambahkan sinyal ke antarmuka jaringan, pilih satu atau lebih sinyal dari daftar.
5. Pilih sinyal decoder untuk sinyal yang Anda tambahkan pada langkah sebelumnya. Untuk memberikan informasi decoding, unggah file.dbc. Setiap sinyal dalam model kendaraan harus dipasangkan dengan sinyal decoder yang dapat Anda pilih dari daftar.
6. Untuk menambahkan antarmuka jaringan lain, pilih Tambahkan antarmuka jaringan. Setelah selesai menambahkan antarmuka jaringan, pilih Berikutnya.
7. Tinjau konfigurasi Anda dan kemudian pilih Buat. Notifikasi muncul mengatakan manifes decoder Anda berhasil dibuat.

Langkah 5: Buat kendaraan

Dalam AWS IoT FleetWise, kendaraan adalah representasi virtual dari kehidupan nyata, kendaraan fisik Anda. Semua kendaraan yang dibuat dari model kendaraan yang sama mewarisi kelompok sinyal yang sama, dan setiap kendaraan yang Anda buat sesuai dengan hal IoT yang baru dibuat. Anda harus mengaitkan semua kendaraan dengan manifes decoder.

Prasyarat

1. Verifikasi bahwa Anda telah membuat model kendaraan dan manifes decoder. Juga, verifikasi bahwa status model kendaraan AKTIF.
 - a. Untuk memverifikasi bahwa status model kendaraan AKTIF, buka konsol AWS IoT FleetWise .
 - b. Pada panel navigasi, pilih Model kendaraan.
 - c. Di bagian Ringkasan, di bawah Status, periksa status kendaraan Anda.



Vehicle model created
You successfully created the vehicle model: demo.

AWS IoT FleetWise > Vehicle models > Demo

demo

[Duplicate](#) [Create vehicle](#) [Create decoder manifest](#)

When a decoder manifest is associated with a vehicle model, you can create a vehicle. To use the API to create vehicles with this vehicle model, follow the instructions in the AWS IoT FleetWise Developer Guide. After you create vehicles, you can create campaigns for them.

Summary Info		
Vehicle model ARN arn:aws:iotfleetwise:us-east-1:012345678912:model-manifest/demo	Status ACTIVE	Date created February 01, 2023 at 14:40 (UTC-05)
Signal catalog ARN arn:aws:iotfleetwise:us-east-1:012345678912:signal-catalog/DefaultSignalCatalog	Description -	Last modified February 01, 2023 at 14:40 (UTC-05)

Untuk membuat kendaraan

1. Buka FleetWise konsol AWS.
2. Pada panel navigasi, pilih Kendaraan.
3. Pilih Buat kendaraan.

4. Untuk menentukan properti kendaraan, masukkan nama kendaraan, lalu pilih manifes model (model kendaraan) dan manifes decoder.
5. (Opsional) Untuk menentukan atribut kendaraan, masukkan pasangan kunci-nilai lalu pilih Tambahkan atribut.
6. (Opsional) Untuk memberi label pada sumber daya AWS Anda, tambahkan tag, lalu pilih Tambahkan tag baru.
7. Pilih Berikutnya.
8. Untuk mengonfigurasi sertifikat kendaraan, Anda dapat mengunggah sertifikat Anda sendiri atau memilih Buat otomatis sertifikat baru. Sebaiknya buat sertifikat Anda secara otomatis untuk pengaturan yang lebih cepat. Jika Anda sudah memiliki sertifikat, Anda dapat memilih untuk menggunakannya sebagai gantinya.
9. Unduh file kunci publik dan pribadi lalu pilih Berikutnya.
10. Untuk melampirkan kebijakan ke sertifikat kendaraan, Anda dapat memasukkan nama kebijakan yang ada atau membuat kebijakan baru. Untuk membuat kebijakan baru, pilih Buat kebijakan, lalu pilih Berikutnya.
11. Tinjau konfigurasi Anda. Setelah selesai, pilih Buat kendaraan.

Langkah 6: Buat kampanye

Dalam AWS IoT FleetWise, kampanye digunakan untuk memfasilitasi pemilihan, pengumpulan, dan transfer data dari kendaraan ke cloud. Kampanye berisi skema pengumpulan data yang memberikan instruksi perangkat lunak Agen Edge tentang cara mengumpulkan data dengan skema pengumpulan berbasis kondisi atau skema pengumpulan berbasis waktu.

Untuk membuat kampanye

1. Buka konsol AWS IoT FleetWise .
2. Pada panel navigasi, pilih Kampanye.
3. Pilih Buat kampanye.
4. Masukkan nama kampanye Anda dan deskripsi opsional.
5. Untuk mengonfigurasi skema pengumpulan data kampanye, Anda dapat menentukan skema pengumpulan data secara manual atau mengunggah file.json dari perangkat lokal Anda. Mengunggah file.json secara otomatis menentukan skema pengumpulan data.

- a. Untuk menentukan skema pengumpulan data secara manual, pilih Tentukan Skema Pengumpulan Data dan pilih jenis skema pengumpulan data yang ingin Anda gunakan untuk kampanye Anda. Anda dapat memilih skema pengumpulan berbasis kondisi atau skema pengumpulan berbasis waktu.
 - b. Jika Anda memilih skema pengumpulan berbasis waktu, Anda harus menentukan durasi waktu kampanye Anda akan mengumpulkan data kendaraan.
 - c. Jika Anda memilih skema pengumpulan berbasis kondisi, Anda harus menentukan ekspresi untuk mengenali data apa yang akan dikumpulkan. Pastikan untuk menentukan nama sinyal sebagai variabel, operator perbandingan, dan nilai perbandingan.
 - d. (Opsional) Pilih versi bahasa ekspresi Anda, atau simpan sebagai nilai default 1.
 - e. (Opsional) Tentukan interval pemicu antara dua peristiwa pengumpulan data.
 - f. Untuk mengumpulkan data, pilih kondisi mode Trigger untuk perangkat lunak Edge Agent. Secara default, Edge Agent untuk FleetWise perangkat lunak AWS IoT Selalu mengumpulkan data setiap kali kondisi terpenuhi. Atau, dapat mengumpulkan data hanya ketika kondisi terpenuhi untuk pertama kalinya, Pada pemicu pertama.
 - g. (Opsional) Anda dapat memilih opsi skema yang lebih maju.
6. Untuk menentukan sinyal dari mana skema pengumpulan data akan mengumpulkan data, cari nama sinyal dari menu.
 7. (Opsional) Anda dapat memilih jumlah sampel maksimum atau interval pengambilan sampel minimum. Anda juga dapat menambahkan lebih banyak sinyal.
 8. Pilih Berikutnya.
 9. Tentukan tujuan penyimpanan yang Anda inginkan kampanye untuk mentransfer data. Anda dapat menyimpan data di Amazon S3 atau Amazon Timestream.
 - a. Amazon S3 — Pilih bucket S3 yang AWS IoT FleetWise memiliki izin.
 - b. Amazon Timestream — pilih database Timestream dan nama tabel. Masukkan peran IAM yang memungkinkan AWS IoT FleetWise untuk mengirim data ke Timestream.
 10. Pilih Berikutnya.
 11. Pilih atribut kendaraan atau nama kendaraan dari kotak pencarian.
 12. Masukkan nilai yang terkait dengan atribut atau nama yang Anda pilih untuk kendaraan Anda.
 13. Pilih kendaraan tempat kampanye Anda akan mengumpulkan datanya. Lalu, pilih Selanjutnya.
 14. Tinjau konfigurasi kampanye Anda, lalu pilih Buat kampanye. Anda atau tim Anda harus menyebarkan kampanye ke kendaraan.

Langkah 7: Bersihkan

Untuk menghindari biaya lebih lanjut untuk sumber daya yang Anda gunakan selama tutorial ini, hapus AWS CloudFormation tumpukan dan semua sumber daya tumpukan.

Untuk menghapus AWS CloudFormation tumpukan

1. Buka [konsol AWS CloudFormation](#).
2. Dari daftar Stacks, pilih tumpukan yang Anda buat di langkah 1.
3. Pilih Hapus.
4. Untuk mengonfirmasi penghapusan, pilih Hapus. Tumpukan membutuhkan waktu sekitar 15 menit untuk dihapus.

Langkah selanjutnya

1. Anda dapat memproses dan memvisualisasikan data kendaraan yang dikumpulkan kampanye Anda. Untuk informasi selengkapnya, lihat [Memproses dan memvisualisasikan data kendaraan](#).
2. Anda dapat memecahkan masalah dan menyelesaikan masalah dengan IoT AWS . FleetWise Untuk informasi selengkapnya, lihat [Pemecahan Masalah AWS IoT FleetWise](#).

Menelan data ke cloud

Edge Agent untuk FleetWise perangkat lunak AWS IoT, ketika diinstal dan dijalankan di kendaraan, dirancang untuk memfasilitasi komunikasi yang aman antara kendaraan Anda dan cloud.

Note

- AWSIoT tidak FleetWise dimaksudkan untuk digunakan dalam, atau terkait dengan, pengoperasian lingkungan berbahaya atau sistem kritis apa pun yang dapat menyebabkan cedera tubuh yang serius atau kematian atau menyebabkan kerusakan lingkungan atau properti. Data kendaraan yang dikumpulkan melalui penggunaan AWS IoT FleetWise oleh Anda hanya untuk tujuan informasi, dan Anda tidak boleh menggunakan AWS IoT FleetWise untuk mengontrol atau mengoperasikan fungsi kendaraan.
- Data kendaraan yang dikumpulkan melalui penggunaan AWS IoT oleh Anda FleetWise harus dievaluasi keakuratannya yang sesuai untuk kasus penggunaan Anda, termasuk untuk tujuan memenuhi kewajiban kepatuhan apa pun yang mungkin Anda miliki berdasarkan peraturan keselamatan kendaraan yang berlaku (seperti pemantauan keselamatan dan kewajiban pelaporan). Evaluasi tersebut harus mencakup pengumpulan dan peninjauan informasi melalui sarana dan sumber standar industri lainnya (seperti laporan dari pengemudi kendaraan).

Untuk menelan data ke cloud, lakukan hal berikut:

1. Kembangkan dan instal Edge Agent Anda untuk FleetWise perangkat lunak AWS IoT di kendaraan Anda. Untuk informasi lebih lanjut tentang cara bekerja dengan perangkat lunak Edge Agent, lakukan hal berikut untuk mengunduh [Edge Agent for AWS IoT FleetWise Software Developer Guide](#).
 1. Arahkan ke konsol [AWSIoT FleetWise](#).
 2. Di halaman beranda layanan, di FleetWise bagian Memulai dengan AWS IoT, pilih Explore Edge Agent.
2. Buat atau impor katalog sinyal yang berisi sinyal yang akan Anda gunakan untuk membuat model kendaraan. Untuk informasi selengkapnya, silakan lihat [Buat katalog sinyal \(AWS CLI\)](#) dan [Impor katalog sinyal \(AWS CLI\)](#).

Note

- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan pertama, Anda tidak perlu membuat katalog sinyal secara manual. Saat Anda membuat model kendaraan pertama Anda, AWS IoT FleetWise secara otomatis membuat katalog sinyal untuk Anda. Untuk informasi selengkapnya, lihat [Buat model kendaraan \(konsol\)](#).
- AWS IoT FleetWise saat ini mendukung katalog sinyal untuk setiap AWS akun per akun Wilayah AWS

3. Gunakan sinyal dalam katalog sinyal untuk membuat model kendaraan. Untuk informasi selengkapnya, lihat [Buat model kendaraan](#).

Note

- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan, Anda dapat mengunggah file.dbc untuk mengimpor sinyal. .dbc adalah format file yang didukung oleh database Controller Area Network (CAN bus). Setelah model kendaraan dibuat, sinyal baru secara otomatis ditambahkan ke katalog sinyal. Untuk informasi selengkapnya, lihat [Buat model kendaraan \(konsol\)](#).
- Jika Anda menggunakan operasi `CreateModelManifest` API untuk membuat model kendaraan, Anda harus menggunakan operasi `UpdateModelManifest` API untuk mengaktifkan model kendaraan. Untuk informasi selengkapnya, lihat [Perbarui model kendaraan \(AWS CLI\)](#).
- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan, AWS IoT FleetWise secara otomatis mengaktifkan model kendaraan untuk Anda.

4. Buat manifes decoder. Manifes decoder berisi informasi decoding untuk setiap sinyal yang ditentukan dalam model kendaraan yang Anda buat pada langkah sebelumnya. Manifes dekoder dikaitkan dengan model kendaraan yang Anda buat. Untuk informasi selengkapnya, lihat [Buat dan kelola manifes decoder](#).

Note

- Jika Anda menggunakan operasi `CreateDecoderManifest` API untuk membuat manifes dekoder, Anda harus menggunakan operasi `UpdateDecoderManifest` API

untuk mengaktifkan manifes dekoder. Untuk informasi selengkapnya, lihat [Perbarui manifes decoder \(AWS CLI\)](#).

- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat manifes dekoder, AWS IoT FleetWise secara otomatis mengaktifkan manifes dekoder untuk Anda.

5. Buat kendaraan dari model kendaraan. Kendaraan yang dibuat dari model kendaraan yang sama mewarisi kelompok sinyal yang sama. Anda harus menggunakan AWS IoT Core untuk menyediakan kendaraan Anda sebelum Anda dapat menelan data ke cloud. Untuk informasi selengkapnya, lihat [Membuat, menyediakan, dan mengelola kendaraan](#).
6. (Opsional) Buat armada untuk mewakili sekelompok kendaraan, dan kemudian kaitkan kendaraan individu dengan armada. Ini membantu Anda mengelola beberapa kendaraan secara bersamaan. Untuk informasi selengkapnya, lihat [Membuat dan mengelola armada](#).
7. Buat kampanye. Kampanye dikerahkan ke kendaraan atau armada kendaraan. Kampanye memberikan instruksi perangkat lunak Edge Agent tentang cara memilih, mengumpulkan, dan mentransfer data ke cloud. Untuk informasi selengkapnya, lihat [Mengumpulkan dan mentransfer data dengan kampanye](#).

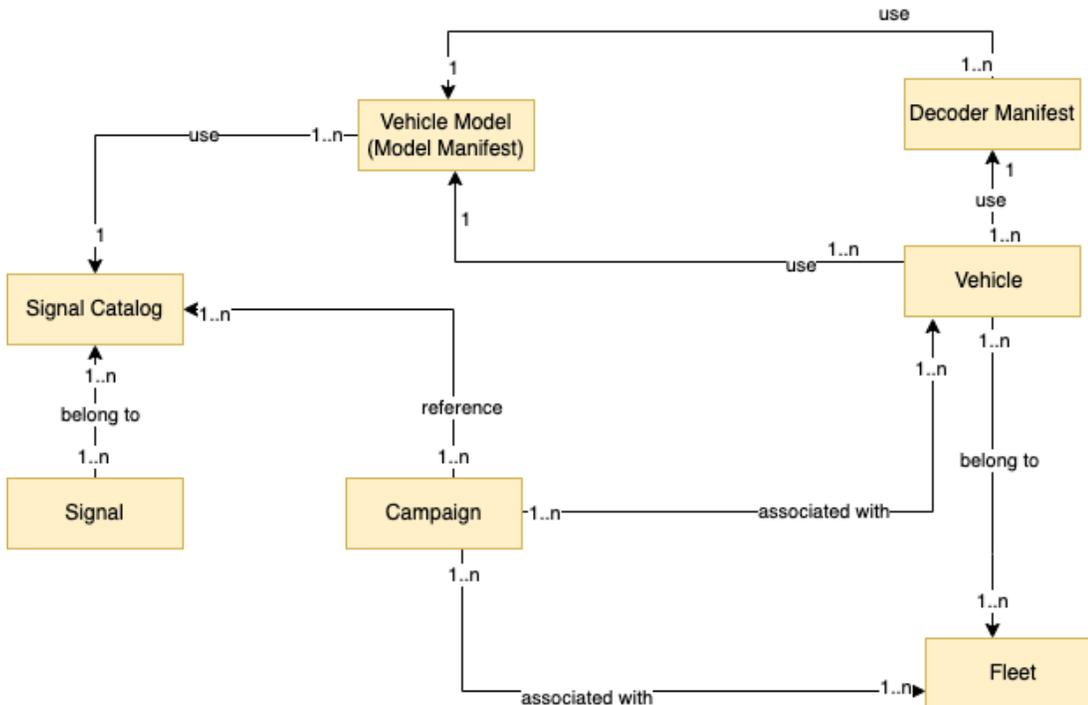
Note

Anda harus menggunakan operasi UpdateCampaign API untuk menyetujui kampanye sebelum AWS FleetWise IoT dapat menerapkannya ke kendaraan atau armada. Untuk informasi selengkapnya, lihat [Memperbarui kampanye \(AWS CLI\)](#).

Perangkat lunak Edge Agent mentransfer data kendaraan untuk AWS IoT Core menggunakan topik yang dicadangkan `$aws/iotfleetwise/vehicles/vehicleName/signals`, yang mengirimkan data ke AWS IoT FleetWise. AWS IoT FleetWise kemudian mengirimkan data ke tabel Timestream atau bucket Amazon S3. Anda dapat menggunakan Timestream untuk menanyakan data Anda, dan menggunakan Amazon QuickSight atau Grafana untuk memvisualisasikan data Anda. Untuk informasi selengkapnya, lihat [Memproses dan memvisualisasikan data kendaraan](#).

Kendaraan pemodelan

AWS IoT FleetWise menyediakan kerangka pemodelan kendaraan yang dapat Anda gunakan untuk membangun representasi virtual kendaraan Anda di cloud. Sinyal, katalog sinyal, model kendaraan, dan manifes decoder adalah komponen inti yang Anda gunakan untuk memodelkan kendaraan Anda.



Sinyal

Sinyal adalah struktur fundamental yang Anda tentukan untuk berisi data kendaraan dan metadatanya. Sinyal dapat berupa atribut, cabang, sensor, atau aktuator. Misalnya, Anda dapat membuat sensor untuk menerima nilai suhu di dalam kendaraan, dan menyimpan metadatanya, termasuk nama sensor, tipe data, dan unit. Untuk informasi selengkapnya, lihat [Membuat dan mengelola katalog sinyal](#).

Katalog sinyal

Katalog sinyal berisi kumpulan sinyal. Sinyal dalam katalog sinyal dapat digunakan untuk memodelkan kendaraan yang menggunakan protokol dan format data yang berbeda. Misalnya, ada dua mobil yang dibuat oleh pembuat mobil yang berbeda: satu menggunakan protokol Control Area Network (CAN bus); yang lain menggunakan protokol On-board Diagnostics (OBD). Anda dapat menentukan sensor dalam katalog sinyal untuk menerima nilai suhu di dalam kendaraan. Sensor ini dapat digunakan untuk mewakili termokopel di kedua mobil. Untuk informasi selengkapnya, lihat [Membuat dan mengelola katalog sinyal](#).

Model kendaraan (manifes model)

Model kendaraan adalah struktur deklaratif yang dapat Anda gunakan untuk membakukan format kendaraan Anda dan untuk menentukan hubungan antara sinyal di kendaraan. Model kendaraan menegakkan informasi yang konsisten di beberapa kendaraan dari jenis yang sama. Anda menambahkan sinyal untuk membuat model kendaraan. Untuk informasi selengkapnya, lihat [Membuat dan mengelola model kendaraan](#).

Manifes dekoder

Manifestasi decoder berisi informasi decoding untuk setiap sinyal dalam model kendaraan. Sensor dan aktuator dalam kendaraan mengirimkan pesan tingkat rendah (data biner). Dengan manifes decoder, AWS FleetWise IoT mampu mengubah data biner menjadi nilai yang dapat dibaca manusia. Setiap manifes decoder dikaitkan dengan model kendaraan. Untuk informasi selengkapnya, lihat [Buat dan kelola manifes decoder](#).

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk memodelkan kendaraan dengan cara berikut.

1. Buat atau impor katalog sinyal yang berisi sinyal yang akan Anda gunakan untuk membuat model kendaraan. Lihat informasi yang lebih lengkap di [Buat katalog sinyal \(AWS CLI\)](#) dan [Impor katalog sinyal \(AWS CLI\)](#).

Note

- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan pertama, Anda tidak perlu membuat katalog sinyal secara manual. Saat Anda membuat model kendaraan pertama Anda, AWS IoT FleetWise secara otomatis membuat katalog sinyal untuk Anda. Untuk informasi selengkapnya, lihat [Buat model kendaraan \(konsol\)](#).
- AWS IoT FleetWise saat ini mendukung katalog sinyal untuk setiap AWS akun per akun. Wilayah AWS

2. Gunakan sinyal dalam katalog sinyal untuk membuat model kendaraan. Untuk informasi selengkapnya, lihat [Buat model kendaraan](#).

Note

- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan, Anda dapat mengunggah file.dbc untuk mengimpor sinyal. .dbc adalah format file yang

didukung oleh database Controller Area Network (CAN bus). Setelah model kendaraan dibuat, sinyal baru secara otomatis ditambahkan ke katalog sinyal. Untuk informasi selengkapnya, lihat [Buat model kendaraan \(konsol\)](#).

- Jika Anda menggunakan operasi `CreateModelManifest` API untuk membuat model kendaraan, Anda harus menggunakan operasi `UpdateModelManifest` API untuk mengaktifkan model kendaraan. Untuk informasi selengkapnya, lihat [Perbarui model kendaraan \(AWS CLI\)](#).
- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan, AWS IoT FleetWise secara otomatis mengaktifkan model kendaraan untuk Anda.

3. Buat manifes decoder. Manifes decoder berisi informasi decoding untuk setiap sinyal yang ditentukan dalam model kendaraan yang Anda buat pada langkah sebelumnya. Manifes dekoder dikaitkan dengan model kendaraan yang Anda buat. Untuk informasi selengkapnya, lihat [Buat dan kelola manifes decoder](#).

Note

- Jika Anda menggunakan operasi `CreateDecoderManifest` API untuk membuat manifes dekoder, Anda harus menggunakan operasi `UpdateDecoderManifest` API untuk mengaktifkan manifes dekoder. Untuk informasi selengkapnya, lihat [Perbarui manifes decoder \(AWS CLI\)](#).
- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat manifes dekoder, AWS IoT FleetWise secara otomatis mengaktifkan manifes dekoder untuk Anda.

Database bus CAN mendukung format file.dbc. Anda dapat mengunggah file.dbc untuk mengimpor sinyal dan sinyal decoder. Untuk mendapatkan contoh file.dbc, lakukan hal berikut.

Untuk mendapatkan file.dbc

1. Unduh [EngineSignals.zip](#).
2. Arahkan ke direktori tempat Anda mengunduh `EngineSignals.zip` file.
3. Buka zip file dan simpan secara lokal sebagai file `EngineSignals.dbc`

Topik

- [Membuat dan mengelola katalog sinyal](#)

- [Membuat dan mengelola model kendaraan](#)
- [Buat dan kelola manifes decoder](#)

Membuat dan mengelola katalog sinyal

Note

Anda dapat mengunduh [skrip demo](#) untuk mengonversi pesan ROS 2 ke file VSS JSON yang kompatibel dengan katalog sinyal. Untuk informasi selengkapnya, lihat [Panduan Pengembang Data Sistem Visi](#).

Katalog sinyal adalah kumpulan sinyal standar yang dapat digunakan kembali untuk membuat model kendaraan. AWS IoT FleetWise mendukung [Spesifikasi Sinyal Kendaraan \(VSS\)](#) yang dapat Anda ikuti untuk menentukan sinyal. Sinyal dapat berupa salah satu dari jenis berikut.

Atribut

Atribut mewakili informasi statis yang umumnya tidak berubah, seperti tanggal pabrikan dan pembuatan.

Cabang

Cabang mewakili sinyal dalam struktur bersarang. Cabang menunjukkan hierarki sinyal. Misalnya, `Vehicle` cabang memiliki cabang anak, `Powertrain`. `Powertrain` cabang memiliki cabang anak, `combustionEngine`. Untuk menemukan `combustionEngine` cabang, gunakan `Vehicle.Powertrain.combustionEngine` ekspresi.

Sensor

Data sensor melaporkan keadaan kendaraan saat ini dan berubah seiring waktu, karena keadaan kendaraan berubah, seperti level cairan, suhu, getaran, atau tegangan.

Aktuator

Data aktuator melaporkan keadaan perangkat kendaraan, seperti motor, pemanas, dan kunci pintu. Mengubah keadaan perangkat kendaraan dapat memperbarui data aktuator. Misalnya, Anda dapat menentukan aktuator untuk mewakili pemanas. Aktuator menerima data baru saat Anda menghidupkan atau mematikan pemanas.

Struktur kustom

Struktur kustom (juga dikenal sebagai struct) mewakili struktur data yang kompleks atau tingkat tinggi. Ini memfasilitasi pengikatan logis atau pengelompokan data yang berasal dari sumber yang sama. Struct digunakan ketika data dibaca atau ditulis dalam operasi atom, seperti untuk mewakili tipe data yang kompleks atau bentuk tingkat tinggi.

Sinyal tipe struct didefinisikan dalam katalog sinyal menggunakan referensi ke tipe data struct alih-alih tipe data primitif. Structs dapat digunakan untuk semua jenis sinyal termasuk sensor, atribut, aktuator, dan tipe data sistem visi. Jika sinyal tipe struct dikirim atau diterima, AWS FleetWise IoT mengharapkan semua item yang disertakan memiliki nilai yang valid, jadi semua item wajib. Misalnya, jika struct berisi item `Vehicle.camera.Image.Height`, `Vehicle.Camera.Image.Width`, dan `Vehicle.Camera.Image.Data` — diharapkan sinyal yang dikirim berisi nilai untuk semua item ini.

Note

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Properti kustom

Properti kustom mewakili anggota struktur data yang kompleks. Tipe data properti dapat berupa primitif atau struct lain.

Saat merepresentasikan bentuk tingkat tinggi menggunakan struct dan properti kustom, bentuk tingkat tinggi yang dimaksudkan selalu didefinisikan dan dilihat sebagai struktur pohon. Properti kustom digunakan untuk mendefinisikan semua node daun sementara struct digunakan untuk mendefinisikan semua node non-daun.

Note

- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan pertama, Anda tidak perlu membuat katalog sinyal secara manual. Saat Anda membuat model kendaraan pertama Anda, AWS IoT FleetWise secara otomatis membuat katalog sinyal untuk Anda. Untuk informasi selengkapnya, lihat [Buat model kendaraan \(konsol\)](#).
- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan, Anda dapat mengunggah file.dbc untuk mengimpor sinyal. .dbc adalah format file yang

didukung oleh database Controller Area Network (CAN bus). Setelah model kendaraan dibuat, sinyal baru secara otomatis ditambahkan ke katalog sinyal. Untuk informasi selengkapnya, lihat [Buat model kendaraan \(konsol\)](#).

- AWS IoT FleetWise saat ini mendukung katalog sinyal untuk masing-masing Akun AWS per Wilayah.

AWS IoT FleetWise menyediakan operasi API berikut yang dapat Anda gunakan untuk membuat dan mengelola katalog sinyal.

- [CreateSignalCatalog](#)— Membuat katalog sinyal baru.
- [ImportSignalCatalog](#)— Mengimpor sinyal untuk membuat katalog sinyal dengan mengunggah file JSON. Sinyal harus ditentukan dengan mengikuti VSS dan disimpan dalam format JSON.
- [UpdateSignalCatalog](#)— Memperbarui katalog sinyal yang ada dengan memperbarui, menghapus, atau menambahkan sinyal.
- [DeleteSignalCatalog](#)— Menghapus katalog sinyal yang ada.
- [ListSignalCatalogs](#)— Mengambil daftar ringkasan paginasi dari semua katalog sinyal.
- [ListSignalCatalogNodes](#)— Mengambil daftar paginasi ringkasan semua sinyal (node) dalam katalog sinyal yang diberikan.
- [GetSignalCatalog](#)— Mengambil informasi tentang katalog sinyal.

Tutorial

- [Konfigurasi sinyal](#)
- [Buat katalog sinyal \(AWS CLI\)](#)
- [Impor katalog sinyal](#)
- [Perbarui katalog sinyal \(AWS CLI\)](#)
- [Hapus katalog sinyal \(AWS CLI\)](#)
- [Dapatkan informasi katalog sinyal \(AWS CLI\)](#)

Konfigurasi sinyal

Bagian ini menunjukkan cara mengonfigurasi cabang, atribut, sensor, dan aktuator.

Topik

- [Konfigurasi cabang](#)
- [Konfigurasi atribut](#)
- [Konfigurasi sensor atau aktuator](#)
- [Konfigurasi tipe data yang kompleks](#)

Konfigurasi cabang

Untuk mengkonfigurasi cabang, tentukan informasi berikut.

- `fullyQualifiedName`— Nama cabang yang sepenuhnya memenuhi syarat adalah jalur ke cabang ditambah nama cabang. Gunakan titik (.) untuk merujuk ke cabang anak. Misalnya, `Vehicle.Chassis.SteeringWheel` adalah nama yang sepenuhnya memenuhi syarat untuk `SteeringWheel` cabang. `Vehicle.Chassis.` adalah jalan menuju cabang ini.

Nama yang sepenuhnya memenuhi syarat dapat memiliki hingga 150 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`, titik dua (:), dan garis bawah (_).

- (Opsional) `Description` — Deskripsi untuk cabang.

Deskripsi dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`,: (titik dua), _ (garis bawah), dan - (tanda hubung).

- (Opsional) `deprecationMessage` - Pesan penghentian untuk node atau cabang yang dipindahkan atau dihapus.

`DeprecationMessage` dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`,: (titik dua), _ (garis bawah), dan - (tanda hubung).

- (Opsional) `comment` — Komentar selain deskripsi. Komentar dapat digunakan untuk memberikan informasi tambahan tentang cabang, seperti alasan cabang atau referensi ke cabang terkait.

Komentar dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`,: (titik dua), _ (garis bawah), dan - (tanda hubung).

Konfigurasi atribut

Untuk mengkonfigurasi atribut, tentukan informasi berikut.

- `dataType`— Tipe data atribut harus salah satu dari yang berikut: `INT8`, `UINT8`, `INT16`, `UINT16`, `INT32`, `UINT32`, `INT64`, `UINT64`, `BOOLEAN`, `FLOAT`, `DOUBLE`, `STRING`, `UNIX_TIMESTAMP`,

INT8_ARRAY, UINT8_ARRAY, INT16_ARRAY, INT32_ARRAY, INT32_ARRAY, INT32_ARRAY, INT32_ARRAY, INT32_ARRAY, INT32_ARRAY, INT32_ARRAY, INT32_ARRAY, INT32_ARRAY, INT32_ARRAY ARRAY, UINT64_ARRAY, BOOLEAN_ARRAY, FLOAT_ARRAY, DOUBLE_ARRAY, STRING_ARRAY, UNIX_TIMESTAMP_ARRAY, UNKNOWN,, atau struct khusus yang ditentukan dalam cabang tipe data. `fullyQualifiedName`

- `fullyQualifiedName`— Nama atribut yang sepenuhnya memenuhi syarat adalah jalur ke atribut ditambah nama atribut. Gunakan titik (.) untuk merujuk ke sinyal anak. Misalnya, `Vehicle.Chassis.SteeringWheel.Diameter` adalah nama yang sepenuhnya memenuhi syarat untuk `Diameter` atribut tersebut. `Vehicle.Chassis.SteeringWheel.` adalah jalan menuju atribut ini.

Nama yang sepenuhnya memenuhi syarat dapat memiliki hingga 150 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`,: (titik dua), dan `_` (garis bawah).

- (Opsional) `Description` — Deskripsi untuk atribut.

Deskripsi dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`,: (titik dua), `_` (garis bawah), dan `-` (tanda hubung).

- (Opsional) `unit` — Unit ilmiah untuk atribut, seperti km atau Celcius.
- (Opsional) `min` - Nilai minimum atribut.
- (Opsional) `max` — Nilai maksimum atribut.
- (Opsional) `defaultValue` - Nilai default atribut.
- (Opsional) `assignedValue` - Nilai yang ditetapkan untuk atribut.
- (Opsional) `allowedValues` - Daftar nilai yang diterima atribut.
- (Opsional) `deprecationMessage` - Pesan penghentian untuk node atau cabang yang sedang dipindahkan atau dihapus.

`DeprecationMessage` dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`,: (titik dua), `_` (garis bawah), dan `-` (tanda hubung).

- (Opsional) `comment` — Komentar selain deskripsi. Komentar dapat digunakan untuk memberikan informasi tambahan tentang atribut, seperti alasan untuk atribut atau referensi ke atribut terkait.

Komentar dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`,: (titik dua), `_` (garis bawah), dan `-` (tanda hubung).

Konfigurasi sensor atau aktuator

Untuk mengkonfigurasi sensor atau aktuator, tentukan informasi berikut.

- `dataType`— Tipe data sinyal harus salah satu dari yang berikut: INT8, UINT8, INT16, UINT16, INT32, UINT32, INT64, UINT64, BOOLEAN, FLOAT, DOUBLE, STRING, UNIX_TIMESTAMP, INT8_ARRAY, UINT8_ARRAY, INT16_ARRAY, INT32_ARRAY, INT32_ARRAY ARRAY, UINT64_ARRAY, BOOLEAN_ARRAY, FLOAT_ARRAY, DOUBLE_ARRAY, STRING_ARRAY, UNIX_TIMESTAMP_ARRAY, UNKNOWN,, atau struct khusus yang ditentukan dalam cabang tipe data. `fullyQualifiedName`
- `fullyQualifiedName`— Nama sinyal yang sepenuhnya memenuhi syarat adalah jalur ke sinyal ditambah nama sinyal. Gunakan titik (.) untuk merujuk ke sinyal anak. Misalnya, `Vehicle.Chassis.SteeringWheel.HandsOff.HandsOffSteeringState` adalah nama yang sepenuhnya memenuhi syarat untuk `HandsOffSteeringState` aktuator. `Vehicle.Chassis.SteeringWheel.HandsOff.` adalah jalan menuju aktuator ini.

Nama yang sepenuhnya memenuhi syarat dapat memiliki hingga 150 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`, `:` (titik dua), dan `_` (garis bawah).

- (Opsional) `Description` — Deskripsi untuk sinyal.

Deskripsi dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`, `:` (titik dua), `_` (garis bawah), dan `-` (tanda hubung).

- (Opsional) `unit` — Unit ilmiah untuk sinyal, seperti km atau celsius.
- (Opsional) `min` — Nilai minimum sinyal.
- (Opsional) `max` — Nilai maksimum sinyal.
- (Opsional) `assignedValue` — Nilai yang ditetapkan untuk sinyal.
- (Opsional) `allowedValues` — daftar nilai yang diterima sinyal.
- (Opsional) `deprecationMessage` - Pesan penghentian untuk node atau cabang yang sedang dipindahkan atau dihapus.

`DeprecationMessage` dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`, `:` (titik dua), `_` (garis bawah), dan `-` (tanda hubung).

- (Opsional) `comment` — Komentar selain deskripsi. Komentar dapat digunakan untuk memberikan informasi tambahan tentang sensor atau aktuator, seperti alasan atau referensi mereka ke sensor atau aktuator terkait.

Komentar dapat memiliki hingga 2048 karakter. Karakter yang valid: a—z, A-Z, 0-9, : (titik dua), _ (garis bawah), dan - (tanda hubung).

Konfigurasi tipe data yang kompleks

Tipe data yang kompleks digunakan saat memodelkan sistem visi. Selain cabang, tipe data ini terdiri dari struktur (juga dikenal sebagai struct) dan properti. Struct adalah sinyal yang dijelaskan oleh beberapa nilai, seperti gambar. Properti mewakili anggota struct, seperti tipe data primitif (seperti UINT8) atau struct lain (seperti stempel waktu). Misalnya, `Vehicle.Cameras.Front` mewakili cabang, `Vehicle.Cameras.Front.Image` mewakili struct, dan `Vehicle.Cameras.Timestamp` mewakili properti.

Contoh tipe data kompleks berikut menunjukkan bagaimana sinyal dan tipe data diekspor ke satu file JSON.

Example tipe data yang kompleks

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

```

        "type": "branch"
      }
    }
  }
}

```

Note

Anda dapat mengunduh [skrip demo](#) untuk mengonversi pesan ROS 2 ke file VSS JSON yang kompatibel dengan katalog sinyal. Untuk informasi selengkapnya, lihat [Panduan Pengembang Data Sistem Visi](#).

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Konfigurasi struct

Untuk mengkonfigurasi struktur kustom (atau struct), tentukan informasi berikut.

- `fullyQualifiedName`- Nama yang sepenuhnya memenuhi syarat dari struktur kustom. Misalnya, nama yang sepenuhnya memenuhi syarat dari struktur kustom mungkin `ComplexDataTypes.VehicleDataTypes.SVMCamera`.

Nama yang sepenuhnya memenuhi syarat dapat memiliki hingga 150 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`, `:` (titik dua), dan `_` (garis bawah).

- (Opsional) `Description` — Deskripsi untuk sinyal.

Deskripsi dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`, `:` (titik dua), `_` (garis bawah), dan `-` (tanda hubung).

- (Opsional) `deprecationMessage` - Pesan penghentian untuk node atau cabang yang sedang dipindahkan atau dihapus.

`DeprecationMessage` dapat memiliki hingga 2048 karakter. Karakter yang valid: `a—z`, `A-Z`, `0-9`, `:` (titik dua), `_` (garis bawah), dan `-` (tanda hubung).

- (Opsional) `comment` — Komentar selain deskripsi. Komentar dapat digunakan untuk memberikan informasi tambahan tentang sensor atau aktuator, seperti alasan atau referensi mereka ke sensor atau aktuator terkait.

- (Opsional) `structFullyQualifiedName` — Nama node structure (struct) yang sepenuhnya memenuhi syarat untuk properti kustom jika tipe data dari properti kustom adalah Struct atau StructArray

Nama yang sepenuhnya memenuhi syarat dapat memiliki hingga 150 karakter. Karakter yang valid: a—z, A-Z, 0-9, : (titik dua), dan _ (garis bawah).

Buat katalog sinyal (AWS CLI)

Anda dapat menggunakan operasi [CreateSignalCatalog](#) API untuk membuat katalog sinyal. Contoh berikut menggunakan AWS CLI.

Untuk membuat katalog sinyal, jalankan perintah berikut.

Ganti *signal-catalog-configuration* dengan nama file JSON yang berisi konfigurasi.

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

- Ganti *signal-catalog-name* dengan nama katalog sinyal yang Anda buat.
- (Opsional) Ganti *deskripsi* dengan deskripsi untuk membantu Anda mengidentifikasi katalog sinyal.

Untuk informasi selengkapnya tentang cara mengonfigurasi cabang, atribut, sensor, dan aktuator, lihat [Konfigurasi sinyal](#)

```
{
  "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

Anda dapat mengunduh [skrip demo](#) untuk mengonversi pesan ROS 2 ke file VSS JSON yang kompatibel dengan katalog sinyal. Untuk informasi selengkapnya, lihat [Panduan Pengembang Data Sistem Visi](#).

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Impor katalog sinyal

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk mengimpor katalog sinyal.

Topik

- [Impor katalog sinyal \(konsol\)](#)
- [Impor katalog sinyal \(AWS CLI\)](#)

Impor katalog sinyal (konsol)

Anda dapat menggunakan FleetWise konsol AWS IoT untuk mengimpor katalog sinyal.

Important

Anda dapat memiliki maksimal satu katalog sinyal. Jika Anda sudah memiliki katalog sinyal, Anda tidak akan melihat opsi untuk mengimpor katalog sinyal di konsol.

Untuk mengimpor katalog sinyal

1. Buka konsol [AWS IoT FleetWise](#) .
2. Pada panel navigasi, pilih Katalog sinyal.
3. Pada halaman ringkasan katalog sinyal, pilih Impor katalog sinyal.
4. Impor file yang berisi sinyal.
 - Untuk mengunggah file dari bucket S3:
 - a. Pilih Impor dari S3.
 - b. Pilih Jelajahi S3.
 - c. Untuk Bucket, masukkan nama atau objek bucket, pilih dari daftar, lalu pilih file dari daftar. Pilih tombol Pilih file.

Atau, untuk URI S3, masukkan URI Amazon Simple Storage Service. Untuk informasi selengkapnya, lihat [Metode untuk mengakses bucket](#) di Panduan Pengguna Amazon S3.

- Untuk mengunggah file dari komputer Anda:
 - a. Pilih Impor dari file.
 - b. Unggah file.json dalam format [Spesifikasi Sinyal Kendaraan \(VSS\)](#).
5. Verifikasi katalog sinyal, lalu pilih Impor file.

Impor katalog sinyal (AWS CLI)

Anda dapat menggunakan operasi [ImportSignalCatalog](#) API untuk mengunggah file JSON yang membantu membuat katalog sinyal. Anda harus mengikuti [Spesifikasi Sinyal Kendaraan \(VSS\)](#) untuk menyimpan sinyal dalam file JSON. Contoh berikut menggunakan AWS CLI.

Untuk mengimpor katalog sinyal, jalankan perintah berikut.

- Ganti *signal-catalog-name* dengan nama katalog sinyal yang Anda buat.
- (Opsional) Ganti *deskripsi dengan deskripsi* untuk membantu Anda mengidentifikasi katalog sinyal.
- Ganti *signal-catalog-configuration-vss* dengan nama file string JSON yang berisi sinyal yang didefinisikan dalam VSS.

Untuk informasi selengkapnya tentang cara mengonfigurasi cabang, atribut, sensor, dan aktuator, lihat [Konfigurasi sinyal](#)

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

JSON harus dirangkai dan melewati lapangan. vssJson Berikut ini adalah contoh sinyal yang didefinisikan dalam VSS.

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

```

    }
  }
}
}
}

```

Contoh berikut menunjukkan sinyal yang sama didefinisikan dalam VSS dalam string JSON.

```

{
  "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

Anda dapat mengunduh [skrip demo](#) untuk mengonversi pesan ROS 2 ke file VSS JSON yang kompatibel dengan katalog sinyal. Untuk informasi selengkapnya, lihat [Panduan Pengembang Data Sistem Visi](#).

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Perbarui katalog sinyal (AWS CLI)

Anda dapat menggunakan operasi [UpdateSignalCatalog](#) API untuk memperbarui katalog sinyal yang ada. Contoh berikut menggunakan AWS CLI.

Untuk memperbarui katalog sinyal yang ada, jalankan perintah berikut.

Ganti *signal-catalog-configuration* dengan nama file JSON yang berisi konfigurasi.

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

Ganti *signal-catalog-name* dengan nama katalog sinyal yang sedang Anda perbarui.

Untuk informasi selengkapnya tentang cara mengonfigurasi cabang, atribut, sensor, dan aktuator, lihat [Konfigurasi sinyal](#)

⚠ Important

Struktur khusus tidak dapat diubah. Jika Anda perlu memesan ulang atau menyisipkan properti ke struktur kustom yang ada (struct), hapus struktur dan buat struktur baru dengan urutan properti yang diinginkan.

Untuk menghapus struktur kustom, tambahkan nama struktur yang sepenuhnya memenuhi syarat `nodesToRemove`. Struktur tidak dapat dihapus jika dirujuk oleh sinyal apa pun. Setiap sinyal yang merujuk pada struktur (tipe datanya didefinisikan sebagai struktur target) harus diperbarui atau dihapus sebelum permintaan untuk memperbarui katalog sinyal.

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

```
}
```

Hapus katalog sinyal (AWS CLI)

Anda dapat menggunakan operasi [DeleteSignalCatalog](#) API untuk menghapus katalog sinyal. Contoh berikut menggunakan AWS CLI.

Important

Sebelum menghapus katalog sinyal, pastikan tidak memiliki model kendaraan terkait, manifes decoder, kendaraan, armada, atau kampanye. Untuk petunjuk, lihat yang berikut ini:

- [Hapus model kendaraan](#)
- [Hapus manifes decoder](#)
- [Hapus kendaraan](#)
- [Menghapus armada \(AWS CLI\)](#)
- [Menghapus kampanye](#)

Untuk menghapus katalog sinyal yang ada, jalankan perintah berikut. Ganti *signal-catalog-name* dengan nama katalog sinyal yang Anda hapus.

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

Note

Perintah ini tidak menghasilkan output.

Dapatkan informasi katalog sinyal (AWS CLI)

Anda dapat menggunakan operasi [ListSignalCatalogs](#) API untuk memverifikasi apakah katalog sinyal telah dihapus. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar paginasi ringkasan semua katalog sinyal, jalankan perintah berikut.

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

Anda dapat menggunakan operasi [ListSignalCatalogNodes](#) API untuk memverifikasi apakah katalog sinyal telah diperbarui. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar paginasi ringkasan semua sinyal (node) dalam katalog sinyal tertentu, jalankan perintah berikut.

Ganti *signal-catalog-name* dengan nama katalog sinyal yang Anda periksa.

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

Anda dapat menggunakan operasi [GetSignalCatalog](#) API untuk mengambil informasi katalog sinyal. Contoh berikut menggunakan AWS CLI.

Untuk mengambil informasi tentang katalog sinyal, jalankan perintah berikut.

Ganti *signal-catalog-name* dengan nama katalog sinyal yang ingin Anda ambil.

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

Note

Operasi ini [pada akhirnya konsisten](#). Dengan kata lain, perubahan pada katalog sinyal mungkin tidak langsung tercermin.

Membuat dan mengelola model kendaraan

Anda menggunakan sinyal untuk membuat model kendaraan yang membantu menstandarisasi format kendaraan Anda. Model kendaraan menerapkan informasi yang konsisten di beberapa kendaraan dengan jenis yang sama, sehingga Anda dapat memproses data dari armada kendaraan. Kendaraan yang dibuat dari model kendaraan yang sama mewarisi kelompok sinyal yang sama. Untuk informasi selengkapnya, lihat [Membuat, menyediakan, dan mengelola kendaraan](#).

Setiap model kendaraan memiliki bidang status yang berisi status model kendaraan. Negara dapat menjadi salah satu dari nilai berikut:

- ACTIVE— Model kendaraan aktif.
- DRAFT— Konfigurasi model kendaraan disimpan.

Important

- Jika Anda ingin menggunakan operasi `CreateModelManifest` API untuk membuat model kendaraan pertama, Anda harus membuat katalog sinyal terlebih dahulu. Untuk informasi selengkapnya, lihat [Buat katalog sinyal \(AWS CLI\)](#).
- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat model kendaraan, AWS IoT FleetWise secara otomatis mengaktifkan model kendaraan untuk Anda.
- Jika Anda menggunakan operasi `CreateModelManifest` API untuk membuat model kendaraan, model kendaraan tetap dalam DRAFT status.
- Anda tidak dapat membuat kendaraan dari model kendaraan yang ada di DRAFT negara bagian. Gunakan operasi `UpdateModelManifest` API untuk mengubah model kendaraan ke ACTIVE status.
- Anda tidak dapat mengedit model kendaraan yang ada di ACTIVE negara bagian.

Topik

- [Buat model kendaraan](#)
- [Perbarui model kendaraan \(AWS CLI\)](#)
- [Hapus model kendaraan](#)
- [Dapatkan informasi model kendaraan \(AWS CLI\)](#)

Buat model kendaraan

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk membuat model kendaraan.

Important

Anda harus memiliki katalog sinyal sebelum Anda dapat membuat model kendaraan dengan menggunakan operasi `CreateModelManifest` API.

Topik

- [Buat model kendaraan \(konsol\)](#)
- [Buat model kendaraan \(AWS CLI\)](#)

Buat model kendaraan (konsol)

Di FleetWise konsol AWS IoT, Anda dapat membuat model kendaraan dengan cara berikut:

- [Gunakan template yang disediakan oleh AWS](#)
- [Buat model kendaraan secara manual](#)
- [Duplikat model kendaraan](#)

Gunakan template yang disediakan oleh AWS

AWS IoT FleetWise menyediakan templat On-board Diagnostic (OBD) II, J1979 yang secara otomatis membuat katalog sinyal, model kendaraan, dan manifes decoder untuk Anda. Template juga menambahkan antarmuka jaringan OBD ke manifes decoder. Untuk informasi selengkapnya, lihat [Buat dan kelola manifes decoder](#).

Untuk membuat model kendaraan dengan menggunakan template

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Pada panel navigasi, pilih Model kendaraan.
3. Pada halaman Model kendaraan, pilih Tambahkan templat yang disediakan.
4. Pilih Diagnostik On-board (OBD) II.
5. Masukkan nama untuk antarmuka jaringan OBD yang dibuat AWS FleetWise IoT.
6. Pilih Tambahkan.

Buat model kendaraan secara manual

Anda dapat menambahkan sinyal dari katalog sinyal atau mengimpor sinyal dengan mengunggah satu atau beberapa file.dbc. File DBC adalah format file yang didukung oleh database Controller Area Network (CAN bus).

Important

Anda tidak dapat membuat model kendaraan dengan sinyal data sistem visi menggunakan konsol AWS IoT FleetWise . Sebaliknya, gunakan AWS CLI untuk membuat model kendaraan.

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Untuk membuat model kendaraan secara manual

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Pada panel navigasi, pilih Model kendaraan.
3. Pada halaman Model kendaraan, pilih Buat model kendaraan, lalu lakukan hal berikut.

Topik

- [Langkah 1: Konfigurasi model kendaraan](#)
- [Langkah 2: Tambahkan sinyal](#)
- [Langkah 3: Impor sinyal](#)
- [\(Opsional\) Langkah 4: Tambahkan atribut](#)
- [Langkah 5: Tinjau dan buat](#)

Langkah 1: Konfigurasi model kendaraan

Secara umum informasi, lakukan hal berikut.

1. Masukkan nama untuk model kendaraan.
2. (Opsional) Masukkan deskripsi.
3. Pilih Berikutnya.

Langkah 2: Tambahkan sinyal

 Note

- Jika ini adalah pertama kalinya Anda menggunakan AWS IoT FleetWise, langkah ini tidak tersedia sampai Anda memiliki katalog sinyal. Ketika model kendaraan pertama dibuat, AWS IoT FleetWise secara otomatis membuat katalog sinyal dengan sinyal yang ditambahkan ke model kendaraan pertama.
- Jika Anda berpengalaman dengan AWS IoT FleetWise, Anda dapat menambahkan sinyal ke model kendaraan Anda dengan memilih sinyal dari katalog sinyal atau mengunggah file.dbc untuk mengimpor sinyal.
- Anda harus memiliki setidaknya satu sinyal untuk membuat model kendaraan.

Untuk menambahkan sinyal

1. Pilih satu atau beberapa sinyal dari katalog sinyal yang Anda tambahkan ke model kendaraan. Anda dapat meninjau sinyal yang dipilih di panel kanan.

Note

Hanya sinyal yang dipilih yang akan ditambahkan ke model kendaraan.

2. Pilih Berikutnya.

Langkah 3: Impor sinyal

Note

- Jika ini adalah pertama kalinya Anda menggunakan AWS IoT FleetWise, Anda harus mengunggah setidaknya satu file.dbc untuk mengimpor sinyal.
- Jika Anda berpengalaman dengan AWS IoT FleetWise, Anda dapat menambahkan sinyal ke model kendaraan Anda dengan memilih sinyal dari katalog sinyal atau mengunggah file.dbc untuk mengimpor sinyal.
- Anda harus memiliki setidaknya satu sinyal untuk membuat model kendaraan.

Untuk mengimpor sinyal

1. Pilih file.
2. Di kotak dialog, pilih file.dbc yang berisi sinyal. Anda dapat mengunggah beberapa file.dbc.
3. AWS IoT FleetWise mem-parsing file.dbc Anda untuk mengambil sinyal.

Di bagian Sinyal, tentukan metadata berikut untuk setiap sinyal.

- Nama - Nama sinyal.

Nama sinyal harus unik. Nama sinyal ditambah jalur dapat memiliki hingga 150 karakter. Karakter yang valid: a—z, A-Z, 0-9, (titik dua), dan _ (garis bawah).

- Tipe data - Tipe data sinyal harus salah satu dari yang berikut: INT8, UINT8, INT16, UINT16, INT32, UUINT32, INT64, UUINT64, BOOLEAN, FLOAT, DOUBLE, STRING,

UNIX_TIMESTAMP, INT8_ARRAY, UINT8_ARRAY, INT16_ARRAY, INT32_ARRAY, 64_ARRAY, UINT64_ARRAY, BOOLEAN_ARRAY, FLOAT_ARRAY, DOUBLE_ARRAY, STRING_ARRAY, UNIX_TIMESTAMP_ARRAY, ATAU TIDAK DIKETAHUI.

- Tipe sinyal — Jenis sinyal, yang dapat berupa Sensor atau Aktuator.
- (Opsional) Unit — Unit ilmiah untuk sinyal, seperti km atau Celcius.
- (Opsional) Jalur — Jalur menuju sinyal. Mirip dengan JsonPath, gunakan titik (.) untuk merujuk ke sinyal anak. Misalnya, **Vehicle.Engine.Light**.

Nama sinyal ditambah jalur dapat memiliki hingga 150 karakter. Karakter yang valid: a—z, A-Z, 0-9, (titik dua), dan _ (garis bawah).

- (Opsional) Min — Nilai minimum sinyal.
- (Opsional) Maks - Nilai maksimum sinyal.
- (Opsional) Deskripsi — Deskripsi untuk sinyal.

Deskripsi dapat memiliki hingga 2048 karakter. Karakter yang valid: a—z, A-Z, 0-9, (titik dua), _ (garis bawah), dan - (tanda hubung).

4. Pilih Berikutnya.

(Opsional) Langkah 4: Tambahkan atribut

Anda dapat menambahkan hingga 100 atribut, termasuk atribut yang ada di katalog sinyal.

Untuk menambahkan atribut

1. Dalam Tambahkan atribut, tentukan metadata berikut untuk setiap atribut.

- Nama - Nama atribut.

Nama sinyal harus unik. Nama sinyal dan jalur dapat memiliki hingga 150 karakter. Karakter yang valid: a—z, A-Z, 0-9, (titik dua), dan _ (garis bawah)

- Tipe data - Tipe data atribut harus salah satu dari yang berikut: INT8, UINT8, INT16, UINT16, INT32, UUINT32, INT64, UUINT64, BOOLEAN, FLOAT, DOUBLE, STRING, UNIX_TIMESTAMP, INT8_ARRAY, UUINT8_ARRAY, INT16_ARRAY, INT32_ARRAY, 64_ARRAY, UUINT64_ARRAY, BOOLEAN_ARRAY, FLOAT_ARRAY, DOUBLE_ARRAY, STRING_ARRAY, UNIX_TIMESTAMP_ARRAY, ATAU TIDAK DIKETAHUI
- (Opsional) Unit — Unit ilmiah untuk atribut, seperti km atau Celcius.

- (Opsional) Jalur — Jalur menuju sinyal. Mirip dengan JsonPath, gunakan titik (.) untuk merujuk ke sinyal anak. Misalnya, **Vehicle.Engine.Light**.

Nama sinyal ditambah jalur dapat memiliki hingga 150 karakter. Karakter yang valid: a—z, A-Z, 0-9, : (titik dua), dan _ (garis bawah)

- (Opsional) Min — Nilai minimum atribut.
- (Opsional) Maks - Nilai maksimum atribut.
- (Opsional) Deskripsi — Deskripsi untuk atribut.

Deskripsi dapat memiliki hingga 2048 karakter. Karakter yang valid: a—z, A-Z, 0-9, : (titik dua), _ (garis bawah), dan - (tanda hubung).

2. Pilih Berikutnya.

Langkah 5: Tinjau dan buat

Verifikasi konfigurasi untuk model kendaraan, lalu pilih Buat.

Duplikat model kendaraan

AWS IoT FleetWise dapat menyalin konfigurasi model kendaraan yang ada untuk membuat model baru. Sinyal yang ditentukan dalam model kendaraan yang dipilih disalin ke model kendaraan baru.

Untuk menduplikasi model kendaraan

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Pada panel navigasi, pilih Model kendaraan.
3. Pilih model dari daftar model kendaraan, lalu pilih Model duplikat.

Untuk mengkonfigurasi model kendaraan, ikuti [Buat model kendaraan secara manual](#) tutorialnya.

Diperlukan beberapa menit bagi AWS IoT FleetWise untuk memproses permintaan Anda untuk membuat model kendaraan. Setelah model kendaraan berhasil dibuat, pada halaman Model kendaraan, kolom Status menunjukkan AKTIF. Ketika model kendaraan menjadi aktif, Anda tidak dapat mengeditnya.

Buat model kendaraan (AWS CLI)

Anda dapat menggunakan operasi [CreateModelManifest](#) API untuk membuat model kendaraan (manifes model). Contoh berikut menggunakan AWS CLI.

⚠ Important

Jika Anda ingin menggunakan AWS IoT FleetWise API untuk membuat model kendaraan pertama, Anda harus membuat katalog sinyal terlebih dahulu. Untuk informasi selengkapnya tentang cara membuat katalog sinyal, lihat [Buat katalog sinyal \(AWS CLI\)](#).

Untuk membuat model kendaraan, jalankan perintah berikut.

Ganti *vehicle-model-configuration* dengan nama file JSON yang berisi konfigurasi.

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

- Ganti *vehicle-model-name* dengan nama model kendaraan yang Anda buat.
- Ganti *Signal-Catalog-ARN* dengan Amazon Resource Name (ARN) dari katalog sinyal.
- (Opsional) Ganti *deskripsi* dengan deskripsi untuk membantu Anda mengidentifikasi model kendaraan.

Untuk informasi selengkapnya tentang cara mengonfigurasi cabang, atribut, sensor, dan aktuator, lihat [Konfigurasi sinyal](#)

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

Perbarui model kendaraan (AWS CLI)

Anda dapat menggunakan operasi [UpdateModelManifest](#) API untuk memperbarui model kendaraan yang ada (manifes model). Contoh berikut menggunakan AWS CLI.

Untuk memperbarui model kendaraan yang ada, jalankan perintah berikut.

Ganti *update-vehicle-model-configuration* dengan nama file JSON yang berisi konfigurasi.

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

- Ganti *vehicle-model-name* dengan nama model kendaraan yang Anda perbarui.
- (Opsional) Untuk mengaktifkan model kendaraan, ganti *vehicle-model-status* dengan ACTIVE.

Important

Setelah model kendaraan diaktifkan, Anda tidak dapat mengubah model kendaraan.

- (Opsional) Ganti *deskripsi* dengan deskripsi yang diperbarui untuk membantu Anda mengidentifikasi model kendaraan.

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

Hapus model kendaraan

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk menghapus model kendaraan.

Important

Kendaraan dan manifes decoder yang terkait dengan model kendaraan harus dihapus terlebih dahulu. Lihat informasi yang lebih lengkap di [Hapus kendaraan](#) dan [Hapus manifes decoder](#).

Hapus model kendaraan (konsol)

Untuk menghapus model kendaraan, gunakan konsol AWS IoT FleetWise .

Untuk menghapus model kendaraan

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Pada panel navigasi, pilih Model kendaraan.
3. Pada halaman Model kendaraan, pilih model kendaraan target.
4. Pilih Hapus.
5. Di Hapus **vehicle-model-name?** , masukkan nama model kendaraan yang akan dihapus, lalu pilih Konfirmasi.

Hapus model kendaraan (AWS CLI)

Anda dapat menggunakan operasi [DeleteModelManifest](#) API untuk menghapus model kendaraan yang ada (manifes model). Contoh berikut menggunakan AWS CLI.

Untuk menghapus model kendaraan, jalankan perintah berikut.

Ganti *model-manifest-name* dengan nama model kendaraan yang Anda hapus.

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

Note

Perintah ini tidak menghasilkan output.

Dapatkan informasi model kendaraan (AWS CLI)

Anda dapat menggunakan operasi [ListModelManifests](#) API untuk memverifikasi apakah model kendaraan telah dihapus. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar ringkasan paginasi semua model kendaraan, jalankan perintah berikut.

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

Anda dapat menggunakan operasi [ListModelManifestNodes](#) API untuk memverifikasi apakah model kendaraan telah diperbarui. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar paginasi ringkasan semua sinyal (node) dalam model kendaraan tertentu, jalankan perintah berikut.

Ganti *vehicle-model-name* dengan nama model kendaraan yang Anda periksa.

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

Untuk mengambil informasi tentang model kendaraan, jalankan perintah berikut.

Ganti *model kendaraan* dengan nama model kendaraan yang ingin Anda ambil.

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

Note

Operasi ini [pada akhirnya konsisten](#). Dengan kata lain, perubahan pada model kendaraan mungkin tidak langsung tercermin.

Buat dan kelola manifes decoder

Manifestasi decoder berisi informasi decoding yang FleetWise digunakan AWS IoT untuk mengubah data kendaraan (data biner) menjadi nilai yang dapat dibaca manusia dan untuk mempersiapkan data Anda untuk analisis data. Antarmuka jaringan dan sinyal decoder adalah komponen inti yang Anda gunakan untuk mengonfigurasi manifes decoder.

Antarmuka jaringan

Berisi informasi tentang protokol yang digunakan jaringan dalam kendaraan. AWS IoT FleetWise mendukung protokol berikut.

Jaringan Area Pengontrol (CAN bus)

Protokol yang mendefinisikan bagaimana data dikomunikasikan antara unit kontrol elektronik (ECU). ECU dapat berupa unit kontrol mesin, airbag, atau sistem audio.

Diagnostik on-board (OBD) II

Protokol yang dikembangkan lebih lanjut yang mendefinisikan bagaimana data diagnostik mandiri dikomunikasikan antara ECU. Ini menyediakan sejumlah kode masalah diagnostik standar (DTC) yang membantu mengidentifikasi apa yang salah dengan kendaraan Anda.

Middleware kendaraan

Middleware kendaraan didefinisikan sebagai jenis antarmuka jaringan. Contoh middleware kendaraan termasuk Robot Operating System (ROS 2) dan Scalable Service-oriented Middleware over IP (SOME/IP).

Note

AWS IoT FleetWise mendukung middleware ROS 2 untuk data sistem visi.

Sinyal decoder

Memberikan informasi decoding terperinci untuk sinyal tertentu. Setiap sinyal yang ditentukan dalam model kendaraan harus dipasangkan dengan sinyal decoder. Jika manifes decoder berisi antarmuka jaringan CAN, itu harus berisi sinyal decoder CAN. Jika manifes decoder berisi antarmuka jaringan OBD, itu harus berisi sinyal decoder OBD.

Manifes decoder harus berisi sinyal decoder pesan jika juga berisi antarmuka middleware kendaraan.

Setiap manifes decoder harus dikaitkan dengan model kendaraan. AWS IoT FleetWise menggunakan manifes decoder terkait untuk memecahkan kode data dari kendaraan yang dibuat berdasarkan model kendaraan.

Setiap manifes decoder memiliki bidang status yang berisi status manifes decoder. Negara dapat menjadi salah satu dari nilai berikut:

- **ACTIVE**— Manifes decoder aktif.
- **DRAFT**— Konfigurasi manifes decoder tidak disimpan.
- **VALIDATING**— Manifes decoder berada di bawah validasi untuk kelayakannya. Ini hanya berlaku untuk manifes decoder yang berisi setidaknya satu sinyal data sistem penglihatan.
- **INVALID**— Manifes dekoder gagal validasi dan belum dapat diaktifkan. Ini hanya berlaku untuk manifes decoder yang berisi setidaknya satu sinyal data sistem penglihatan. Anda dapat

menggunakan `GetDecoderManifest` API `ListDecoderManifests` dan untuk memeriksa alasan validasi gagal.

Important

- Jika Anda menggunakan FleetWise konsol AWS IoT untuk membuat manifes dekoder, AWS IoT FleetWise secara otomatis mengaktifkan manifes dekoder untuk Anda.
- Jika Anda menggunakan operasi `CreateDecoderManifest` API untuk membuat manifes dekoder, manifes dekoder tetap berada dalam status. DRAFT
- Anda tidak dapat membuat kendaraan dari model kendaraan yang terkait dengan DRAFT manifes dekoder. Gunakan operasi `UpdateDecoderManifest` API untuk mengubah manifes dekoder ke status. ACTIVE
- Anda tidak dapat mengedit manifes decoder yang berada dalam status. ACTIVE

Topik

- [Konfigurasi antarmuka jaringan dan sinyal decoder](#)
- [Buat manifes decoder](#)
- [Perbarui manifes decoder \(\)AWS CLI](#)
- [Hapus manifes decoder](#)
- [Dapatkan informasi manifes decoder \(\)AWS CLI](#)

Konfigurasi antarmuka jaringan dan sinyal decoder

Setiap manifes decoder memiliki setidaknya antarmuka jaringan dan sinyal decoder yang dipasangkan dengan sinyal yang ditentukan dalam model kendaraan terkait.

Jika manifes decoder berisi antarmuka jaringan CAN, itu harus berisi sinyal decoder CAN. Jika manifes decoder berisi antarmuka jaringan OBD, itu harus berisi sinyal decoder OBD.

Topik

- [Konfigurasi antarmuka jaringan](#)
- [Konfigurasi sinyal decoder](#)

Konfigurasi antarmuka jaringan

Untuk mengkonfigurasi antarmuka jaringan CAN, tentukan informasi berikut.

- `name`— Nama antarmuka CAN.

Nama antarmuka harus unik dan dapat memiliki 1-100 karakter.

- (Opsional) `protocolName` — Nama protokol.

Nilai yang valid: CAN-FD dan CAN

- (Opsional) `protocolVersion` - AWS IoT FleetWise saat ini mendukung CAN-FD dan CAN 2.0b.

Nilai yang valid: 1.0 dan 2.0b

Untuk mengkonfigurasi antarmuka jaringan OBD, tentukan informasi berikut.

- `name`— Nama antarmuka OBD.

Nama antarmuka harus unik dan dapat memiliki 1-100 karakter.

- `requestMessageId`— ID pesan yang meminta data.

- (Opsional) `dtcRequestIntervalSeconds` — Seberapa sering meminta kode masalah diagnostik (DTC) dari kendaraan dalam hitungan detik. Misalnya, jika nilai yang ditentukan adalah 120, perangkat lunak Edge Agent mengumpulkan DTC yang disimpan setiap 2 menit sekali.

- (Opsional) `hasTransmissionEcu` — Apakah kendaraan memiliki modul kontrol transmisi (TCM).

Nilai yang valid: `true` dan `false`

- (Opsional) `obdStandard` - Standar OBD yang didukung AWS FleetWise IoT. AWS IoT FleetWise saat ini mendukung standar World Wide Harmonization On-Board Diagnostics (WWH-OBD) ISO15765-4.

- (Opsional) `pidRequestIntervalSeconds` — Seberapa sering meminta PID OBD II dari kendaraan. Misalnya, jika nilai yang ditentukan adalah 120, perangkat lunak Edge Agent mengumpulkan PID OBD II setiap 2 menit sekali.

- (Opsional) `useExtendedIds` — Apakah akan menggunakan ID yang diperluas dalam pesan.

Nilai yang valid: `true` dan `false`

Untuk mengkonfigurasi antarmuka jaringan middleware kendaraan, tentukan informasi berikut.

- `name`— Nama antarmuka middleware kendaraan.

Nama antarmuka harus unik dan dapat memiliki 1-100 karakter.

- `protocolName`- Nama protokol.

Nilai yang valid: `R0S_2`

Konfigurasi sinyal decoder

Untuk mengkonfigurasi sinyal decoder CAN, tentukan informasi berikut.

- `factor`— Pengganda yang digunakan untuk memecahkan kode pesan.
- `isBigEndian`— Apakah urutan byte pesan adalah big-endian. Jika besar-endian, nilai paling signifikan dalam urutan disimpan terlebih dahulu, di alamat penyimpanan terendah.
- `isSigned`Apakah pesan tersebut ditandatangani. Jika ditandatangani, pesan dapat mewakili angka positif dan negatif.
- `length`— Panjang pesan dalam byte.
- `messageId`— ID pesan.
- `offset`— Offset yang digunakan untuk menghitung nilai sinyal. Dikombinasikan dengan faktor, perhitungannya adalah $value = raw_value * factor + offset$.
- `startBit`— Menunjukkan lokasi bit pertama pesan.
- (Opsional) `name` — Nama sinyal.

Untuk mengkonfigurasi sinyal decoder OBD, tentukan informasi berikut.

- `byteLength`— Panjang pesan dalam byte.
- `offset`— Offset yang digunakan untuk menghitung nilai sinyal. Dikombinasikan dengan penskalaan, perhitungannya adalah $value = raw_value * scaling + offset$.
- `pid`— Kode diagnostik yang digunakan untuk meminta pesan dari kendaraan untuk sinyal ini.
- `pidResponseLength`— Panjang pesan yang diminta.
- `scaling`— Pengganda yang digunakan untuk memecahkan kode pesan.
- `serviceMode`— Mode operasi (layanan diagnostik) dalam pesan.
- `startByte`— Menunjukkan awal pesan.
- (Opsional) `bitMaskLength` — Jumlah bit yang ditutupi dalam pesan.

- (Opsional) `bitRightShift` — Jumlah posisi bergeser ke kanan.

Untuk mengkonfigurasi sinyal decoder pesan, tentukan informasi berikut.

- `topicName`— Nama topik untuk sinyal pesan. Ini sesuai dengan topik di ROS 2. Untuk informasi selengkapnya tentang objek pesan terstruktur, lihat [StructuredMessage](#).
- `structuredMessage`— Pesan terstruktur untuk sinyal pesan. Hal ini dapat didefinisikan dengan baik `primitiveMessageDefinition`, `structuredMessageList` Definisi, atau `structuredMessageDefinition` rekursif.

Buat manifes decoder

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk membuat manifes dekoder untuk model kendaraan Anda.

Important

Anda harus memiliki model kendaraan sebelum Anda dapat membuat manifes decoder. Setiap manifes decoder harus dikaitkan dengan model kendaraan. Untuk informasi selengkapnya, lihat [Membuat dan mengelola model kendaraan](#).

Topik

- [Buat manifes dekoder \(konsol\)](#)
- [Buat manifes decoder \(\)AWS CLI](#)

Buat manifes dekoder (konsol)

Anda dapat menggunakan FleetWise konsol AWS IoT untuk membuat manifes decoder yang terkait dengan model kendaraan Anda.

Important

Anda tidak dapat mengonfigurasi sinyal data sistem penglihatan dalam manifes decoder menggunakan konsol IoT AWS . FleetWise Sebagai gantinya, gunakan AWS CLI. Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Untuk membuat manifes decoder

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Pada panel navigasi, pilih Model kendaraan.
3. Pilih model kendaraan target.
4. Pada halaman ringkasan model kendaraan, pilih Buat manifes decoder, lalu lakukan hal berikut.

Topik

- [Langkah 1: Konfigurasi manifes decoder](#)
- [Langkah 2: Tambahkan antarmuka jaringan](#)
- [Langkah 3: Tinjau dan buat](#)

Langkah 1: Konfigurasi manifes decoder

Secara umum informasi, lakukan hal berikut.

1. Masukkan nama unik untuk manifes decoder.
2. (Opsional) Masukkan deskripsi.
3. Pilih Berikutnya.

Langkah 2: Tambahkan antarmuka jaringan

Setiap manifes decoder harus memiliki setidaknya satu antarmuka jaringan. Anda dapat menambahkan beberapa antarmuka jaringan ke manifes decoder.

Untuk menambahkan antarmuka jaringan

- Di antarmuka Jaringan, lakukan hal berikut.
 - a. Untuk jenis antarmuka Jaringan, pilih CAN_INTERFACE atau OBD_INTERFACE.
 - b. Masukkan nama unik untuk antarmuka jaringan Anda.
 - c. Masukkan ID antarmuka jaringan yang unik. Anda dapat menggunakan ID yang dihasilkan oleh AWS IoT FleetWise.
 - d. Pilih satu atau lebih sinyal yang ditentukan dalam model kendaraan Anda untuk dipasangkan dengan sinyal decoder.

- e. Untuk memberikan informasi decoding, unggah file.dbc. AWS IoT FleetWise mem-parsing file.dbc untuk mengambil sinyal decoder.
- f. Di bagian Sinyal berpasangan, pastikan bahwa setiap sinyal dipasangkan dengan sinyal decoder.
- g. Pilih Berikutnya.

Note

- Anda hanya dapat mengunggah satu file.dbc untuk setiap antarmuka jaringan.
- Pastikan bahwa setiap sinyal yang ditentukan dalam model kendaraan Anda dipasangkan dengan sinyal decoder.
- Setelah Anda memilih untuk menambahkan antarmuka jaringan lain, Anda tidak dapat mengedit salah satu yang Anda edit. Anda dapat menghapus antarmuka jaringan yang ada.

Langkah 3: Tinjau dan buat

Verifikasi konfigurasi untuk manifes dekoder, lalu pilih Buat.

Buat manifes decoder ()AWS CLI

Anda dapat menggunakan operasi [CreateDecoderManifest](#) API untuk membuat manifes decoder. Contoh berikut menggunakan AWS CLI.

Important

Sebelum Anda membuat manifes decoder, buat model kendaraan terlebih dahulu. Untuk informasi selengkapnya, lihat [Buat model kendaraan](#).

Untuk membuat manifes decoder, jalankan perintah berikut.

Ganti *decoder-manifest-configuration* dengan nama file JSON yang berisi konfigurasi.

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

- Ganti *decoder-manifest-name* dengan nama manifes decoder yang Anda buat.
- Ganti *Vehicle-model-ARN* dengan Amazon Resource Name (ARN) dari model kendaraan.
- (Opsional) Ganti *deskripsi* dengan deskripsi untuk membantu Anda mengidentifikasi manifes decoder.

Untuk informasi selengkapnya tentang cara mengonfigurasi cabang, atribut, sensor, dan aktuator, lihat [Konfigurasi antarmuka jaringan dan sinyal decoder](#)

```
{
  "name": "decoder-manifest-name",
  "modelManifestArn": "vehicle-model-arn",
  "description": "description",
  "networkInterfaces": [
    {
      "canInterface": {
        "name": "myNetworkInterface",
        "protocolName": "CAN",
        "protocolVersion": "2.0b"
      },
      "interfaceId": "Qq1acaenBy0B3sSM39SYm",
      "type": "CAN_INTERFACE"
    }
  ],
  "signalDecoders": [
    {
      "canSignal": {
        "name": "Engine_Idle_Time",
        "factor": 1,
        "isBigEndian": true,
        "isSigned": false,
        "length": 24,
        "messageId": 271343712,
        "offset": 0,
        "startBit": 16
      },
      "fullyQualified_name": "Vehicle.EngineIdleTime",
      "interfaceId": "Qq1acaenBy0B3sSM39SYm",
      "type": "CAN_SIGNAL"
    },
    {
      "canSignal": {
```

```

        "name": "Engine_Run_Time",
        "factor": 1,
        "isBigEndian": true,
        "isSigned": false,
        "length": 24,
        "messageId": 271343712,
        "offset": 0,
        "startBit": 40
    },
    "fullyQualifiedNames": "Vehicle.EngineRunTime",
    "interfaceId": "Qq1acaenByOB3sSM39SYm",
    "type": "CAN_SIGNAL"
}
]
}

```

- Ganti *decoder-manifest-name* dengan nama manifes decoder yang Anda buat.
- Ganti *Vehicle-model-ARN* dengan Amazon Resource Name (ARN) dari model kendaraan.
- (Opsional) Ganti *deskripsi* dengan deskripsi untuk membantu Anda mengidentifikasi manifes decoder.

Urutan node properti dalam struktur (struct) harus tetap konsisten seperti yang didefinisikan dalam katalog sinyal dan model kendaraan (manifes model). Untuk informasi selengkapnya tentang cara mengonfigurasi cabang, atribut, sensor, dan aktuator, lihat [Konfigurasi antarmuka jaringan dan sinyal decoder](#)

```

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

Note

Anda dapat mengunduh [skrip demo](#) untuk membuat manifes decoder dengan sinyal sistem penglihatan. Untuk informasi selengkapnya, lihat [Panduan Pengembang Data Sistem Visi](#). Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Perbarui manifes decoder ()AWS CLI

Anda dapat menggunakan operasi [UpdateDecoderManifest](#) API untuk memperbarui manifes decoder. Anda dapat menambahkan, menghapus, dan memperbarui antarmuka jaringan dan decoder sinyal. Anda juga dapat mengubah status manifes decoder. Contoh berikut menggunakan AWS CLI.

Untuk memperbarui manifes decoder, jalankan perintah berikut.

Ganti *decoder-manifest-name* dengan nama manifes decoder yang Anda perbarui.

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

Important

Setelah Anda mengaktifkan manifes decoder, Anda tidak dapat mengeditnya.

Hapus manifes decoder

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk menghapus manifes dekoder.

Important

Kendaraan yang terkait dengan manifes decoder harus dihapus terlebih dahulu. Untuk informasi selengkapnya, lihat [Hapus kendaraan](#).

Topik

- [Hapus manifes dekoder \(konsol\)](#)
- [Hapus manifes dekoder \(\)AWS CLI](#)

Hapus manifes dekoder (konsol)

Anda dapat menggunakan FleetWise konsol AWS IoT untuk menghapus manifes decoder.

Untuk menghapus manifes decoder

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Pada panel navigasi, pilih Model kendaraan.
3. Pilih model kendaraan target.
4. Pada halaman ringkasan model kendaraan, pilih tab manifes Decoder.
5. Pilih manifes dekoder target, lalu pilih Hapus.
6. Di Hapus **decoder-manifest-name?** , masukkan nama manifes dekoder untuk dihapus, lalu pilih Konfirmasi.

Hapus manifes dekoder ()AWS CLI

Anda dapat menggunakan operasi [DeleteDecoderManifest](#) API untuk menghapus manifes decoder. Contoh berikut menggunakan AWS CLI.

Important

Sebelum Anda menghapus manifes dekoder, hapus kendaraan terkait terlebih dahulu. Untuk informasi selengkapnya, lihat [Hapus kendaraan](#).

Untuk menghapus manifes decoder, jalankan perintah berikut.

Ganti *decoder-manifest-name* dengan nama manifes decoder yang Anda hapus.

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

Dapatkan informasi manifes decoder ()AWS CLI

Anda dapat menggunakan operasi [ListDecoderManifests](#) API untuk memverifikasi apakah manifes decoder telah dihapus. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar paginasi ringkasan dari semua manifes decoder, jalankan perintah berikut.

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

Anda dapat menggunakan operasi [ListDecoderManifestSignals](#) API untuk memverifikasi apakah sinyal dekoder dalam manifes decoder telah diperbarui. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar ringkasan dari semua sinyal decoder (node) dalam manifes decoder tertentu, jalankan perintah berikut.

Ganti *decoder-manifest-name* dengan nama manifes decoder yang Anda periksa.

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

Anda dapat menggunakan operasi [ListDecoderManifestNetworkInterfaces](#) API untuk memverifikasi apakah antarmuka jaringan dalam manifes decoder telah diperbarui. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar paginasi ringkasan semua antarmuka jaringan dalam manifes decoder tertentu, jalankan perintah berikut.

Ganti *decoder-manifest-name* dengan nama manifes decoder yang Anda periksa.

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

Anda dapat menggunakan operasi [GetDecoderManifest](#) API untuk memverifikasi apakah antarmuka jaringan dan sinyal dekoder dalam manifes decoder telah diperbarui. Contoh berikut menggunakan AWS CLI.

Untuk mengambil informasi tentang manifes decoder, jalankan perintah berikut.

Ganti *decoder-manifest* dengan nama manifes decoder yang ingin Anda ambil.

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

Note

Operasi ini [pada akhirnya konsisten](#). Dengan kata lain, perubahan pada manifes decoder mungkin tidak segera tercermin.

Membuat, menyediakan, dan mengelola kendaraan

Kendaraan adalah contoh model kendaraan. Kendaraan harus dibuat dari model kendaraan dan dikaitkan dengan manifes decoder. Kendaraan mengunggah satu atau lebih aliran data ke cloud. Misalnya, kendaraan dapat mengirim jarak tempuh, suhu mesin, dan status data pemanas ke cloud. Setiap kendaraan berisi informasi berikut:

`vehicleName`

ID yang mengidentifikasi kendaraan.

Jangan menambahkan informasi identitas pribadi (PII) atau informasi rahasia atau sensitif lainnya dalam nama kendaraan Anda. Nama kendaraan dapat diakses oleh AWS layanan lain, termasuk Amazon CloudWatch. Nama kendaraan tidak dimaksudkan untuk digunakan untuk data pribadi atau sensitif.

`modelManifestARN`

Nama Sumber Daya Amazon (ARN) dari model kendaraan (manifes model). Setiap kendaraan dibuat dari model kendaraan. Kendaraan yang dibuat dari model kendaraan yang sama terdiri dari kelompok sinyal yang sama yang diwarisi dari model kendaraan. Sinyal-sinyal ini didefinisikan dan distandarisasi dalam katalog sinyal.

`decoderManifestArn`

ARN dari manifes decoder. Manifes decoder menyediakan informasi decoding yang FleetWise dapat digunakan AWS IoT untuk mengubah data sinyal mentah (data biner) menjadi nilai yang dapat dibaca manusia. Manifes dekoder harus dikaitkan dengan model kendaraan. AWS IoT FleetWise menggunakan manifes decoder yang sama untuk memecahkan kode data mentah dari kendaraan yang dibuat berdasarkan model kendaraan yang sama.

`attributes`

Atribut adalah pasangan kunci-nilai yang berisi informasi statis. Kendaraan dapat berisi atribut yang diwarisi dari model kendaraan. Anda dapat menambahkan atribut tambahan untuk membedakan kendaraan individu dari kendaraan lain yang dibuat dari model kendaraan yang sama. Misalnya, jika Anda memiliki mobil hitam, Anda dapat menentukan nilai berikut untuk atribut: `{"color": "black"}`.

⚠ Important

Atribut harus didefinisikan dalam model kendaraan terkait sebelum Anda dapat menambahkannya ke kendaraan individu.

Untuk informasi selengkapnya tentang model kendaraan, manifes decoder, dan atribut, lihat.

[Kendaraan pemodelan](#)

AWS IoT FleetWise menyediakan operasi API berikut yang dapat Anda gunakan untuk membuat dan mengelola kendaraan.

- [CreateVehicle](#)— Menciptakan kendaraan baru.
- [BatchCreateVehicle](#)— Membuat satu atau lebih kendaraan baru.
- [UpdateVehicle](#)— Memperbarui kendaraan yang ada.
- [BatchUpdateVehicle](#)— Memperbarui satu atau lebih kendaraan yang ada.
- [DeleteVehicle](#)— Menghapus kendaraan yang ada.
- [ListVehicles](#)— Mengambil daftar ringkasan semua kendaraan yang diberi halaman.
- [GetVehicle](#)— Mengambil informasi tentang kendaraan.

Tutorial

- [Kendaraan penyediaan](#)
- [Topik yang dipesan](#)
- [Buat kendaraan](#)
- [Perbarui kendaraan \(AWS CLI\)](#)
- [Perbarui beberapa kendaraan \(AWS CLI\)](#)
- [Hapus kendaraan](#)
- [Dapatkan informasi kendaraan \(AWS CLI\)](#)

Kendaraan penyediaan

Agan Edge untuk FleetWise perangkat lunak AWS IoT yang berjalan di kendaraan Anda mengumpulkan dan mentransfer data ke cloud. AWS IoT FleetWise terintegrasi dengan AWS IoT Core untuk mendukung komunikasi yang aman antara perangkat lunak Edge Agent dan cloud melalui

MQTT. Setiap kendaraan sesuai dengan suatu AWS IoT hal. Anda dapat menggunakan AWS IoT benda yang sudah ada untuk membuat kendaraan atau mengatur AWS IoT FleetWise untuk secara otomatis membuat AWS IoT sesuatu untuk kendaraan Anda. Untuk informasi selengkapnya, lihat [Buat kendaraan \(AWS CLI\)](#).

AWS IoT Core mendukung [otentikasi](#) dan [otorisasi](#) yang membantu mengontrol akses ke sumber daya AWS IoT dengan aman. FleetWise Kendaraan dapat menggunakan sertifikat X.509 untuk mendapatkan otentikasi (masuk) untuk menggunakan AWS IoT FleetWise dan AWS IoT Core kebijakan untuk mendapatkan otorisasi (memiliki izin) untuk melakukan tindakan tertentu.

Otentikasi kendaraan

Anda dapat membuat AWS IoT Core kebijakan untuk mengautentikasi kendaraan Anda.

Untuk mengautentikasi kendaraan Anda

- Untuk membuat AWS IoT Core kebijakan, jalankan perintah berikut.
 - Ganti *nama kebijakan* dengan nama kebijakan yang ingin Anda buat.
 - Ganti *nama file* dengan nama file JSON yang berisi kebijakan. AWS IoT Core

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

Sebelum Anda menggunakan kebijakan contoh, lakukan hal berikut:

- Ganti *wilayah* dengan AWS Wilayah tempat Anda membuat sumber daya AWS IoT FleetWise.
- Ganti *AWSaccount* dengan ID AWS akun Anda.

Contoh ini mencakup topik yang dicadangkan oleh AWS IoT FleetWise. Anda harus menambahkan topik ke kebijakan. Untuk informasi selengkapnya, lihat [Topik yang dipesan](#).

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

```
        "Action": [
            "iot:Connect"
        ],
        "Resource": [
            "arn:aws:iot:region:awsAccount:client/
${iot:Connection.Thing.ThingName}"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "iot:Publish"
        ],
        "Resource": [
            "arn:aws:iot:region:awsAccount:topic/$aws/iotfleetwise/vehicles/
${iot:Connection.Thing.ThingName}/checkins",
            "arn:aws:iot:region:awsAccount:topic/$aws/iotfleetwise/vehicles/
${iot:Connection.Thing.ThingName}/signals"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "iot:Subscribe"
        ],
        "Resource": [
            "arn:aws:iot:region:awsAccount:topicfilter/$aws/iotfleetwise/
vehicles/${iot:Connection.Thing.ThingName}/collection_schemes",
            "arn:aws:iot:region:awsAccount:topicfilter/$aws/iotfleetwise/
vehicles/${iot:Connection.Thing.ThingName}/decoder_manifests"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "iot:Receive"
        ],
        "Resource": [
            "arn:aws:iot:region:awsAccount:topic/$aws/iotfleetwise/vehicles/
${iot:Connection.Thing.ThingName}/collection_schemes",
            "arn:aws:iot:region:awsAccount:topic/$aws/iotfleetwise/vehicles/
${iot:Connection.Thing.ThingName}/decoder_manifests"
        ]
    }
}
```

```
]
}
```

Otorisasi kendaraan

Anda dapat membuat sertifikat X.509 untuk mengotorisasi kendaraan Anda.

Untuk mengotorisasi kendaraan Anda

Important

Kami menyarankan Anda membuat sertifikat baru untuk setiap kendaraan.

1. Untuk membuat key pair RSA dan mengeluarkan sertifikat X.509, jalankan perintah berikut.
 - Ganti *sertifikat* dengan nama file yang menyimpan isi output perintah CertificatePEM.
 - Ganti *public-key* dengan nama file yang menyimpan isi output perintah KeyPair. PublicKey.
 - Ganti *private-key* dengan nama file yang menyimpan isi output perintah KeyPair. 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. Salin Nama Sumber Daya Amazon (ARN) sertifikat dari output.
3. Untuk melampirkan kebijakan ke sertifikat, jalankan perintah berikut.
 - Ganti *nama kebijakan* dengan nama AWS IoT Core kebijakan yang Anda buat.
 - Ganti *sertifikat-arn dengan* ARN dari sertifikat yang Anda salin.

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

4. Untuk melampirkan sertifikat ke benda itu, jalankan perintah berikut.
- Ganti *nama barang* dengan nama AWS IoT barang Anda atau ID kendaraan Anda.
 - Ganti *sertifikat-arn dengan* ARN dari sertifikat yang Anda salin.

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

Topik yang dipesan

AWS IoT FleetWise mencadangkan penggunaan topik-topik berikut. Jika topik yang dicadangkan memungkinkan, Anda dapat berlangganan atau memublikasikannya. Namun, Anda tidak dapat membuat topik baru yang dimulai dengan tanda dolar (\$). Jika Anda menggunakan operasi publikasi atau berlangganan yang tidak didukung dengan topik yang dicadangkan, itu dapat mengakibatkan koneksi berakhir.

Topik	Operasi klien diizinkan	Deskripsi
\$aws/iotfleetwise/vehicles/ <i>vehicleName</i> /checkins	Publikasikan	Perangkat lunak Edge Agent menerbitkan informasi status kendaraan untuk topik ini. Informasi status kendaraan dipertukarkan dalam format buffer protokol (protobuf). Untuk informasi selengkapnya, lihat Panduan Pengembang FleetWise perangkat

Topik	Operasi klien diizinkan	Deskripsi
		lunak Edge Agent for AWS IoT.
\$aws/iotfleetwise/vehicles/ <i>vehicleName</i> /signals	Publikasikan	Perangkat lunak Edge Agent menerbitkan sinyal untuk topik ini. Informasi sinyal dipertukarkan dalam format buffer protokol (protobuf). Untuk informasi selengkapnya, lihat Panduan Pengembang FleetWise perangkat lunak Edge Agent for AWS IoT.
\$aws/iotfleetwise/vehicles/ <i>vehicleName</i> /collection_schemes	Langganan	AWS IoT FleetWise menerbitkan skema pengumpulan data untuk topik ini. Kendaraan mengkonsumsi skema pengumpulan data ini.
\$aws/iotfleetwise/vehicles/ <i>vehicleName</i> /decoder_manifests	Langganan	AWS IoT FleetWise menerbitkan manifes decoder untuk topik ini. Kendaraan mengkonsumsi manifes decoder ini.

Buat kendaraan

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk membuat kendaraan.

Important

Sebelum Anda mulai, periksa yang berikut ini:

- Anda harus memiliki model kendaraan dan status model kendaraan harus ACTIVE. Untuk informasi selengkapnya, lihat [Membuat dan mengelola model kendaraan](#).
- Model kendaraan Anda harus dikaitkan dengan manifes decoder, dan status manifes decoder harus ACTIVE. Untuk informasi selengkapnya, lihat [Buat dan kelola manifes decoder](#).

Topik

- [Buat kendaraan \(konsol\)](#)
- [Buat kendaraan \(AWS CLI\)](#)
- [Buat beberapa kendaraan \(AWS CLI\)](#)

Buat kendaraan (konsol)

Anda dapat menggunakan FleetWise konsol AWS IoT untuk membuat kendaraan.

Important

Sebelum Anda mulai, periksa yang berikut ini:

- Anda harus memiliki model kendaraan dan status model kendaraan harus ACTIVE. Untuk informasi selengkapnya, lihat [Membuat dan mengelola model kendaraan](#).
- Model kendaraan Anda harus dikaitkan dengan manifes decoder, dan status manifes decoder harus ACTIVE. Untuk informasi selengkapnya, lihat [Buat dan kelola manifes decoder](#).

Untuk membuat kendaraan

1. Buka konsol [AWS IoT FleetWise](#) .
2. Pada panel navigasi, pilih Kendaraan.
3. Pada halaman ringkasan kendaraan, pilih Buat kendaraan, lalu lakukan langkah-langkah berikut.

Topik

- [Langkah 1: Tentukan properti kendaraan](#)
- [Langkah 2: Konfigurasi sertifikat kendaraan](#)
- [Langkah 3: Lampirkan kebijakan ke sertifikat](#)
- [Langkah 4: Tinjau dan buat](#)

Langkah 1: Tentukan properti kendaraan

Pada langkah ini, Anda memberi nama kendaraan dan mengaitkannya dengan manifes model dan manifes dekoder.

1. Masukkan nama unik untuk kendaraan.

Important

Kendaraan sesuai dengan AWS IoT sesuatu. Jika sesuatu sudah ada dengan nama itu, pilih Kaitkan kendaraan dengan IoT untuk memperbarui barang dengan kendaraan. Atau, pilih nama kendaraan yang berbeda dan AWS IoT FleetWise akan secara otomatis membuat hal baru untuk kendaraan.

2. Pilih model kendaraan (manifes model) dari daftar.
3. Pilih manifes decoder dari daftar. Manifes decoder dikaitkan dengan model kendaraan.
4. (Opsional) Untuk mengaitkan atribut kendaraan, pilih Tambahkan atribut. Jika Anda melewati langkah ini, Anda harus menambahkan atribut setelah kendaraan dibuat sebelum Anda dapat menerapkannya ke kampanye.
5. (Opsional) Untuk mengaitkan tag dengan kendaraan, pilih Tambahkan tag baru. Anda juga dapat menambahkan tag setelah kendaraan dibuat.
6. Pilih Selanjutnya.

Langkah 2: Konfigurasi sertifikat kendaraan

Untuk menggunakan kendaraan Anda sebagai AWS IoT sesuatu, Anda harus mengonfigurasi sertifikat kendaraan dengan kebijakan terlampir. Jika Anda melewati langkah ini, Anda harus mengonfigurasi sertifikat setelah kendaraan dibuat sebelum Anda dapat menerapkannya ke kampanye.

1. Pilih Buat otomatis sertifikat baru (disarankan).
2. Pilih Selanjutnya.

Langkah 3: Lampirkan kebijakan ke sertifikat

Lampirkan kebijakan ke sertifikat yang Anda konfigurasi pada langkah sebelumnya.

1. Untuk Kebijakan, masukkan nama kebijakan yang ada. Untuk membuat kebijakan baru, pilih Buat kebijakan.
2. Pilih Selanjutnya.

Langkah 4: Tinjau dan buat

Verifikasi konfigurasi untuk kendaraan, lalu pilih Buat kendaraan.

Important

Setelah kendaraan dibuat, Anda harus mengunduh sertifikat dan kunci. Anda akan menggunakan sertifikat dan kunci pribadi untuk menghubungkan kendaraan di Edge Agent untuk perangkat lunak AWS IoT FleetWise .

Buat kendaraan (AWS CLI)

Saat Anda membuat kendaraan, Anda harus menggunakan model kendaraan yang dikaitkan dengan manifes dekoder. Anda dapat menggunakan operasi [CreateVehicle](#) API untuk membuat kendaraan. Contoh berikut menggunakan AWS CLI.

Important

Sebelum Anda mulai, periksa yang berikut ini:

- Anda harus memiliki model kendaraan dan status model kendaraan harus ACTIVE. Untuk informasi selengkapnya, lihat [Membuat dan mengelola model kendaraan](#).
- Model kendaraan Anda harus dikaitkan dengan manifes decoder, dan status manifes decoder harus ACTIVE. Untuk informasi selengkapnya, lihat [Buat dan kelola manifes decoder](#).

Untuk membuat kendaraan, jalankan perintah berikut.

Ganti *nama file* dengan nama file JSON yang berisi konfigurasi kendaraan.

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

Example konfigurasi kendaraan

- (Opsional) `associationBehavior` Nilai dapat berupa salah satu dari berikut ini:
 - `CreateIotThing`— Ketika kendaraan Anda dibuat, AWS IoT FleetWise secara otomatis membuat AWS IoT sesuatu dengan nama ID kendaraan Anda untuk kendaraan Anda.
 - `ValidateIotThingExists`— Gunakan AWS IoT hal yang sudah ada untuk membuat kendaraan.

Untuk membuat AWS IoT sesuatu, jalankan perintah berikut. Ganti *thing-name* dengan nama benda yang ingin Anda buat.

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

Jika tidak ditentukan, AWS IoT FleetWise secara otomatis menciptakan AWS IoT sesuatu untuk kendaraan Anda.

Important

Pastikan AWS IoT barang itu disediakan setelah kendaraan dibuat. Untuk informasi selengkapnya, lihat [Kendaraan penyediaan](#).

- Ganti *nama kendaraan* dengan salah satu dari berikut ini.
 - Nama AWS IoT barang Anda jika `associationBehavior` dikonfigurasi ke `ValidateIotThingExists`.

- ID kendaraan yang akan dibuat jika `associationBehavior` dikonfigurasi ke `CreateIotThing`.

ID kendaraan dapat memiliki 1-100 karakter. Karakter yang valid: a—z, A-Z, 0—9, dasbor (-), garis bawah (_), dan titik dua (:).

- Ganti `Model-MANIFEST-ARN` dengan ARN model kendaraan Anda (manifes model).
- Ganti `decoder-manifest-ARN` dengan ARN dari manifes decoder yang terkait dengan model kendaraan yang ditentukan.
- (Opsional) Anda dapat menambahkan atribut tambahan untuk membedakan kendaraan ini dari kendaraan lain yang dibuat dari model kendaraan yang sama. Misalnya, jika Anda memiliki mobil listrik, Anda dapat menentukan nilai berikut untuk atribut: `{"fuelType": "electric"}`.

Important

Atribut harus didefinisikan dalam model kendaraan terkait sebelum Anda dapat menambahkannya ke kendaraan individu.

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

Buat beberapa kendaraan (AWS CLI)

Anda dapat menggunakan operasi [BatchCreateVehicle](#) API untuk membuat beberapa kendaraan sekaligus. Contoh berikut menggunakan AWS CLI.

Untuk membuat beberapa kendaraan, jalankan perintah berikut.

Ganti `nama file` dengan nama file JSON yang berisi konfigurasi beberapa kendaraan.

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

Example konfigurasi kendaraan

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

Anda dapat membuat hingga 10 kendaraan untuk setiap operasi batch. Untuk informasi lebih lanjut tentang konfigurasi kendaraan, lihat [Buat kendaraan \(AWS CLI\)](#).

Perbarui kendaraan (AWS CLI)

Anda dapat menggunakan operasi [UpdateVehicle](#) API untuk memperbarui kendaraan yang ada. Contoh berikut menggunakan AWS CLI.

Untuk memperbarui kendaraan, jalankan perintah berikut.

Ganti *nama file* dengan nama file JSON yang berisi konfigurasi kendaraan Anda.

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

Example konfigurasi kendaraan

- Ganti *nama kendaraan* dengan ID kendaraan yang ingin Anda perbarui.

- (Opsional) Ganti *Model-MANIFEST-ARN* dengan ARN model kendaraan (manifes model) yang Anda gunakan untuk mengganti model kendaraan yang digunakan.
- (Opsional) Ganti *Decoder-manifest-ARN dengan ARN* manifes dekoder Anda yang terkait dengan model kendaraan baru yang Anda tentukan.
- (Opsional) Ganti *attribute-update-mode* dengan atribut kendaraan.
 - Merge— Gabungkan atribut baru ke atribut yang ada dengan memperbarui atribut yang ada dengan nilai baru dan menambahkan atribut baru jika tidak ada.

Misalnya, jika kendaraan memiliki atribut berikut: `{"color": "black", "fuelType": "electric"}`, dan Anda memperbarui kendaraan dengan atribut berikut: `{"color": "", "fuelType": "gasoline", "model": "x"}`, kendaraan yang diperbarui memiliki atribut berikut: `{"fuelType": "gasoline", "model": "x"}`.

- Overwrite— Ganti atribut yang ada dengan atribut baru.

Misalnya, jika kendaraan memiliki atribut berikut: `{"color": "black", "fuelType": "electric"}`, dan Anda memperbarui kendaraan dengan `{"model": "x"}` atribut, kendaraan yang diperbarui memiliki `{"model": "x"}` atribut.

Ini diperlukan jika atribut hadir dalam input.

- (Opsional) Untuk menambahkan atribut baru atau memperbarui yang sudah ada dengan nilai baru, konfigurasi `attributes`. Misalnya, jika Anda memiliki mobil listrik, Anda dapat menentukan nilai berikut untuk atribut: `{"fuelType": "electric"}`.

Untuk menghapus atribut, konfigurasi `attributeUpdateMode` ke `Merge`.

Important

Atribut harus didefinisikan dalam model kendaraan terkait sebelum Anda dapat menambahkannya ke kendaraan individu.

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

```
}
```

Perbarui beberapa kendaraan (AWS CLI)

Anda dapat menggunakan operasi [BatchUpdateVehicle](#) API untuk memperbarui beberapa kendaraan yang ada sekaligus. Contoh berikut menggunakan AWS CLI.

Untuk memperbarui beberapa kendaraan, jalankan perintah berikut.

Ganti *nama file* dengan nama file JSON yang berisi konfigurasi beberapa kendaraan.

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

Example konfigurasi kendaraan

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

Anda dapat memperbarui hingga 10 kendaraan untuk setiap operasi batch. Untuk informasi lebih lanjut tentang konfigurasi setiap kendaraan, lihat [Perbarui kendaraan \(AWS CLI\)](#).

Hapus kendaraan

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk menghapus kendaraan.

Important

Setelah kendaraan dihapus, AWS IoT FleetWise secara otomatis menghapus kendaraan dari armada dan kampanye terkait. Untuk informasi selengkapnya, lihat [Membuat dan mengelola armada](#) dan [Mengumpulkan dan mentransfer data dengan kampanye](#). Namun, kendaraan masih ada sebagai benda atau masih dikaitkan dengan sesuatu di dalamnya AWS IoT Core. Untuk petunjuk cara menghapus sesuatu, lihat [Menghapus sesuatu](#) di Panduan AWS IoT Core Pengembang.

Hapus kendaraan (konsol)

Anda dapat menggunakan FleetWise konsol AWS IoT untuk menghapus kendaraan.

Untuk menghapus kendaraan

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Pada panel navigasi, pilih Kendaraan.
3. Pada halaman Kendaraan, pilih tombol di sebelah kendaraan yang ingin Anda hapus.
4. Pilih Hapus.
5. Di Hapus **vehicle-name**, masukkan nama kendaraan, lalu pilih Hapus.

Hapus kendaraan (AWS CLI)

Anda dapat menggunakan operasi [DeleteVehicle](#) API untuk menghapus kendaraan. Contoh berikut menggunakan AWS CLI.

Untuk menghapus kendaraan, jalankan perintah berikut.

Ganti *nama kendaraan* dengan ID kendaraan yang ingin Anda hapus.

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

Dapatkan informasi kendaraan (AWS CLI)

Anda dapat menggunakan operasi [ListVehicles](#) API untuk memverifikasi apakah kendaraan telah dihapus. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar paginasi ringkasan semua kendaraan, jalankan perintah berikut.

```
aws iotfleetwise list-vehicles
```

Anda dapat menggunakan operasi [GetVehicle](#) API untuk mengambil informasi kendaraan. Contoh berikut menggunakan AWS CLI.

Untuk mengambil metadata kendaraan, jalankan perintah berikut.

Ganti *nama kendaraan* dengan ID kendaraan yang ingin Anda ambil.

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

Note

Operasi ini [pada akhirnya konsisten](#). Dengan kata lain, perubahan pada kendaraan mungkin tidak langsung tercermin.

Membuat dan mengelola armada

Armada mewakili sekelompok kendaraan. Armada tanpa kendaraan terkait adalah entitas kosong. Sebelum Anda dapat menggunakan armada untuk mengelola beberapa kendaraan pada saat yang sama, Anda harus mengasosiasikan kendaraan dengan armada. Sebuah kendaraan dapat menjadi bagian dari beberapa armada. Anda dapat mengontrol data apa yang akan dikumpulkan dari armada kendaraan dan kapan harus mengumpulkan data dengan menyebarkan kampanye. Untuk informasi selengkapnya, lihat [Mengumpulkan dan mentransfer data dengan kampanye](#).

Armada berisi informasi berikut.

`fleetId`

ID armada.

(Opsional) `description`

Deskripsi yang membantu Anda menemukan armada.

`signalCatalogArn`

Amazon Resource Name (ARN) dari katalog sinyal.

AWS IoT FleetWise menyediakan operasi API berikut yang dapat Anda gunakan untuk membuat dan mengelola armada.

- [CreateFleet](#)- Membuat sekelompok kendaraan yang berisi kelompok sinyal yang sama.
- [AssociateVehicleFleet](#)— Mengaitkan kendaraan ke armada.
- [DisassociateVehicleFleet](#)— Disassociates kendaraan dari armada.
- [UpdateFleet](#)- Update deskripsi untuk armada yang sudah ada.
- [DeleteFleet](#)- Menghapus armada yang ada.
- [ListFleets](#)- Mengambil daftar paginasi ringkasan dari semua armada.
- [ListFleetsForVehicle](#)- Mengambil daftar paginasi ID dari semua armada bahwa kendaraan milik.
- [ListVehiclesInFleet](#)- Mengambil daftar paginasi ringkasan semua kendaraan dalam armada.
- [GetFleet](#)- Mengambil informasi tentang armada.

Topik

- [Buat armada \(AWS CLI\)](#)
- [Mengaitkan kendaraan dengan armada \(AWS CLI\)](#)
- [Putus kendaraan dari armada \(AWS CLI\)](#)
- [Memperbarui armada \(AWS CLI\)](#)
- [Menghapus armada \(AWS CLI\)](#)
- [Dapatkan informasi armada \(AWS CLI\)](#)

Buat armada (AWS CLI)

Anda dapat menggunakan operasi [CreateFleet](#) API untuk membuat armada kendaraan. Contoh berikut menggunakan AWS CLI.

Important

Anda harus memiliki katalog sinyal sebelum dapat membuat armada. Untuk informasi selengkapnya, lihat [Buat katalog sinyal \(AWS CLI\)](#).

Untuk membuat armada, jalankan perintah berikut.

- Ganti *fleet-id* dengan ID armada yang Anda buat.

ID armada harus unik dan memiliki 1-100 karakter. karakter yang valid: huruf (A-Z dan a-z z, angka (0-9), titik (:), tanda hubung (-), dan garis bawah (_).

- (Opsional) Ganti *deskripsi* dengan deskripsi.

Deskripsi dapat memiliki 1-2048 karakter.

- Ganti *signal-catalog-arn* dengan ARN katalog sinyal.

```
aws iotfleetwise create-fleet \  
  --fleet-id fleet-id \  
  --description deskripsi \  
  --signal-catalog-arn signal-catalog-arn
```

Mengaitkan kendaraan dengan armada (AWS CLI)

Anda dapat menggunakan operasi [AssociateVehicleFleet](#) API untuk mengaitkan kendaraan dengan armada. Contoh berikut menggunakan AWS CLI.

Important

- Anda harus memiliki kendaraan dan armada sebelum Anda dapat mengasosiasikan kendaraan dengan armada. Untuk informasi selengkapnya, lihat [Membuat, menyediakan, dan mengelola kendaraan](#).
- Jika Anda mengaitkan kendaraan dengan armada yang ditargetkan oleh kampanye, AWS IoT FleetWise secara otomatis menyebarkan kampanye ke kendaraan.

Untuk mengaitkan kendaraan dengan armada, jalankan perintah berikut.

- Ganti *armada id* dengan ID armada.
- Ganti *nama kendaraan* dengan ID kendaraan.

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

Putus kendaraan dari armada (AWS CLI)

Anda dapat menggunakan operasi [DisassociateVehicleFleet](#) API untuk memisahkan kendaraan dari armada. Contoh berikut menggunakan AWS CLI.

Untuk memisahkan kendaraan dengan armada, jalankan perintah berikut.

- Ganti *armada id* dengan ID armada.
- Ganti *nama kendaraan* dengan ID kendaraan.

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

Memperbarui armada (AWS CLI)

Anda dapat menggunakan operasi [UpdateFleet](#) API untuk memperbarui deskripsi armada. Contoh berikut menggunakan AWS CLI.

Untuk memperbarui armada, jalankan perintah berikut.

- Ganti *fleet-id* dengan ID armada yang Anda perbarui.
- Ganti *deskripsi* dengan deskripsi baru.

Deskripsi dapat memiliki 1-2048 karakter.

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

Menghapus armada (AWS CLI)

Anda dapat menggunakan operasi [DeleteFleet](#) API untuk menghapus armada. Contoh berikut menggunakan AWS CLI.

Important

Sebelum Anda menghapus armada, pastikan armada tidak memiliki kendaraan terkait.

Untuk petunjuk tentang cara memisahkan kendaraan dari armada, lihat [Putus kendaraan dari armada \(AWS CLI\)](#).

Untuk menghapus armada, jalankan perintah berikut.

Ganti *fleet-id* dengan ID armada yang Anda hapus.

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

Dapatkan informasi armada (AWS CLI)

Anda dapat menggunakan operasi [ListFleets](#) API untuk memverifikasi apakah armada telah dihapus. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar ringkasan dari semua armada, jalankan perintah berikut.

```
aws iotfleetwise list-fleets
```

Anda dapat menggunakan operasi [ListFleetsForVehicle](#) API untuk mengambil daftar paginasi ID dari semua armada yang dimiliki kendaraan. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar ID dari semua armada yang menjadi milik kendaraan, jalankan perintah berikut.

Ganti *nama kendaraan* dengan ID kendaraan.

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

Anda dapat menggunakan operasi [ListVehiclesInFleet](#) API untuk mengambil daftar paginasi ringkasan semua kendaraan dalam armada. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar ringkasan dari semua kendaraan di armada, jalankan perintah berikut.

Ganti *armada id* dengan ID armada.

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

Anda dapat menggunakan operasi [GetFleet](#) API untuk mengambil informasi armada. Contoh berikut menggunakan AWS CLI.

Untuk mengambil metadata armada, jalankan perintah berikut.

Ganti *armada id* dengan ID armada.

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

Note

Operasi ini [akhirnya konsisten](#). Dengan kata lain, perubahan pada armada mungkin tidak segera tercermin.

Mengumpulkan dan mentransfer data dengan kampanye

Kampanye adalah orkestrasi aturan pengumpulan data. Kampanye memberikan instruksi FleetWise perangkat lunak Agen Edge untuk AWS IoT tentang cara memilih, mengumpulkan, dan mentransfer data ke cloud.

Anda membuat kampanye di cloud. Setelah Anda atau tim Anda menyetujui kampanye, AWS IoT FleetWise secara otomatis menyebarkannya ke kendaraan. Anda dapat memilih untuk menyebarkan kampanye ke kendaraan atau armada kendaraan. Perangkat lunak Edge Agent tidak mulai mengumpulkan data sampai kampanye yang sedang berjalan diterapkan ke kendaraan.

Note

Kampanye tidak akan berfungsi sampai Anda memiliki yang berikut ini.

- Perangkat lunak Edge Agent berjalan di kendaraan Anda. Untuk informasi lebih lanjut tentang cara mengembangkan, menginstal, dan bekerja dengan perangkat lunak Edge Agent, lakukan hal berikut.
 1. Arahkan ke konsol [AWS IoT FleetWise](#).
 2. Di halaman beranda layanan, di FleetWise bagian Memulai dengan AWS IoT, pilih Explore Edge Agent.
- Anda telah mengatur AWS IoT Core untuk menyediakan kendaraan Anda. Untuk informasi selengkapnya, lihat [Kendaraan penyediaan](#).

Setiap kampanye berisi informasi berikut.

`signalCatalogArn`

Nama Sumber Daya Amazon (ARN) dari katalog sinyal yang terkait dengan kampanye.

(Opsional) `tags`

Tag adalah metadata yang dapat digunakan untuk mengelola kampanye. Anda dapat menetapkan tag yang sama ke sumber daya dari layanan yang berbeda untuk menunjukkan bahwa sumber daya terkait.

TargetArn

ARN kendaraan atau armada tempat kampanye dikerahkan.

name

Nama unik yang membantu mengidentifikasi kampanye.

collectionScheme

Skema pengumpulan data memberikan instruksi perangkat lunak Edge Agent tentang data apa yang harus dikumpulkan atau kapan harus mengumpulkannya. AWS IoT FleetWise saat ini mendukung skema pengumpulan berbasis kondisi dan skema pengumpulan berbasis waktu.

conditionBasedCollectionScheme

Skema pengumpulan berbasis kondisi menggunakan ekspresi logis untuk mengenali data apa yang akan dikumpulkan. Perangkat lunak Edge Agent mengumpulkan data ketika kondisi terpenuhi.

expression

Ekspresi logis digunakan untuk mengenali data apa yang akan dikumpulkan. Misalnya, jika `$variable.`myVehicle.InVehicleTemperature` > 50.0` ekspresi ditentukan, perangkat lunak Edge Agent mengumpulkan nilai suhu yang lebih besar dari 50,0. Untuk petunjuk tentang cara menulis ekspresi, lihat [Ekspresi logis untuk kampanye](#).

(Opsional) `triggerMode` dapat menjadi salah satu dari nilai-nilai berikut.

- `RISING_EDGE`— Perangkat lunak Edge Agent mengumpulkan data hanya ketika kondisi terpenuhi untuk pertama kalinya. Sebagai contoh, `$variable.`myVehicle.AirBagDeployed` == true`.
- `ALWAYS`— Perangkat lunak Edge Agent mengumpulkan data setiap kali kondisi terpenuhi.

(Opsional) `minimumTriggerIntervalMs`

Durasi minimum waktu antara dua peristiwa pengumpulan data, dalam milidetik. Jika sinyal sering berubah, Anda mungkin mengumpulkan data pada tingkat yang lebih lambat.

(Opsional) `conditionLanguageVersion`

Versi bahasa ekspresi bersyarat.

`timeBasedCollectionScheme`

Saat Anda menentukan skema pengumpulan berbasis waktu, tentukan periode waktu dalam milidetik. Perangkat lunak Edge Agent menggunakan periode waktu untuk memutuskan seberapa

sering mengumpulkan data. Misalnya, jika periode waktunya 120.000 milidetik, perangkat lunak Edge Agent mengumpulkan data setiap dua menit sekali.

(Opsional) `compression`

Untuk menghemat bandwidth nirkabel dan mengurangi lalu lintas jaringan, Anda dapat menentukan [SNAPPY](#) untuk mengompres data di kendaraan.

Secara default (OFF), perangkat lunak Edge Agent tidak memampatkan data.

`dataDestinationConfigs`

Pilih tujuan di mana kampanye akan mentransfer data kendaraan. Anda dapat memilih untuk menyimpan data di Amazon S3 atau Amazon Timestream.

S3 adalah mekanisme penyimpanan data hemat biaya yang menawarkan kemampuan manajemen data yang tahan lama dan layanan data hilir. Anda dapat menggunakan S3 untuk data yang terkait dengan perilaku mengemudi atau menganalisis pemeliharaan jangka panjang.

Timestream adalah mekanisme persistensi data yang dapat membantu Anda mengidentifikasi tren dan pola dalam waktu dekat. Anda dapat menggunakan Timestream untuk data deret waktu, seperti untuk menganalisis tren historis dalam kecepatan kendaraan atau pengereman.

(Opsional) `dataExtraDimensions`

Anda dapat menambahkan satu atau beberapa atribut untuk memberikan informasi tambahan untuk sinyal.

(Opsional) `description`

Anda dapat menambahkan deskripsi untuk membantu mengidentifikasi tujuan kampanye.

(Opsional) `diagnosticsMode`

Saat mode diagnostik dikonfigurasi `SEND_ACTIVE_DTCS`, kampanye mengirimkan kode masalah diagnostik standar (DTC) tersimpan yang membantu mengidentifikasi apa yang salah dengan kendaraan Anda. Misalnya, P0097 menunjukkan modul kontrol mesin (ECM) telah menentukan bahwa input intake air temperature sensor 2 (IAT2) lebih rendah dari kisaran sensor normal.

Secara default (OFF), perangkat lunak Edge Agent tidak mengirim kode diagnostik.

(Opsional) `expiryTime`

Anda dapat menentukan tanggal kedaluwarsa untuk kampanye Anda. Saat kampanye kedaluwarsa, perangkat lunak Agen Edge berhenti mengumpulkan data sebagaimana ditentukan

dalam kampanye ini. Jika beberapa kampanye diterapkan ke kendaraan, perangkat lunak Edge Agent menggunakan kampanye lain untuk mengumpulkan data.

Nilai default: 253402243200 (31 Desember 9999, 00:00:00 UTC)

(Opsional) `postTriggerCollectionDuration`

Anda dapat menentukan durasi pengumpulan pasca-pemicu, sehingga perangkat lunak Edge Agent terus mengumpulkan data untuk periode tertentu setelah skema dipanggil. Misalnya, jika skema pengumpulan berbasis kondisi dengan ekspresi berikut dipanggil: `$variable.`myVehicle.Engine.RPM` > 7000.0`, perangkat lunak Edge Agent terus mengumpulkan nilai putaran per menit (RPM) untuk mesin. Bahkan jika RPM hanya lebih tinggi dari 7000 sekali, itu mungkin menunjukkan bahwa ada masalah mekanis. Dalam hal ini, Anda mungkin ingin perangkat lunak Edge Agent terus mengumpulkan data untuk membantu memantau kondisi.

Nilai default: 0

(Opsional) `priority`

Anda dapat menentukan bilangan bulat untuk menunjukkan tingkat prioritas kampanye. Kampanye dengan jumlah yang lebih kecil adalah prioritas yang lebih tinggi. Jika Anda menerapkan beberapa kampanye ke kendaraan, kampanye yang memiliki prioritas lebih tinggi akan dimulai terlebih dahulu.

Nilai default: 0

(Opsional) `signalsToCollect`

Daftar sinyal dari mana data dikumpulkan ketika skema pengumpulan data dipanggil.

Important

Sinyal yang digunakan dalam ekspresi untuk skema pengumpulan berbasis kondisi harus ditentukan dalam bidang ini.

`name`

Nama sinyal dari mana data dikumpulkan ketika skema pengumpulan data dipanggil.

(Opsional) `maxSampleCount`

Jumlah maksimum sampel data yang dikumpulkan dan ditransfer oleh perangkat lunak Edge Agent ke cloud saat skema pengumpulan data dipanggil.

(Opsional) `minimumSamplingIntervalMs`

Durasi minimum waktu antara dua peristiwa pengumpulan sampel data, dalam milidetik. Jika sinyal sering berubah, Anda dapat menggunakan parameter ini untuk mengumpulkan data pada tingkat yang lebih lambat.

Rentang yang valid: 0-4294967295

(Opsional) `spoolingMode`

Jika `spoolingMode` dikonfigurasi `TO_DISK`, perangkat lunak Edge Agent untuk sementara menyimpan data secara lokal saat kendaraan tidak terhubung ke cloud. Setelah koneksi dibangun kembali, data yang disimpan secara lokal ditransfer secara otomatis ke cloud.

Nilai default: `OFF`

(Opsional) `startTime`

Kampanye yang disetujui diaktifkan pada waktu mulai.

Nilai default: `0`

Status kampanye dapat menjadi salah satu dari nilai berikut.

- **CREATING**— AWS IoT FleetWise sedang memproses permintaan Anda untuk membuat kampanye.
- **WAITING_FOR_APPROVAL**— Setelah kampanye dibuat, ia memasuki `WAITING_FOR_APPROVAL` negara bagian. Untuk menyetujui kampanye, gunakan operasi `UpdateCampaign` API. Setelah kampanye disetujui, AWS IoT FleetWise secara otomatis menyebarkan kampanye ke kendaraan atau armada target. Untuk informasi selengkapnya, lihat [Memperbarui kampanye \(AWS CLI\)](#).
- **RUNNING** Kampanye ini aktif.
- **SUSPENDED** Kampanye ditangguhkan. Untuk melanjutkan kampanye, gunakan operasi `UpdateCampaign` API.

AWS IoT FleetWise menyediakan operasi API berikut yang dapat Anda gunakan untuk membuat dan mengelola kampanye.

- [CreateCampaign](#)— Membuat kampanye baru.
- [UpdateCampaign](#)— Memperbarui kampanye yang ada. Setelah kampanye dibuat, Anda harus menggunakan operasi API ini untuk menyetujui kampanye.
- [DeleteCampaign](#)— Menghapus kampanye yang ada.
- [ListCampaigns](#)— Mengambil daftar ringkasan paginasi untuk semua kampanye.
- [GetCampaign](#)— Mengambil informasi tentang kampanye.

Tutorial

- [Buat kampanye](#)
- [Memperbarui kampanye \(AWS CLI\)](#)
- [Menghapus kampanye](#)
- [Dapatkan informasi kampanye \(AWS CLI\)](#)

Buat kampanye

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk membuat kampanye untuk mengumpulkan data kendaraan.

Important

Agar kampanye Anda berfungsi, Anda harus memiliki yang berikut:

- Perangkat lunak Edge Agent berjalan di kendaraan Anda. Untuk informasi lebih lanjut tentang cara mengembangkan, menginstal, dan bekerja dengan perangkat lunak Edge Agent, lakukan hal berikut:
 1. Arahkan ke konsol [AWS IoT FleetWise](#).
 2. Di halaman beranda layanan, di FleetWise bagian Memulai dengan AWS IoT, pilih Explore Edge Agent.
- Anda telah mengatur AWS IoT Core untuk menyediakan kendaraan Anda. Untuk informasi selengkapnya, lihat [Kendaraan penyediaan](#).

Topik

- [Buat kampanye \(konsol\)](#)

- [Buat kampanye \(AWS CLI\)](#)
- [Ekspresi logis untuk kampanye](#)

Buat kampanye (konsol)

Anda dapat menggunakan FleetWise konsol AWS IoT untuk membuat kampanye untuk memilih, mengumpulkan, dan mentransfer data kendaraan ke cloud.

Untuk membuat kampanye

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Pada panel navigasi, pilih Kampanye.
3. Pada halaman Kampanye, pilih Buat kampanye, lalu selesaikan langkah-langkah dalam topik berikut.

Topik

- [Langkah 1: Konfigurasi kampanye](#)
- [Langkah 2: Tentukan tujuan penyimpanan](#)
- [Langkah 3: Tambahkan kendaraan](#)
- [Langkah 4: Tinjau dan buat](#)
- [Langkah 5: Menyebarkan kampanye](#)

Important

- Anda harus memiliki katalog sinyal dan kendaraan sebelum Anda membuat kampanye. Untuk informasi selengkapnya, silakan lihat [Membuat dan mengelola katalog sinyal](#) dan [Membuat, menyediakan, dan mengelola kendaraan](#).
- Setelah kampanye dibuat, Anda harus menyetujui kampanye. Untuk informasi selengkapnya, lihat [Langkah 5: Menyebarkan kampanye](#).

Langkah 1: Konfigurasi kampanye

Secara umum informasi, lakukan hal berikut:

1. Masukkan nama untuk kampanye.
2. (Opsional) Masukkan deskripsi.

Konfigurasi skema pengumpulan data kampanye. Skema pengumpulan data memberikan instruksi perangkat lunak Edge Agent tentang data apa yang harus dikumpulkan atau kapan harus mengumpulkannya. Di FleetWise konsol AWS IoT, Anda dapat mengonfigurasi skema pengumpulan data dengan cara berikut:

- Tentukan skema pengumpulan data secara manual.
- Unggah file untuk secara otomatis menentukan skema pengumpulan data.

Di opsi Konfigurasi, pilih salah satu dari berikut ini:

- Untuk menentukan jenis skema pengumpulan data secara manual dan menentukan opsi untuk menyesuaikan skema, pilih Tentukan skema pengumpulan data.

Tentukan jenis skema pengumpulan data secara manual dan tentukan opsi untuk menyesuaikan skema.

1. Di bagian Rincian skema pengumpulan data, pilih jenis skema pengumpulan data yang ingin digunakan kampanye ini. Untuk menggunakan ekspresi logis untuk mengenali data kendaraan apa yang akan dikumpulkan, pilih Berbasis kondisi. Untuk menggunakan periode waktu tertentu untuk memutuskan seberapa sering mengumpulkan data kendaraan, pilih Berbasis waktu.
2. Tentukan durasi waktu kampanye mengumpulkan data.

 Note

Secara default, kampanye yang disetujui segera diaktifkan dan tidak memiliki waktu akhir yang ditetapkan. Untuk menghindari biaya tambahan, Anda harus menentukan rentang waktu.

3. Jika Anda menentukan skema pengumpulan data berbasis kondisi, Anda harus menentukan ekspresi logis untuk mengenali data apa yang akan dikumpulkan. AWS IoT FleetWise menggunakan ekspresi logis untuk mengenali data apa yang akan dikumpulkan untuk skema berbasis kondisi. Ekspresi harus menentukan nama sinyal yang sepenuhnya memenuhi syarat sebagai variabel, operator perbandingan, dan nilai perbandingan.

Misalnya, jika Anda menentukan `$variable.`myVehicle.InVehicleTemperature` > 50.0` ekspresi, AWS IoT FleetWise mengumpulkan nilai suhu yang lebih besar dari 50,0. Untuk petunjuk tentang cara menulis ekspresi, lihat [Ekspresi logis untuk kampanye](#).

Masukkan ekspresi logis yang digunakan untuk mengenali data apa yang akan dikumpulkan.

4. (Opsional) Anda dapat menentukan versi bahasa dari ekspresi bersyarat. Nilai default adalah 1.
5. (Opsional) Anda dapat menentukan interval pemicu minimum, yang merupakan durasi waktu terkecil antara dua peristiwa pengumpulan data. Misalnya, jika sinyal sering berubah, Anda mungkin ingin mengumpulkan data dengan kecepatan yang lebih lambat.
6. Tentukan kondisi mode Pemicu untuk perangkat lunak Edge Agent untuk mengumpulkan data. Secara default, Edge Agent untuk FleetWise perangkat lunak AWS IoT Selalu mengumpulkan data setiap kali kondisi terpenuhi. Atau, dapat mengumpulkan data hanya ketika kondisi terpenuhi untuk pertama kalinya, Pada pemicu pertama.
7. Jika Anda menentukan skema pengumpulan data berbasis waktu, Anda harus menentukan Periode waktu, dalam milidetik, dari 10.000 - 60.000 milidetik. Perangkat lunak Edge Agent menggunakan periode waktu untuk memutuskan seberapa sering mengumpulkan data.
8. (Opsional) Anda dapat mengedit opsi skema lanjutan skema.
 - a. Untuk menghemat bandwidth nirkabel dan mengurangi lalu lintas jaringan dengan mengompresi data, pilih Snappy.
 - b. (Opsional) Untuk menentukan berapa lama, dalam milidetik, untuk melanjutkan pengumpulan data setelah peristiwa pengumpulan data, Anda dapat menentukan durasi pengumpulan pemicu Post.
 - c. (Opsional) Untuk menunjukkan tingkat prioritas kampanye, Anda dapat menentukan Prioritas kampanye. Kampanye dengan jumlah prioritas yang lebih kecil diterapkan terlebih dahulu dan dianggap memiliki prioritas yang lebih tinggi.
 - d. Perangkat lunak Edge Agent dapat menyimpan data sementara secara lokal ketika kendaraan tidak terhubung ke cloud. Setelah koneksi dibangun kembali, data yang disimpan secara lokal ditransfer secara otomatis ke cloud. Tentukan apakah Anda ingin Agen Edge Menyimpan data secara lokal selama koneksi terputus.
 - e. (Opsional) Untuk memberikan informasi tambahan untuk sinyal, tambahkan hingga lima atribut sebagai dimensi data tambahan.

- Untuk mengunggah file untuk menentukan skema pengumpulan data, pilih Unggah file.json dari perangkat lokal Anda. AWS IoT FleetWise secara otomatis menentukan opsi mana yang dapat Anda tentukan dalam file. Anda dapat meninjau dan memperbarui opsi yang dipilih.

Unggah file.json dengan detail tentang skema pengumpulan data.

1. Untuk mengimpor informasi tentang skema pengumpulan data, pilih Pilih file. Untuk informasi selengkapnya tentang format file yang diperlukan, lihat dokumentasi [CreateCampaignAPI](#).

 Note

AWSIoT FleetWise saat ini mendukung ekstensi format file.json.

2. AWSIoT FleetWise secara otomatis mendefinisikan skema pengumpulan data berdasarkan informasi dalam file Anda. Tinjau opsi yang AWS IoT FleetWise pilih untuk Anda. Anda dapat memperbarui opsi, jika diperlukan.

Tentukan sinyal

Anda dapat menentukan sinyal untuk mengumpulkan data dari saat skema pengumpulan data dipanggil.

 Important

Sinyal yang digunakan dalam ekspresi untuk skema pengumpulan berbasis kondisi harus ditentukan dalam bidang ini.

Untuk menentukan sinyal untuk mengumpulkan data dari

1. Cari Nama sinyal yang sepenuhnya memenuhi syarat.

 Note

Nama sinyal yang sepenuhnya memenuhi syarat adalah jalur ke sinyal ditambah nama sinyal. Gunakan titik (.) untuk merujuk ke sinyal anak.

Misalnya,

`Vehicle.Chassis.SteeringWheel.HandsOff.HandsOffSteeringState` adalah

nama yang sepenuhnya memenuhi syarat untuk `HandsOffSteeringState` aktuator. `Vehicle.Chassis.SteeringWheel.HandsOff` adalah jalan menuju aktuator ini.

2. (Opsional) Untuk jumlah sampel Maks, masukkan jumlah maksimum sampel data yang dikumpulkan dan ditransfer oleh perangkat lunak Edge Agent ke cloud saat skema pengumpulan data dipanggil.
3. (Opsional) Untuk interval pengambilan sampel Min, masukkan durasi waktu minimum antara dua peristiwa pengumpulan sampel data, dalam milidetik. Jika sinyal sering berubah, Anda dapat menggunakan parameter ini untuk mengumpulkan data pada tingkat yang lebih lambat.
4. Untuk menambahkan sinyal lain, pilih Tambahkan lebih banyak sinyal. Anda dapat menambahkan hingga 999 sinyal.
5. Pilih Selanjutnya.

Langkah 2: Tentukan tujuan penyimpanan

Note

Anda hanya dapat mentransfer data kendaraan ke Amazon S3 jika kampanye berisi sinyal data sistem penglihatan.

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Pilih tujuan tempat Anda ingin menyimpan data yang dikumpulkan oleh kampanye. Anda dapat mentransfer data kendaraan ke Amazon S3 atau Amazon Timestream.

Di Pengaturan tujuan, lakukan hal berikut:

- Pilih S3 atau Timestream dari daftar dropdown.

Untuk menyimpan data kendaraan dalam bucket S3, pilih Amazon S3. S3 adalah layanan penyimpanan objek yang menyimpan data sebagai objek dalam ember. Untuk informasi selengkapnya, lihat [Membuat, mengonfigurasi, dan bekerja dengan bucket Amazon S3](#) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

S3 mengoptimalkan biaya penyimpanan data dan menyediakan mekanisme tambahan untuk menggunakan data kendaraan, seperti data lake, penyimpanan data terpusat, pipa pemrosesan data, dan analitik. Anda dapat menggunakan S3 untuk menyimpan data untuk pemrosesan dan analisis

batch. Misalnya, Anda dapat membuat laporan peristiwa pengereman keras untuk model machine learning (ML) Anda. Data kendaraan yang masuk disangga selama 10 menit sebelum pengiriman.

Amazon S3

Important

Anda hanya dapat mentransfer data ke S3 jika AWS FleetWise IoT memiliki izin untuk menulis ke dalam bucket S3. Untuk informasi selengkapnya tentang pemberian akses, lihat [Mengontrol akses dengan AWS IoT FleetWise](#).

Di pengaturan tujuan S3, lakukan hal berikut:

1. Untuk bucket S3, pilih bucket yang AWS IoT FleetWise memiliki izin.
2. (Opsional) Masukkan awalan khusus yang dapat Anda gunakan untuk mengatur data yang disimpan di bucket S3.
3. Pilih format output, yang merupakan file format yang disimpan seperti pada bucket S3.
4. Pilih apakah Anda ingin mengompres data yang disimpan di bucket S3 sebagai file.gzip. Kami merekomendasikan mengompresi data karena meminimalkan biaya penyimpanan.
5. Opsi yang Anda pilih di pengaturan tujuan S3 mengubah URI objek Contoh S3. Ini adalah contoh file apa yang disimpan seperti di S3.

Untuk menyimpan data kendaraan dalam tabel Timestream, pilih Amazon Timestream. Anda dapat menggunakan Timestream untuk menanyakan data kendaraan sehingga Anda dapat mengidentifikasi tren dan pola. Misalnya, Anda dapat menggunakan Timestream untuk membuat alarm untuk tingkat bahan bakar kendaraan. Data kendaraan yang masuk ditransfer ke Timestream dalam waktu dekat. Untuk informasi selengkapnya, lihat [Apa itu Amazon Timestream?](#) di Panduan Pengembang Amazon Timestream.

Amazon Timestream

Important

Anda hanya dapat mentransfer data ke tabel jika AWS IoT FleetWise memiliki izin untuk menulis data ke Timestream. Untuk informasi selengkapnya tentang pemberian akses, lihat [Mengontrol akses dengan AWS IoT FleetWise](#).

Dalam pengaturan tabel Timestream, lakukan hal berikut:

1. Untuk nama database Timestream, pilih nama database Timestream Anda dari daftar dropdown.
2. Untuk nama tabel Timestream, pilih nama tabel Timestream Anda dari daftar dropdown.

Dalam akses Layanan untuk Timestream, lakukan hal berikut:

- Pilih peran IAM dari daftar dropdown.
- Pilih Selanjutnya.

Langkah 3: Tambahkan kendaraan

Untuk memilih kendaraan mana yang akan digunakan kampanye Anda, pilih di daftar kendaraan. Filter kendaraan dengan mencari atribut dan nilainya yang Anda tambahkan saat membuat kendaraan, atau dengan nama kendaraan.

Di kendaraan Filter, lakukan hal berikut:

1. Di kotak pencarian, temukan atribut atau nama kendaraan dan pilih dari daftar.

Note

Setiap atribut hanya dapat digunakan sekali.

2. Masukkan nilai atribut atau nama kendaraan yang ingin Anda gunakan untuk kampanye. Misalnya, jika nama atribut yang sepenuhnya memenuhi syarat adalah `fuelType`, masukkan `gasoline` sebagai nilainya.
3. Untuk mencari atribut kendaraan lain, ulangi langkah sebelumnya. Anda dapat mencari hingga lima atribut kendaraan dan jumlah nama kendaraan yang tidak terbatas.
4. Kendaraan yang cocok dengan pencarian Anda tercantum di bawah nama Kendaraan. Pilih kendaraan yang Anda inginkan untuk disebar oleh kampanye.

Note

Hingga 100 kendaraan ditampilkan di hasil pencarian. Pilih Pilih semua untuk menambahkan semua kendaraan ke kampanye.

5. Pilih Selanjutnya.

Langkah 4: Tinjau dan buat

Verifikasi konfigurasi untuk kampanye, lalu pilih Buat kampanye.

Note

Setelah kampanye dibuat, Anda atau tim Anda harus menyebarkan kampanye ke kendaraan.

Langkah 5: Menyebarkan kampanye

Setelah Anda membuat kampanye, Anda atau tim Anda harus menyebarkan kampanye ke kendaraan.

Untuk menyebarkan kampanye

1. Pada halaman Ringkasan kampanye, pilih Terapkan.
2. Tinjau dan konfirmasi bahwa Anda ingin memulai penyebaran dan mulai mengumpulkan data dari kendaraan yang terhubung ke kampanye.
3. Pilih Deploy.

Jika Anda ingin menunda pengumpulan data dari kendaraan yang terhubung ke kampanye, pada halaman Ringkasan kampanye, pilih Tangguhkan. Untuk melanjutkan pengumpulan data dari kendaraan yang terhubung ke kampanye, pilih Lanjutkan.

Buat kampanye (AWS CLI)

Anda dapat menggunakan operasi [CreateCampaign](#) API untuk membuat kampanye. Contoh berikut menggunakan AWS CLI.

Saat Anda membuat kampanye, data yang dikumpulkan dari kendaraan dapat disimpan di Amazon S3 (S3) atau Amazon Timestream. Pilih Timestream untuk database deret waktu yang cepat, dapat diskalakan, dan tanpa server, seperti untuk menyimpan data yang memerlukan pemrosesan hampir waktu nyata. Pilih S3 untuk penyimpanan objek dengan skalabilitas, ketersediaan data, keamanan, dan kinerja terdepan di industri.

⚠ Important

Anda hanya dapat mentransfer data kendaraan jika AWS IoT FleetWise memiliki izin untuk menulis data ke S3 atau Timestream. Untuk informasi selengkapnya tentang pemberian akses, lihat [Mengontrol akses dengan AWS IoT FleetWise](#).

Buat kampanye

⚠ Important

- Anda harus memiliki katalog sinyal dan kendaraan atau armada sebelum Anda membuat kampanye. Untuk informasi selengkapnya, lihat [Membuat dan mengelola katalog sinyal](#), [Membuat, menyediakan, dan mengelola kendaraan](#), dan [Membuat dan mengelola armada](#).
- Setelah kampanye dibuat, Anda harus menggunakan operasi UpdateCampaign API untuk menyetujui kampanye. Untuk informasi selengkapnya, silakan lihat [Memperbarui kampanye \(AWS CLI\)](#)

Untuk membuat kampanye, jalankan perintah berikut.

Ganti *nama file* dengan nama file JSON yang berisi konfigurasi kampanye.

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

- Ganti *nama-kampanye* dengan nama kampanye yang Anda buat.
- Ganti *signal-catalog-arn* dengan Nama Sumber Daya Amazon (ARN) dari katalog sinyal.
- Ganti *target-arn* dengan ARN armada atau kendaraan yang Anda buat.
- Ganti *bucket-arn* dengan ARN bucket S3.

```
{
  "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"
        }
    }
]
}

```

- Ganti *nama-kampanye* dengan nama kampanye yang Anda buat.
- Ganti *signal-catalog-arn* dengan Nama Sumber Daya Amazon (ARN) dari katalog sinyal.
- Ganti *target-arn* dengan ARN armada atau kendaraan yang Anda buat.
- Ganti *role-arn* dengan ARN dari peran eksekusi tugas yang memberikan FleetWise izin AWS IoT untuk mengirimkan data ke tabel Timestream.
- Ganti *table-arn* dengan ARN dari tabel Timestream.

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

Ekspresi logis untuk kampanye

AWS IoT FleetWise menggunakan ekspresi logis untuk mengenali data apa yang akan dikumpulkan sebagai bagian dari kampanye. Untuk informasi selengkapnya tentang ekspresi, lihat [Ekspresi](#) dalam Panduan AWS IoT Events Pengembang.

Variabel ekspresi harus dibangun untuk mematuhi aturan untuk jenis data yang dikumpulkan. Untuk data sistem telemetri, variabel ekspresi harus menjadi nama sinyal yang sepenuhnya memenuhi syarat. Untuk data sistem visi, ekspresi menggabungkan nama sinyal yang sepenuhnya memenuhi syarat dengan jalur yang mengarah dari tipe data sinyal ke salah satu propertinya.

Misalnya, jika katalog sinyal berisi node berikut:

```
{
  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"
}
```

Jika node mengikuti definisi ROS 2:

```
{
  Vehicle.ADAS.CameraStruct.msg:
    boolean obstaclesExists
    uint8[] image
    Obstacle[30] obstacles
}
{
  Vehicle.ADAS.Obstacle.msg:
    float32: probability
    uint8 o_type
    float32: distance
}
```

Berikut ini adalah semua variabel ekspresi peristiwa yang mungkin:

```
{
```

```
...
  $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`
}
```

Memperbarui kampanye (AWS CLI)

Anda dapat menggunakan operasi [UpdateCampaign](#) API untuk memperbarui kampanye yang ada. Perintah berikut menggunakan AWS CLI.

- Ganti *nama-kampanye* dengan nama kampanye yang Anda perbarui.
- Ganti *tindakan* dengan salah satu dari berikut ini:
 - APPROVE— Menyetujui kampanye untuk memungkinkan AWS FleetWise IoT menyebarkannya ke kendaraan atau armada.
 - SUSPEND— Menangguhkan kampanye. Kampanye dihapus dari kendaraan dan semua kendaraan dalam kampanye yang ditangguhkan akan berhenti mengirim data.
 - RESUME— Mengaktifkan kembali kampanye. SUSPEND Kampanye ini dipindahkan ke semua kendaraan dan kendaraan akan melanjutkan pengiriman data.
 - UPDATE— Memperbarui kampanye dengan mendefinisikan atribut dan mengaitkannya dengan sinyal.

```
aws iotfleetwise update-campaign \  
    --name campaign-name \  
    --action action
```

Menghapus kampanye

Anda dapat menggunakan FleetWise konsol AWS IoT atau API untuk menghapus kampanye.

Menghapus kampanye (konsol)

Untuk menghapus kampanye, gunakan konsol AWS IoT FleetWise .

Untuk menghapus kampanye

1. Arahkan ke konsol [AWSIoT FleetWise](#).
2. Pada panel navigasi, pilih Kampanye.
3. Pada halaman Kampanye, pilih kampanye target.
4. Pilih Hapus.
5. Di Hapus **campaign-name?** , masukkan nama kampanye yang akan dihapus, lalu pilih Konfirmasi.

Menghapus kampanye (AWS CLI)

Anda dapat menggunakan operasi [DeleteCampaign](#) API untuk menghapus kampanye. Contoh berikut menggunakan AWS CLI.

Untuk menghapus kampanye, jalankan perintah berikut.

Ganti *nama-kampanye* dengan nama kendaraan yang Anda hapus.

```
aws iotfleetwise delete-campaign --name campaign-name
```

Dapatkan informasi kampanye (AWS CLI)

Anda dapat menggunakan operasi [ListCampaigns](#) API untuk memverifikasi apakah kampanye telah dihapus. Contoh berikut menggunakan AWS CLI.

Untuk mengambil daftar ringkasan paginasi untuk semua kampanye, jalankan perintah berikut.

```
aws iotfleetwise list-campaigns
```

Anda dapat menggunakan operasi [GetCampaign](#) API untuk mengambil informasi kendaraan. Contoh berikut menggunakan AWS CLI.

Untuk mengambil metadata kampanye, jalankan perintah berikut.

Ganti *nama-kampanye* dengan nama kampanye yang ingin Anda ambil.

```
aws iotfleetwise get-campaign --name campaign-name
```

 Note

Operasi ini [pada akhirnya konsisten](#). Dengan kata lain, perubahan kampanye mungkin tidak segera tercermin.

Memproses dan memvisualisasikan data kendaraan

Edge Agent untuk FleetWise perangkat lunak AWS IoT mentransfer data kendaraan yang dipilih ke Amazon Timestream atau Amazon Simple Storage Service (Amazon S3). Setelah data Anda tiba di tujuan data, Anda dapat menggunakan AWS layanan lain untuk memvisualisasikan dan membagikannya.

Memproses data kendaraan di Timestream

Timestream adalah database deret waktu yang dikelola sepenuhnya yang dapat menyimpan dan menganalisis triliunan titik data deret waktu per hari. Data Anda disimpan dalam tabel Timestream yang dikelola pelanggan. Anda dapat menggunakan Timestream untuk menanyakan data kendaraan sehingga Anda dapat memperoleh wawasan tentang kendaraan Anda. Untuk informasi selengkapnya, lihat [Apa itu Amazon Timestream?](#)

Skema default data yang ditransfer ke Timestream berisi bidang berikut.

Nama bidang	Tipe data	Deskripsi
eventId	varchar	ID acara pengumpulan data.
vehicleName	varchar	ID kendaraan dari mana data dikumpulkan.
name	varchar	Nama kampanye yang digunakan perangkat lunak Edge Agent untuk mengumpulkan data.
time	stempel waktu	Stempel waktu titik data.
measure_name	varchar	Nama sinyalnya.

Nama bidang	Tipe data	Deskripsi
measure_value::bigint	bigint	Nilai sinyal tipe Integer.
measure_value::double	double	Nilai sinyal tipe Double.
measure_value::boolean	boolean	Nilai sinyal tipe Boolean.

Memvisualisasikan data kendaraan yang disimpan di Timestream

Setelah data kendaraan Anda ditransfer ke Timestream, Anda dapat menggunakan AWS layanan berikut untuk memvisualisasikan, memantau, menganalisis, dan membagikan data Anda.

- Visualisasikan dan pantau data di dasbor menggunakan Grafana [atau Grafana Terkelola Amazon](#). Anda dapat memvisualisasikan data dari berbagai AWS sumber (seperti Amazon CloudWatch dan Timestream) dan sumber data lainnya dengan satu dasbor Grafana.
- [Analisis dan visualisasikan data di dasbor dengan menggunakan Amazon. QuickSight](#)

Memproses data kendaraan di S3

Amazon S3 adalah layanan penyimpanan objek yang menyimpan dan melindungi sejumlah data. Anda dapat menggunakan S3 untuk berbagai kasus penggunaan, seperti data lake, backup dan restore, arsip, aplikasi perusahaan, AWS IoT perangkat, dan analisis data besar. Data Anda disimpan di S3 sebagai objek dalam ember. Untuk informasi lebih lanjut, lihat [Apa itu Amazon S3?](#)

Skema default data yang ditransfer ke Amazon S3 berisi bidang berikut.

Nama bidang	Tipe data	Deskripsi
eventId	varchar	ID acara pengumpulan data.

Nama bidang	Tipe data	Deskripsi
vehicleName	varchar	ID kendaraan dari mana data dikumpulkan.
name	varchar	Nama kampanye yang digunakan perangkat lunak Edge Agent untuk mengumpulkan data.
time	stempel waktu	Stempel waktu titik data.
measure_name	varchar	Nama sinyalnya.
measure_value_BIGINT	bigint	Nilai sinyal tipe Integer.
measure_value_DOUBLE	double	Nilai sinyal tipe Double.
measure_value_BOOLEAN	boolean	Nilai sinyal tipe Boolean.
measure_value_STRUCT	struct	Nilai sinyal tipe Struct.

Format objek S3

AWS IoT FleetWise mentransfer data kendaraan ke S3 di mana ia disimpan sebagai objek. Anda dapat menggunakan URI objek yang secara unik mengidentifikasi data untuk menemukan data dari kampanye. Format URI objek S3 tergantung pada apakah data yang dikumpulkan adalah data yang tidak terstruktur atau diproses.

Data tidak terstruktur

Data tidak terstruktur disimpan dalam S3 dengan cara yang tidak ditentukan sebelumnya. Bisa dalam berbagai format, seperti gambar atau video.

Pesan kendaraan diteruskan ke AWS IoT FleetWise dengan data sinyal dari file Amazon Ion diterjemahkan dan ditransfer ke S3 sebagai objek. Objek S3 mewakili setiap sinyal dan dikodekan biner.

URI objek S3 data tidak terstruktur menggunakan format berikut:

```
s3://bucket-name/prefix/unstructured-data/random-ID-yyyy-MM-dd-HH-mm-ss-SSS-vehicleName-signalName-fieldName
```

Data yang diproses

Data yang diproses disimpan dalam S3 dan menjalani langkah-langkah pemrosesan yang memvalidasi, memperkaya, dan mengubah pesan. Daftar objek dan kecepatan adalah contoh data yang diproses.

Data yang ditransfer ke S3 disimpan sebagai objek yang mewakili catatan yang disangga untuk jangka waktu sekitar 10 menit. Secara default, AWS IoT FleetWise menambahkan awalan waktu UTC dalam format `year=YYYY/month=MM/date=DD/hour=HH` sebelum menulis objek ke S3. Awalan ini menciptakan hierarki logis di bucket di mana setiap garis miring maju (/) menciptakan level dalam hierarki. Data yang diproses juga berisi URI objek S3 ke data tidak terstruktur.

URI objek S3 data yang diproses menggunakan format berikut:

```
s3://bucket-name/prefix/processed-data/year=YYYY/month=MM/day=DD/hour=HH/part-0000-random-ID.gz.parquet
```

Data mentah

Data mentah, juga dikenal sebagai data primer, adalah data yang dikumpulkan dari file Amazon Ion. Anda dapat menggunakan data mentah untuk memecahkan masalah apa pun atau untuk melakukan root penyebab kesalahan.

URI objek S3 data mentah menggunakan format berikut:

```
s3://bucket-name/prefix/raw-data/vehicle-name/eventID-timestamp.10n
```

Menganalisis data kendaraan yang disimpan di S3

Setelah data kendaraan Anda ditransfer ke S3, Anda dapat menggunakan AWS layanan berikut untuk memantau, menganalisis, dan membagikan data Anda.

Ekstrak dan analisis data menggunakan Amazon SageMaker untuk alur kerja pelabelan hilir dan pembelajaran mesin (ML).

Untuk informasi selengkapnya, lihat topik berikut di Panduan SageMaker Pengembang Amazon:

- [Data proses](#)
- [Melatih model pembelajaran mesin](#)
- [Gambar Label](#)

Katalog data Anda menggunakan Perayap AWS Glue dan menganalisisnya di Amazon Athena. Secara default, objek yang ditulis ke S3 memiliki partisi waktu gaya Apache Hive, dengan jalur data yang berisi pasangan nilai kunci yang dihubungkan dengan tanda yang sama.

Untuk informasi selengkapnya, lihat topik berikut di Panduan Pengguna Amazon Athena:

- [Partisi data di Athena](#)
- [Menggunakan AWS Glue untuk terhubung ke sumber data di Amazon S3](#)
- [Praktik terbaik saat menggunakan Athena dengan AWS Glue](#)

Visualisasikan data menggunakan Amazon QuickSight dengan membaca tabel Athena atau bucket S3 Anda secara langsung.

Tip

Jika Anda membaca dari S3 secara langsung, konfirmasi bahwa data kendaraan Anda dalam format JSON karena Amazon QuickSight tidak mendukung format Apache Parquet.

Untuk informasi selengkapnya, lihat topik berikut di Panduan QuickSight Pengguna Amazon:

- [Sumber data yang didukung](#)
- [Membuat sumber data](#)

AWS CLI dan AWS SDK

Bagian ini menyediakan informasi tentang pembuatan AWS IoT FleetWise Permintaan API. Untuk informasi lebih lanjut tentang AWS IoT FleetWise [tipe operasi dan data](#), lihat AWS IoT FleetWise Referensi API.

Untuk menggunakan AWS IoT FleetWise dengan berbagai bahasa pemrograman, gunakan [AWS SDK](#), yang berisi fungsi otomatis berikut:

- Secara kriptografi menandatangani permintaan layanan Anda
- Mencoba kembali permintaan
- Menangani respons kesalahan

Untuk akses baris perintah, gunakan AWS IoT FleetWise dengan [AWS CLI](#). Anda dapat mengontrol AWS IoT FleetWise, dan layanan Anda yang lain, dari baris perintah, mengotomatiskan layanan melalui skrip.

Pemecahan Masalah AWS IoT FleetWise

Gunakan informasi dan solusi pemecahan masalah di bagian ini untuk membantu menyelesaikan masalah dengan IoT AWS. FleetWise

Informasi berikut dapat membantu Anda memecahkan masalah umum dengan AWS IoT. FleetWise

Topik

- [Masalah manifes decoder](#)
- [Edge Agent untuk AWS masalah perangkat lunak IoT FleetWise](#)

Masalah manifes decoder

Memecahkan masalah manifes decoder.

Mendiagnosis panggilan API manifes decoder

Kesalahan	Pedoman pemecahan masalah
<code>UpdateOperationFailure.ConflictingDecoderUpdate</code>	Manifes decoder yang sama memiliki beberapa permintaan pembaruan. Tunggu dan coba lagi.
<code>UpdateOperationFailure.InternalFailure</code>	InternalFailure diluncurkan sebagai pengecualian yang dikapsulasi. Masalahnya sendiri tergantung pada pengecualian yang dikapsulasi.
<code>UpdateOperationFailure.ActiveDecoderUpdate</code>	Manifes dekoder dalam Active keadaan dan tidak dapat diperbarui. Ubah status manifes decoder menjadi DRAFT, dan kemudian coba lagi.
<code>UpdateOperationFailure.ConflictingModelUpdate</code>	AWS IoT FleetWise mencoba memvalidasi terhadap model kendaraan (manifes model) yang sedang dimodifikasi oleh orang lain. Tunggu dan coba lagi.

Kesalahan	Pedoman pemecahan masalah
<pre>UpdateOperationFailure.Mode ManifestValidationResponse : FailureReason.MODEL_DATA_ENTRIES_NOT_FOUND</pre>	<p>Model kendaraan tidak memiliki sinyal yang terkait dengannya. Tambahkan sinyal ke model kendaraan dan verifikasi bahwa sinyal dapat ditemukan di katalog sinyal terkait.</p>
<pre>UpdateOperationFailure.Mode ManifestValidationResponse : FailureReason.MODEL_NOT_ACTIVE</pre>	<p>Perbarui model kendaraan sehingga dalam ACTIVE keadaan, dan kemudian coba lagi.</p>
<pre>UpdateOperationFailure.Mode ManifestValidationResponse : FailureReason.MODEL_NOT_FOUND</pre>	<p>AWSIoT tidak FleetWise dapat menemukan model kendaraan yang terkait dengan manifes decoder. Verifikasi Nama Sumber Daya Amazon (ARN) dari model kendaraan dan coba lagi.</p>
<pre>UpdateOperationFailure.Mode ManifestValidationResponse (FailureReason.MODEL_DATA_ENTRIES_READ_FAILURE</pre>	<p>Validasi model kendaraan gagal karena nama sinyal dari model kendaraan tidak ditemukan dalam katalog sinyal. Verifikasi bahwa sinyal dalam model kendaraan semuanya termasuk dalam katalog sinyal terkait.</p>
<pre>UpdateOperationFailure.ValidationFailure</pre>	<p>Sinyal atau antarmuka jaringan yang tidak valid ditemukan dalam permintaan untuk memperbarui manifes decoder. Verifikasi bahwa semua sinyal dan antarmuka jaringan yang dikembalikan oleh pengecualian ada, bahwa semua sinyal yang digunakan terkait dengan antarmuka yang tersedia, dan bahwa Anda tidak akan menghapus antarmuka yang memiliki sinyal yang terkait dengannya.</p>
<pre>UpdateOperationFailure.KmsKeyAccessDenied</pre>	<p>Ada masalah izin pada kunci AWS Key Management Service (AWS KMS) yang digunakan untuk operasi. Verifikasi bahwa Anda menggunakan peran yang memiliki akses ke kunci dan coba lagi.</p>

Kesalahan	Pedoman pemecahan masalah
<code>UpdateOperationFailure.DecoderDoesNotExist</code>	Manifes decoder tidak ada. Verifikasi nama manifes decoder dan coba lagi.

Pesan kesalahan data sistem visi dengan

`SIGNAL_DECODER_INCOMPATIBLE_WITH_SIGNAL_CATALOG` alasannya akan menyertakan petunjuk dalam respons yang memberikan informasi tentang mengapa permintaan gagal. Anda dapat menggunakan petunjuk untuk menentukan pedoman pemecahan masalah mana yang harus diikuti.

 Note

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Mendiagnosis validasi data sistem visi manifes decoder

Kesalahan	Pedoman pemecahan masalah
<code>InvalidSignalDecoder.withReason(SignalDecoderFailureReason.NO_SIGNAL_IN_CATALOG_FOR_DECODER_SIGNAL)</code>	AWSIoT FleetWise tidak menemukan struktur sinyal root yang digunakan dalam decoder sinyal menggunakan katalog sinyal. Verifikasi bahwa sinyal root struktur didefinisikan dengan benar dalam katalog sinyal.
<code>InvalidSignalDecoder.withReason(SignalDecoderFailureReason.SIGNAL_DECODER_TYPE_INCOMPATIBLE_WITH_MESSAGE_SIGNAL_TYPE)</code>	Pesan primitif dalam katalog sinyal tidak ditentukan dengan tipe data yang sama dalam permintaan pembaruan manifes decoder. Verifikasi bahwa pesan primitif yang ditentukan dalam permintaan cocok dengan definisi katalog sinyal yang sesuai.
<code>InvalidSignalDecoder.withReason(SignalDecoderFailureReason.STRUCT_SIZE_MISMATCH)</code>	Jumlah properti yang ditentukan dalam struct dalam katalog sinyal tidak cocok dengan jumlah properti yang Anda coba dekode dalam manifes decoder. Verifikasi bahwa Anda memiliki jumlah sinyal yang benar untuk

Kesalahan	Pedoman pemecahan masalah
<pre>InvalidSignalDecoder.withReason(SignalDecoderFailureReason.SIGNAL_DECODER_INCOMPATIBLE_WITH_SIGNAL_CATALOG)</pre>	<p>AWSIoT FleetWise menemukan sinyal yang didefinisikan sebagai STRUCT dalam katalog sinyal tanpa structuredMessageDefinition didefinisikan dalam permintaan manifes decoder. Pastikan bahwa setiap struct didefinisikan sebagai permintaan pembaruan structure dMessageDefinition manifes decoder.</p>
<pre>InvalidSignalDecoder.withReason(SignalDecoderFailureReason.SIGNAL_DECODER_INCOMPATIBLE_WITH_SIGNAL_CATALOG)</pre>	<p>Sinyal akar dari struktur yang digunakan dalam manifes decoder tidak didefinisikan dengan benar sebagai struktur dalam katalog sinyal. Struktur sinyal root yang digunakan dalam manifes decoder harus memiliki structFullyQualified nama bidangnya yang ditentukan. Itu juga membutuhkan simpul STRUCT dengan itu fullyQualifiedName.</p>
<pre>InvalidSignalDecoder.withReason(SignalDecoderFailureReason.SIGNAL_DECODER_INCOMPATIBLE_WITH_SIGNAL_CATALOG)</pre>	<p>Salah satu pesan daun yang digunakan dalam permintaan manifes decoder tidak didefinisikan sebagai pesan primitif. Verifikasi bahwa semua objek daun dalam permintaan didefinisikan sebagai pesan primitif.</p>
<pre>InvalidSignalDecoder.withReason(SignalDecoderFailureReason.SIGNAL_DECODER_INCOMPATIBLE_WITH_SIGNAL_CATALOG)</pre>	<p>Objek array dalam katalog sinyal tidak didefinisikan sebagai structuredMessageList Definisi dalam permintaan pembaruan manifes dekoder. Verifikasi bahwa semua properti array didefinisikan sebagai structuredMessageList Definisi dalam permintaan pembaruan manifes decoder.</p>

Edge Agent untuk AWS masalah perangkat lunak IoT FleetWise

Memecahkan masalah perangkat lunak Edge Agent.

Masalah

- [Masalah: Perangkat lunak Edge Agent tidak dimulai.](#)
- [Masalah: \[ERROR\] \[IoTFleetWiseEngine: :connect\]: \[Gagal memuat perpustakaan persistensi\]](#)
- [Masalah: Perangkat lunak Edge Agent tidak mengumpulkan PID diagnostik on-board \(OBD\) II dan kode masalah diagnostik \(DTC\).](#)
- [Masalah: Agen Edge untuk FleetWise perangkat lunak AWS IoT tidak mengumpulkan data dari jaringan atau tidak dapat menerapkan aturan pemeriksaan data.](#)
- [Masalah: \[ERROR\] \[AwsIotConnectivityModule: :connect\]: \[Koneksi gagal dengan kesalahan\] atau \[WARN\] \[AwsIotChannel: :send\]: \[Tidak ada Koneksi MQTT yang hidup.\]](#)

Masalah: Perangkat lunak Edge Agent tidak dimulai.

Anda mungkin melihat kesalahan berikut ketika perangkat lunak Edge Agent tidak dimulai.

- ```
Error from reader: * Line 1, Column 1
Syntax error: value, object or array expected.
```

Solusi: Pastikan Edge Agent untuk file konfigurasi FleetWise perangkat lunak AWS IoT menggunakan format JSON yang valid. Sebagai contoh, pastikan bahwa koma digunakan dengan benar. Untuk informasi lebih lanjut tentang file konfigurasi, lakukan hal berikut untuk mengunduh Edge Agent for AWS IoT FleetWise Software Developer Guide.

1. Arahkan ke konsol [AWS IoT FleetWise](#).
2. Di halaman beranda layanan, di FleetWise bagian Memulai dengan AWS IoT, pilih Explore Edge Agent.

- ```
[ERROR] [SocketCANBusChannel::connect]: [ SocketCan with name xxx is not accessible]
[ERROR] [IoTFleetWiseEngine::connect]: [ Failed to Bind Consumers to Producers ]
```

Solusi: Anda mungkin melihat kesalahan ini ketika perangkat lunak Edge Agent gagal membangun komunikasi soket dengan antarmuka jaringan yang ditentukan dalam file konfigurasi.

Untuk memeriksa apakah setiap antarmuka jaringan yang ditentukan dalam konfigurasi tersedia, jalankan perintah berikut.

```
ip link show
```

Untuk membawa antarmuka jaringan online, jalankan perintah berikut. Ganti *network-interface-id* dengan ID antarmuka jaringan.

```
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]
```

Solusi: Anda mungkin melihat kesalahan ini ketika perangkat lunak Edge Agent gagal membuat koneksi MQTT ke AWS IoT Core. Periksa apakah berikut ini dikonfigurasi dengan benar dan restart perangkat lunak Edge Agent.

- `mqtConnection::endpointUrl`— AWS titik akhir perangkat IoT akun.
- `mqtConnection::clientId`— ID kendaraan tempat perangkat lunak Edge Agent berjalan.
- `mqtConnection::certificateFilename`— Jalur ke file sertifikat kendaraan.
- `mqtConnection::privateKeyFilename`— Jalur ke file kunci pribadi kendaraan.
- Anda telah AWS IoT Core terbiasa menyediakan kendaraan. Untuk informasi selengkapnya, lihat [Kendaraan penyediaan](#).

Untuk informasi pemecahan masalah lainnya, lihat [AWS IoT Device SDK for C++Pertanyaan yang Sering Diajukan](#).

Masalah: [ERROR] [IoTFleetWiseEngine: :connect]: [Gagal memuat perpustakaan persistensi]

Solusi: Anda mungkin melihat kesalahan ini ketika perangkat lunak Edge Agent gagal menemukan penyimpanan persistensi. Periksa apakah berikut ini dikonfigurasi dengan benar dan restart perangkat lunak Edge Agent.

`persistency:persistencyPath`— Jalur lokal yang digunakan untuk mempertahankan skema pengumpulan, manifes decoder, dan snapshot data.

Masalah: Perangkat lunak Edge Agent tidak mengumpulkan PID diagnostik on-board (OBD) II dan kode masalah diagnostik (DTC).

Solusi: Anda mungkin melihat kesalahan ini jika `obdInterface:pidRequestIntervalSeconds` atau `obdInterface:dtcRequestIntervalSeconds` dikonfigurasi ke 0.

Jika perangkat lunak Edge Agent berjalan di kendaraan transmisi otomatis, pastikan `obdInterface:hasTransmissionEcu` dikonfigurasi untuk `true`.

Jika kendaraan Anda mendukung ID arbitrase Controller Area Network (CAN bus) yang diperluas, pastikan `obdInterface:useExtendedIds` sudah dikonfigurasi. `true`

Masalah: Agen Edge untuk FleetWise perangkat lunak AWS IoT tidak mengumpulkan data dari jaringan atau tidak dapat menerapkan aturan pemeriksaan data.

Solusi: Anda mungkin melihat kesalahan ini ketika kuota default dilanggar.

Sumber daya	Kuota	Dapat disesuaikan	Catatan
Nilai ID sinyal	ID sinyal harus kurang dari atau sama dengan 50.000	Ya	Perangkat lunak Edge Agent tidak akan mengumpulkan data dari sinyal yang memiliki ID lebih dari 50.000. Kami menyarankan Anda memeriksa berapa banyak sinyal yang terkandung dalam katalog sinyal sebelum Anda mengubah kuota ini.

Sumber daya	Kuota	Dapat disesuaikan	Catatan
Jumlah skema pengumpulan data aktif per kendaraan	256	Ya	Kami menyarankan Anda memeriksa berapa banyak kampanye yang telah Anda buat di cloud dan berapa banyak skema yang terkandung di setiap kampanye sebelum Anda mengubah kuota ini.
Ukuran buffer riwayat sinyal	20 MB	Ya	Jika kuota dilanggar, perangkat lunak Edge Agent berhenti mengumpulkan data baru.

Masalah: [ERROR] [AwsIotConnectivityModule: :connect]: [Koneksi gagal dengan kesalahan] atau [WARN] [AwsIotChannel: :send]: [Tidak ada Koneksi MQTT yang hidup.]

Solusi: Anda mungkin melihat kesalahan ini ketika perangkat lunak Edge Agent tidak terhubung ke cloud. Secara default, perangkat lunak Edge Agent mengirimkan permintaan ping ke AWS IoT Core setiap menit dan menunggu selama tiga menit. Jika tidak ada respons, perangkat lunak Edge Agent secara otomatis membangun kembali koneksi ke cloud.

Keamanan di AWS IoT FleetWise

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara AWS dan kamu. [Model tanggung jawab bersama](#) menjelaskan hal ini sebagai keamanan cloud dan keamanan dalam cloud:

- Keamanan Cloud — AWS Bertanggung jawab untuk melindungi infrastruktur yang berjalan AWS layanan di AWS Awan. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari [AWS Program Kepatuhan](#) . Untuk mempelajari tentang program kepatuhan yang berlaku AWS IoT FleetWise, lihat [AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan](#) .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan yang Anda gunakan. Anda juga bertanggung jawab atas faktor lain, yang mencakup kepekaan data Anda, persyaratan perusahaan, serta peraturan perundangan yang berlaku

Dokumentasi ini membantu Anda memahami cara menerapkan model tanggung jawab bersama saat menggunakan AWS IoT FleetWise. Ini menunjukkan kepada Anda cara mengkonfigurasi AWS IoT FleetWise untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan yang lain AWS layanan yang membantu Anda memantau dan mengamankan AWS Sumber daya IoT FleetWise .

Daftar Isi

- [Perlindungan data di AWS IoT FleetWise](#)
- [Mengontrol akses dengan AWS IoT FleetWise](#)
- [Identity and Access Management untuk AWS IoT FleetWise](#)
- [Validasi Kepatuhan untuk AWS IoT FleetWise](#)
- [Ketahanan di AWS IoT FleetWise](#)
- [Keamanan infrastruktur di AWS IoT FleetWise](#)
- [Analisis konfigurasi dan kerentanan di AWS IoT FleetWise](#)
- [Praktik terbaik keamanan untuk AWS IoT FleetWise](#)

Perlindungan data di AWS IoT FleetWise

Bagian AWS [model tanggung jawab bersama model](#) berlaku untuk perlindungan data di AWS IoT FleetWise. Seperti yang dijelaskan dalam model ini, AWS bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk menjaga kontrol atas konten Anda yang di-host di infrastruktur ini. Anda juga bertanggung jawab atas konfigurasi keamanan dan tugas manajemen untuk Layanan AWS yang Anda gunakan. Untuk informasi selengkapnya tentang privasi data, lihat [Privasi Data FAQ](#). Untuk informasi tentang perlindungan data di Eropa, lihat [AWS Model Tanggung Jawab Bersama dan posting GDPR](#) blog di AWS Blog Keamanan.

Untuk tujuan perlindungan data, kami menyarankan Anda untuk melindungi Akun AWS kredensi dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan otentikasi multi-faktor (MFA) dengan setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan AWS sumber daya. Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang menggunakan CloudTrail jalur untuk menangkap AWS kegiatan, lihat [Bekerja dengan CloudTrail jalan setapak](#) di AWS CloudTrail Panduan Pengguna.
- Gunakan AWS solusi enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan FIPS 140-3 modul kriptografi yang divalidasi saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan FIPS titik akhir. Untuk informasi selengkapnya tentang FIPS titik akhir yang tersedia, lihat [Federal Information Processing Standard \(FIPS\) 140-3](#).

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk ketika Anda bekerja dengan AWS IoT FleetWise atau lainnya Layanan AWS menggunakan konsol, API, AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log

penagihan atau log diagnostik. Jika Anda memberikan URL ke server eksternal, kami sangat menyarankan agar Anda tidak menyertakan informasi kredensial dalam URL untuk memvalidasi permintaan Anda ke server tersebut.

AWS IoT FleetWise dimaksudkan untuk digunakan dengan Agen Edge yang Anda kembangkan dan instal pada perangkat keras kendaraan yang didukung untuk mengirimkan data kendaraan ke AWS Awan. Mengekstrak data dari kendaraan mungkin tunduk pada peraturan privasi data di yurisdiksi tertentu. Sebelum menggunakan AWS IoT FleetWise dan menginstal Agen Edge Anda, kami sangat menyarankan Anda menilai kewajiban kepatuhan Anda berdasarkan hukum yang berlaku. Ini termasuk persyaratan hukum yang berlaku untuk memberikan pemberitahuan privasi yang memadai secara hukum dan mendapatkan persetujuan yang diperlukan untuk mengekstraksi data kendaraan.

Enkripsi diam

Data yang dikumpulkan dari kendaraan ditransmisikan ke cloud melalui AWS IoT Core pesan dengan protokol MQTT pesan. AWS IoT FleetWise mengirimkan data ke database Amazon Timestream Anda. Di Timestream, data Anda dienkripsi. Semua Layanan AWS mengenkripsi data saat istirahat secara default.

Enkripsi saat istirahat terintegrasi dengan AWS Key Management Service (AWS KMS) untuk mengelola kunci enkripsi yang digunakan untuk mengenkripsi data Anda. Anda dapat memilih untuk menggunakan kunci yang dikelola pelanggan untuk mengenkripsi data yang dikumpulkan oleh AWS IoT FleetWise. Anda dapat membuat, mengelola, dan melihat kunci enkripsi Anda melalui AWS KMS. Untuk informasi lebih lanjut, lihat [Apa itu AWS Key Management Service?](#) di AWS Key Management Service Panduan Pengembang.

Enkripsi bergerak

Semua data dipertukarkan dengan AWS IoT layanan dienkripsi dalam perjalanan dengan menggunakan Transport Layer Security (TLS). Untuk informasi selengkapnya, lihat [Keamanan transportasi](#) di AWS IoT Panduan Pengembang.

Juga, AWS IoT Core mendukung [otentikasi](#) dan [otorisasi](#) untuk membantu mengontrol akses ke AWS Sumber daya IoT FleetWise . Kendaraan dapat menggunakan sertifikat X.509 untuk mendapatkan otentikasi (masuk) untuk digunakan AWS IoT FleetWise dan penggunaan AWS IoT Core kebijakan untuk mendapatkan otorisasi (memiliki izin) untuk melakukan tindakan tertentu. Untuk informasi selengkapnya, lihat [the section called “Kendaraan penyediaan”](#).

Enkripsi data

Enkripsi data mengacu pada perlindungan data saat dalam perjalanan (saat bepergian ke dan dari AWS IoT FleetWise, dan antara gateway dan server), dan saat istirahat (saat disimpan di perangkat lokal atau di Layanan AWS). Anda dapat melindungi data saat istirahat menggunakan enkripsi sisi klien.

Note

AWS Ekspos pemrosesan FleetWise tepi IoT yang di-host di APIs dalamnya AWS FleetWise Gateway IoT dan dapat diakses melalui jaringan lokal. Ini APIs diekspos melalui TLS koneksi yang didukung oleh server-sertifikat yang dimiliki oleh AWS Konektor IoT FleetWise Edge. Untuk otentikasi klien, ini APIs menggunakan kata sandi kontrol akses. Server-certificate private-key dan password akses-kontrol keduanya disimpan pada disk. AWS Pemrosesan FleetWise tepi IoT bergantung pada enkripsi sistem file untuk keamanan kredensial ini saat istirahat.

Untuk informasi selengkapnya tentang enkripsi di sisi server dan enkripsi di sisi klien, tinjau topik berikut.

Daftar Isi

- [Enkripsi diam](#)
- [Manajemen kunci](#)

Enkripsi diam

AWS IoT FleetWise menyimpan data Anda di AWS Cloud dan di gateway.

Data saat istirahat di AWS Cloud

AWS IoT FleetWise menyimpan data di tempat lain Layanan AWS yang mengenkripsi data saat istirahat secara default. Enkripsi saat istirahat terintegrasi dengan [AWS Key Management Service \(AWS KMS\)](#) untuk mengelola kunci enkripsi yang digunakan untuk mengenkripsi nilai properti aset Anda dan nilai agregat di AWS IoT FleetWise. Anda dapat memilih untuk menggunakan kunci terkelola pelanggan untuk mengenkripsi nilai properti aset dan nilai agregat AWS IoT FleetWise. Anda dapat membuat, mengelola, dan melihat kunci enkripsi Anda melalui AWS KMS.

Anda dapat memilih sebuah Kunci milik AWS atau kunci yang dikelola pelanggan untuk mengenkripsi data Anda.

Cara kerjanya

Enkripsi saat istirahat terintegrasi dengan AWS KMS untuk mengelola kunci enkripsi yang digunakan untuk mengenkripsi data Anda.

- Kunci milik AWS — Kunci enkripsi default. AWS IoT FleetWise memiliki kunci ini. Anda tidak dapat melihat, mengelola, atau menggunakan kunci ini di Akun AWS. Anda juga tidak dapat melihat operasi pada kunci di AWS CloudTrail log. Anda dapat menggunakan kunci ini tanpa biaya tambahan.
- Kunci yang dikelola pelanggan — Kunci disimpan di akun Anda, yang Anda buat, miliki, dan kelola. Anda memiliki kendali penuh atas KMS kunci. Tambahan AWS KMS dikenakan biaya.

Kunci milik AWS

Kunci milik AWS tidak disimpan di akun Anda. Mereka adalah bagian dari kumpulan KMS kunci yang AWS memiliki dan mengelola untuk digunakan dalam banyak Akun AWS. Layanan AWS dapat menggunakan Kunci milik AWS untuk melindungi data Anda.

Anda tidak dapat melihat, mengelola, atau menggunakan Kunci milik AWS atau mengaudit penggunaannya. Namun, Anda tidak perlu mengambil tindakan apa pun atau mengubah program apa pun untuk melindungi kunci yang mengenkripsi data Anda.

Anda tidak akan dikenakan biaya jika Anda menggunakan Kunci milik AWS, dan mereka tidak menghitung AWS KMS kuota untuk akun Anda.

Kunci yang dikelola pelanggan

Kunci yang dikelola pelanggan adalah KMS kunci di akun Anda yang Anda buat, miliki, dan kelola. Anda memiliki kontrol penuh atas KMS kunci-kunci ini, seperti berikut ini:

- Menetapkan dan memelihara kebijakan, IAM kebijakan, dan hibah utama mereka
- Mengaktifkan dan menonaktifkannya
- Memutar materi kriptografi mereka
- Menambahkan tanda
- Membuat alias yang merujuk kepada mereka

- Menjadwalkan mereka untuk dihapus

Anda juga dapat menggunakan CloudTrail dan Amazon CloudWatch Logs untuk melacak permintaan yang AWS IoT mengirim FleetWise ke AWS KMS atas nama Anda.

Jika Anda menggunakan kunci yang dikelola pelanggan, Anda harus memberikan AWS FleetWise Akses IoT ke KMS kunci yang disimpan di akun Anda. AWS IoT FleetWise menggunakan enkripsi amplop dan hierarki kunci untuk mengenkripsi data. Klaster AWS KMS kunci enkripsi digunakan untuk mengenkripsi kunci root dari hierarki kunci ini. Untuk informasi selengkapnya, lihat [Enkripsi amplop](#) di AWS Key Management Service Panduan Pengembang.

Contoh hibah kebijakan berikut AWS FleetWise Izin IoT untuk membuat kunci terkelola pelanggan atas nama Anda.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmnt1603902045292",
      "Action": [
        "kms:GenerateDataKey*",
        "kms:Decrypt",
        "kms:DescribeKey",
        "kms:CreateGrant",
        "kms:RetireGrant",
        "kms:RevokeGrant"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

Important

Saat Anda menambahkan bagian baru ke kebijakan KMS utama Anda, jangan mengubah bagian yang ada dalam kebijakan. AWS IoT FleetWise tidak dapat melakukan operasi ke data Anda jika enkripsi diaktifkan untuk AWS IoT FleetWise dan salah satu dari berikut ini benar:

- KMSKuncinya dinonaktifkan atau dihapus.
- Kebijakan KMS kunci tidak dikonfigurasi dengan benar untuk layanan.

Menggunakan data sistem visi dengan enkripsi saat istirahat

Note

Data sistem visi dalam rilis pratinjau dan dapat berubah sewaktu-waktu.

Jika Anda memiliki enkripsi yang dikelola pelanggan dengan AWS KMS kunci diaktifkan pada AWS FleetWise Akun IoT, dan Anda ingin menggunakan data sistem visi, setel ulang pengaturan enkripsi Anda agar kompatibel dengan tipe data yang kompleks. Hal ini memungkinkan AWS IoT FleetWise untuk menetapkan izin tambahan yang diperlukan untuk data sistem visi.

Note

Manifes dekoder Anda dapat terjebak dalam status validasi jika Anda belum mengatur ulang pengaturan enkripsi untuk data sistem visi.

1. Gunakan [GetEncryptionConfiguration](#) API operasi untuk memeriksa apakah AWS KMS enkripsi diaktifkan. Tidak diperlukan tindakan lebih lanjut jika jenis enkripsi `FLEETWISE_DEFAULT_ENCRYPTION`.
2. Jika jenis enkripsi `KMS_BASED_ENCRYPTION`, gunakan [PutEncryptionConfiguration](#) API operasi untuk mengatur ulang jenis enkripsi ke `FLEETWISE_DEFAULT_ENCRYPTION`.

```
{
  aws iotfleetwise put-encryption-configuration --encryption-type
  FLEETWISE_DEFAULT_ENCRYPTION
}
```

3. Gunakan [PutEncryptionConfiguration](#) API operasi untuk mengaktifkan kembali jenis enkripsi ke `KMS_BASED_ENCRYPTION`.

```
{
```

```
aws iotfleetwise put-encryption-configuration \
  --encryption-type "KMS_BASED_ENCRYPTION"
  --kms-key-id kms_key_id
}
```

Untuk informasi selengkapnya tentang mengaktifkan enkripsi, lihat [Manajemen kunci](#).

Manajemen kunci

AWS Manajemen kunci FleetWise cloud IoT

Secara default, AWS Penggunaan IoT FleetWise Kunci yang dikelola AWS untuk melindungi data Anda di AWS Cloud. Anda dapat memperbarui pengaturan untuk menggunakan kunci terkelola pelanggan untuk mengenkripsi data di AWS IoT FleetWise. Anda dapat membuat, mengelola, dan melihat kunci enkripsi Anda melalui AWS Key Management Service (AWS KMS).

AWS IoT FleetWise mendukung enkripsi sisi server dengan kunci terkelola pelanggan yang disimpan di AWS KMS untuk mengenkripsi data untuk sumber daya berikut.

AWS Sumber daya IoT FleetWise	Tipe data	Bidang yang dienkripsi saat istirahat dengan kunci yang dikelola pelanggan
Katalog sinyal		deskripsi
	Atribut	deskripsi, allowedValues, defaultValue, min, maks
	Aktuator	deskripsi, allowedValues, min, maks
	Sensor	deskripsi, allowedValues, min, maks
Model kendaraan (manifes model)		deskripsi
Manifes dekoder		deskripsi
	CanInterface	protocolName, protocolVersion

AWS Sumber daya IoT FleetWise	Tipe data	Bidang yang dienkrpsi saat istirahat dengan kunci yang dikelola pelanggan
	ObdInterface	requestMessageId, dtcRequestInterval Detik, hasTransmissionEcu, obdStandard, pidRequestInterval Detik, useExtendedIds
	CanSignal	faktor, isBigEndian, isSigned, panjang, messageId, offset, startBit
	ObdSignal	byteLength, offset, pid, penskalaan pidResponseLength, serviceMode, startByte bitMaskLength bitRightShift
Kendaraan		atribut
Kampanye		deskripsi
	conditionBasedCollectionSkema	ekspresi, conditionLanguageVersion, minimumTriggerInterval Ms, triggerMode
	TimeBasedCollectionScheme	periodMs

Note

Data dan sumber daya lainnya dienkrpsi menggunakan enkripsi default dengan kunci yang dikelola oleh AWS IoT FleetWise. Kunci ini dibuat dan disimpan di AWS Akun IoT FleetWise .

Untuk informasi lebih lanjut, lihat [Apa itu AWS Key Management Service?](#) di AWS Key Management Service Panduan Pengembang.

Aktifkan enkripsi menggunakan KMS kunci (konsol)

Untuk menggunakan kunci yang dikelola pelanggan dengan AWS IoT FleetWise, Anda harus memperbarui AWS Pengaturan IoT FleetWise .

Untuk mengaktifkan enkripsi menggunakan KMS kunci (konsol)

1. Buka [AWS Konsol IoT FleetWise](#) .
2. Arahkan ke Pengaturan.
3. Di Enkripsi, pilih Edit untuk membuka halaman Edit enkripsi.
4. Untuk jenis kunci Enkripsi, pilih Pilih yang berbeda AWS KMS kunci. Ini memungkinkan enkripsi dengan kunci terkelola pelanggan yang disimpan di AWS KMS.

 Note

Anda hanya dapat menggunakan enkripsi kunci terkelola pelanggan untuk AWS Sumber daya IoT FleetWise . Ini termasuk katalog sinyal, model kendaraan (manifes model), manifes decoder, kendaraan, armada, dan kampanye.

5. Pilih KMS kunci Anda dengan salah satu opsi berikut:
 - Untuk menggunakan KMS kunci yang ada — Pilih alias KMS kunci Anda dari daftar.
 - Untuk membuat KMS kunci baru — Pilih Buat AWS KMS kunci.

 Note

Ini membuka AWS KMS konsol. Untuk informasi selengkapnya tentang membuat KMS kunci, lihat [Membuat kunci](#) di AWS Key Management Service Panduan Pengembang.

6. Pilih Simpan untuk memperbarui pengaturan Anda.

Aktifkan enkripsi menggunakan KMS kunci (AWS CLI)

Anda dapat menggunakan [PutEncryptionConfiguration](#) API operasi untuk mengaktifkan enkripsi AWS Akun IoT FleetWise . Contoh berikut menggunakan AWS CLI.

Untuk mengaktifkan enkripsi, jalankan perintah berikut.

- Ganti *KMS key id* dengan ID KMS kunci.

```
aws iotfleetwise put-encryption-configuration --kms-key-id KMS key id --encryption-type KMS_BASED_ENCRYPTION
```

Example response

```
{
  "kmsKeyId": "customer_kms_key_id",
  "encryptionStatus": "PENDING",
  "encryptionType": "KMS_BASED_ENCRYPTION"
}
```

Kebijakan kunci KMS

Setelah Anda membuat KMS kunci, Anda harus, setidaknya, menambahkan pernyataan berikut ke kebijakan KMS kunci Anda agar dapat berfungsi AWS IoT FleetWise.

```
{
  "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:Decrypt",
    "kms:DescribeKey",
    "kms:CreateGrant",
    "kms:RetireGrant",
    "kms:RevokeGrant"
  ],
  "Resource": "*"
}
```

Untuk informasi selengkapnya tentang mengedit kebijakan KMS utama untuk digunakan dengan AWS IoT FleetWise, lihat [Mengubah kebijakan utama](#) di AWS Key Management Service Panduan Pengembang.

Important

Saat Anda menambahkan bagian baru ke kebijakan KMS utama Anda, jangan mengubah bagian yang ada dalam kebijakan. AWS IoT FleetWise tidak dapat melakukan operasi ke data Anda jika enkripsi diaktifkan untuk AWS IoT FleetWise dan salah satu dari berikut ini benar:

- KMSKuncinya dinonaktifkan atau dihapus.
- Kebijakan KMS kunci tidak dikonfigurasi dengan benar untuk layanan.

Mengontrol akses dengan AWS IoT FleetWise

Bagian berikut mencakup cara mengontrol akses ke dan dari AWS IoT FleetWise sumber daya. Informasi yang mereka cakup mencakup cara memberikan akses aplikasi Anda AWS IoT FleetWise dapat mentransfer data kendaraan selama kampanye. Mereka juga menjelaskan bagaimana Anda dapat memberikan AWS IoT FleetWise akses ke bucket Amazon S3 (S3) atau database dan tabel Amazon Timestream untuk menyimpan data.

Teknologi untuk mengelola semua bentuk akses ini adalah AWS Identity and Access Management (IAM). Untuk informasi lebih lanjut tentang IAM, lihat [Apa itu IAM?](#) .

Daftar Isi

- [Pemberian Izin AWS IoT FleetWise akses ke tujuan Amazon S3](#)
- [Pemberian Izin AWS IoT FleetWise akses ke tujuan Amazon Timestream](#)

Pemberian Izin AWS IoT FleetWise akses ke tujuan Amazon S3

Saat Anda menggunakan tujuan Amazon S3, AWS IoT FleetWise mengirimkan data kendaraan ke bucket S3 Anda dan secara opsional dapat menggunakan AWS KMS kunci yang Anda miliki untuk enkripsi data. Jika pencatatan kesalahan diaktifkan, AWS IoT FleetWise juga mengirimkan kesalahan pengiriman data ke grup CloudWatch log dan aliran Anda. Anda harus memiliki IAM peran saat membuat aliran pengiriman.

AWS IoT FleetWise menggunakan kebijakan bucket dengan prinsip layanan untuk tujuan S3. Untuk informasi selengkapnya tentang menambahkan kebijakan bucket, lihat [Menambahkan kebijakan bucket menggunakan konsol Amazon S3](#) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

Gunakan kebijakan akses berikut untuk mengaktifkan AWS IoT FleetWise untuk mengakses bucket S3 Anda. Jika Anda tidak memiliki bucket S3, tambahkan `s3:PutObjectACL` ke daftar tindakan Amazon S3. Ini memberi pemilik ember akses penuh ke objek yang dikirim oleh AWS IoT FleetWise. Untuk informasi selengkapnya tentang cara mengamankan akses ke objek di bucket, lihat [contoh kebijakan Bucket](#) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

```

{
  "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": "account-id"
        }
      }
    }
  ]
}

```

Kebijakan bucket berikut adalah untuk semua kampanye di akun di AWS Wilayah.

```

{
  "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:account-id:campaign/*",
      "aws:SourceAccount": "account-id"
    }
  }
}
]
}

```

Jika Anda memiliki KMS kunci yang terpasang pada bucket S3 Anda, kunci tersebut memerlukan kebijakan berikut. Untuk informasi tentang manajemen kunci, lihat [Melindungi data menggunakan enkripsi sisi server dengan AWS Key Management Service keys \(SSE-KMS\)](#) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon.

```

{
  "Version": "2012-10-17",
  "Effect": "Allow",
  "Principal": {

```

```
"Service": "iotfleetwise.amazonaws.com",
},
"Action": [
  "kms:GenerateDataKey",
  "kms:Decrypt"
],
"Resource": "key-arn"
}
```

Important

Saat Anda membuat bucket, S3 akan membuat daftar kontrol akses default (ACL) yang memberi pemilik sumber daya kontrol penuh atas sumber daya. Jika AWS IoT tidak FleetWise dapat mengirimkan data ke S3, pastikan Anda menonaktifkan ACL pada bucket S3. Untuk informasi selengkapnya, lihat [Menonaktifkan ACLs semua bucket baru dan menerapkan Kepemilikan Objek di Panduan Pengguna](#) Layanan Penyimpanan Sederhana Amazon.

Pemberian Izin AWS IoT FleetWise akses ke tujuan Amazon Timestream

Saat Anda menggunakan tujuan Timestream, AWS IoT FleetWise mengirimkan data kendaraan ke tabel Timestream. Anda harus melampirkan kebijakan ke IAM peran untuk memungkinkan AWS IoT FleetWise untuk mengirim data ke Timestream.

Jika Anda menggunakan konsol untuk [membuat kampanye](#), AWS IoT FleetWise secara otomatis melampirkan kebijakan yang diperlukan ke peran tersebut.

Sebelum Anda mulai, periksa yang berikut ini:

Important

- Anda harus menggunakan yang sama AWS Wilayah saat Anda membuat sumber daya Timestream untuk AWS IoT FleetWise. Jika Anda beralih AWS Wilayah, Anda mungkin mengalami masalah saat mengakses sumber daya Timestream.
- AWS IoT FleetWise tersedia di AS Timur (Virginia N.) dan Eropa (Frankfurt).
- Untuk daftar Wilayah yang didukung, lihat [Titik akhir dan kuota Timestream](#) di Referensi Umum AWS.

- Anda harus memiliki database Timestream. Untuk tutorial, lihat [Membuat database di Panduan Pengembang Amazon Timestream](#).
- Anda harus memiliki tabel yang dibuat dalam database Timestream yang ditentukan. Untuk tutorial, lihat [Membuat tabel di Panduan Pengembang Amazon Timestream](#).

Anda dapat menggunakan AWS CLI untuk membuat IAM peran dengan kebijakan kepercayaan untuk Timestream. Untuk membuat IAM peran, jalankan perintah berikut.

Untuk membuat IAM peran dengan kebijakan kepercayaan

- Ganti *TimestreamExecutionRole* dengan nama peran yang Anda buat.
- Ganti *trust-policy* dengan JSON file yang berisi kebijakan kepercayaan.

```
aws iam create-role --role-name TimestreamExecutionRole --assume-role-policy-document file://trust-policy.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:account-id:campaign/campaign-name"
          ],
          "aws:SourceAccount": [
            "account-id"
          ]
        }
      }
    }
  ]
}
```

Membuat kebijakan izin untuk diberikan AWS FleetWise Izin IoT untuk menulis data ke Timestream. Untuk membuat kebijakan izin, jalankan perintah berikut.

Untuk membuat kebijakan izin

- Ganti *AWSIoT Fleetwise Access Timestream Permissions Policy* dengan nama kebijakan yang Anda buat.
- Ganti *permissions-policy* dengan nama JSON file yang berisi kebijakan izin.

```
aws iam create-policy --policy-name AWSIoT Fleetwise Access Timestream Permissions Policy --policy-document file://permissions-policy.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": "*"
    }
  ]
}
```

Untuk melampirkan kebijakan izin ke peran Anda IAM

1. Dari output, salin Amazon Resource Name (ARN) dari kebijakan izin.
2. Untuk melampirkan kebijakan IAM izin ke IAM peran Anda, jalankan perintah berikut.

- Ganti *permissions-policy-arn* dengan ARN yang Anda salin pada langkah sebelumnya.
- Ganti *TimestreamExecutionRole* dengan nama IAM peran yang Anda buat.

```
aws iam attach-role-policy --policy-arn permissions-policy-arn --role-name TimestreamExecutionRole
```

Untuk informasi selengkapnya, lihat Manajemen [akses untuk AWS sumber daya](#) dalam Panduan IAM Pengguna.

Identity and Access Management untuk AWS IoT FleetWise

AWS Identity and Access Management (IAM) adalah Layanan AWS yang membantu administrator mengontrol akses ke AWS sumber daya. IAM administrator mengontrol siapa yang dapat diautentikasi (masuk) dan diotorisasi (memiliki izin) untuk digunakan AWS Sumber daya IoT FleetWise . IAM adalah sebuah Layanan AWS yang dapat Anda gunakan tanpa biaya tambahan.

Topik

- [Audiens](#)
- [Mengautentikasi dengan identitas](#)
- [Mengelola akses menggunakan kebijakan](#)
- [Bagaimana AWS IoT bekerja FleetWise dengan IAM](#)
- [Contoh kebijakan berbasis identitas untuk AWS IoT FleetWise](#)
- [Pemecahan Masalah AWS FleetWise Identitas dan akses IoT](#)

Audiens

Bagaimana Anda menggunakan AWS Identity and Access Management (IAM) berbeda, tergantung pada pekerjaan yang Anda lakukan AWS IoT FleetWise.

Pengguna layanan — Jika Anda menggunakan AWS FleetWise Layanan IoT untuk melakukan pekerjaan Anda, maka administrator Anda memberi Anda kredensi dan izin yang Anda butuhkan. Saat Anda menggunakan lebih banyak AWS FleetWise Fitur IoT untuk melakukan pekerjaan Anda, Anda mungkin memerlukan izin tambahan. Memahami cara akses dikelola dapat membantu Anda

meminta izin yang tepat dari administrator Anda. Jika Anda tidak dapat mengakses fitur di AWS IoT FleetWise, lihat. [Pemecahan Masalah AWS FleetWise Identitas dan akses IoT](#)

Administrator layanan — Jika Anda bertanggung jawab AWS FleetWise Sumber daya IoT di perusahaan Anda, Anda mungkin memiliki akses penuh ke AWS IoT FleetWise. Adalah tugas Anda untuk menentukan yang AWS FleetWise Fitur dan sumber daya IoT yang harus diakses pengguna layanan Anda. Anda kemudian harus mengirimkan permintaan ke IAM administrator Anda untuk mengubah izin pengguna layanan Anda. Tinjau informasi di halaman ini untuk memahami konsep dasar IAM. Untuk mempelajari lebih lanjut tentang bagaimana perusahaan Anda dapat menggunakan IAM AWS IoT FleetWise, lihat. [Bagaimana AWS IoT bekerja FleetWise dengan IAM](#)

IAM administrator - Jika Anda seorang IAM administrator, Anda mungkin ingin mempelajari detail tentang cara menulis kebijakan untuk mengelola akses AWS IoT FleetWise. Untuk melihat contoh AWS Kebijakan FleetWise berbasis identitas IoT yang dapat Anda gunakan, lihat. IAM [Contoh kebijakan berbasis identitas untuk AWS IoT FleetWise](#)

Mengautentikasi dengan identitas

Otentikasi adalah cara Anda masuk AWS menggunakan kredensi identitas Anda. Anda harus diautentikasi (masuk ke AWS) sebagai Pengguna root akun AWS, sebagai IAM pengguna, atau dengan mengambil IAM peran.

Anda dapat masuk ke AWS sebagai identitas federasi dengan menggunakan kredensial yang disediakan melalui sumber identitas. AWS IAM Identity Center Pengguna (Pusat IAM Identitas), autentikasi masuk tunggal perusahaan Anda, dan kredensi Google atau Facebook Anda adalah contoh identitas federasi. Saat Anda masuk sebagai identitas federasi, administrator Anda sebelumnya menyiapkan federasi identitas menggunakan IAM peran. Saat Anda mengakses AWS dengan menggunakan federasi, Anda secara tidak langsung mengambil peran.

Tergantung pada jenis pengguna Anda, Anda dapat masuk ke AWS Management Console atau AWS portal akses. Untuk informasi lebih lanjut tentang masuk AWS, lihat [Cara masuk ke Akun AWS](#) di AWS Sign-In Panduan Pengguna.

Jika Anda mengakses AWS secara terprogram, AWS menyediakan kit pengembangan perangkat lunak (SDK) dan antarmuka baris perintah (CLI) untuk menandatangani permintaan Anda secara kriptografis dengan menggunakan kredensial Anda. Jika Anda tidak menggunakan AWS alat, Anda harus menandatangani permintaan sendiri. Untuk informasi selengkapnya tentang menggunakan metode yang disarankan untuk menandatangani permintaan sendiri, lihat [Menandatangani AWS API permintaan](#) di Panduan IAM Pengguna.

Apa pun metode autentikasi yang digunakan, Anda mungkin diminta untuk menyediakan informasi keamanan tambahan. Misalnya, AWS merekomendasikan agar Anda menggunakan otentikasi multi-faktor (MFA) untuk meningkatkan keamanan akun Anda. Untuk mempelajari selengkapnya, lihat [Autentikasi multi-faktor](#) di AWS IAM Identity Center Panduan Pengguna dan [Menggunakan otentikasi multi-faktor \(\) MFA di AWS](#) di Panduan Pengguna IAM.

Akun AWS pengguna root

Saat Anda membuat Akun AWS, Anda mulai dengan satu identitas masuk yang memiliki akses lengkap ke semua Layanan AWS dan sumber daya di akun. Identitas ini disebut Akun AWS pengguna root dan diakses dengan masuk dengan alamat email dan kata sandi yang Anda gunakan untuk membuat akun. Kami sangat menyarankan agar Anda tidak menggunakan pengguna root untuk tugas sehari-hari. Lindungi kredensial pengguna root Anda dan gunakan kredensial tersebut untuk melakukan tugas yang hanya dapat dilakukan pengguna root. Untuk daftar lengkap tugas yang mengharuskan Anda masuk sebagai pengguna root, lihat [Tugas yang memerlukan kredensi pengguna root](#) di IAMPanduan Pengguna.

Identitas gabungan

Sebagai praktik terbaik, mewajibkan pengguna manusia, termasuk pengguna yang memerlukan akses administrator, untuk menggunakan federasi dengan penyedia identitas untuk mengakses Layanan AWS dengan menggunakan kredensi sementara.

Identitas federasi adalah pengguna dari direktori pengguna perusahaan Anda, penyedia identitas web, AWS Directory Service, direktori Pusat Identitas, atau pengguna mana pun yang mengakses Layanan AWS dengan menggunakan kredensi yang disediakan melalui sumber identitas. Ketika akses identitas federasi Akun AWS, mereka mengambil peran, dan peran memberikan kredensyal sementara.

Untuk manajemen akses terpusat, kami sarankan Anda menggunakan AWS IAM Identity Center. Anda dapat membuat pengguna dan grup di Pusat IAM Identitas, atau Anda dapat menghubungkan dan menyinkronkan ke sekumpulan pengguna dan grup di sumber identitas Anda sendiri untuk digunakan di semua Akun AWS dan aplikasi. Untuk informasi tentang Pusat IAM Identitas, lihat [Apa itu Pusat IAM Identitas?](#) di AWS IAM Identity Center Panduan Pengguna.

Pengguna dan grup IAM

[IAMPengguna](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus untuk satu orang atau aplikasi. Jika memungkinkan, sebaiknya mengandalkan kredensi sementara daripada

membuat IAM pengguna yang memiliki kredensi jangka panjang seperti kata sandi dan kunci akses. Namun, jika Anda memiliki kasus penggunaan khusus yang memerlukan kredensi jangka panjang dengan IAM pengguna, kami sarankan Anda memutar kunci akses. Untuk informasi selengkapnya, lihat [Memutar kunci akses secara teratur untuk kasus penggunaan yang memerlukan kredensi jangka panjang](#) di IAMPanduan Pengguna.

[IAMGrup](#) adalah identitas yang menentukan kumpulan IAM pengguna. Anda tidak dapat masuk sebagai grup. Anda dapat menggunakan grup untuk menentukan izin bagi beberapa pengguna sekaligus. Grup mempermudah manajemen izin untuk sejumlah besar pengguna sekaligus. Misalnya, Anda dapat memiliki grup bernama IAMAdmins dan memberikan izin grup tersebut untuk mengelola sumber daya IAM.

Pengguna berbeda dari peran. Pengguna secara unik terkait dengan satu orang atau aplikasi, tetapi peran dimaksudkan untuk dapat digunakan oleh siapa pun yang membutuhkannya. Pengguna memiliki kredensial jangka panjang permanen, tetapi peran memberikan kredensial sementara. Untuk mempelajari lebih lanjut, lihat [Kapan membuat IAM pengguna \(bukan peran\)](#) di Panduan IAM Pengguna.

IAMperan

[IAMPeran](#) adalah identitas dalam diri Anda Akun AWS yang memiliki izin khusus. Ini mirip dengan IAM pengguna, tetapi tidak terkait dengan orang tertentu. Anda dapat mengambil IAM peran sementara dalam AWS Management Console dengan [beralih peran](#). Anda dapat mengambil peran dengan memanggil AWS CLI atau AWS API operasi atau dengan menggunakan kustom URL. Untuk informasi selengkapnya tentang metode penggunaan peran, lihat [Menggunakan IAM peran](#) di Panduan IAM Pengguna.

IAMperan dengan kredensi sementara berguna dalam situasi berikut:

- Akses pengguna terfederasi – Untuk menetapkan izin ke identitas terfederasi, Anda membuat peran dan menentukan izin untuk peran tersebut. Ketika identitas terfederasi mengotentikasi, identitas tersebut terhubung dengan peran dan diberi izin yang ditentukan oleh peran. Untuk informasi tentang peran untuk federasi, lihat [Membuat peran untuk Penyedia Identitas pihak ketiga](#) di Panduan IAM Pengguna. Jika Anda menggunakan Pusat IAM Identitas, Anda mengonfigurasi set izin. Untuk mengontrol apa yang dapat diakses identitas Anda setelah diautentikasi, Pusat IAM Identitas mengkorelasikan izin yang disetel ke peran. Untuk informasi tentang set izin, lihat [Set izin](#) di AWS IAM Identity Center Panduan Pengguna.
- Izin IAM pengguna sementara — IAM Pengguna atau peran dapat mengambil IAM peran untuk sementara mengambil izin yang berbeda untuk tugas tertentu.

- Akses lintas akun — Anda dapat menggunakan IAM peran untuk memungkinkan seseorang (prinsipal tepercaya) di akun lain mengakses sumber daya di akun Anda. Peran adalah cara utama untuk memberikan akses lintas akun. Namun, dengan beberapa Layanan AWS, Anda dapat melampirkan kebijakan langsung ke sumber daya (alih-alih menggunakan peran sebagai proxy). Untuk mempelajari perbedaan antara peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Akses sumber daya lintas akun di IAM](#) Panduan Pengguna. IAM
- Akses lintas layanan - Beberapa Layanan AWS menggunakan fitur di lain Layanan AWS. Misalnya, saat Anda melakukan panggilan dalam suatu layanan, biasanya layanan tersebut menjalankan aplikasi di Amazon EC2 atau menyimpan objek di Amazon S3. Sebuah layanan mungkin melakukannya menggunakan izin prinsipal yang memanggil, menggunakan peran layanan, atau peran terkait layanan.
- Teruskan sesi akses (FAS) — Saat Anda menggunakan IAM pengguna atau peran untuk melakukan tindakan AWS Anda dianggap sebagai kepala sekolah. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari prinsipal yang memanggil Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. FAS permintaan hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan saat membuat FAS permintaan, lihat [Meneruskan sesi akses](#).
- Peran layanan — Peran layanan adalah [IAM peran](#) yang diasumsikan layanan untuk melakukan tindakan atas nama Anda. IAM Administrator dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) di Panduan Pengguna IAM.
- Peran terkait layanan — Peran terkait layanan adalah jenis peran layanan yang ditautkan ke Layanan AWS. Layanan dapat mengambil peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Akun AWS dan dimiliki oleh layanan. IAM Administrator dapat melihat, tetapi tidak mengedit izin untuk peran terkait layanan.
- Aplikasi yang berjalan di Amazon EC2 — Anda dapat menggunakan IAM peran untuk mengelola kredensial sementara untuk aplikasi yang berjalan pada EC2 instance dan membuat AWS CLI atau AWS API permintaan. Ini lebih baik untuk menyimpan kunci akses dalam EC2 instance. Untuk menetapkan AWS peran ke sebuah EC2 instance dan membuatnya tersedia untuk semua aplikasinya, Anda membuat profil instance yang dilampirkan ke instance. Profil instance berisi peran dan memungkinkan program yang berjalan pada EC2 instance untuk mendapatkan kredensi

sementara. Untuk informasi selengkapnya, lihat [Menggunakan IAM peran untuk memberikan izin ke aplikasi yang berjalan di EC2 instans Amazon](#) di IAMPanduan Pengguna.

Untuk mempelajari apakah akan menggunakan IAM peran atau IAM pengguna, lihat [Kapan membuat IAM peran \(bukan pengguna\)](#) di Panduan IAM Pengguna.

Mengelola akses menggunakan kebijakan

Anda mengontrol akses di AWS dengan membuat kebijakan dan melampirkannya AWS identitas atau sumber daya. Kebijakan adalah objek di AWS bahwa, ketika dikaitkan dengan identitas atau sumber daya, mendefinisikan izin mereka. AWS mengevaluasi kebijakan ini ketika prinsipal (pengguna, pengguna root, atau sesi peran) membuat permintaan. Izin dalam kebijakan menentukan apakah permintaan diizinkan atau ditolak. Sebagian besar kebijakan disimpan di AWS sebagai JSON dokumen. Untuk informasi selengkapnya tentang struktur dan isi dokumen JSON kebijakan, lihat [Ringkasan JSON kebijakan](#) di Panduan IAM Pengguna.

Administrator dapat menggunakan AWS JSONkebijakan untuk menentukan siapa yang memiliki akses ke apa. Yaitu, principal dapat melakukan tindakan pada suatu sumber daya, dan dalam suatu syarat.

Secara default, pengguna dan peran tidak memiliki izin. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang mereka butuhkan, IAM administrator dapat membuat IAM kebijakan. Administrator kemudian dapat menambahkan IAM kebijakan ke peran, dan pengguna dapat mengambil peran.

IAMkebijakan menentukan izin untuk tindakan terlepas dari metode yang Anda gunakan untuk melakukan operasi. Misalnya, anggaplah Anda memiliki kebijakan yang mengizinkan tindakan `iam:GetRole`. Pengguna dengan kebijakan tersebut dapat memperoleh informasi peran dari AWS Management Console, AWS CLI, atau AWS API.

Kebijakan berbasis identitas

Kebijakan berbasis identitas adalah dokumen kebijakan JSON izin yang dapat Anda lampirkan ke identitas, seperti pengguna, grup IAM pengguna, atau peran. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Membuat IAM kebijakan di Panduan](#) Pengguna. IAM

Kebijakan berbasis identitas dapat dikategorikan lebih lanjut sebagai kebijakan inline atau kebijakan yang dikelola. Kebijakan inline disematkan langsung ke satu pengguna, grup, atau peran. Kebijakan terkelola adalah kebijakan mandiri yang dapat Anda lampirkan ke beberapa pengguna, grup, dan peran di Akun AWS. Kebijakan terkelola meliputi AWS kebijakan terkelola dan kebijakan yang dikelola pelanggan. Untuk mempelajari cara memilih antara kebijakan terkelola atau kebijakan sebaris, lihat [Memilih antara kebijakan terkelola dan kebijakan sebaris](#) di IAMPanduan Pengguna.

Kebijakan berbasis sumber daya

Kebijakan berbasis sumber daya adalah dokumen JSON kebijakan yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan IAM peran dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus [menentukan prinsipal](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau Layanan AWS.

Kebijakan berbasis sumber daya merupakan kebijakan inline yang terletak di layanan tersebut. Anda tidak dapat menggunakan AWS kebijakan terkelola dari IAM dalam kebijakan berbasis sumber daya.

Daftar kontrol akses (ACLs)

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan. JSON

Amazon S3, AWS WAF, dan Amazon VPC adalah contoh layanan yang mendukung ACLs. Untuk mempelajari selengkapnya ACLs, lihat [Ikhtisar daftar kontrol akses \(ACL\)](#) di Panduan Pengembang Layanan Penyimpanan Sederhana Amazon.

Jenis-jenis kebijakan lain

AWS mendukung jenis kebijakan tambahan yang kurang umum. Jenis-jenis kebijakan ini dapat mengatur izin maksimum yang diberikan kepada Anda oleh jenis kebijakan yang lebih umum.

- **Batas izin** — Batas izin adalah fitur lanjutan tempat Anda menetapkan izin maksimum yang dapat diberikan oleh kebijakan berbasis identitas kepada entitas (pengguna atau peran). IAM IAM Anda dapat menetapkan batasan izin untuk suatu entitas. Izin yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas milik entitas dan batasan izinnya. Kebijakan

berbasis sumber daya yang menentukan pengguna atau peran dalam bidang `Principal` tidak dibatasi oleh batasan izin. Penolakan eksplisit dalam salah satu kebijakan ini akan menggantikan pemberian izin. Untuk informasi selengkapnya tentang batas izin, lihat [Batas izin untuk IAM entitas](#) di [IAM Panduan Pengguna](#).

- Kebijakan kontrol layanan (SCPs) — SCPs adalah JSON kebijakan yang menentukan izin maksimum untuk organisasi atau unit organisasi (OU) di AWS Organizations. AWS Organizations adalah layanan untuk mengelompokkan dan mengelola beberapa Akun AWS yang dimiliki bisnis Anda. Jika Anda mengaktifkan semua fitur dalam suatu organisasi, maka Anda dapat menerapkan kebijakan kontrol layanan (SCPs) ke salah satu atau semua akun Anda. SCP Membatasi izin untuk entitas di akun anggota, termasuk masing-masing Pengguna root akun AWS. Untuk informasi selengkapnya tentang Organizations dan SCPs, lihat [Kebijakan kontrol layanan](#) di [AWS Organizations Panduan Pengguna](#).
- Kebijakan sesi – Kebijakan sesi adalah kebijakan lanjutan yang Anda berikan sebagai parameter ketika Anda membuat sesi sementara secara programatis untuk peran atau pengguna terfederasi. Izin sesi yang dihasilkan adalah perpotongan antara kebijakan berbasis identitas pengguna atau peran dan kebijakan sesi. Izin juga bisa datang dari kebijakan berbasis sumber daya. Penolakan secara tegas dalam salah satu kebijakan ini membatalkan izin. Untuk informasi selengkapnya, lihat [Kebijakan sesi](#) di [Panduan IAM Pengguna](#).

Berbagai jenis kebijakan

Ketika beberapa jenis kebijakan berlaku pada suatu permintaan, izin yang dihasilkan lebih rumit untuk dipahami. Untuk mempelajari caranya AWS menentukan apakah akan mengizinkan permintaan saat beberapa jenis kebijakan terlibat, lihat [Logika evaluasi kebijakan](#) di [Panduan IAM Pengguna](#).

Bagaimana AWS IoT bekerja FleetWise dengan IAM

Sebelum Anda gunakan IAM untuk mengelola akses ke AWS IoT FleetWise, pelajari IAM fitur apa yang tersedia untuk digunakan AWS IoT FleetWise.

IAM fitur yang dapat Anda gunakan dengan AWS IoT FleetWise

IAM fitur	AWS Dukungan IoT FleetWise
Kebijakan berbasis identitas	Ya

IAM fitur	AWS Dukungan IoT FleetWise
Kebijakan berbasis sumber daya	Tidak
Tindakan kebijakan	Ya
Sumber daya kebijakan	Ya
Kunci kondisi kebijakan	Ya
ACLs	Tidak
ABAC(tag dalam kebijakan)	Parsial
Kredensial sementara	Ya
Izin prinsipal	Ya
Peran layanan	Tidak
Peran terkait layanan	Tidak

Untuk mendapatkan pandangan tingkat tinggi tentang bagaimana AWS IoT FleetWise dan lainnya AWS layanan bekerja dengan sebagian besar IAM fitur, lihat [AWS layanan yang bekerja dengan IAM](#) dalam Panduan IAM Pengguna.

Kebijakan berbasis identitas untuk AWS IoT FleetWise

Mendukung kebijakan berbasis identitas: Ya

Kebijakan berbasis identitas adalah dokumen kebijakan JSON izin yang dapat Anda lampirkan ke identitas, seperti pengguna, grup IAM pengguna, atau peran. Kebijakan ini mengontrol jenis tindakan yang dapat dilakukan oleh pengguna dan peran, di sumber daya mana, dan berdasarkan kondisi seperti apa. Untuk mempelajari cara membuat kebijakan berbasis identitas, lihat [Membuat IAM kebijakan di Panduan](#) Pengguna. IAM

Dengan kebijakan IAM berbasis identitas, Anda dapat menentukan tindakan dan sumber daya yang diizinkan atau ditolak serta kondisi di mana tindakan diizinkan atau ditolak. Anda tidak dapat menentukan secara spesifik prinsipal dalam sebuah kebijakan berbasis identitas karena prinsipal berlaku bagi pengguna atau peran yang melekat kepadanya. Untuk mempelajari semua elemen yang

dapat Anda gunakan dalam JSON kebijakan, lihat [referensi elemen IAM JSON kebijakan](#) di Panduan IAM Pengguna.

Contoh kebijakan berbasis identitas untuk AWS IoT FleetWise

Untuk melihat contoh AWS Kebijakan FleetWise berbasis identitas IoT, lihat. [Contoh kebijakan berbasis identitas untuk AWS IoT FleetWise](#)

Kebijakan berbasis sumber daya dalam AWS IoT FleetWise

Mendukung kebijakan berbasis sumber daya: Tidak

Kebijakan berbasis sumber daya adalah dokumen JSON kebijakan yang Anda lampirkan ke sumber daya. Contoh kebijakan berbasis sumber daya adalah kebijakan kepercayaan IAM peran dan kebijakan bucket Amazon S3. Dalam layanan yang mendukung kebijakan berbasis sumber daya, administrator layanan dapat menggunakannya untuk mengontrol akses ke sumber daya tertentu. Untuk sumber daya tempat kebijakan dilampirkan, kebijakan menentukan tindakan apa yang dapat dilakukan oleh prinsipal tertentu pada sumber daya tersebut dan dalam kondisi apa. Anda harus [menentukan prinsipal](#) dalam kebijakan berbasis sumber daya. Prinsipal dapat mencakup akun, pengguna, peran, pengguna federasi, atau Layanan AWS.

Untuk mengaktifkan akses lintas akun, Anda dapat menentukan seluruh akun atau IAM entitas di akun lain sebagai prinsipal dalam kebijakan berbasis sumber daya. Menambahkan prinsipal akun silang ke kebijakan berbasis sumber daya hanya setengah dari membangun hubungan kepercayaan. Ketika kepala sekolah dan sumber daya berbeda Akun AWS, IAM administrator di akun tepercaya juga harus memberikan izin entitas utama (pengguna atau peran) untuk mengakses sumber daya. Mereka memberikan izin dengan melampirkan kebijakan berbasis identitas kepada entitas. Namun, jika kebijakan berbasis sumber daya memberikan akses ke prinsipal dalam akun yang sama, tidak diperlukan kebijakan berbasis identitas tambahan. Untuk informasi selengkapnya, lihat [Akses sumber daya lintas akun IAM di](#) Panduan IAM Pengguna.

Tindakan kebijakan untuk AWS IoT FleetWise

Mendukung tindakan kebijakan: Ya

Administrator dapat menggunakan AWS JSONkebijakan untuk menentukan siapa yang memiliki akses ke apa. Yaitu, principal dapat melakukan tindakan pada suatu sumber daya, dan dalam suatu syarat.

ActionElemen JSON kebijakan menjelaskan tindakan yang dapat Anda gunakan untuk mengizinkan atau menolak akses dalam kebijakan. Tindakan kebijakan biasanya memiliki nama yang sama dengan yang terkait AWS API operasi. Ada beberapa pengecualian, seperti tindakan khusus izin yang tidak memiliki operasi yang cocok. API Ada juga beberapa operasi yang memerlukan beberapa tindakan dalam suatu kebijakan. Tindakan tambahan ini disebut tindakan dependen.

Menyertakan tindakan dalam kebijakan untuk memberikan izin untuk melakukan operasi terkait.

Untuk melihat daftar AWS FleetWise Tindakan IoT, lihat [Tindakan yang Ditentukan oleh AWS IoT FleetWise](#) dalam Referensi Otorisasi Layanan.

Tindakan kebijakan di AWS IoT FleetWise menggunakan awalan berikut sebelum tindakan:

```
iotfleetwise
```

Untuk menetapkan secara spesifik beberapa tindakan dalam satu pernyataan, pisahkan tindakan tersebut dengan koma.

```
"Action": [  
  "iotfleetwise:action1",  
  "iotfleetwise:action2"  
]
```

Anda juga dapat menentukan beberapa tindakan menggunakan wildcard (*). Sebagai contoh, untuk menentukan semua tindakan yang dimulai dengan kata List, sertakan tindakan berikut:

```
"Action": "iotfleetwise:List*"
```

Untuk melihat contoh AWS Kebijakan FleetWise berbasis identitas IoT, lihat [Contoh kebijakan berbasis identitas untuk AWS IoT FleetWise](#)

Sumber daya kebijakan untuk AWS IoT FleetWise

Mendukung sumber daya kebijakan: Ya

Administrator dapat menggunakan AWS JSONkebijakan untuk menentukan siapa yang memiliki akses ke apa. Yaitu, principal dapat melakukan tindakan pada suatu sumber daya, dan dalam suatu syarat.

Elemen `Resource` JSON kebijakan menentukan objek atau objek yang tindakan tersebut berlaku. Pernyataan harus menyertakan elemen `Resource` atau `NotResource`. Sebagai praktik terbaik, tentukan sumber daya menggunakan [Amazon Resource Name \(ARN\)](#). Anda dapat melakukan ini untuk tindakan yang mendukung jenis sumber daya tertentu, yang dikenal sebagai izin tingkat sumber daya.

Untuk tindakan yang tidak mendukung izin di tingkat sumber daya, misalnya operasi pencantuman, gunakan wildcard (*) untuk menunjukkan bahwa pernyataan tersebut berlaku untuk semua sumber daya.

```
"Resource": "*" 
```

Untuk melihat daftar AWS Jenis FleetWise sumber daya IoT dan jenisnya ARNs, lihat Sumber [Daya yang Ditentukan oleh AWS IoT FleetWise](#) dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan mana yang dapat Anda tentukan ARN dari setiap sumber daya, lihat [Tindakan yang Ditentukan oleh AWS IoT FleetWise](#).

Untuk melihat contoh AWS Kebijakan FleetWise berbasis identitas IoT, lihat. [Contoh kebijakan berbasis identitas untuk AWS IoT FleetWise](#)

Kunci kondisi kebijakan untuk AWS IoT FleetWise

Mendukung kunci kondisi kebijakan khusus layanan: Ya

Administrator dapat menggunakan AWS JSON kebijakan untuk menentukan siapa yang memiliki akses ke apa. Yaitu, di mana utama dapat melakukan tindakan pada sumber daya, dan dalam kondisi apa.

Elemen `Condition` (atau blok `Condition`) akan memungkinkan Anda menentukan kondisi yang menjadi dasar suatu pernyataan berlaku. Elemen `Condition` bersifat opsional. Anda dapat membuat ekspresi bersyarat yang menggunakan [operator kondisi](#), misalnya sama dengan atau kurang dari, untuk mencocokkan kondisi dalam kebijakan dengan nilai-nilai yang diminta.

Jika Anda menentukan beberapa `Condition` elemen dalam pernyataan, atau beberapa kunci dalam satu `Condition` elemen, AWS mengevaluasi mereka menggunakan AND operasi logis. Jika Anda menentukan beberapa nilai untuk satu kunci kondisi, AWS mengevaluasi kondisi menggunakan OR operasi logis. Semua kondisi harus dipenuhi sebelum izin pernyataan diberikan.

Anda juga dapat menggunakan variabel placeholder saat menentukan kondisi. Misalnya, Anda dapat memberikan izin IAM pengguna untuk mengakses sumber daya hanya jika ditandai dengan nama IAM pengguna mereka. Untuk informasi selengkapnya, lihat [elemen IAM kebijakan: variabel dan tag](#) di Panduan IAM Pengguna.

AWS mendukung kunci kondisi global dan kunci kondisi khusus layanan. Untuk melihat semua AWS kunci kondisi global, lihat [AWS kunci konteks kondisi global](#) di Panduan IAM Pengguna.

Untuk melihat daftar AWS Kunci FleetWise kondisi IoT, lihat Kunci [Kondisi untuk AWS IoT FleetWise](#) dalam Referensi Otorisasi Layanan. Untuk mempelajari tindakan dan sumber daya yang dapat Anda gunakan kunci kondisi, lihat [Tindakan yang Ditentukan oleh AWS IoT FleetWise](#).

Untuk melihat contoh AWS Kebijakan FleetWise berbasis identitas IoT, lihat. [Contoh kebijakan berbasis identitas untuk AWS IoT FleetWise](#)

Daftar kontrol akses (ACLs) di AWS IoT FleetWise

Mendukung ACLs: Tidak

Access control lists (ACLs) mengontrol prinsipal mana (anggota akun, pengguna, atau peran) yang memiliki izin untuk mengakses sumber daya. ACLs mirip dengan kebijakan berbasis sumber daya, meskipun mereka tidak menggunakan format dokumen kebijakan. JSON

Kontrol akses berbasis atribut () ABAC dengan AWS IoT FleetWise

Mendukung ABAC (tag dalam kebijakan): Sebagian

Attribute-based access control (ABAC) adalah strategi otorisasi yang mendefinisikan izin berdasarkan atribut. Masuk AWS, atribut ini disebut tag. Anda dapat melampirkan tag ke IAM entitas (pengguna atau peran) dan ke banyak AWS sumber daya. Menandai entitas dan sumber daya adalah langkah pertama dari ABAC. Kemudian Anda merancang ABAC kebijakan untuk mengizinkan operasi ketika tag prinsipal cocok dengan tag pada sumber daya yang mereka coba akses.

ABAC membantu dalam lingkungan yang berkembang pesat dan membantu dengan situasi di mana manajemen kebijakan menjadi rumit.

Untuk mengendalikan akses berdasarkan tag, berikan informasi tentang tag di [elemen kondisi](#) dari kebijakan menggunakan kunci kondisi `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, atau `aws:TagKeys`.

Jika sebuah layanan mendukung ketiga kunci kondisi untuk setiap jenis sumber daya, nilainya adalah Ya untuk layanan tersebut. Jika suatu layanan mendukung ketiga kunci kondisi untuk hanya beberapa jenis sumber daya, nilainya adalah Parsial.

Untuk informasi lebih lanjut tentang ABAC, lihat [Apa itu ABAC?](#) dalam IAM User Guide. Untuk melihat tutorial dengan langkah-langkah penyiapan ABAC, lihat [Menggunakan kontrol akses berbasis atribut \(ABAC\)](#) di IAM Panduan Pengguna.

Note

AWS IoT FleetWise hanya mendukung `iam:PassRole`, yang diperlukan untuk operasi `CreateCampaign API`

Menggunakan kredensi sementara dengan AWS IoT FleetWise

Mendukung kredensi sementara: Ya

Beberapa Layanan AWS tidak berfungsi saat Anda masuk menggunakan kredensi sementara. Untuk informasi tambahan, termasuk yang Layanan AWS bekerja dengan kredensi sementara, lihat [Layanan AWS yang bekerja dengan IAM](#) dalam Panduan IAM Pengguna.

Anda menggunakan kredensi sementara jika Anda masuk ke AWS Management Console menggunakan metode apa pun kecuali nama pengguna dan kata sandi. Misalnya, ketika Anda mengakses AWS menggunakan tautan single sign-on (SSO) perusahaan Anda, proses itu secara otomatis membuat kredensi sementara. Anda juga akan secara otomatis membuat kredensial sementara ketika Anda masuk ke konsol sebagai seorang pengguna lalu beralih peran. Untuk informasi selengkapnya tentang beralih peran, lihat [Beralih ke peran \(konsol\)](#) di Panduan IAM Pengguna.

Anda dapat secara manual membuat kredensi sementara menggunakan AWS CLI atau AWS API. Anda kemudian dapat menggunakan kredensi sementara tersebut untuk mengakses AWS. AWS merekomendasikan agar Anda secara dinamis menghasilkan kredensi sementara alih-alih menggunakan kunci akses jangka panjang. Untuk informasi selengkapnya, lihat [Kredensi keamanan sementara](#) di IAM

Izin utama lintas layanan untuk AWS IoT FleetWise

Mendukung sesi akses maju (FAS): Ya

Saat Anda menggunakan IAM pengguna atau peran untuk melakukan tindakan AWS Anda dianggap sebagai kepala sekolah. Ketika Anda menggunakan beberapa layanan, Anda mungkin melakukan sebuah tindakan yang kemudian menginisiasi tindakan lain di layanan yang berbeda. FAS menggunakan izin dari prinsipal yang memanggil Layanan AWS, dikombinasikan dengan permintaan Layanan AWS untuk membuat permintaan ke layanan hilir. FAS permintaan hanya dibuat ketika layanan menerima permintaan yang memerlukan interaksi dengan orang lain Layanan AWS atau sumber daya untuk diselesaikan. Dalam hal ini, Anda harus memiliki izin untuk melakukan kedua tindakan tersebut. Untuk detail kebijakan saat membuat FAS permintaan, lihat [Meneruskan sesi akses](#).

Peran layanan untuk AWS IoT FleetWise

Mendukung peran layanan: Tidak

Peran layanan adalah [IAM peran](#) yang diasumsikan layanan untuk melakukan tindakan atas nama Anda. IAM Administrator dapat membuat, memodifikasi, dan menghapus peran layanan dari dalam IAM. Untuk informasi selengkapnya, lihat [Membuat peran untuk mendelegasikan izin ke Layanan AWS](#) di Panduan Pengguna IAM.

Warning

Mengubah izin untuk peran layanan mungkin rusak AWS Fungsionalitas IoT FleetWise. Edit peran layanan hanya ketika AWS IoT FleetWise memberikan panduan untuk melakukannya.

Peran terkait layanan untuk AWS IoT FleetWise

Mendukung peran terkait layanan: Tidak

Peran terkait layanan adalah jenis peran layanan yang ditautkan ke Layanan AWS. Layanan dapat mengambil peran untuk melakukan tindakan atas nama Anda. Peran terkait layanan muncul di Akun AWS dan dimiliki oleh layanan. IAM Administrator dapat melihat, tetapi tidak mengedit izin untuk peran terkait layanan.

Untuk detail tentang membuat atau mengelola peran terkait layanan, lihat [AWS layanan yang bekerja dengan IAM](#). Cari layanan dalam tabel yang memiliki Yes di kolom Peran terkait layanan. Pilih tautan Ya untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut.

Menggunakan peran terkait layanan untuk IoT AWS FleetWise

Penggunaan IoT FleetWise AWS Identity and Access Management (IAM) peran [terkait layanan](#). Peran terkait layanan adalah jenis peran unik yang ditautkan langsung ke IoT AWS. IAM FleetWise Peran terkait layanan telah ditentukan sebelumnya oleh IoT FleetWise dan menyertakan izin yang dibutuhkan AWS IoT untuk mengirim metrik ke Amazon AWS. FleetWise CloudWatch Untuk informasi selengkapnya, lihat [Memantau AWS IoT FleetWise dengan Amazon CloudWatch](#).

Peran terkait layanan membuat pengaturan AWS IoT FleetWise lebih cepat karena Anda tidak perlu menambahkan izin yang diperlukan secara manual. AWS IoT FleetWise mendefinisikan izin peran terkait layanan, dan kecuali ditentukan lain, hanya AWS IoT yang dapat mengambil perannya. FleetWise Izin yang ditentukan mencakup kebijakan kepercayaan dan kebijakan izin. Kebijakan izin ini tidak dapat dilampirkan ke IAM entitas lain.

Anda dapat menghapus peran tertaut layanan hanya setelah menghapus sumber daya terkait terlebih dahulu. Ini melindungi FleetWise sumber daya AWS IoT Anda karena Anda tidak dapat secara tidak sengaja menghapus izin untuk mengakses sumber daya.

Untuk informasi tentang layanan lain yang mendukung peran terkait layanan, lihat [AWS layanan yang bekerja dengan IAM](#), dan mencari layanan yang memiliki Ya di kolom Peran terkait layanan. Untuk melihat dokumentasi peran terkait layanan untuk layanan tersebut, pilih Ya dengan tautan.

Izin peran terkait layanan untuk IoT AWS FleetWise

AWS IoT FleetWise menggunakan peran terkait layanan bernama `AWSIoTServiceRoleForIoT FleetWise`— Kebijakan AWS terkelola yang digunakan untuk semua izin out-of-the-box IoT. AWS FleetWise

Peran `AWSIoTServiceRoleForIoT FleetWise` terkait layanan mempercayai layanan berikut untuk mengambil peran:

- `IoT FleetWise`

Kebijakan izin peran bernama `AWSIoT FleetWise ServiceRolePolicy` memungkinkan AWS FleetWise IoT untuk menyelesaikan tindakan berikut pada sumber daya yang ditentukan:

- Tindakan: `cloudwatch:PutMetricData` pada sumber daya: *

Anda harus mengonfigurasi izin untuk mengizinkan IAM entitas (seperti pengguna, grup, atau peran) membuat, mengedit, atau menghapus peran terkait layanan. Untuk informasi selengkapnya, lihat [Izin peran terkait layanan di Panduan Pengguna](#). IAM

Membuat peran terkait layanan untuk IoT AWS FleetWise

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda mendaftarkan akun di FleetWise konsol AWS IoT, AWS CLI, atau AWS API, AWS IoT FleetWise menciptakan peran terkait layanan untuk Anda. Untuk informasi selengkapnya, lihat [Mengkonfigurasi pengaturan](#).

Membuat peran terkait layanan di AWS FleetWise IoT (konsol)

Anda tidak perlu membuat peran terkait layanan secara manual. Saat Anda mendaftarkan akun di FleetWise konsol AWS IoT, AWS CLI, atau AWS API, AWS IoT FleetWise menciptakan peran terkait layanan untuk Anda.

Mengedit peran terkait layanan untuk IoT AWS FleetWise

Anda tidak dapat mengedit peran `AWSServiceRoleForIoT FleetWise` terkait layanan di IoT AWS. FleetWise Karena berbagai entitas mungkin mereferensikan peran terkait layanan apa pun yang Anda buat, Anda tidak dapat mengubah nama peran tersebut. Namun, Anda dapat mengedit deskripsi peran dengan menggunakan IAM. Untuk informasi selengkapnya, lihat [Mengedit peran terkait layanan](#) di IAMPanduan Pengguna.

Membersihkan peran terkait layanan

Sebelum dapat digunakan IAM untuk menghapus peran terkait layanan, Anda harus menghapus sumber daya apa pun yang digunakan oleh peran tersebut terlebih dahulu.

Note

Jika AWS IoT FleetWise menggunakan peran saat Anda mencoba menghapus sumber daya, maka penghapusan mungkin gagal. Jika hal itu terjadi, tunggu beberapa menit dan coba mengoperasikannya lagi. Untuk mempelajari cara menghapus service-linked-role melalui konsol, AWS CLI, atau AWS API, lihat [Menggunakan peran terkait layanan](#) di IAMPanduan Pengguna.

Jika Anda menghapus peran terkait layanan ini, dan kemudian perlu membuatnya lagi, Anda dapat mendaftarkan akun dengan AWS IoT. FleetWise AWSIoT FleetWise kemudian menciptakan peran terkait layanan untuk Anda lagi.

Contoh kebijakan berbasis identitas untuk AWS IoT FleetWise

Secara default, pengguna dan peran tidak memiliki izin untuk membuat atau memodifikasi AWS Sumber daya IoT FleetWise. Mereka juga tidak dapat melakukan tugas dengan menggunakan AWS Management Console, AWS Command Line Interface (AWS CLI), atau AWS API. Untuk memberikan izin kepada pengguna untuk melakukan tindakan pada sumber daya yang mereka butuhkan, IAM administrator dapat membuat IAM kebijakan. Administrator kemudian dapat menambahkan IAM kebijakan ke peran, dan pengguna dapat mengambil peran.

Untuk mempelajari cara membuat kebijakan IAM berbasis identitas menggunakan contoh dokumen kebijakan ini, lihat [Membuat JSON IAM kebijakan di Panduan Pengguna](#). IAM

Untuk detail tentang tindakan dan jenis sumber daya yang ditentukan oleh AWS IoT FleetWise, termasuk format ARNs untuk setiap jenis sumber daya, lihat [Tindakan, Sumber Daya, dan Kunci Kondisi untuk AWS IoT FleetWise](#) dalam Referensi Otorisasi Layanan.

Topik

- [Praktik terbaik kebijakan](#)
- [Menggunakan AWS Konsol IoT FleetWise](#)
- [Mengizinkan pengguna melihat izin mereka sendiri](#)
- [Akses sumber daya di Amazon Timestream](#)

Praktik terbaik kebijakan

Kebijakan berbasis identitas menentukan apakah seseorang dapat membuat, mengakses, atau menghapus AWS FleetWise Sumber daya IoT di akun Anda. Tindakan ini dapat menimbulkan biaya untuk Akun AWS. Saat Anda membuat atau mengedit kebijakan berbasis identitas, ikuti panduan dan rekomendasi berikut:

- Memulai dengan AWS kebijakan terkelola dan beralih ke izin hak istimewa terkecil — Untuk memulai pemberian izin kepada pengguna dan beban kerja Anda, gunakan AWS kebijakan terkelola yang memberikan izin untuk banyak kasus penggunaan umum. Mereka tersedia di Akun AWS. Kami menyarankan Anda mengurangi izin lebih lanjut dengan mendefinisikan AWS kebijakan terkelola pelanggan yang khusus untuk kasus penggunaan Anda. Untuk informasi selengkapnya, silakan lihat [AWS kebijakan terkelola](#) atau [AWS kebijakan terkelola untuk fungsi pekerjaan](#) di Panduan IAM Pengguna.

- Menerapkan izin hak istimewa paling sedikit — Saat Anda menetapkan izin dengan IAM kebijakan, berikan hanya izin yang diperlukan untuk melakukan tugas. Anda melakukannya dengan mendefinisikan tindakan yang dapat diambil pada sumber daya tertentu dalam kondisi tertentu, yang juga dikenal sebagai izin dengan hak akses paling rendah. Untuk informasi selengkapnya tentang penggunaan IAM untuk menerapkan izin, lihat [Kebijakan dan izin IAM di IAM](#) Panduan Pengguna.
- Gunakan ketentuan dalam IAM kebijakan untuk membatasi akses lebih lanjut — Anda dapat menambahkan kondisi ke kebijakan Anda untuk membatasi akses ke tindakan dan sumber daya. Misalnya, Anda dapat menulis kondisi kebijakan untuk menentukan bahwa semua permintaan harus dikirim menggunakan SSL. Anda juga dapat menggunakan ketentuan untuk memberikan akses ke tindakan layanan jika digunakan melalui tindakan tertentu Layanan AWS, seperti AWS CloudFormation. Untuk informasi selengkapnya, lihat [elemen IAM JSON kebijakan: Kondisi](#) dalam Panduan IAM Pengguna.
- Gunakan IAM Access Analyzer untuk memvalidasi IAM kebijakan Anda guna memastikan izin yang aman dan fungsional — IAM Access Analyzer memvalidasi kebijakan baru dan yang sudah ada sehingga kebijakan mematuhi bahasa IAM kebijakan () JSON dan praktik terbaik. IAM IAMAccess Analyzer menyediakan lebih dari 100 pemeriksaan kebijakan dan rekomendasi yang dapat ditindaklanjuti untuk membantu Anda membuat kebijakan yang aman dan fungsional. Untuk informasi selengkapnya, lihat [Validasi kebijakan IAM Access Analyzer](#) di IAMPanduan Pengguna.
- Memerlukan otentikasi multi-faktor (MFA) - Jika Anda memiliki skenario yang mengharuskan IAM pengguna atau pengguna root di Akun AWS, nyalakan MFA untuk keamanan tambahan. Untuk meminta MFA kapan API operasi dipanggil, tambahkan MFA kondisi ke kebijakan Anda. Untuk informasi selengkapnya, lihat [Mengonfigurasi API akses MFA yang dilindungi](#) di IAMPanduan Pengguna.

Untuk informasi selengkapnya tentang praktik terbaik di IAM, lihat [Praktik terbaik keamanan IAM di](#) Panduan IAM Pengguna.

Menggunakan AWS Konsol IoT FleetWise

Untuk mengakses AWS FleetWise Konsol IoT, Anda harus memiliki set izin minimum. Izin ini harus memungkinkan Anda untuk membuat daftar dan melihat detail tentang AWS FleetWise Sumber daya IoT di Anda Akun AWS. Jika Anda membuat kebijakan berbasis identitas yang lebih ketat daripada izin minimum yang diperlukan, konsol tidak akan berfungsi sebagaimana dimaksud untuk entitas (pengguna atau peran) dengan kebijakan tersebut.

Anda tidak perlu mengizinkan izin konsol minimum untuk pengguna yang melakukan panggilan hanya ke AWS CLI atau AWS API. Sebagai gantinya, izinkan akses hanya ke tindakan yang cocok dengan API operasi yang mereka coba lakukan.

Untuk memastikan bahwa pengguna dan peran masih dapat menggunakan AWS FleetWise Konsol IoT, juga pasang AWS IoT FleetWise ConsoleAccess atau ReadOnly AWS kebijakan yang dikelola untuk entitas. Untuk informasi selengkapnya, lihat [Menambahkan izin ke pengguna](#) di Panduan IAM Pengguna.

Mengizinkan pengguna melihat izin mereka sendiri

Contoh ini menunjukkan cara Anda membuat kebijakan yang memungkinkan IAM pengguna melihat kebijakan sebaris dan terkelola yang dilampirkan pada identitas pengguna mereka. Kebijakan ini mencakup izin untuk menyelesaikan tindakan ini di konsol atau secara terprogram menggunakan AWS CLI atau AWS API.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "ViewOwnUserInfo",
      "Effect": "Allow",
      "Action": [
        "iam:GetUserPolicy",
        "iam:ListGroupsWithUser",
        "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": "*"
    }
  ]
}
```

Akses sumber daya di Amazon Timestream

Sebelum menggunakan AWS IoT FleetWise, Anda harus mendaftarkan AWS akun, IAM, dan sumber daya Amazon Timestream untuk diberikan AWS FleetWise Izin IoT untuk mengirim data kendaraan ke AWS Cloud atas nama Anda. Untuk mendaftar, Anda perlu:

- Database Amazon Timestream.
- Tabel yang dibuat dalam database Amazon Timestream yang ditentukan.
- IAM Peran yang memungkinkan AWS IoT FleetWise untuk mengirim data ke Amazon Timestream.

Untuk informasi selengkapnya, termasuk prosedur dan contoh kebijakan, lihat [Mengkonfigurasi Pengaturan](#).

Pemecahan Masalah AWS FleetWise Identitas dan akses IoT

Gunakan informasi berikut untuk membantu Anda mendiagnosis dan memperbaiki masalah umum yang mungkin Anda temui saat bekerja dengan AWS IoT FleetWise dan IAM

Topik

- [Saya tidak berwenang untuk melakukan tindakan di AWS IoT FleetWise](#)
- [Saya tidak berwenang untuk melakukan iam: PassRole](#)
- [Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses saya AWS Sumber daya IoT FleetWise](#)

Saya tidak berwenang untuk melakukan tindakan di AWS IoT FleetWise

Jika AWS Management Console memberitahu Anda bahwa Anda tidak berwenang untuk melakukan tindakan, maka Anda harus menghubungi administrator Anda untuk bantuan. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Contoh kesalahan berikut terjadi ketika mateojackson IAM pengguna mencoba menggunakan konsol untuk melihat detail tentang *myVehicle* sumber daya fiksi tetapi tidak memiliki `iotfleetwise:GetVehicleStatus` izin.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
iotfleetwise:GetVehicleStatus on resource: myVehicle
```

Dalam hal ini, Mateo meminta administratornya untuk memperbarui kebijakannya untuk mengizinkan dia mengakses sumber daya *myVehicle* menggunakan tindakan `iotfleetwise:GetVehicleStatus`.

Saya tidak berwenang untuk melakukan iam: PassRole

Jika Anda menerima kesalahan yang tidak diizinkan untuk melakukan `iam:PassRole` tindakan, kebijakan Anda harus diperbarui agar Anda dapat meneruskan peran AWS IoT FleetWise.

Beberapa Layanan AWS memungkinkan Anda untuk meneruskan peran yang ada ke layanan tersebut alih-alih membuat peran layanan baru atau peran terkait layanan. Untuk melakukannya, Anda harus memiliki izin untuk meneruskan peran ke layanan.

Contoh kesalahan berikut terjadi ketika IAM pengguna bernama marymajor mencoba menggunakan konsol untuk melakukan tindakan di AWS IoT FleetWise. Namun, tindakan tersebut memerlukan layanan untuk mendapatkan izin yang diberikan oleh peran layanan. Mary tidak memiliki izin untuk meneruskan peran tersebut pada layanan.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
iam:PassRole
```

Dalam kasus ini, kebijakan Mary harus diperbarui agar dia mendapatkan izin untuk melakukan tindakan `iam:PassRole` tersebut.

Jika Anda membutuhkan bantuan, hubungi AWS administrator. Administrator Anda adalah orang yang memberi Anda kredensial masuk.

Saya ingin mengizinkan orang di luar saya Akun AWS untuk mengakses saya AWS Sumber daya IoT FleetWise

Anda dapat membuat peran yang dapat digunakan pengguna di akun lain atau orang-orang di luar organisasi Anda untuk mengakses sumber daya Anda. Anda dapat menentukan siapa saja yang dipercaya untuk mengambil peran tersebut. Untuk layanan yang mendukung kebijakan berbasis

sumber daya atau daftar kontrol akses (ACLs), Anda dapat menggunakan kebijakan tersebut untuk memberi orang akses ke sumber daya Anda.

Untuk mempelajari selengkapnya, periksa referensi berikut:

- Untuk mempelajari apakah AWS IoT FleetWise mendukung fitur-fitur ini, lihat [Bagaimana AWS IoT bekerja FleetWise dengan IAM](#)
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda Akun AWS yang Anda miliki, lihat [Menyediakan akses ke IAM pengguna lain Akun AWS yang Anda miliki](#) di Panduan IAM Pengguna.
- Untuk mempelajari cara menyediakan akses ke sumber daya Anda ke pihak ketiga Akun AWS, lihat [Menyediakan akses ke Akun AWS dimiliki oleh pihak ketiga](#) dalam Panduan IAM Pengguna.
- Untuk mempelajari cara menyediakan akses melalui federasi identitas, lihat [Menyediakan akses ke pengguna yang diautentikasi secara eksternal \(federasi identitas\) di Panduan Pengguna](#). IAM
- Untuk mempelajari perbedaan antara menggunakan peran dan kebijakan berbasis sumber daya untuk akses lintas akun, lihat [Akses sumber daya lintas akun di IAM](#) Panduan Pengguna. IAM

Validasi Kepatuhan untuk AWS IoT FleetWise

Note

AWS IoT FleetWise tidak dalam lingkup apa pun AWS program kepatuhan.

Untuk mengetahui apakah sebuah Layanan AWS berada dalam lingkup program kepatuhan tertentu, lihat [Layanan AWS dalam Lingkup oleh Program Kepatuhan](#) dan pilih program kepatuhan yang Anda minati. Untuk informasi umum, lihat [AWS Program Kepatuhan](#) .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh Laporan di AWS Artifact](#).

Tanggung jawab kepatuhan Anda saat menggunakan Layanan AWS ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- [Panduan Memulai Cepat Keamanan dan Kepatuhan — Panduan](#) penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar AWS yang berfokus pada keamanan dan kepatuhan.

- [Arsitektur untuk HIPAA Keamanan dan Kepatuhan di Amazon Web Services](#) — Whitepaper ini menjelaskan bagaimana perusahaan dapat menggunakan AWS untuk membuat aplikasi HIPAA yang memenuhi syarat.

 Note

Tidak semua Layanan AWS HIPAA memenuhi syarat. Untuk informasi selengkapnya, lihat [Referensi Layanan yang HIPAA Memenuhi Syarat](#).

- [AWS Sumber Daya Kepatuhan](#) — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- [AWS Panduan Kepatuhan Pelanggan](#) - Memahami model tanggung jawab bersama melalui lensa kepatuhan. Panduan merangkum praktik terbaik untuk mengamankan Layanan AWS dan memetakan panduan untuk kontrol keamanan di berbagai kerangka kerja (termasuk Institut Standar dan Teknologi Nasional (NIST), Dewan Standar Keamanan Industri Kartu Pembayaran (PCI), dan Organisasi Internasional untuk Standardisasi (ISO)).
- [Mengevaluasi Sumber Daya dengan Aturan](#) di AWS Config Panduan Pengembang — The AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- [AWS Security Hub](#)— Ini Layanan AWS memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalam AWS. Security Hub menggunakan kontrol keamanan untuk mengevaluasi AWS sumber daya dan untuk memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik. Untuk daftar layanan dan kontrol yang didukung, lihat [Referensi kontrol Security Hub](#).
- [Amazon GuardDuty](#) - Ini Layanan AWS Mendeteksi potensi ancaman terhadap Akun AWS, beban kerja, kontainer, dan data dengan memantau lingkungan Anda untuk aktivitas yang mencurigakan dan berbahaya. GuardDuty dapat membantu Anda mengatasi berbagai persyaratan kepatuhan, seperti PCIDSS, dengan memenuhi persyaratan deteksi intrusi yang diamanatkan oleh kerangka kerja kepatuhan tertentu.
- [AWS Audit Manager](#)— Ini Layanan AWS membantu Anda terus mengaudit AWS penggunaan untuk menyederhanakan cara Anda mengelola risiko dan kepatuhan terhadap peraturan dan standar industri.

Ketahanan di AWS IoT FleetWise

Bagian AWS Infrastruktur global dibangun di sekitar AWS Wilayah dan Zona Ketersediaan. Wilayah memberikan beberapa Zona Ketersediaan yang terpisah dan terisolasi secara fisik, yang terkoneksi melalui jaringan latensi rendah, throughput tinggi, dan sangat redundan. Dengan Zona Ketersediaan, Anda dapat merancang serta mengoperasikan aplikasi dan basis data yang secara otomatis melakukan fail over di antara zona tanpa gangguan. Zona Ketersediaan memiliki ketersediaan dan toleransi kesalahan yang lebih baik, dan dapat diskalakan dibandingkan infrastruktur pusat data tunggal atau multi tradisional.

Untuk informasi lebih lanjut tentang AWS Wilayah dan Availability Zone, lihat [AWS Infrastruktur Global](#).

Note

Data diproses oleh AWS IoT FleetWise disimpan dalam database Amazon Timestream. Timestream mendukung backup ke yang lain AWS Zona Ketersediaan atau Wilayah. Namun, Anda dapat menulis aplikasi Anda sendiri menggunakan Timestream SDK untuk menanyakan data dan menyimpannya ke tujuan pilihan Anda.

Untuk informasi selengkapnya tentang Amazon Timestream, lihat [di Panduan Pengembang Amazon Timestream](#).

Keamanan infrastruktur di AWS IoT FleetWise

Sebagai layanan yang dikelola, AWS IoT dilindungi FleetWise oleh AWS keamanan jaringan global. Untuk informasi tentang AWS Layanan keamanan dan bagaimana AWS melindungi infrastruktur, lihat [AWS Keamanan Cloud](#). Untuk mendesain Anda AWS lingkungan menggunakan praktik terbaik untuk keamanan infrastruktur, lihat [Perlindungan Infrastruktur](#) di Pilar Keamanan AWS Kerangka Kerja yang Diarsiteksikan dengan Baik.

Anda menggunakan AWS API panggilan yang dipublikasikan untuk mengakses AWS IoT FleetWise melalui jaringan. Klien harus mendukung hal-hal berikut:

- Keamanan Lapisan Transportasi (TLS). Kami membutuhkan TLS 1.2 dan merekomendasikan TLS 1.3.

- Suite cipher dengan kerahasiaan maju yang sempurna (PFS) seperti (Ephemeral Diffie-Hellman) atau DHE (Elliptic Curve Ephemeral Diffie-Hellman). ECDHE Sebagian besar sistem modern seperti Java 7 dan versi lebih baru mendukung mode-mode ini.

Selain itu, permintaan harus ditandatangani dengan menggunakan ID kunci akses dan kunci akses rahasia yang terkait dengan IAM prinsipal. Atau Anda dapat menggunakan [AWS Security Token Service](#) (AWS STS) untuk menghasilkan kredensial keamanan sementara untuk menandatangani permintaan.

Anda dapat memanggil API operasi ini dari lokasi jaringan mana pun, tetapi AWS IoT FleetWise memang mendukung kebijakan akses berbasis sumber daya, yang dapat mencakup pembatasan berdasarkan alamat IP sumber. Anda juga dapat menggunakan AWS FleetWise Kebijakan IoT untuk mengontrol akses dari titik akhir Amazon Virtual Private Cloud VPC (Amazon) tertentu atau spesifik. VPCs Secara efektif, ini mengisolasi akses jaringan ke suatu AWS FleetWise Sumber daya IoT hanya dari yang spesifik VPC di dalam AWS jaringan.

Topik

- [Menghubungkan ke AWS IoT FleetWise melalui titik akhir antarmuka VPC](#)

Menghubungkan ke AWS IoT FleetWise melalui titik akhir antarmuka VPC

Anda dapat terhubung langsung ke AWS IoT FleetWise dengan menggunakan titik akhir [antarmuka VPC \(AWS PrivateLink\)](#) di Virtual Private Cloud (VPC) Anda, alih-alih terhubung melalui internet. Saat Anda menggunakan VPC titik akhir antarmuka, komunikasi antara Anda VPC dan AWS IoT dilakukan FleetWise sepenuhnya di dalam AWS jaringan. Setiap VPC titik akhir diwakili oleh satu atau lebih [antarmuka jaringan Elastic](#) (ENIs) dengan alamat IP pribadi di subnet AndaVPC.

VPCTitik akhir antarmuka menghubungkan Anda VPC langsung ke AWS IoT FleetWise tanpa gateway internet, NAT perangkat, VPN koneksi, atau AWS Direct Connect koneksi. Contoh di Anda VPC tidak memerlukan alamat IP publik untuk berkomunikasi dengan AWS IoT FleetWiseAPI.

Untuk menggunakan AWS IoT FleetWise melalui AndaVPC, Anda harus terhubung dari instans yang ada di dalam VPC atau menghubungkan jaringan pribadi Anda ke Anda VPC dengan menggunakan AWS Virtual Private Network (VPN) atau AWS Direct Connect. Untuk informasi tentang AmazonVPN, lihat [VPNkoneksi](#) di Panduan Pengguna Amazon Virtual Private Cloud. Untuk informasi tentang AWS Direct Connect, lihat [Membuat koneksi](#) di AWS Direct Connect Panduan Pengguna.

Anda dapat membuat VPC titik akhir antarmuka untuk terhubung AWS IoT FleetWise dengan menggunakan AWS konsol atau AWS Command Line Interface (AWS CLI) perintah. Untuk informasi selengkapnya, lihat [Membuat titik akhir antarmuka](#).

Setelah Anda membuat VPC titik akhir antarmuka, jika Anda mengaktifkan DNS nama host pribadi untuk titik akhir, default AWS FleetWise Titik akhir IoT menyelesaikan ke titik akhir Anda. VPC Titik akhir nama layanan default untuk AWS IoT FleetWise dalam format berikut.

```
iotfleetwise.Region.amazonaws.com
```

Jika Anda tidak mengaktifkan DNS nama host pribadi, Amazon VPC menyediakan nama DNS titik akhir yang dapat Anda gunakan dalam format berikut.

```
VPCE_ID.iotfleetwise.Region.vpce.amazonaws.com
```

Untuk informasi selengkapnya, lihat VPC Endpoint [antarmuka \(AWS PrivateLink\)](#) di Panduan VPC Pengguna Amazon.

AWS IoT FleetWise mendukung panggilan ke semua [API tindakannya](#) di dalam Anda. VPC

Anda dapat melampirkan kebijakan VPC titik akhir ke VPC titik akhir untuk mengontrol akses bagi IAM prinsipal. Anda juga dapat mengaitkan grup keamanan dengan VPC titik akhir untuk mengontrol akses masuk dan keluar berdasarkan asal dan tujuan lalu lintas jaringan, seperti berbagai alamat IP. Untuk informasi selengkapnya, lihat [Mengontrol akses ke layanan dengan VPC titik akhir](#).

Membuat kebijakan VPC endpoint untuk AWS IoT FleetWise

Anda dapat membuat kebijakan untuk VPC titik akhir Amazon AWS IoT FleetWise untuk menentukan yang berikut:

- Kepala sekolah yang bisa atau tidak bisa melakukan tindakan
- Tindakan yang bisa atau tidak bisa dilakukan

Untuk informasi selengkapnya, lihat [Mengontrol akses ke layanan dengan VPC titik akhir](#) di Panduan VPC Pengguna Amazon.

Example — kebijakan VPC endpoint untuk menolak semua akses dari yang ditentukan AWS akun

Kebijakan VPC endpoint berikut menyangkal AWS akun **123456789012** semua API panggilan menggunakan titik akhir.

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": "*"
    },
    {
      "Action": "*",
      "Effect": "Deny",
      "Resource": "*",
      "Principal": {
        "AWS": [
          "123456789012"
        ]
      }
    }
  ]
}
```

Example — kebijakan VPC endpoint untuk mengizinkan VPC akses hanya ke IAM prinsipal tertentu (pengguna)

Kebijakan VPC endpoint berikut memungkinkan akses penuh hanya ke pengguna *lijuan* in AWS akun *123456789012*. Ini menyangkal semua akses IAM kepala sekolah lainnya ke titik akhir.

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456789012:user/lijuan"
        ]
      }
    }
  ]
}
```

Example — kebijakan VPC titik akhir untuk AWS Tindakan IoT FleetWise

Berikut ini adalah contoh kebijakan endpoint untuk AWS IoT FleetWise. Saat dilampirkan ke titik akhir, kebijakan ini memberikan akses ke yang tercantum AWS FleetWise Tindakan IoT untuk pengguna IAM *fleetWise* di Akun AWS *123456789012*.

```
{
  "Statement": [
    {
      "Principal": {
        "AWS": [
          "arn:aws:iam::123456789012:user/fleetWise"
        ],
      },
      "Resource": "*",
      "Effect": "Allow",
      "Action": [
        "iotfleetwise:ListFleets",
        "iotfleetwise:ListCampaigns",
        "iotfleetwise:CreateVehicle",
      ]
    }
  ]
}
```

Analisis konfigurasi dan kerentanan di AWS IoT FleetWise

Lingkungan IoT dapat terdiri atas sejumlah besar perangkat yang memiliki beragam kemampuan, berumur panjang, dan didistribusikan secara geografis. Karakteristik ini membuat penyiapan perangkat menjadi kompleks dan rawan kesalahan. Juga, karena perangkat sering dibatasi dalam daya komputasi, memori, dan kemampuan penyimpanan, penggunaan enkripsi dan bentuk keamanan lainnya pada perangkat terbatas. Perangkat sering menggunakan perangkat lunak dengan kerentanan yang diketahui. Faktor-faktor ini membuat perangkat IoT, termasuk kendaraan yang mengumpulkan data untuk AWS IoT FleetWise, target yang menarik bagi peretas dan membuatnya sulit untuk mengamankan mereka secara berkelanjutan.

Konfigurasi dan kontrol TI adalah tanggung jawab bersama antara AWS dan Anda, pelanggan kami. Untuk informasi lebih lanjut, lihat AWS [model tanggung jawab bersama](#).

Praktik terbaik keamanan untuk AWS IoT FleetWise

AWS IoT FleetWise menyediakan sejumlah fitur keamanan untuk dipertimbangkan saat Anda mengembangkan dan menerapkan kebijakan keamanan Anda sendiri. Praktik terbaik berikut adalah pedoman umum dan tidak mewakili solusi keamanan yang lengkap. Karena praktik terbaik ini mungkin tidak sesuai atau cukup untuk lingkungan Anda, anggap sebagai pertimbangan yang membantu dan bukan sebagai resep.

Untuk belajar tentang keamanan di AWS IoT lihat Praktik terbaik [keamanan di AWS IoT Core](#) di AWS IoT Panduan Pengembang

Berikan izin minimum yang memungkinkan

Ikuti prinsip hak istimewa terkecil dengan menggunakan set izin minimum dalam IAM peran. Batasi penggunaan * wildcard untuk Resource properti Action dan properti dalam IAM kebijakan Anda. Sebaliknya, nyatakan serangkaian terbatas tindakan dan sumber daya bila memungkinkan. Untuk informasi lebih lanjut tentang hak istimewa minimum dan praktik terbaik kebijakan lainnya, lihat [the section called “Praktik terbaik kebijakan”](#).

Jangan log informasi sensitif

Anda harus mencegah pencatatan kredensi dan informasi identitas pribadi lainnya (). PII Kami menyarankan Anda menerapkan pengamanan berikut:

- Jangan gunakan informasi sensitif dalam nama perangkat.
- Jangan gunakan informasi sensitif dalam nama IDs dan AWS FleetWise Sumber daya IoT, misalnya dalam nama kampanye, manifes decoder, model kendaraan, dan katalog sinyal, atau kendaraan dan armada. IDs

Gunakan AWS CloudTrail untuk melihat riwayat API panggilan

Anda dapat melihat riwayat AWS FleetWise API Panggilan IoT dilakukan di akun Anda untuk analisis keamanan dan tujuan pemecahan masalah operasional. Untuk menerima sejarah AWS FleetWise API Panggilan IoT yang dilakukan di akun Anda, cukup nyalakan di CloudTrail AWS Management Console Untuk informasi selengkapnya, lihat [the section called “Log CloudTrail”](#).

Sinkronkan jam perangkat Anda

Penting untuk memiliki waktu yang akurat di perangkat Anda. Sertifikat X.509 memiliki tanggal dan waktu kedaluwarsa. Jam di perangkat Anda digunakan untuk memverifikasi bahwa sertifikat server masih valid. Jam perangkat dapat melayang dari waktu ke waktu atau baterai dapat habis.

Untuk informasi selengkapnya, lihat praktik terbaik [Menyinkronkan jam perangkat Anda](#) di AWS IoT Core Panduan Pengembang.

Pemantauan AWS IoT FleetWise

Pemantauan adalah bagian penting untuk menjaga keandalan, ketersediaan, dan kinerja AWS IoT FleetWise dan solusi Anda yang lain AWS. AWS menyediakan alat pemantauan berikut untuk menonton AWS IoT FleetWise, melaporkan ketika ada sesuatu yang salah, dan mengambil tindakan otomatis bila perlu:

- Amazon CloudWatch memantau AWS sumber daya Anda dan aplikasi yang Anda jalankan AWS secara real time. Anda dapat mengumpulkan dan melacak metrik, membuat dasbor yang disesuaikan, dan mengatur alarm yang memberi tahu Anda atau mengambil tindakan saat metrik tertentu mencapai ambang batas yang Anda tentukan. Misalnya, Anda dapat CloudWatch melacak penggunaan CPU atau metrik lain dari instans Amazon EC2 Anda dan secara otomatis meluncurkan instans baru bila diperlukan. Untuk informasi selengkapnya, lihat [Panduan CloudWatch Pengguna Amazon](#).
- Amazon CloudWatch Logs dapat digunakan untuk memantau, menyimpan, dan mengakses file log Anda dari instans Amazon EC2, CloudTrail, dan sumber lainnya. CloudWatch Log dapat memantau informasi dalam file log dan memberi tahu Anda ketika ambang batas tertentu terpenuhi. Anda juga dapat mengarsipkan data log Anda dalam penyimpanan yang sangat tahan lama. Untuk informasi selengkapnya, lihat [Panduan Pengguna Amazon CloudWatch Logs](#).
- AWS CloudTrail menangkap panggilan API dan kejadian terkait yang dilakukan oleh atau atas nama Akun AWS Anda. Kemudian, ia mengirimkan file log ke bucket Amazon S3 yang Anda tentukan. Anda dapat mengidentifikasi pengguna dan akun mana yang memanggil AWS, alamat IP sumber yang melakukan panggilan, dan kapan panggilan tersebut terjadi. Untuk mengetahui informasi selengkapnya, lihat [Panduan Pengguna AWS CloudTrail](#).

Memantau AWS IoT FleetWise dengan Amazon CloudWatch

CloudWatch Metrik Amazon adalah cara untuk memantau AWS sumber daya Anda dan kinerjanya. AWS IoT FleetWise mengirimkan metrik ke CloudWatch Anda dapat menggunakan, API AWS Management Console AWS CLI, atau API untuk membuat daftar metrik yang dikirimkan AWS FleetWise IoT. CloudWatch Untuk informasi selengkapnya, lihat [Panduan CloudWatch Pengguna Amazon](#).

⚠ Important

Anda harus mengonfigurasi pengaturan sehingga AWS IoT FleetWise dapat mengirim metrik ke CloudWatch Untuk informasi selengkapnya, lihat [Mengkonfigurasi pengaturan](#).

Namespace AWS/IoTFleetWise mencakup metrik berikut.

Metrik sinyal

Metrik	Deskripsi
IllegalMessageFromEdge	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise tidak sesuai dengan format yang diperlukan.</p> <p>Unit: Jumlah</p> <p>Dimensi: VehicleName</p> <p>Statistik yang valid: Sum</p>
MessageThrottled	<p>Sebuah pesan yang dikirim dari kendaraan ke AWS IoT FleetWise dibatasi. Ini karena Anda melampaui batas layanan untuk akun ini di Wilayah saat ini.</p> <p>Unit: Jumlah</p> <p>Dimensi: VehicleName</p> <p>Statistik yang valid: Sum</p>
ModelingError	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise berisi sinyal yang gagal divalidasi terhadap model kendaraan.</p> <p>Unit: Jumlah</p> <p>Dimensi: ModelManifestName</p>

Metrik	Deskripsi
	Statistik yang valid: Sum
DecodingError	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise berisi sinyal yang gagal mendekoder terhadap manifes decoder kendaraan.</p> <p>Unit: Jumlah</p> <p>Dimensi: DecoderName</p> <p>Statistik yang valid: Sum</p>

Metrik kampanye

Metrik	Deskripsi
VehicleNotFound	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise, di mana kendaraan tidak diketahui.</p> <p>Unit: Jumlah</p> <p>Dimensi: VehicleName</p> <p>Statistik yang valid: Sum</p>
CampaignInvalid	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise, di mana kampanye tidak valid.</p> <p>Unit: Jumlah</p> <p>Dimensi: CampaignName</p> <p>Statistik yang valid: Sum</p>

Metrik	Deskripsi
CampaignNotFound	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise, di mana kampanye tidak diketahui.</p> <p>Unit: Jumlah</p> <p>Dimensi: CampaignName</p> <p>Statistik yang valid: Sum</p>

Metrik tujuan data kampanye

Metrik	Deskripsi
TimestreamWriteError	<p>AWSIoT FleetWise tidak dapat menulis pesan dari kendaraan ke tabel Amazon Timestream.</p> <p>Unit: Jumlah</p> <p>Dimensi: DatabaseName, TableName</p> <p>Statistik yang valid: Sum</p>
S3 WriteError	<p>AWSIoT FleetWise tidak dapat menulis pesan dari kendaraan ke bucket Amazon Simple Storage Service (Amazon S3).</p> <p>Unit: Jumlah</p> <p>Dimensi: BucketName</p> <p>Statistik yang valid: Sum</p>
S3 ReadError	<p>AWSIoT FleetWise tidak dapat membaca kunci objek dari kendaraan di bucket Amazon Simple Storage Service (Amazon S3).</p> <p>Unit: Jumlah</p>

Metrik	Deskripsi
	Dimensi: BucketName
	Statistik yang valid: Sum

Metrik AWS KMS kunci yang dikelola pelanggan

Metrik	Deskripsi
KMS KeyAccessDenied	<p>AWSIoT FleetWise tidak dapat menulis pesan dari kendaraan ke tabel Timestream atau bucket Amazon S3 karena kesalahan AWS KMS akses kunci ditolak.</p> <p>Unit: Jumlah</p> <p>Dimensi: KMS KeyId</p> <p>Statistik yang valid: Sum</p>

Memantau AWS IoT dengan FleetWise Amazon Logs CloudWatch

Amazon CloudWatch Logs memantau peristiwa yang terjadi di sumber daya Anda dan memberi tahu Anda jika ada masalah. Jika Anda menerima peringatan, Anda dapat mengakses file log untuk mendapatkan informasi tentang peristiwa tertentu. Untuk informasi selengkapnya, lihat [Panduan Pengguna Amazon CloudWatch Logs](#).

Melihat FleetWise log AWS IoT di konsol CloudWatch

Important

Sebelum Anda dapat melihat grup FleetWise log AWS IoT di CloudWatch konsol, pastikan bahwa yang berikut ini benar:

- Anda telah mengaktifkan login di AWS IoT FleetWise. Untuk informasi selengkapnya tentang pencatatan, lihat [Konfigurasi AWS pencatatan IoT FleetWise](#).

- Sudah ada entri log yang ditulis oleh AWS IoT operasi.

Untuk melihat FleetWise log AWS IoT Anda di konsol CloudWatch

1. Buka [konsol CloudWatch](#).
2. Pada panel navigasi, pilih Log, Grup log.
3. Pilih grup log.
4. Pilih Search log group (Cari grup log). Anda akan melihat daftar lengkap peristiwa log yang dibuat untuk akun Anda.
5. Pilih ikon perluas untuk melihat aliran individual dan temukan semua log yang memiliki tingkat logERROR.

Anda juga dapat memasukkan kueri di kotak pencarian Filter peristiwa. Misalnya, Anda dapat mencoba kueri berikut:

```
{ $.logLevel = "ERROR" }
```

Untuk informasi selengkapnya tentang membuat ekspresi filter, lihat [Filter dan sintaks pola](#) di Panduan Pengguna Amazon CloudWatch Logs.

Example entri log

```
{
  "accountId": "123456789012",
  "vehicleName": "test-vehicle",
  "message": "Unrecognized signal ID",
  "eventType": "MODELING_ERROR",
  "logLevel": "ERROR",
  "timestamp": 1685743214239,
  "campaignName": "test-campaign",
  "signalCatalogName": "test-catalog",
  "signalId": 10242
}
```

Jenis peristiwa sinyal

Tipe peristiwa	Deskripsi
MODELING_ERROR	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise berisi sinyal yang gagal divalidasi terhadap model kendaraan.</p> <p>Atribut: VehicleName, CampaignName,, SignalValue signalCatalogName, SignalValue, Min, Max, signalValueRange signalValueRange modelManifestName</p>
ILLEGAL_MESSAGE_FROM_EDGE	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise tidak sesuai dengan format yang diperlukan.</p> <p>Atribut: VehicleName, CampaignName, signalCatalogName</p>
DECODING_ERROR	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise berisi sinyal yang gagal mendekoder terhadap manifes decoder kendaraan.</p> <p>Atribut: campaignName,, signalCatalogName decoderManifestName, (opsional) signalName, (opsional) S3uri</p>

Jenis acara kampanye

Tipe peristiwa	Deskripsi
KENDARAAN_NOT_FOUND	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise, di mana kendaraan itu tidak diketahui.</p> <p>Atribut: VehicleName, CampaignName</p>

Tipe peristiwa	Deskripsi
CAMPAIGN_NOT_FOUND	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise, di mana kampanye tidak diketahui.</p> <p>Atribut: VehicleName (opsional), campaignName</p>
CAMPAIGN_INVALID	<p>Pesan yang dikirim dari kendaraan dan diterima oleh AWS IoT FleetWise, di mana kampanye tidak valid.</p> <p>Atribut: VehicleName (opsional), campaignName</p>

Jenis acara tujuan data kampanye

Tipe peristiwa	Deskripsi
TIMESTREAM_WRITE_ERROR	<p>AWSIoT FleetWise tidak dapat menulis pesan dari kendaraan ke tabel Amazon Timestream.</p> <p>Atribut: VehicleName, CampaignName,, timestreamDatabaseName timestreamTableName</p>
S3_WRITE_ERROR	<p>AWSIoT FleetWise tidak dapat menulis pesan dari kendaraan ke bucket Amazon Simple Storage Service (Amazon S3).</p> <p>Atribut: CampaignName, DestinationName</p>
S3_READ_ERROR	<p>AWSIoT FleetWise tidak dapat membaca kunci objek dari kendaraan di bucket Amazon Simple Storage Service (Amazon S3).</p> <p>Atribut: CampaignName, DestinationName</p>

Jenis acara AWS KMS kunci yang dikelola pelanggan

Tipe peristiwa	Deskripsi
KMS_KEY_ACCESS_DENIED	AWSIoT FleetWise tidak dapat menulis pesan dari kendaraan ke tabel Timestream atau bucket Amazon S3 karena kesalahan AWS KMS akses kunci ditolak.

Atribut

Semua entri CloudWatch Log menyertakan atribut ini:

`accountId`

Akun AWSID Anda.

`eventType`

Jenis peristiwa yang log dihasilkan. Nilai jenis acara tergantung pada peristiwa yang menghasilkan entri log. Setiap deskripsi entri log mencakup nilai `eventType` untuk entri log tersebut.

`logLevel`

Level log yang sedang digunakan. Untuk informasi selengkapnya, lihat [Tingkat log](#) di Panduan AWS IoT Core Pengembang.

`pesan`

Berisi detail spesifik tentang log.

`stempel waktu`

Stempel waktu milidetik epoch saat IoT AWS memproses log. FleetWise

Atribut opsional

CloudWatch Entri log secara opsional menyertakan atribut ini, tergantung pada: `eventType`

`decoderManifestName`

Nama manifes decoder yang berisi sinyal.

DestinationName

Nama tujuan untuk data kendaraan. Misalnya, nama bucket Amazon S3.

Nama Kampanye

Nama kampanye.

signalCatalogName

Nama katalog sinyal yang berisi sinyal.

Signalid

ID dari sinyal kesalahan.

Signalids

Daftar ID sinyal kesalahan.

SignalName

Nama sinyalnya.

signalTimestampEpochNona

Stempel waktu dari sinyal kesalahan.

SignalValue

Nilai sinyal kesalahan.

signalValueRangeMaks

Rentang maksimum sinyal kesalahan.

signalValueRangeMin

Rentang minimum sinyal kesalahan.

S3uri

Pengidentifikasi unik Amazon S3 dari file Amazon Ion dari pesan kendaraan.

timestreamDatabaseName

Nama basis data Timestream.

timestreamTableName

Nama tabel Timestream.

Nama Kendaraan

Nama kendaraannya.

Konfigurasi AWS pencatatan IoT FleetWise

Anda dapat mengirim data FleetWise log AWS IoT Anda ke grup CloudWatch log. CloudWatch Log memberikan visibilitas jika AWS FleetWise IoT gagal memproses pesan dari kendaraan. Misalnya, ini dapat terjadi karena konfigurasi yang salah atau kesalahan klien lainnya. Anda diberi tahu tentang kesalahan apa pun sehingga Anda dapat mengidentifikasi dan mengurangi masalah.

Sebelum Anda dapat mengirim log ke CloudWatch, Anda harus membuat grup CloudWatch log. Konfigurasi grup log dengan akun yang sama dan di Wilayah yang sama yang Anda gunakan dengan AWS IoT FleetWise. Saat Anda mengaktifkan login di AWS IoT FleetWise, berikan nama grup log. Setelah logging diaktifkan, AWS IoT FleetWise mengirimkan log ke grup log dalam CloudWatch aliran log.

Anda dapat melihat data log yang dikirim dari AWS IoT FleetWise di konsol. CloudWatch Untuk informasi selengkapnya tentang mengonfigurasi grup CloudWatch log dan melihat data log, lihat [Bekerja dengan Grup Log](#).

Izin untuk mempublikasikan log ke CloudWatch

Mengkonfigurasi logging untuk grup CloudWatch log memerlukan pengaturan izin yang dijelaskan di bagian ini. Untuk informasi tentang mengelola izin, lihat [Manajemen akses untuk AWS sumber daya](#) di Panduan Pengguna IAM.

Dengan izin ini, Anda dapat mengubah konfigurasi logging, mengonfigurasi pengiriman log untuk CloudWatch, dan mengambil informasi tentang grup log Anda.

```
{
  "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"
  }
]
```

Ketika tindakan diizinkan pada semua AWS sumber daya, itu ditunjukkan dalam kebijakan dengan "Resource" pengaturan "*". Ini berarti bahwa tindakan diizinkan pada semua AWS sumber daya yang didukung oleh setiap tindakan.

Konfigurasi login di AWS IoT FleetWise (konsol)

Bagian ini menjelaskan cara menggunakan FleetWise konsol AWS IoT untuk mengonfigurasi logging.

Untuk menggunakan FleetWise konsol AWS IoT untuk mengonfigurasi logging

1. Buka konsol [AWS IoT FleetWise](#).
2. Di panel kiri, pilih Pengaturan.
3. Di bagian Logging pada halaman Pengaturan, pilih Edit.
4. Di bagian CloudWatch logging, masukkan grup Log.
5. Untuk menyimpan perubahan Anda, pilih Kirim.

Setelah mengaktifkan logging, Anda dapat melihat data log Anda di [CloudWatch konsol](#).

Konfigurasi logging default di AWS IoT (FleetWise CLI)

Bagian ini menjelaskan cara mengonfigurasi logging untuk AWS IoT FleetWise dengan menggunakan CLI.

Anda juga dapat melakukan prosedur ini dengan API dengan menggunakan metode di AWS API yang sesuai dengan perintah CLI yang ditampilkan di sini. Anda dapat menggunakan operasi [GetLoggingOptions](#) API untuk mengambil konfigurasi saat ini dan operasi [PutLoggingOptions](#) API untuk memodifikasi konfigurasi.

Untuk menggunakan CLI untuk mengonfigurasi logging untuk IoT AWS FleetWise

1. Untuk mendapatkan opsi logging untuk akun Anda, gunakan `get-logging-options` perintah.

```
aws iotfleetwise get-logging-options
```

2. Untuk mengaktifkan logging, gunakan `put-logging-options` perintah.

```
aws iotfleetwise put-logging-options --cloud-watch-log-delivery  
logType=ERROR,logGroupName=MyLogGroup
```

di mana:

`logType`

Jenis log untuk mengirim data ke CloudWatch Log. Untuk menonaktifkan logging, ubah nilainya menjadi `OFF`.

`logGroupName`

Grup CloudWatch Log tempat operasi mengirimkan data ke. Pastikan Anda membuat nama grup log sebelum mengaktifkan logging untuk AWS IoT FleetWise.

Setelah Anda mengaktifkan logging, lihat [Cari entri log menggunakan AWS CLI](#).

Penebangan AWS IoT FleetWise Panggilan API menggunakan AWS CloudTrail

AWS IoT FleetWise terintegrasi dengan AWS CloudTrail, layanan yang menyediakan catatan tindakan yang diambil oleh pengguna, peran, atau AWS layanan di AWS IoT FleetWise. CloudTrail menangkap

semua panggilan API untuk AWS IoT FleetWise sebagai peristiwa. Panggilan yang diambil termasuk panggilan dari AWS IoT FleetWise konsol dan kode panggilan ke AWS IoT FleetWise Operasi API. Jika Anda membuat jejak, Anda dapat mengaktifkan pengiriman berkelanjutan CloudTrail peristiwa ke bucket Amazon S3, termasuk acara untuk AWS IoT FleetWise. Jika Anda tidak membuat konfigurasi jejak, Anda masih dapat melihat kejadian terbaru dalam konsol CloudTrail di Riwayat peristiwa. Menggunakan informasi yang dikumpulkan oleh CloudTrail, Anda dapat menentukan permintaan yang dibuat untuk AWS IoT FleetWise, alamat IP dari mana permintaan dibuat, yang membuat permintaan, ketika dibuat, dan detail tambahan.

Untuk mempelajari lebih lanjut tentang CloudTrail, lihat [AWS CloudTrail Panduan Pengguna](#).

AWS IoT FleetWise informasi dalam CloudTrail

CloudTrail diaktifkan pada akun AWS Anda saat Anda membuat akun tersebut. Ketika aktivitas terjadi di AWS IoT FleetWise, aktivitas itu dicatat dalam a CloudTrail acara bersama dengan lainnya AWS secara layanan di Riwayat acara. Anda dapat melihat, mencari, dan mengunduh peristiwa terbaru di akun AWS Anda. Untuk informasi lain, lihat [Melihat Peristiwa dengan Riwayat Peristiwa CloudTrail](#).

Untuk catatan acara yang sedang berlangsung di AWS akun, termasuk acara untuk AWS IoT FleetWise, buat jejak. Jejak memungkinkan CloudTrail untuk mengirim berkas log ke bucket Amazon S3. Secara default, ketika Anda membuat jejak di konsol tersebut, jejak diterapkan ke semua Wilayah AWS. Jejak mencatat kejadian dari semua Wilayah di partisi AWS dan mengirimkan berkas log ke bucket Amazon S3 yang Anda tentukan. Selain itu, Anda dapat membuat konfigurasi layanan AWS lainnya untuk menganalisis lebih lanjut dan bertindak berdasarkan data peristiwa yang dikumpulkan di log CloudTrail. Untuk informasi selengkapnya, lihat yang berikut:

- [Ikhtisar untuk membuat jejak](#)
- [CloudTrail layanan dan integrasi yang didukung](#)
- [Mengonfigurasi notifikasi Amazon SNS untuk CloudTrail](#)
- [Menerima CloudTrail log file dari beberapa Daerah](#)
- [Menerima CloudTrail log file dari beberapa akun](#)

Semua AWS IoT FleetWise tindakan dicatat oleh CloudTrail dan didokumentasikan dalam [AWS IoT FleetWise Referensi API](#). Misalnya, panggilan untuk tindakan `CreateCampaign`, `AssociateVehicleFleet`, dan `GetModelManifest` menghasilkan entri dalam file log CloudTrail.

Setiap entri peristiwa atau log berisi informasi tentang siapa yang membuat permintaan tersebut. Informasi identitas membantu Anda menentukan hal berikut:

- Jika permintaan tersebut dibuat dengan kredensial pengguna root atau IAM.
- Jika permintaan tersebut dibuat dengan kredensial keamanan sementara untuk peran atau pengguna federasi.
- Bahwa permintaan dibuat oleh layanan AWS lain.

Untuk informasi selengkapnya, lihat [Elemen userIdentity CloudTrail](#).

MemahamiAWSIoT FleetWiseentri berkas log

Jejak adalah konfigurasi yang memungkinkan pengiriman peristiwa sebagai berkas log ke bucket Amazon S3 yang telah Anda tentukan. Berkas log CloudTrail berisi satu atau beberapa entri log. Peristiwa mewakili satu permintaan dari sumber apa pun dan mencakup informasi tentang tindakan yang diminta, tanggal dan waktu tindakan, parameter permintaan, dan sebagainya. Berkas log CloudTrail bukan jejak tumpukan terurut dari panggilan API publik, sehingga berkas tersebut tidak muncul dalam urutan tertentu.

Contoh berikut menunjukkan entri log CloudTrail yang menunjukkan operasi *AssociateVehicleFleet*.

```
{
  "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"  
}
```

Riwayat dokumen untuk Panduan AWS Pengembang IoT FleetWise

Tabel berikut menjelaskan rilis dokumentasi untuk AWS IoT FleetWise.

Perubahan	Deskripsi	Tanggal
Pratinjau data sistem visi	Anda dapat menggunakan pratinjau data sistem penglihatan dari AWS IoT FleetWise untuk mengumpulkan dan mengatur data dari sistem penglihatan kendaraan, termasuk dari kamera, radar, dan lidar. Ini membuat data sistem visi terstruktur dan tidak terstruktur, metadata (ID peristiwa, kampanye, kendaraan), dan sensor standar (data telemetri) secara otomatis disinkronkan di cloud.	26 November 2023
AWS KMSkunci yang dikelola pelanggan	AWSIoT FleetWise sekarang mendukung kunci yang dikelola AWS KMS pelanggan . Anda dapat menggunakan kunci KMS untuk mengenkripsi data sisi server yang terkait dengan FleetWise sumber daya AWS IoT (katalog sinyal, model kendaraan, manifes decoder, kendaraan , dan konfigurasi kampanye pengumpulan data) yang	16 Oktober 2023

disimpan di dalamnya. AWS
Cloud

[Penyimpanan objek di
Amazon S3](#)

AWSIoT FleetWise sekarang mendukung penyimpanan data menggunakan Amazon Simple Storage Service (Amazon S3). Anda dapat menyimpan data yang dikumpulkan selama kampanye di Amazon S3, selain Amazon Timestream.

1 Juni 2023

[Ketersediaan umum](#)

Ini adalah rilis publik AWS IoT FleetWise.

September 27, 2022

[Rilis awal](#)

Ini adalah rilis pratinjau dari Panduan FleetWise Pengembang AWS IoT.

30 November 2021

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.