Contents
Overview .......................................................................................................................... 4
Cost ................................................................................................................................. 4
Architecture Overview ................................................................................................. 5
Solution Components ..................................................................................................... 6
Device Simulator Microservices ..................................................................................... 6
  Admin Microservice ....................................................................................................... 6
  Metrics Microservice .................................................................................................... 6
  Device Microservice ..................................................................................................... 6
  Profile Microservice ..................................................................................................... 7
Amazon DynamoDB ......................................................................................................... 7
Device Simulation ........................................................................................................... 11
  Device Types ................................................................................................................ 12
  Widgets .......................................................................................................................... 13
  Simulation Stage ........................................................................................................... 13
  Logging Level .............................................................................................................. 13
Device Simulator Console .............................................................................................. 14
  Dashboard .................................................................................................................. 14
  Device View ................................................................................................................ 15
  Device Types View ...................................................................................................... 16
  Custom Widget View .................................................................................................. 17
Authentication ................................................................................................................ 17
User Management .......................................................................................................... 17
Logging and Metrics ..................................................................................................... 17
Amazon ECS .................................................................................................................. 18
Considerations ................................................................................................................ 18
  Solution Updates ........................................................................................................ 18
  Regional Deployments ............................................................................................... 18
AWS CloudFormation Template ..................................................................................... 19
Automated Deployment ................................................................................................. 19
Prerequisites .................................................................................................................. 19
What We’ll Cover .................................................................................................................. 19

Step 1. Launch the Stack ........................................................................................................ 20

Step 2. Define Your Device Types .......................................................................................... 21

Step 3. Create a Pool of Widgets ............................................................................................ 22

Step 4. Test Your IoT Backend Service ................................................................................. 22

Getting Started ...................................................................................................................... 22

Access the IoT Device Simulator ........................................................................................... 23

Create a weather station device type ..................................................................................... 23

Create weather station widgets ............................................................................................ 24

View the simulated data .......................................................................................................... 25

  IoT Device Simulator console ............................................................................................... 25

  AWS IoT console .................................................................................................................. 25

Security .................................................................................................................................... 25

  IAM Roles ............................................................................................................................. 26

  Amazon CloudFront ............................................................................................................ 26

Additional Resources .............................................................................................................. 26

Appendix A: Automotive Module ........................................................................................... 26

  Add Your Mapbox Token (Optional) .................................................................................... 27

  Launch Simulated Vehicles .................................................................................................. 27

  Manage Simulated Vehicles ................................................................................................. 28

Appendix B: Managing Widgets ............................................................................................... 29

Appendix C: Managing Device Types ..................................................................................... 29

Appendix D: User Administration ........................................................................................... 30

  Add a new user ..................................................................................................................... 30

  Remove a user ...................................................................................................................... 30

  Disable a user ....................................................................................................................... 31

  Enable a user ......................................................................................................................... 31

  Change an existing user’s role ............................................................................................... 31

Appendix E: Collection of Operational Metrics ....................................................................... 32

Source Code ............................................................................................................................ 33
About This Guide

This implementation guide discusses architectural considerations and configuration steps for deploying the IoT Device Simulator on the Amazon Web Services (AWS) Cloud. It includes a link to a AWS CloudFormation template that launches, configures, and runs the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects, administrators, and DevOps professionals who have practical experience with IoT devices, and the AWS Cloud.

Overview

Amazon Web Services (AWS) provides many services to help customers build serverless IoT applications that gather, process, analyze, and act on connected device data, without having to manage any infrastructure. With AWS, customers also build a secure, agile, and scalable backend for their IoT applications. This eliminates the need for customers to develop and manage their own backend resources and can help reduce costs and increase productivity and innovation. But, it can be a challenge to test IoT applications and backend services without a large pool of physical, connected devices.

To help customers more easily test device integration and IoT backend services, AWS offers the IoT Device Simulator solution. This solution provides a web-based graphical user interface (GUI) console that enables customers to create and simulate hundreds of connected devices, without having to configure and manage physical devices, or develop time-consuming scripts. This solution is designed to work out-of-the-box, or you can use this solution as a reference implementation to build a custom simulation engine for your specific use case.

IoT Device Simulator provides a console that enables users to build a large fleet of virtual connected devices (widgets) from a user-defined template and simulate those widgets publishing data at regular intervals to AWS IoT. You can also monitor individual widgets from the simulator or observe how backend services are processing the data.

Cost

You are responsible for the cost of the AWS services used while running this solution. The total cost for running this solution depends on the amount of data being simulated, collected, stored, processed, and presented. For full details, see the pricing webpage for each AWS service you will be using in this solution.
Architecture Overview

Deploying this solution builds the following environment in the AWS Cloud.

Figure 1: IoT Device Simulator architecture

The AWS CloudFormation template deploys a device simulator API, which leverages Amazon API Gateway to invoke the solution’s microservices (AWS Lambda functions). These microservices provide the business logic to perform CRUD operations on virtual devices and device types, record simulation metrics, and perform administration tasks. These microservices interact with Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, AWS Identity and Access Management (IAM), and Amazon CloudWatch Logs to provide data storage, management, and logging functions.

The solution also deploys an Amazon Virtual Private Cloud (Amazon VPC) network topology with two public subnets and two private subnets that contains the solution’s simulation engine, which runs in Amazon Elastic Container Service (Amazon ECS) containers provisioned by AWS Fargate. The VPC also includes a NAT gateway.

The solution creates a web console and deploys it into an Amazon S3 bucket configured for static website hosting. Amazon CloudFront is used to restrict access to the solution’s website bucket contents. During initial configuration, the solution also creates a default administrator role and sends an access invite to a customer-specified user email. The solution uses an Amazon Cognito user pool to manage user access to the console and the device simulator API.
When the device simulator API receives an authorized request, Amazon API Gateway invokes the appropriate Lambda function. The Lambda function returns the execution results to the API, which returns the results to the simulator console.

When a device simulation request is received, the device microservice sends the request to a simulation queue in Amazon Simple Queue Service (Amazon SQS). The simulation engine polls the simulation queue for simulation start and stop requests.

When a start simulation request is received, the solution will spawn a virtual device based on the request and start publishing simulated data to the defined AWS IoT endpoint for the duration defined in the device type definition. Each simulation runs until the defined execution duration expires or a stop request is received. When a stop simulation request is received, the solution will terminate the simulation based on the request and update the device catalog.

Solution Components

Device Simulator Microservices
The IoT Device Simulator microservices are a series of AWS Lambda functions that provide the business logic and data access layer for all device simulation operations. Each Lambda function assumes an AWS Identity and Access Management (IAM) role with least privilege access (minimum permissions necessary) to perform its designated functions.

Admin Microservice
The `iot-sim-admin-service` Lambda function processes device simulator API requests sent to the `/admin/*` endpoints. All `/admin/*` API endpoints are configured as Lambda proxy endpoints that pass the full request payload to the `iot-sim-admin-service` function. The admin microservice handles all administrative services, including users and general settings.

Metrics Microservice
The `iot-sim-metrics-service` Lambda function processes device simulator API requests sent to the `/metrics/*` endpoints. All `/metrics/*` API endpoints are configured as Lambda proxy endpoints that pass the full request payload to the `iot-sim-metrics-service` function. The metrics microservice handles all metrics operations for the IoT Device Simulator.

Device Microservice
The `iot-sim-device-service` Lambda function processes device simulator API requests sent to the `/device/*` endpoints. All `/device/*` API endpoints are configured as Lambda
proxy endpoints that pass the full request payload to the `iot-sim-device-service` function. The device microservice handles all device and device type operations, including list, add device, remove device, list device types, add device type, update device type, start simulation, and stop simulation.

**Profile Microservice**

The `iot-sim-profile-service` Lambda function processes device simulator API requests sent to the `/profile/*` endpoints. All `/profile/*` API endpoints are configured as Lambda proxy endpoints that pass the full request payload to the `iot-sim-profile-service` function. The profile microservice handles reading user profile information for the IoT Device Simulator.

**Amazon DynamoDB**

The IoT Device Simulator uses Amazon DynamoDB to persist metadata for devices, device types, settings, and metrics. Each DynamoDB table is provisioned using Amazon DynamoDB on-demand, a flexible billing option for DynamoDB capable of serving thousands of requests per second without capacity planning. DynamoDB on-demand offers simple pay-per-request pricing for read and write requests so that you only pay for what you use, making it easy to balance costs and performance. On-demand mode instantly accommodates workloads as they ramp up or down to any previously observed traffic level. If the level of traffic hits a new peak, DynamoDB adapts rapidly to accommodate the workload.


The `iot-sim-device-widgets` table stores the following information on virtual devices:

- **Category**: The category of the device widget. This is defined by the device type.
- **CreatedAt**: The date and time (in UTC) when the device widget was created
- **EndedAt**: The date and time (in UTC) when the last simulation run ended
- **Id**: The unique identifier for the device widget. This is generated automatically.
- **Metadata**: The metadata associated with the device widget
- **Runs**: The number of simulation runs performed by the device widget
- **Stage**: The current simulation stage of the device widget. For more information, see [Simulation Stage](#).
- **StartedAt**: The date and time (in UTC) when the last simulation run started
- **Subcategory**: The sub category of the device widget. This is defined by the device type.
- **TypeId**: The device type identifier
- **UpdatedAt**: The date and time (in UTC) when the device widget was updated
- **UserId**: The user id of the owner of the device widget
- **Visibility**: The visibility (private or shared) of the device types for other IoT Device Simulator users. Use `private` to prevent other users from creating devices using the device type. Use `shared` to allow other users to create devices using the device type.

```json
{
    "category": "custom widget",
    "createdAt": "2018-04-09T20:01:58Z",
    "endedAt": "2018-04-09T20:04:13Z",
    "id": "B1OJ6UHtoG",
    "metadata": {},
    "runs": 1,
    "stage": "sleeping",
    "startedAt": "2018-04-09T20:02:10Z",
    "subCategory": "weather station",
    "typeId": "rkbZDGso6f",
    "shared": "private",
    "updatedAt": "2018-04-09T20:04:13Z",
    "userId": "sample_user"
}
```

**Figure 2: Sample device record**

The `iot-sim-device-types` table stores the following information on device types:

- **CreatedAt**: The date and time (in UTC) when the device type was created
- **Custom**: Identifies whether the device type is user created (true) or system generated (false)
- **Name**: The name of the device widget
- **Spec**: The specification of the device type
- **Duration**: The duration a simulation will run for a particular device type
- **Interval**: The interval at which devices will publish data to AWS IoT
- **Payload**: The JSON definition of the attributes to be simulated
- **Topic**: The AWS IoT topic where the device will publish data
- **TypeId**: The device type identifier
- **UpdatedAt**: The date and time (in UTC) when the device type was updated
- **UserId**: The user id of the owner of the device type
{  
  "createdAt": "2018-03-05T19:14:00Z",
  "custom": true,
  "name": "weather station",
  "spec": {
    "duration": "120000",
    "interval": 2000,
    "payload": [
      {
        "_id": "HJpJGapuf",
        "name": "stationid",
        "smax": 20,
        "smin": 10,
        "static": true,
        "type": "string"
      },
      {
        "_id": "H1ea1fTTuM",
        "dmax": 99,
        "dmin": 0,
        "imax": 200,
        "imin": 0,
        "name": "temperature",
        "precision": 2,
        "type": "float"
      },
      {
        "_id": "H1f61z6aOz",
        "max": 100,
        "min": 0,
        "name": "humidity",
        "type": "int"
      },
      {
        "_id": "HkQa1G6p_M",
        "name": "timestamp",
        "tsformat": "default",
        "type": "timestamp"
      },
      {
        "_id": "r1IfpsQ3M",
        "arr": [
          "running",
          "stopped",
          "starting",
          "error",
          "warning"
        ],
        "name": "state",
        "type": "pickOne"
      }
    ],
  }
}
Figure 3: Sample device type record

The `iot-sim-metrics` table stores the following logging information:

- **CreatedAt**: The date and time (in UTC) when the metric record was created
- **DeviceBreakdown**: A breakdown of the total number of simulation runs per device type by a specific user for the current month
- **Id**: The unique identifier for the device widget. This is generated automatically.
- **Simulations**: The number of simulation runs broken down by category
- **MonthlyRuns**: A breakdown of the last six months of simulation runs
- **TotalDuration**: The total number of simulation minutes by a user
- **TotalRuns**: The total number of simulation runs by a user
- **UpdatedAt**: The date and time (in UTC) when the metric record was updated
- **UserId**: The user id of the simulation user
Figure 4: Sample metrics record

The `iot-sim-settings` table stores the following settings information:

- **SettingID**: The unique identifier of the setting record
- **CreatedAt**: The date and time (in UTC) when the setting was created
- **Setting**: The JSON description of the configuration setting details
- **Type**: The type of setting record (config)
- **UpdatedAt**: The date and time (in UTC) when the setting was updated

**Device Simulation**

The IoT Device Simulator leverages Amazon Simple Queue Service (Amazon SQS), Amazon Elastic Container Service (Amazon ECS), and AWS Fargate to simulate virtual devices sending data to AWS IoT endpoints.

Users make simulation requests via the included web console. Simulation requests are added to the Amazon SQS queue where they are stored until they are processed.
Amazon ECS containers provisioned by AWS Fargate contain a simulation engine that periodically polls the simulation queue for simulation requests. The simulation engine launches a virtual device and starts the device publishing simulated data to the AWS IoT endpoint. After the specified duration, the simulation engine stops the simulation, terminates the virtual device, and updates the device state and metrics in the iot-sim-device-widgets table.

Device Types
Device types are used to define the type of data your simulated IoT devices will send. When you create a device type, you define the structure of the data and specific attributes for each item in the payload. Device types contain the follow properties:

- **Name**: The name of the device type
- **Data topic**: The topic where the data will be sent in AWS IoT
- **Data transmission duration**: The duration, in milliseconds, your device will send data
- **Data transmission interval**: The interval, in milliseconds, at which your device will send data

Attributes
Attributes define what each payload contains. Attributes include the follow fields:

- **Attribute name**: The name of the attribute
- **Attribute data type**: The data type of the payload. The following data types are available:
  - **Boolean**: Sends a random true or false value based on seeding. You set a minimum value, a maximum value, and seed value.
  - **Decay**: Generates a number that exponentially decreases at a rate proportional to its current value.
  - **Float**: Sends a random decimal value. You set the precision of the value, a minimum value, a maximum value, a decimal precision minimum value, and a decimal precision maximum value.
  - **Integer**: Sends a random integer value. You set a minimum value and a maximum value.
  - **Object**: Adds a nested object to the payload.
  - **Sinusoidal**: Generates values on a mathematical curve that represents periodic oscillation with noise.
- **String**: Sends a random string value. You set a minimum length and a maximum length of the string.

- **Unique Identifier**: Sends a random UUID value

- **Unique Short Identifier**: Sends a random short UUID value

- **UTC Timestamp**: Sends a random UTC timestamp in `YYYY-MM-DDTHH:mm:ss`. For example, `2018-04-10T12:21:22`. You can also specify a timestamp in Unix time format. For example, `1523377324`.

- **Location**: Sends a random latitude and longitude coordinated within a radius of a defined point. You specify a latitude for the center position, a longitude for the center position, and a radius from the center position.

- **Pick One from Array**: Sends a random string value from a user-defined, comma-separated list of strings.

- **Static**: Choose whether the value stays the same for the duration of the simulation

- **Default Value**: A fixed value that will always be sent in the payload

### Widgets

Widgets are virtual IoT devices that send simulated data to AWS IoT endpoints. With the IoT Device Simulator, you can create up to 100 widgets at a time. For example, to launch 400 widgets, you can create four batches of 100 widgets. The solution allows you to run up to 1,000 simultaneous simulations across the simulation engine. If you request more than 1,000 simulations at a given time, they will be queued and executed when the number of current simulations is less than 1,000.

### Simulation Stage

Device widgets can be in one of three stages:

- **Simulating**: The device widget is sending simulated data to the IoT endpoint

- **Provisioning**: The device widget is launching

- **Sleeping**: The device widget is stopped and not sending simulated data

You can view each widget’s simulation stage as well as a breakdown of the total number of widgets in each stage in the web console’s device view.

### Logging Level

By default, this solution logs informational messages for the simulation engine. To change the logging level, open the IoT Device Simulator console and navigate to the **Settings** menu.
option. On the **Simulation Engine** tab, you can change the **Logging Level** to **INFO**, **ROBUST**, or **DEBUG**.

<table>
<thead>
<tr>
<th>Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFO</td>
<td>Logs will include high-level information about how the simulation engine is operating.</td>
</tr>
<tr>
<td>ROBUST</td>
<td>Logs will include information on anything that can potentially cause inconsistencies in the simulation engine but might not necessarily cause the engine to fail.</td>
</tr>
<tr>
<td>DEBUG</td>
<td>Logs will include information that might be helpful when debugging a problem with the simulation engine.</td>
</tr>
</tbody>
</table>

**Device Simulator Console**

The IoT Device Simulator features a web console that you use to manage simulations. The console displays information about virtual devices and device types, simulation states, and user profiles. You use the console to create and terminate virtual devices, start and stop simulations, and view metrics.

**Dashboard**

The console includes a dashboard you can use to view how many simulations you have run, how many simulation hours you have run, and a breakdown of the different device types you have created.
Figure 5: Sample dashboard

Device View
The console also includes a device view that shows all devices in your pool of created devices, as well as the simulation state of the devices.
Device Types View
The console includes a device types view that displays a list of device types, and allows you to create new device types.
Custom Widget View
The console includes a custom widget view that displays your list of widgets, and allows you to create and delete widgets, and start and stop simulations. This view also provides a count of the widgets in various simulation stages.

![Sample widgets view](image)

**Figure 8: Sample widgets view**

Authentication
This solution takes advantage of the user authentication features of Amazon Cognito User Pools. After successfully authenticating a user, Amazon Cognito issues a JSON web token that is used to allow the console to submit requests to the device simulator API. HTTPS requests are sent by the console to the simulator API with the authorization header that includes the token.

User Management
After the device simulator is deployed, administrators can invite privileged users and customize their permissions to implement granular access-control policies. Administrators can also change settings. For more information, see Appendix D.

Logging and Metrics
The device simulator solution logs API calls, latency, and error rates to Amazon CloudWatch which you can use to set alarms based on defined thresholds. The solution also monitors traffic at the REST API level. Optionally, you can enable detailed metrics for each method of
the device simulator REST API from the Amazon API Gateway deployment configuration console. Detailed metrics will incur an extra cost.

Amazon ECS

This solution leverages Amazon Elastic Container Service (Amazon ECS) for the solution’s simulation engine. Amazon ECS uses AWS Identity and Access Management (IAM) service-linked roles. A service-linked role is a unique type of IAM role that is linked directly to an AWS service. Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf.

Amazon ECS uses the AWSServiceRoleForECS role to allow the service to manage your cluster. If you have not already created a service-linked role for Amazon ECS, you must create one before you use this solution. For instructions on how to create the role, see Creating a Service-Linked Role for Amazon ECS in the Amazon ECS Developer Guide. For more information on service-linked roles, see Using Service Linked Roles in the AWS Identity and Access Management User Guide.

Considerations

Solution Updates

IoT Device Simulator version 2.1.1 uses the most up-to-date Node.js runtime. Version 2.0 uses the Node.js 8.10 runtime, which reaches end-of-life on December 31, 2019. In January, AWS Lambda will block the create operation and, in February, Lambda will block the update operation. For more information, see Runtime Support Policy in the AWS Lambda Developer Guide.

To continue using this solution with the latest features and improvements, you can update the stack.

Regional Deployments

This solution uses the AWS Fargate service, which is currently available in specific AWS Regions only. Therefore, you must launch this solution in a region where AWS Fargate is available.

1 For the most current AWS Fargate availability by region, see https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/
AWS CloudFormation Template

This solution uses AWS CloudFormation to automate the deployment of the IoT Device Simulator. It includes the following AWS CloudFormation template, which you can download before deployment:

[iot-device-simulator.template]: Use this template to launch the solution and all associated components. The default configuration deploys an Amazon Virtual Private Cloud network topology, Amazon API Gateway, AWS Lambda functions, Amazon Simple Storage Service buckets, Amazon DynamoDB tables, AWS Identity and Access Management roles, Amazon CloudWatch Logs, a device simulator graphical user interface, an Amazon Cognito user pool, Amazon Simple Queue Service, Amazon Elastic Container Service, an Amazon CloudFront distribution, and AWS Fargate. You can also customize the template based on your specific network needs.

Automated Deployment

Before you launch the automated deployment, please review the architecture, configuration, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the IoT Device Simulator into your account.

**Time to deploy:** Approximately 10 minutes

**Prerequisites**

Before you use this solution, you must have an existing service-linked role for Amazon ECS. If you have not already created one, see [Creating a Service-Linked Role for Amazon ECS](#) in the [Amazon ECS Developer Guide](#).

**What We’ll Cover**

The procedure for deploying this architecture on AWS consists of the following steps. For detailed instructions, follow the links for each step.

**Step 1. Launch the Stack**

- Launch the AWS CloudFormation template into your AWS account
- Enter values for required parameters: **Stack Name, Administrator Name** and **Administrator Email**

**Step 2. Define Your Device Types**

- Define the data your simulated IoT devices will send
Step 3. Create a Pool of Widgets

- Create a pool of simulated IoT devices

Step 4. Test Your IoT Backend Services

- Subscribe to the applicable IoT topic and view data

Step 1. Launch the Stack

This automated AWS CloudFormation template deploys the device simulator.

**Note:** You are responsible for the cost of the AWS services used while running this solution. See the Cost section for more details. For full details, see the pricing webpage for each AWS service you will be using in this solution.

1. Sign in to the AWS Management Console and click the button to the right to launch the `iot-device-simulator` AWS CloudFormation template.

   You can also download the template as a starting point for your own implementation.

2. The template is launched in the US East (N. Virginia) Region by default. To launch this solution in a different AWS Region, use the region selector in the console navigation bar.

   **Note:** This solution uses the AWS Fargate service, which is currently available in specific AWS Regions only. Therefore, you must launch this solution in a region where AWS Fargate is available. For the most current service availability by region, see AWS service offerings by region.

3. On the Select Template page, verify that you selected the correct template and choose Next.

4. On the Specify Details page, assign a name to your solution stack.

5. Under Parameters, review the parameters for the template and modify them as necessary. This solution uses the following default values.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator Name</td>
<td><code>&lt;Requires input&gt;</code></td>
<td>The user name for the initial solution administrator. After the solution is deployed, this administrator can create and manage other users, including additional administrators.</td>
</tr>
<tr>
<td>Administrator Email</td>
<td><code>&lt;Requires input&gt;</code></td>
<td>Email address of the administrator user. After launch, an email will be sent to this address with console login instructions.</td>
</tr>
</tbody>
</table>
6. Choose **Next**.

7. On the **Options** page, choose **Next**.

8. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.

9. Choose **Create** to deploy the stack.

   You can view the status of the stack in the AWS CloudFormation Console in the **Status** column. You should see a status of **CREATE_COMPLETE** in approximately 10 minutes.

   The solution sends an email invitation to join the IoT Device Simulator console.

10. In the email, follow the instructions log in to the console.

    **Note:** In addition to the AWS Lambda functions that make up the device simulator microservices, this solution includes the `iot-device-sim-helper` Lambda function, which runs only during initial configuration or when resources are updated or deleted.

    When running this solution, the `iot-device-sim-helper` function is inactive. However, do not delete the `iot-device-sim-helper` function as it is necessary to manage associated resources.

**Step 2. Define Your Device Types**

Use this procedure to define the data each of your simulated IoT devices will send. Create the structure of the data and the specific properties and type for each item in the payload.

1. Sign in to the IoT Device Simulator console and in the navigation pane, choose **Device Types**.

2. Select + **Add Device Types**.

3. On the **Device Type Definition** page, enter the appropriate information. For more information, see **Device Types**. Note the value you entered for the **Data topic** attribute.

4. To format the payload, select **Add Attribute**.
5. In the **Message Attribute** window, enter the appropriate information. For more information, see **Attributes**.

6. Select **Submit**.

7. Repeat steps 4-6 for each attribute for your device payload.

8. Select **Save**.

### Step 3. Create a Pool of Widgets

Use this procedure to define the number of simulated IoT devices (widgets) you will launch.

1. On the IoT Device Simulator console page, in the navigation menu, select **Widgets**.

2. Select **+ Add Widgets**.

3. In the **Create a widget** window, specify a device type and the number of widgets you want to launch. You can create up to 100 widgets at a time.

4. Select **Submit**.

5. To view a specific device, select **View** for the specific widget. This shows metadata about the widget and the messages being received by AWS IoT during the simulation.

### Step 4. Test Your IoT Backend Service

Use this procedure to test your IoT backend service.

1. Sign in to the AWS IoT Console and in the left navigation pane, choose **Test**.

2. Subscribe to the applicable topic. You can find the topic in the **Data topic** attribute in your device type definition.

3. View the simulated data flowing into the AWS IoT Core service in your account.

### Getting Started

Get started with the IoT Device Simulator by creating a device type and starting a simulation of 25 devices. For this tutorial, you will create a weather station device type that generates temperature, wind speed, and humidity data that will be sent to an AWS IoT topic (/weather/data), and 25 weather station widgets.
Access the IoT Device Simulator
After you [deploy the solution](#), sign into the IoT Device Simulator console using the link in the email invitation you receive. When you sign in to the console for the first time, use the temporary password that is included in the email invitation. You will be prompted to change your password upon successful authentication.

Create a weather station device type
Use the following procedure to create a weather station device type.

1. Sign in to the IoT Device Simulator console and in the navigation pane, choose **Device Types**.

2. Select **Add Device Types**.

3. In the **Device Type Definition** section, enter the following information:
   - For **Device Type Name**, enter *WeatherStation*.
   - For **Visibility**, enter *private*.
   - For **Data Topic**, enter /weather/data
   - For **Data Transmission Duration**, enter 120000.
   - For **Data Transmission Interval**, enter 2000.

4. Under **Message Payload**, select **Add Attribute**.

5. In the **Message Attribute** window, enter the following information:
   - For **Attribute name**, enter *deviceId*.
   - For **Attribute data type**, select *DEVICE ID*.

6. Select **Submit**.

7. Repeat steps 4 – 6 to create the following attributes:
   - **Attribute name**: temperature
     - **Attribute data type**: FLOAT
     - **Float precision**: 2
     - **Integer minimum value**: 30
     - **Integer maximum value**: 130
• Decimal precision minimum value: 0
• Decimal precision maximum value: 99
• **Attribute name:** windspeed
  • **Attribute data type:** FLOAT
  • Float precision: 2
  • Integer minimum value: 0
  • Integer maximum value: 300
  • Decimal precision minimum value: 0
  • Decimal precision maximum value: 99
• **Attribute name:** humidity
  • **Attribute data type:** INTEGER
  • Minimum value: 0
  • Maximum value: 300
• **Attribute name:** timestamp
  • **Attribute data type:** UTC TIMESTAMP
  • **Timestamp format:** Default (YYYY-MM-DDTHH:mm:ss)

8. Select **Save.**

**Create weather station widgets**

Use this procedure to create 25 weather station widgets.

1. On the IoT Device Simulator console page, in the navigation pane, select **Widgets.**

2. Select **+ Add Widgets.**

3. In the **Create a widget** window, enter the following information:
   - For **Device Type**, select **WeatherStation.**
   - For **Number of widgets**, enter **25.**

4. Select **Submit.**

The solution provisions 25 weather station widgets and starts sending simulated data for those widgets to the AWS IoT topic `/weather/data.`
View the simulated data
You can view simulated IoT Device Simulator data in the device simulator console or in the AWS IoT Console.

IoT Device Simulator console
To view simulated data in the solution console, use the following procedure:

1. On the IoT Device Simulator console page, in the navigation menu, select **Widgets**.

2. On the **Device Widgets** page, find a weather station widget with a **Stage** value of **RUNNING**.

3. Select **View**.

   The simulated data will be displayed.

   **Note:** If none of the weather station widgets have a **Stage** value of **RUNNING**, select **Start** next to any weather station widget to start a simulation. Then, select **View** to view the simulated data for the running widget.

AWS IoT console
To view simulated data in the AWS IoT console, use the following procedure:

1. Navigate to the **AWS IoT console**.

2. In the navigation pane, choose **Test**.

3. For **Subscription topic**, enter `/weather/data`.

4. Select **Subscribe to topic**.

   If your weather station widgets are simulating data, the simulated data will be displayed on the page.

   **Note:** If the data is not displayed, verify that a simulation is running. In a separate browser window, navigate to the **Device Widgets** page in the IoT Device Simulator console. Select **Start** next to any weather station widget to start a simulation. Check the `/weather/data` topic in the AWS IoT console. The data will be displayed.

Security
When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. For more information about security on AWS, visit the AWS Security Center.

IAM Roles
AWS Identity and Access Management (IAM) roles enable customers to assign granular access policies and permissions to services and users on the AWS Cloud. The IoT Device Simulator creates several IAM roles, including roles that grant the device simulator AWS Lambda functions access to the other AWS services used in this solution. These roles are necessary to allow the services to simulate devices in your account.

Amazon CloudFront
This solution deploys a static website hosted in an Amazon S3 bucket. To help reduce latency and improve security, this solution includes an Amazon CloudFront distribution with an origin access identity, which is a special CloudFront user that helps restrict access to the solution’s website bucket contents. For more information, see Restricting Access to Amazon S3 Content by Using an Origin Access Identity.

Additional Resources
AWS services
- AWS Lambda
- AWS Fargate
- Amazon Elastic Container Service
- Amazon DynamoDB
- Amazon Cognito
- Amazon CloudWatch
- Amazon Virtual Private Cloud
- AWS CloudFormation
- Amazon Simple Queue Service
- AWS IoT
- Amazon Simple Storage Service
- Amazon API Gateway
- AWS Identity and Access Management
- Amazon CloudFront

Appendix A: Automotive Module
The IoT Device Simulator includes a pre-built automotive module that you can use to simulate vehicle telemetry data using pre-defined device types. The automotive module uses a power train simulation model to generate simulated vehicle telemetry data.

**Add Your Mapbox Token (Optional)**

The module leverages the location features of Mapbox to provide a map for your simulated vehicles. To display the map in the automotive module, you must register for a free Mapbox developer account. After you register, add your Mapbox token to the IoT Device Simulator.

1. On the IoT Device Simulator console page, in the navigation menu, select **Settings**.

2. In the **General** tab, enter your Mapbox token.

3. Select **Save**.

   **Note:** The Mapbox Token setting is cached when you log in to the device simulator console. If you change to a new token, you must log out of the console and log back in for the change to take effect.

**Launch Simulated Vehicles**

Use the following procedure to launch virtual connected vehicles.

1. On the IoT Device Simulator console page, in the navigation menu, select **Automotive**.

2. On the **My Automotive Fleet** page, select **+ Add Vehicles**.

3. In the **Create vehicles** window, specify the number of vehicles you want to simulate. Enter a value between 1 and 100.

4. Select **Submit**.
Manage Simulated Vehicles

To monitor a specific vehicle, on the My Automotive Fleet page, find the applicable vehicle and select View. This shows metadata about the vehicle, the route it is driving, and the telemetry messages being received by AWS IoT during the simulation.

To start a specific vehicle, find the applicable vehicle and select Start. This starts sending simulated data for that vehicle to the AWS IoT topic. To stop a specific vehicle, find the applicable vehicle and select Stop.

To start multiple vehicles, on the My Automotive Fleet page, select the checkbox next to each applicable vehicle. Then, select Start Vehicles. To start all vehicles on the page, select the checkbox in the table header. Then, select Start Vehicles. To stop multiple vehicles, select the checkbox next to each applicable vehicle. Then, select Stop Vehicles. To stop all vehicles on the page, select the checkbox in the table header. Then, select Stop Vehicles.

To remove a vehicle, on the My Automotive Fleet page, find the applicable vehicle and select Delete. In the popup window, select Yes, delete it! to delete the vehicle, or Cancel to abort the deletion.

While the simulated vehicles this solution creates are designed to be used out-of-the-box, you can configure how the automotive module sends data using the Configuration button at the top of the screen.
Appendix B: Managing Widgets

Widgets are virtual IoT devices that send simulated data to AWS IoT endpoints. The IoT Device Simulator console helps make it easier to manage your widgets. You can create, edit, and delete single widgets, or multiple widgets at a time. You can also start and stop simulations for a single widget, or multiple widgets at a time.

To view your widgets, navigate to the IoT Device Simulator console. In the navigation pane, select **Widgets**. A list of all widgets associated with your account will be displayed on the **Device Widgets** page.

To view the details of a specific widget, on the **Device Widgets** page, find the applicable widget and select **View**. This shows the widget details including ID, device type, status, the number of simulations run, the time the last simulation ran, and the time the widget was created.

To start a specific widget, find the applicable widget and select **Start**. This starts sending simulated data for that widget to the AWS IoT topic. To stop a specific widget, find the applicable widget and select **Stop**.

To start multiple widgets, on the **Device Widgets** page, select the checkbox next to each applicable widget. Then, select **Start Devices**. To start all widgets on the page, select the checkbox in the table header. Then, select **Start Devices**. To stop multiple widgets, select the checkbox next to each applicable widget. Then, select **Stop Devices**. To stop all widgets on the page, select the checkbox in the table header. Then, select **Stop Devices**.

To remove a widget, on the **Device Widgets** page, find the applicable widget and select **Delete**. In the popup window, select **Yes, delete it!** to delete the widget, or **Cancel** to abort the deletion.

Appendix C: Managing Device Types

Device types are used to define the type of data your simulated IoT devices will send. The IoT Device Simulator console helps make it easier to manage your device types. With the console, you can view and edit your device types.

To view your device types, navigate to the IoT Device Simulator console. In the navigation pane, select **Device Types**. A list of all device types associated with your account will be displayed on the **Device Types** page.
To view the details of a specific device type, on the Device Types page, find the applicable device type and select Edit. This shows the device type definition details including the name, the visibility, the data topic, the data transmission duration and interval, and the message payload. To edit the device type, change the applicable values. You can also remove existing attributes from or add new attributes to the message payload. Select Save to save the changes or Cancel to abort the changes.

Appendix D: User Administration

Administrators can invite privileged users and customize their permissions to implement granular access-control policies. Administrators can also change settings.

Add a new user
Use this procedure to add a new user to the IoT Device Simulator console.

1. On the IoT Device Simulator console page, in the navigation menu, select Users.

2. On the Users page, select Invite User.

3. In the Invite User window, enter the following information:
   - For Name, enter the name of the user.
   - For Email Address, enter the user’s email address.

4. Select Submit.

   The solution creates a user record and sends an email invitation to the email address you entered.

Remove a user
Use this procedure to remove a user from the IoT Device Simulator console.

1. On the IoT Device Simulator console page, in the navigation menu, select Users.

2. On the Users page, select Review.

3. On the User:<name> page, select Disable User.

4. Select Delete User.
Disable a user
Use this procedure to prevent a user from signing into the IoT Device Simulator console. When you disable a user, they will not be able to sign into the console, but their account information will still be listed on the Users page. You can enable a user that has been disabled.

1. On the IoT Device Simulator console page, in the navigation menu, select Users.

2. On the Users page, select Review.

3. On the User:<name> page, select Disable User.

Enable a user
Use this procedure to allow user that has been disabled to sign into the IoT Device Simulator console.

1. On the IoT Device Simulator console page, in the navigation menu, select Users.

2. On the Users page, select Review.


Change an existing user’s role
Use this procedure to change an existing user’s role. There are two roles available: administrator and member. Members can only run simulations. They cannot create device types or widgets.

1. On the IoT Device Simulator console page, in the navigation menu, select Users.

2. On the Users page, select Review.

3. On the User:<name> page, under Group Assignments, select the applicable checkbox.

Note: You cannot change the group assignment for users that have been disabled. To change their role, you must enable them first.

4. Select Save User.
Appendix E: Collection of Operational Metrics

This solution includes an option to send anonymous operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. When enabled, the following information is collected and sent to AWS:

- **Solution ID**: The AWS solution identifier
- **Unique ID (UUID)**: Randomly generated, unique identifier for each solution deployment
- **Timestamp**: Data-collection timestamp
- **Simulations Run**: The number of simulations started
- **Simulation Duration**: The duration, in minutes, of the simulation
- **Simulation Category**: The device widget category of the simulation

Note that AWS will own the data gathered via this survey. Data collection will be subject to the [AWS Privacy Policy](#). To opt out of this feature, complete one of the following tasks:

Modify the AWS CloudFormation template mapping section as follows:

```yaml
Mappings:
  Send:
    AnonymousUsage:
      Data: "Yes"
```

or

```yaml
Mappings:
  Send:
    AnonymousUsage:
      Data: "No"
```
Source Code

You can visit our GitHub repository to download the templates and scripts for this solution, and to share your customizations with others.

Document Revisions

<table>
<thead>
<tr>
<th>Date</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 2018</td>
<td>Initial release</td>
</tr>
<tr>
<td>December 2018</td>
<td>Added information about the Amazon CloudFront distribution for the static website hosted in the Amazon S3 bucket</td>
</tr>
<tr>
<td>March 2019</td>
<td>Added information about Amazon DynamoDB on-demand, the Amazon ECS service-linked role, additional device type attributes and functionality, and managing device types, widgets, and users</td>
</tr>
<tr>
<td>December 2019</td>
<td>Added information on support for Node.js update</td>
</tr>
</tbody>
</table>

Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS’s current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS’s products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. AWS’s responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

The IoT Device Simulator is licensed under the terms of the Apache License Version 2.0 available at https://www.apache.org/licenses/LICENSE-2.0.

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