

Tutorial: Sending an Amazon SNS notification

[PDF \(iot-dg.pdf#iot-sns-rule\)](#)

[Kindle \(https://www.amazon.com/dp/B07JBRCWWZ\)](https://www.amazon.com/dp/B07JBRCWWZ)

This tutorial demonstrates how to create an AWS IoT rule that sends MQTT message data to an Amazon SNS topic so that it can be sent as an SMS text message.

In this tutorial, you create a rule that sends message data from a weather sensor to all subscribers of an Amazon SNS topic, whenever the temperature exceeds the value set in the rule. The rule detects when the reported temperature exceeds the value set by the rule, and then creates a new message payload that includes only the device ID, the reported temperature, and the temperature limit that was exceeded. The rule sends the new message payload as a JSON document to an SNS topic, which notifies all subscribers to the SNS topic.

What you'll learn in this tutorial:

- How to create and test an Amazon SNS notification
- How to call an Amazon SNS notification from an AWS IoT rule
- How to use simple SQL queries and functions in a rule query statement
- How to use the MQTT client to test an AWS IoT rule

This tutorial takes about 30 minutes to complete.

In this tutorial, you'll:

- [Step 1: Create an Amazon SNS topic that sends an SMS text message \(#iot-sns-rule-create-sns-topic\)](#)
- [Step 2: Create an AWS IoT rule to send the text message \(#iot-sns-rule-create-rule\)](#)
- [Step 3: Test the AWS IoT rule and Amazon SNS notification \(#iot-sns-rule-test-rule\)](#)
- [Step 4: Review the results and next steps \(#iot-sns-rule-review-results\)](#)

Before you start this tutorial, make sure that you have:

- [Set up your AWS account \(./setting-up.html\)](#)

You'll need your AWS account and AWS IoT console to complete this tutorial.

- Reviewed [View MQTT messages with the AWS IoT MQTT client \(./view-mqtt-messages.html\)](#)

Be sure you can use the MQTT client to subscribe and publish to a topic. You'll use the MQTT client to test your new rule in this procedure.

- Reviewed the [Amazon Simple Notification Service \(https://docs.aws.amazon.com/sns/latest](https://docs.aws.amazon.com/sns/latest)


[/dg/welcome.html](#))

If you haven't used Amazon SNS before, review [Setting up access for Amazon SNS](#) (<https://docs.aws.amazon.com/sns/latest/dg/sns-setting-up.html>) . If you've already completed other AWS IoT tutorials, your AWS account should already be configured correctly.

Step 1: Create an Amazon SNS topic that sends an SMS text message

To create an Amazon SNS topic that sends an SMS text message

1. Create an Amazon SNS topic.

1. Sign in to the [Amazon SNS console](#)  (<https://console.aws.amazon.com/sns/home>) .
2. In the left navigation pane, choose **Topics**.
3. On the **Topics** page, choose **Create topic**.
4. In **Details**, choose the **Standard** type. By default, the console creates a FIFO topic.
5. In **Name**, enter the SNS topic name. For this tutorial, enter `high_temp_notice`.
6. Scroll to the end of the page and choose **Create topic**.

The console opens the new topic's **Details** page.


2. Create an Amazon SNS subscription.

Note

The phone number that you use in this subscription might incur text messaging charges from the messages you will send in this tutorial.

1. In the `high_temp_notice` topic's details page, choose **Create subscription**.
2. In **Create subscription**, in the **Details** section, in the **Protocol** list, choose **SMS**.
3. In **Endpoint**, enter the number of a phone that can receive text messages. Be sure to enter it such that it starts with a +, includes the country and area code, and doesn't include any other punctuation characters.
4. Choose **Create subscription**.

3. Test the Amazon SNS notification.

1. In the [Amazon SNS console](#)  (<https://console.aws.amazon.com/sns/home>) , in the left navigation pane, choose **Topics**.
2. To open the topic's details page, in **Topics**, in the list of topics, choose `high_temp_notice`.

3. To open the **Publish message to topic** page, in the **high_temp_notice** details page, choose **Publish message**.
4. In **Publish message to topic**, in the **Message body** section, in **Message body to send to the endpoint**, enter a short message.
5. Scroll down to the bottom of the page and choose **Publish message**.
6. On the phone with the number you used earlier when creating the subscription, confirm that the message was received.

If you did not receive the test message, double check the phone number and your phone's settings.

Make sure you can publish test messages from the [Amazon SNS console](https://console.aws.amazon.com/sns/home) (https://console.aws.amazon.com/sns/home) before you continue the tutorial.

Step 2: Create an AWS IoT rule to send the text message

The AWS IoT rule that you'll create in this tutorial subscribes to the device/device_id/data MQTT topics where device_id is the ID of the device that sent the message. These topics are described in a topic filter as device+/data, where the + is a wildcard character that matches any string between the two forward slash characters. This rule also tests the value of the temperature field in the message payload.

When the rule receives a message from a matching topic, it takes the device_id from the topic name, the temperature value from the message payload, and adds a constant value for the limit it's testing, and sends these values as a JSON document to an Amazon SNS notification topic.

For example, an MQTT message from weather sensor device number 32 uses the device/32/data topic and has a message payload that looks like this:

```
{
  "temperature": 38,
  "humidity": 80,
  "barometer": 1013,
  "wind": {
    "velocity": 22,
    "bearing": 255
  }
}
```

The rule's rule query statement takes the temperature value from the message payload, the device_id from the topic name, and adds the constant max_temperature value to send a message payload that looks like this to the Amazon SNS topic:

```
{
  "device_id": "32",
  "reported_temperature": 38,
  "max_temperature": 30
}
```

To create an AWS IoT rule to detect an over-limit temperature value and create the data to send to the Amazon SNS topic

1. Open [the Rules hub of the AWS IoT console](https://console.aws.amazon.com/iot/home#/rulehub) (https://console.aws.amazon.com/iot/home#/rulehub).
2. If this is your first rule, choose **Create**, or **Create a rule**.
3. In **Create a rule**:

1. In **Name**, enter temp_limit_notify.

Remember that a rule name must be unique within your AWS account and Region, and it can't have any spaces. We've used an underscore character in this name to separate the words in the rule's name.

2. In **Description**, describe the rule.

A meaningful description makes it easier to remember what this rule does and why you created it. The description can be as long as needed, so be as detailed as possible.

4. In **Rule query statement of Create a rule**:

1. In **Using SQL version**, select **2016-03-23**.
2. In the **Rule query statement** edit box, enter the statement:

```
SELECT topic(2) as device_id,
       temperature as reported_temperature,
       30 as max_temperature
FROM 'device/+/data'
WHERE temperature > 30
```

This statement:

- Listens for MQTT messages with a topic that matches the device/+/data topic filter and that have a temperature value greater than 30.
- Selects the second element from the topic string and assigns it to the device_id field.
- Selects the value temperature field from the message payload and assigns it to the

reported_temperature field.

- Creates a constant value 30 to represent the limit value and assigns it to the max_temperature field.

5. To open up the list of rule actions for this rule, in **Set one or more actions**, choose **Add action**.

6. In **Select an action**, choose **Send a message as an SNS push notification**.

7. To open the selected action's configuration page, at the bottom of the action list, choose **Configure action**.

8. In **Configure action**:

1. In **SNS target**, choose **Select**, find your SNS topic named **high_temp_notice**, and choose **Select**.
2. In **Message format**, choose **RAW**.
3. In **Choose or create a role to grant AWS IoT access to perform this action**, choose **Create Role**.
4. In **Create a new role**, in **Name**, enter a unique name for the new role. For this tutorial, use `sns_rule_role`.
5. Choose **Create role**.

If you're repeating this tutorial or reusing an existing role, choose **Update role** before continuing. This updates the role's policy document to work with the SNS target.

9. Choose **Add action** and return to the **Create a rule** page.

In the new action's tile, below **Send a message as an SNS push notification**, you can see the SNS topic that your rule will call.

This is the only rule action you'll add to this rule.

10. To create the rule and complete this step, in **Create a rule**, scroll down to the bottom and choose **Create rule**.

Step 3: Test the AWS IoT rule and Amazon SNS notification

To test your new rule, you'll use the MQTT client to publish and subscribe to the MQTT messages used by this rule.

Open the [MQTT client in the AWS IoT console](https://console.aws.amazon.com/iot/home#/test) (https://console.aws.amazon.com/iot/home#/test) in a new window. This will let you edit the rule without losing the configuration of your MQTT client. If you leave the MQTT client to go to another page in the console, it won't retain any subscriptions

or message logs.

To use the MQTT client to test your rule

1. In the [MQTT client in the AWS IoT console](https://console.aws.amazon.com/iot/home#/test) (<https://console.aws.amazon.com/iot/home#/test>) , subscribe to the input topics, in this case, `device/+ /data`.

1. In the MQTT client, under **Subscriptions**, choose **Subscribe to a topic**.
2. In **Subscription topic**, enter the topic of the input topic filter, `device/+ /data`.
3. Keep the rest of the fields at their default settings.
4. Choose **Subscribe to topic**.

In the **Subscriptions** column, under **Publish to a topic**, `device/+ /data` appears.

2. Publish a message to the input topic with a specific device ID, `device/32 /data`. You can't publish to MQTT topics that contain wildcard characters.

1. In the MQTT client, under **Subscriptions**, choose **Publish to topic**.
2. In the **Publish** field, enter the input topic name, `device/32 /data`.
3. Copy the sample data shown here and, in the edit box below the topic name, paste the sample data.

```
{
  "temperature": 38,
  "humidity": 80,
  "barometer": 1013,
  "wind": {
    "velocity": 22,
    "bearing": 255
  }
}
```

4. Choose **Publish to topic** to publish your MQTT message.
3. Confirm that the text message was sent.
 1. In the MQTT client, under **Subscriptions**, there is a green dot next to the topic to which you subscribed earlier.

The green dot indicates that one or more new messages have been received since the last time you looked at them.

2. Under **Subscriptions**, choose **device/+ /data** to check that the message payload matches what you just published and looks like this:

```
{
  "temperature": 38,
  "humidity": 80,
  "barometer": 1013,
  "wind": {
    "velocity": 22,
    "bearing": 255
  }
}
```

3. Check the phone that you used to subscribe to the SNS topic and confirm the contents of the message payload look like this:

```
{"device_id":"32","reported_temperature":38,"max_temperature":30}
```

Notice that the `device_id` value is a quoted string and the temperature value is numeric. This is because the `topic()` ([./html#](#)) function extracted the string from the input message's topic name while the temperature value uses the numeric value from the input message's payload.

If you want to make the `device_id` value a numeric value, replace `topic(2)` in the rule query statement with:

```
cast(topic(2) AS DECIMAL)
```

Note that casting the `topic(2)` value to a numeric, DECIMAL value will only work if that part of the topic contains only numeric characters.

4. Try sending an MQTT message in which the temperature does not exceed the limit.

1. In the MQTT client, under **Subscriptions**, choose **Publish to topic**.
2. In the **Publish** field, enter the input topic name, `device/33/data`.
3. Copy the sample data shown here and, in the edit box below the topic name, paste the sample data.

```
{
  "temperature": 28,
  "humidity": 80,
  "barometer": 1013,
```

```
"wind": {  
  "velocity": 22,  
  "bearing": 255  
}  
}
```

4. To send your MQTT message, choose **Publish to topic**.

You should see the message that you sent in the `device/+ /data` subscription. However, because the temperature value is below the max temperature in the rule query statement, you shouldn't receive a text message.

If you don't see the correct behavior, check the troubleshooting tips.

Troubleshooting your SNS message rule

Here are some things to check, in case you're not seeing the results you expect.

- **You got an error banner**

If an error appeared when you published the input message, correct that error first. The following steps might help you correct that error.

- **You don't see the input message in the MQTT client**

Every time you publish your input message to the `device/22/data` topic, that message should appear in the MQTT client, if you subscribed to the `device/+ /data` topic filter as described in the procedure.

Things to check

- **Check the topic filter you subscribed to**

If you subscribed to the input message topic as described in the procedure, you should see a copy of the input message every time you publish it.

If you don't see the message, check the topic name you subscribed to and compare it to the topic to which you published. Topic names are case sensitive and the topic to which you subscribed must be identical to the topic to which you published the message payload.

- **Check the message publish function**

In the MQTT client, under **Subscriptions**, choose **device/+ /data**, check the topic of the publish message, and then choose **Publish to topic**. You should see the message payload from the edit box below the topic appear in the message list.

- **You don't receive an SMS message**

For your rule to work, it must have the correct policy that authorizes it to receive a message

and send an SNS notification, and it must receive the message.

Things to check

- **Check the AWS Region of your MQTT client and the rule that you created**

The console in which you're running the MQTT client must be in the same AWS Region as the rule you created.

- **Check that the temperature value in the message payload exceeds the test threshold**

If the temperature value is less than or equal to 30, as defined in the rule query statement, the rule will not perform any of its actions.

- **Check the input message topic in the rule query statement**

For the rule to work, it must receive a message with the topic name that matches the topic filter in the FROM clause of the rule query statement.

Check the spelling of the topic filter in the rule query statement with that of the topic in the MQTT client. Topic names are case sensitive and the message's topic must match the topic filter in the rule query statement.

- **Check the contents of the input message payload**

For the rule to work, it must find the data field in the message payload that is declared in the SELECT statement.

Check the spelling of the temperature field in the rule query statement with that of the message payload in the MQTT client. Field names are case sensitive and the temperature field in the rule query statement must be identical to the temperature field in the message payload.

Make sure that the JSON document in the message payload is correctly formatted. If the JSON has any errors, such as a missing comma, the rule will not be able to read it.

- **Check the republished message topic in the rule action**

The topic to which the Republish rule action publishes the new message must match the topic to which you subscribed in the MQTT client.

Open the rule you created in the console and check the topic to which the rule action will republish the message.

- **Check the role being used by the rule**

The rule action must have permission to receive the original topic and publish the new topic.

The policies that authorize the rule to receive message data and republish it are specific to

the topics used. If you change the topic used to republish the message data, you must update the rule action's role to update its policy to match the current topic.

If you suspect this is the problem, edit the Republish rule action and create a new role. New roles created by the rule action receive the authorizations necessary to perform these actions.

Step 4: Review the results and next steps

In this tutorial:

- You created and tested an Amazon SNS notification topic and subscription.
- You used a simple SQL query and functions in a rule query statement to create a new message for your notification.
- You created an AWS IoT rule to send an Amazon SNS notification that used your customized message payload.
- You used the MQTT client to test your AWS IoT rule.

Next steps

After you send a few text messages with this rule, try experimenting with it to see how changing some aspects of the tutorial affect the message and when it's sent. Here are some ideas to get you started.

- Change the `device_id` in the input message's topic and observe the effect in the text message contents.
- Change the fields selected in the rule query statement and observe the effect in the text message contents.
- Change the test in the rule query statement to test for a minimum temperature instead of a maximum temperature. Remember to change the name of `max_temperature`!
- Add a republish rule action to send an MQTT message when an SNS notification is sent.
- Try the next tutorial in this series and learn how to [Tutorial: Storing device data in a DynamoDB table \(./iot-ddb-rule.html\)](https://docs.aws.amazon.com/iot/latest/developerguide/iot-ddb-rule.html).