



Connectivity Guide

Kepware MQTT Agent and Google Cloud IoT Core

September 2021
Ref. 1.005

Table of Contents

- 1. Configure Google Cloud Platform3
 - 1.1 Create a New Project.....3
 - 1.2 Create a Registry4
 - 1.3 Create Certificates.....5
 - 1.4 Create Device6
- 2. Configure MQTT Client in KEPServerEX7
 - 2.1 Create JWT7
 - 2.2 Configure MQTT Client7
- 3. Review Data Flow on Google Cloud8
 - 3.1 Subscribe to Topic.....8
 - 3.2 Activate Cloud Shell.....9
- 4. Update a Tag from Google Cloud9
 - 4.1 Configure MQTT Agent.....9
 - 4.2 Send Messages from Google Cloud.....9

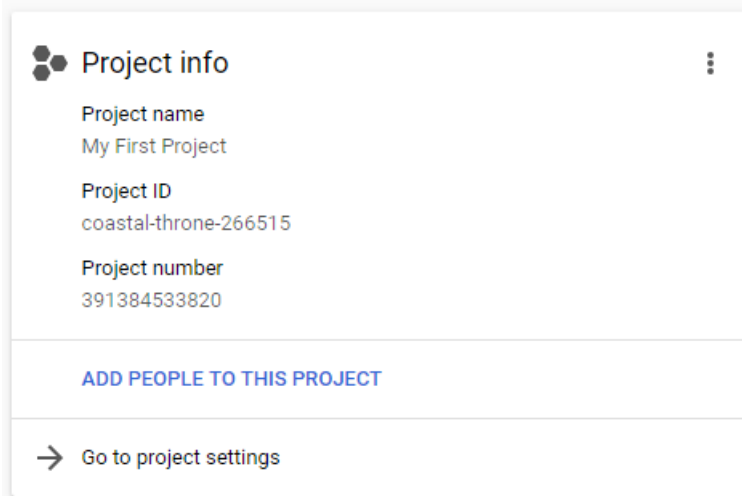
Kepware MQTT Agent and Google Cloud IoT Core

This document instructs users how to connect KEPServerEX® MQTT Agent and Google Cloud IoT Core to manage and deliver data. These instructions assume the user is registered with Google Cloud, and that OpenSSL is available to create self-signed certificates.

1. Configure Google Cloud Platform

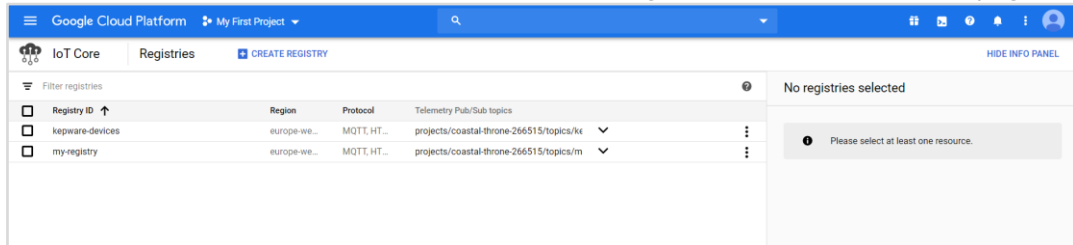
1.1 Create a New Project

Once registered to Google Cloud, an initial project is created on the account. Use this project or create a new project for this exercise. Note the Project ID, which will be required later.



To enable the Cloud IoT Core and Cloud Pub/Sub APIs for the project, access the link below and select the desired project from the dropdown menu:
<https://console.cloud.google.com/flows/enableapi?apiid=cloudiot.googleapis.com,pubsub& ga=2.96525144.639288887.1580223926-1334733146.1576681398>

Search and select **IoT Core** in the search bar to navigate to the IoT Core home page.



1.2 Create a Registry

Create a registry by entering a Registry ID, and selecting a region, and selecting a Cloud Pub/Sub topic.

Google Cloud Platform My First Project

IoT Core Create a registry

Define how devices in this registry will send data to Cloud IoT Core. After you create your registry, you can start adding devices to it. [Learn more](#)

Registry properties

Registry ID
registry1

Permanent identifier for your registry. 3-255 characters. Start with a letter. You can also include numbers and the following characters: + . % - _ ~

Region
europe-west1

Determines where data is stored for devices in this registry. Choice is permanent.

Cloud Pub/Sub topics

Cloud IoT Core routes device messages to Cloud Pub/Sub for aggregation. You can route messages to different topics and subfolders in Cloud Pub/Sub based on the type of data in the messages. [Learn more](#)

Select a Cloud Pub/Sub topic
projects/coastal-throne-266515/topics/kep-topic

Device telemetry events will be published to this topic by default.

+ ADD ADDITIONAL TOPIC

SHOW ADVANCED OPTIONS

CREATE CANCEL

Review **SHOW ADVANCED OPTIONS** to ensure the MQTT protocol is selected.

Create a topic to add to the Pub/Sub for the device registry. Enable **Google-managed key** for encryption.

Create a topic

Add a topic to Pub/Sub so that you can use it in your device registry.

Topic ID * ?

Topic name: projects/coastal-throne-266515/topics/kep-topic

Encryption

- Google-managed key
No configuration required
- Customer-managed key
Manage via Google Cloud Key Management Service

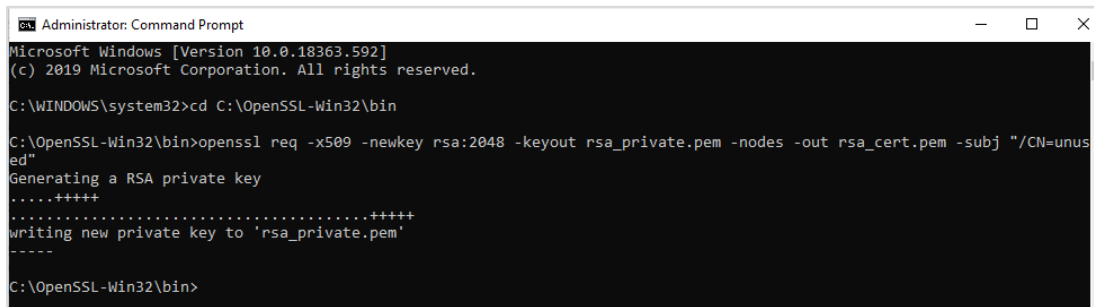
CANCEL CREATE TOPIC

1.3 Create Certificates

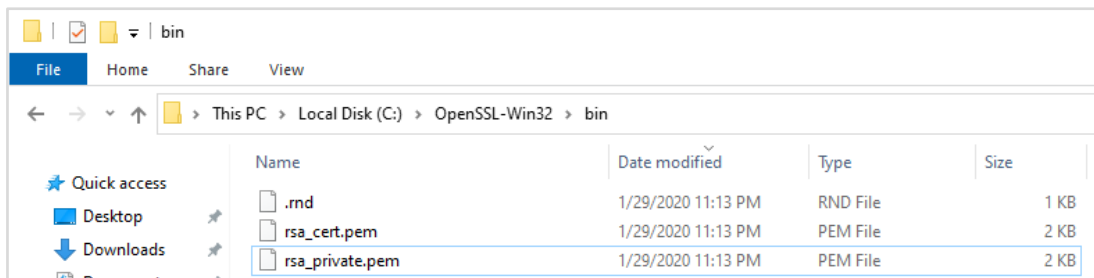
Certificates are required for both device and JSON Web Token (JWT) authentication. To create self-signed certificates, download and install OpenSSL: <https://www.openssl.org/>

Once the OpenSSL is installed, open a command prompt and run the following command to create a certificate and private key:

```
openssl req -x509 -newkey rsa:2048 -keyout rsa_private.pem -nodes -out  
rsa_cert.pem -subj "/CN=unused"
```

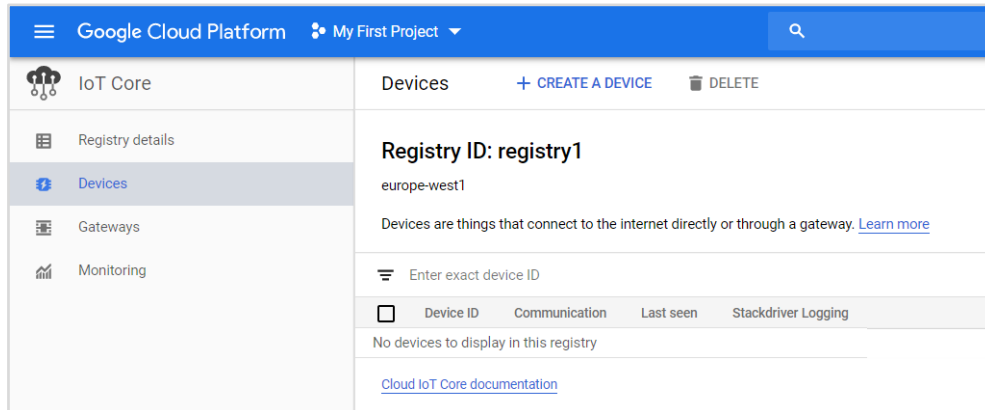


Locate the certificates in the designated folder to verify that both public key and private keys have been created.

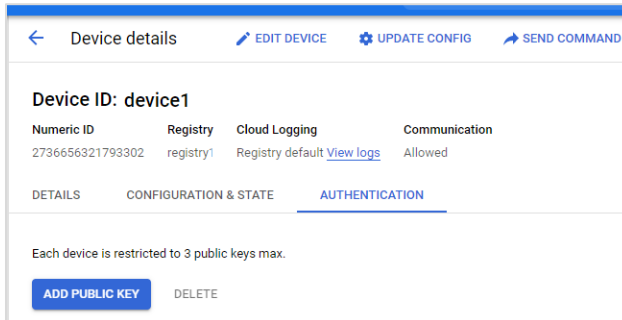


1.4 Create Device

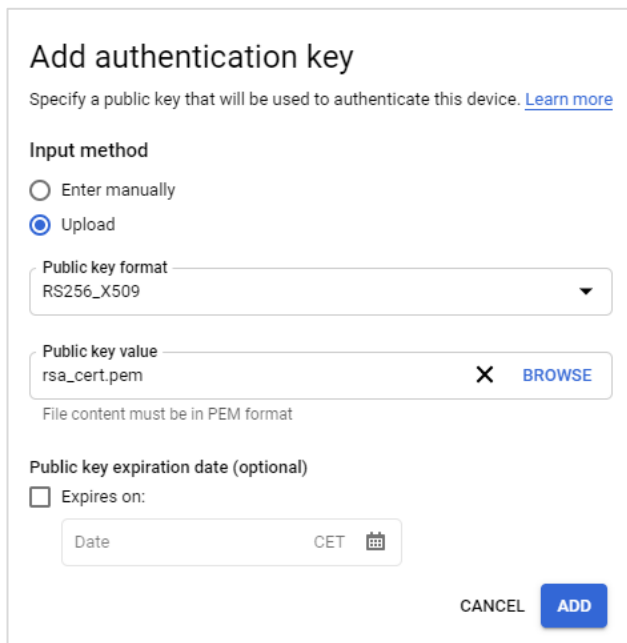
Return to the Google Cloud Platform and select **+ CREATE A DEVICE** in the **Devices** section.



Finish adding a device with default settings and access the Authentication settings for the newly created device. Click **ADD PUBLIC KEY**.



Choose **Upload** as the input method and select **RS256_X509** from the Public key format dropdown. For the Public key value, browse for the `rsa_cert.pem` file created in [step 1.3](#), then click **ADD**.



This completes the Google Cloud Platform configuration.

2. Configure MQTT Client in KEPServerEX

2.1 Create JWT

For the MQTT Client, a JSON Web Token (JWT) is required.

• For JWT creation examples, refer to the following Google Cloud documentation: <https://cloud.google.com/iot/docs/how-tos/credentials/jwts>

• **Note:** Per the Google documentation linked above, the JWT required for authentication with Google IoT Core is a short-lived token, meaning the token will expire after a short period of time. A new token must be created and placed into the appropriate field within KEPServerEX prior to the expiration of the previous token in order to avoid connection disruption. This JWT token generation and KEPServerEX update can happen in two ways:

1. Manually create the JWT and manually update KEPServerEX.
2. Using scripts (Powershell, Python, Javascript, etc.), automatically create a new JWT and automatically update KEPServerEX at a designated time interval.

• Example reference scripts are available from Kepware. For more information, contact sales@kepware.com or presales.support@kepware.com.

2.2 Configure MQTT Client

• For more information, see the [IoT Gateway Manual](#).

In KEPServerEX, click **Add Agent...** under the IoT Gateway Plug-in tree node.

Enter a name and select **MQTT Client** as the agent type.

Through the configuration wizard, set the following parameters:

Property	Value
URL	ssl://mqtt.googleapis.com:8883
Topic	/devices/<DEVICE_ID>/events
Client ID	projects/<PROJECT_ID>/locations/<REGION>/registries/<REGISTRY_ID>/devices/<DEVICE_ID>
Username	<USERNAME>
Password	<JWT>

• **Note:** A username is required when setting a password. For this scenario, any username is acceptable.

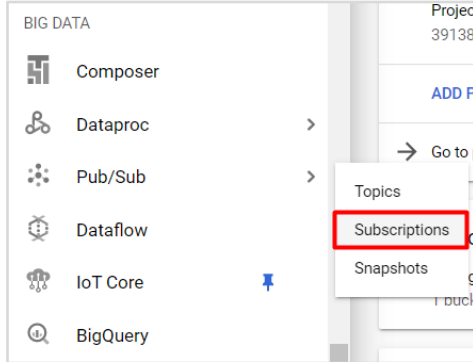
Add tags to the MQTT Agent. Once added, a message on Event Log should confirm a connection to Google Cloud IoT Core.

Date	Time	Source	Event
29/01/2020	23:37:12	KEPServerEX\Runtime	Local Historian Plug-in V6.8.796.0
29/01/2020	23:37:12	KEPServerEX\Runtime	IDF for Splunk V6.8.796.0
29/01/2020	23:37:12	KEPServerEX\Runtime	Scheduler Plug-in V6.8.796.0
29/01/2020	23:37:12	KEPServerEX\Runtime	IoT Gateway V6.8.796.0
29/01/2020	23:37:12	KEPServerEX\Runtime	Runtime project replaced.
29/01/2020	23:37:13	KEPServerEX\IoT Gateway	Running with Java 1.8.0_241 [Oracle Corporation Java HotSpot(TM) Client VM version 25.241-b07].
29/01/2020	23:37:25	Licensing	Feature IoT Gateway is time limited and will expire at 30/01/2020 01:37.
29/01/2020	23:37:32	KEPServerEX\Runtime	MQTT agent 'Google Cloud' is connected to broker 'ssl://mqtt.googleapis.com:8883'

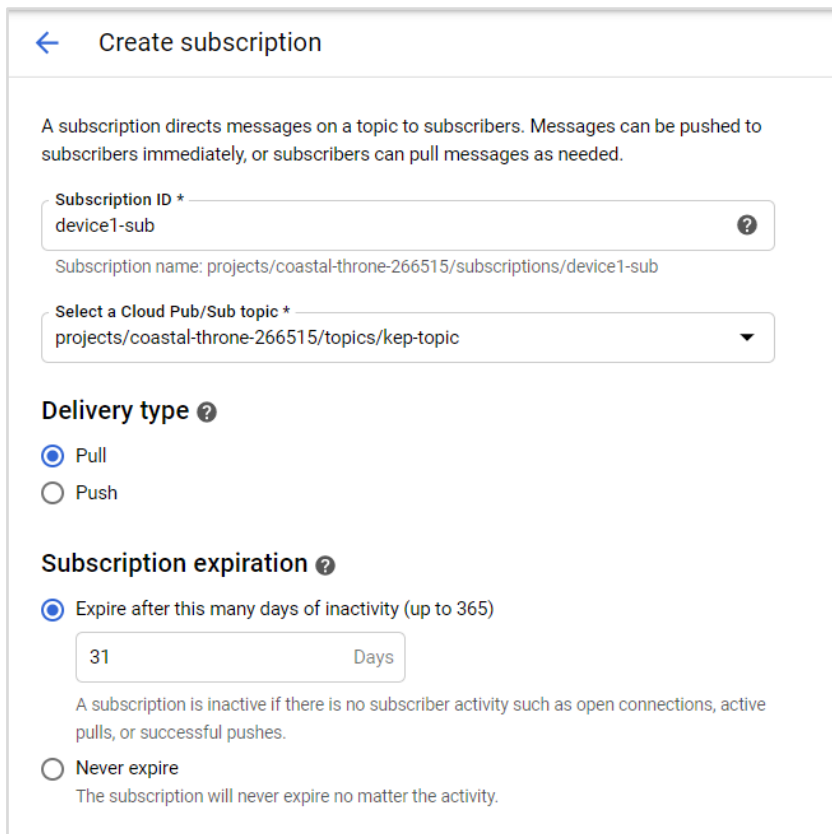
3. Review Data Flow on Google Cloud

3.1 Subscribe to Topic

To view the data flow on Google Cloud, create a subscription for the topic from the registry. On Google Cloud, navigate to **Pub/Sub | Subscriptions** in the project.



Choose the appropriate Cloud Pub/Sub topic from the dropdown menu (see [Step 1.2](#)), and set delivery type to **Pull**.

A screenshot of the 'Create subscription' form in the Google Cloud console. The form includes a back arrow and the title 'Create subscription'. Below the title is a descriptive paragraph: 'A subscription directs messages on a topic to subscribers. Messages can be pushed to subscribers immediately, or subscribers can pull messages as needed.' The form contains several fields: 'Subscription ID *' with the value 'device1-sub' and a help icon; 'Subscription name: projects/coastal-throne-266515/subscriptions/device1-sub'; 'Select a Cloud Pub/Sub topic *' with a dropdown menu showing 'projects/coastal-throne-266515/topics/kep-topic'; 'Delivery type' with radio buttons for 'Pull' (selected) and 'Push'; and 'Subscription expiration' with radio buttons for 'Expire after this many days of inactivity (up to 365)' (selected) and 'Never expire'. The selected option has a text input field with '31' and 'Days' next to it. A note explains that a subscription is inactive if there is no subscriber activity such as open connections, active pulls, or successful pushes.

3.2 Activate Cloud Shell

In the Google Cloud Console, click **Activate Cloud Shell**.



In Cloud Shell, run the following command:

```
gcloud pubsub subscriptions pull --auto-ack
projects/<PROJECT_ID>/subscriptions/<SUBSCRIPTION_ID>
```

Under **DATA**, the MQTT Client should deliver the JSON payload. In this example, the *Simulation Examples.Functions.Ramp1* tag is published by the MQTT Agent.

DATA	MESSAGE_ID	ATTRIBUTES
<pre>{ "timestamp":1580374936574, "values":{ "id":"Simulation Examples.Functions.Ramp1", "v":35, "q":true, "t":1580374935705 } }</pre>	949918307208407	<pre>deviceId=device1 deviceNumId=2892641596482375 deviceRegistryId=registry1 deviceRegistryLocation=europe-west1 projectId=coastal-throne-266515 subFolder=</pre>

4. Update a Tag from Google Cloud

4.1 Configure MQTT Agent

For more information, see the [IoT Gateway Manual](#).

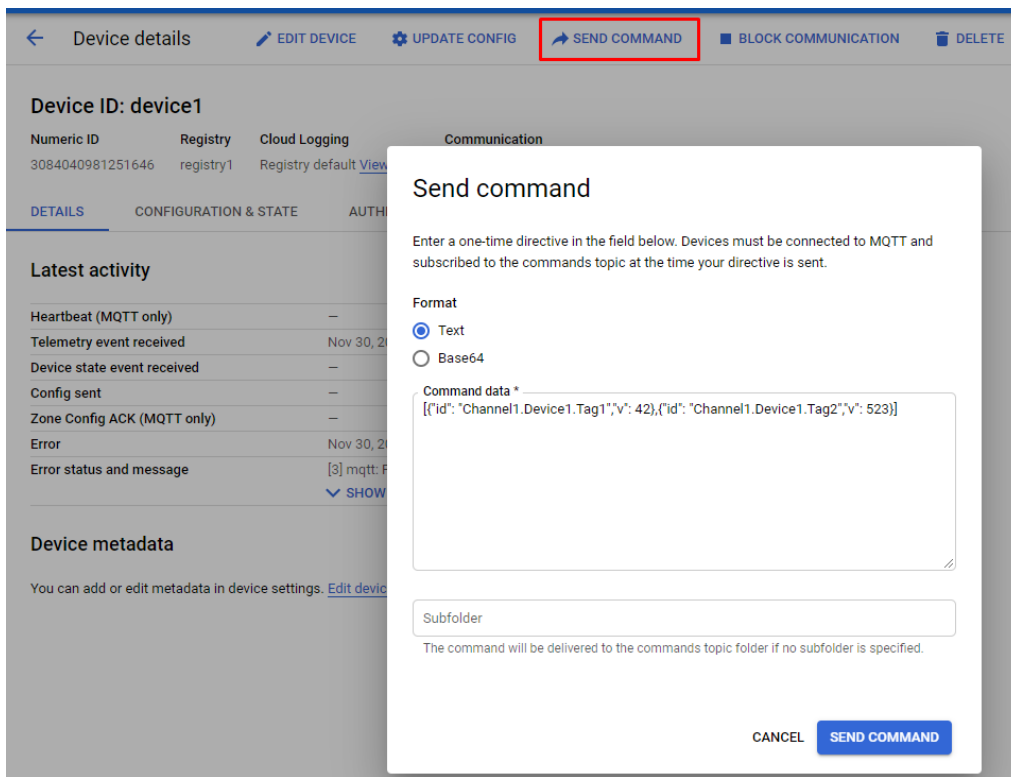
The MQTT Agent can be configured to receive messages from Google IoT Core. In order to do that navigate to “Subscriptions” in MQTT Agent Properties and configure according the following parameters:

Property	Value
Listen for Write Requests	Yes
Topic	/devices/<DEVICE_ID>/commands/#

4.2 Send Messages from Google Cloud

For more information: <https://cloud.google.com/iot/docs/how-tos/commands>

Go to the **Registries** page in Cloud Console. Click the ID of the registry for the device. In the registry menu on the left, click **Devices**. Click the ID of the device you want to send the command to. At the top of the page, click **Send command**. Select the format of the command as Text. In the **Command data** field, enter the message. The message should be in correct format for MQTT Agent to receive.



For example, a payload to update two tags:

```
[{"id": "Channel1.Device1.Tag1", "v": 42}, {"id": "Channel1.Device1.Tag2", "v": 523}]
```

Click **Send command**. Verify that the desired tags have been successfully updated.