



Connectivity Guide

Kepware MQTT Agent and Google Cloud IoT Core

March 2020
Ref. 1.001

Table of Contents

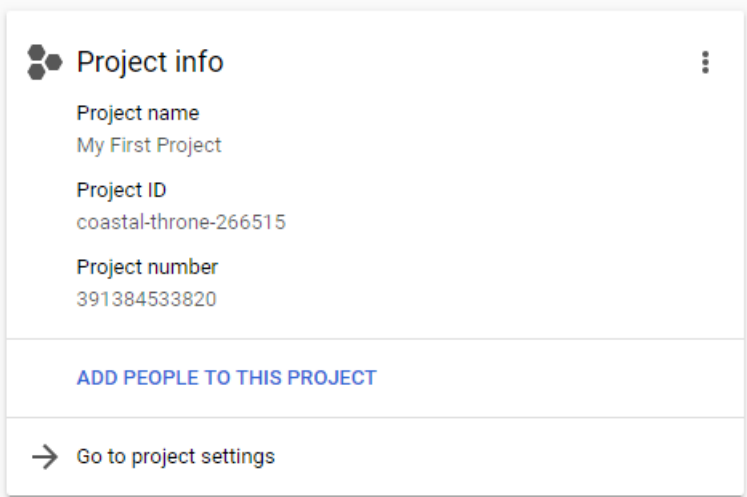
1.	Configure Google Cloud Platform.....	3
1.1	Create a New Project.....	3
1.2	Create a Registry.....	4
1.3	Create Certificates.....	5
1.4	Create Device.....	6
2.	Configure MQTT Client in KEPServerEX.....	7
2.1	Create JWT.....	7
2.2	Configure MQTT Client.....	7
3.	Review Data Flow on Google Cloud.....	8
3.1	Subscribe to Topic.....	8
3.2	Activate Cloud Shell.....	9

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

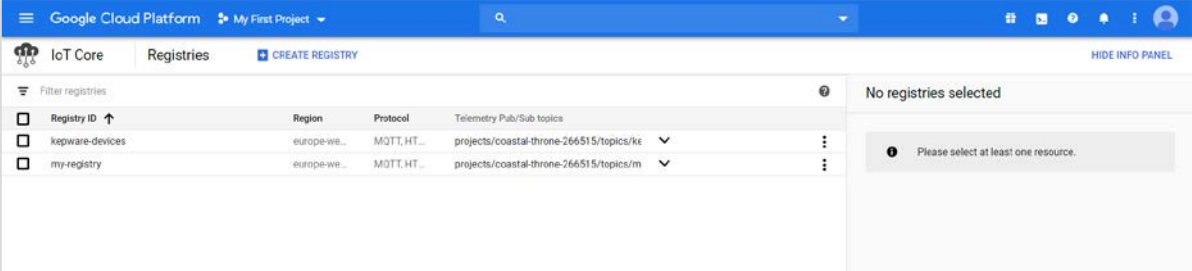
Once registered to Google Cloud, an initial project is created on the account. Use this project or create a new project for this exercise. Note the Project ID, which will be required later.



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.

The screenshot shows the 'Create a registry' page in the Google Cloud Platform IoT Core console. The page is titled 'IoT Core' and 'Create a registry'. It contains the following sections:

- Registry properties**
 - Registry ID:** A text input field containing 'registry1'. Below it, a note states: 'Permanent identifier for your registry. 3-255 characters. Start with a letter. You can also include numbers and the following characters: + . % - _ ~'.
 - Region:** A dropdown menu showing 'europe-west1'. Below it, a note states: 'Determines where data is stored for devices in this registry. Choice is permanent.'
- Cloud Pub/Sub topics**
 - A text input field with a dropdown arrow, containing 'projects/coastal-throne-266515/topics/kep-topic'. Above the field is the label 'Select a Cloud Pub/Sub topic'. Below the field, a note states: 'Device telemetry events will be published to this topic by default.'
 - A button labeled '+ ADD ADDITIONAL TOPIC'.
 - A link labeled 'SHOW ADVANCED OPTIONS' with a downward arrow.
 - Two buttons at the bottom: 'CREATE' (in a blue box) and 'CANCEL'.

Review **SHOW ADVANCED OPTIONS** to ensure MQTT and HTTP protocols are 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

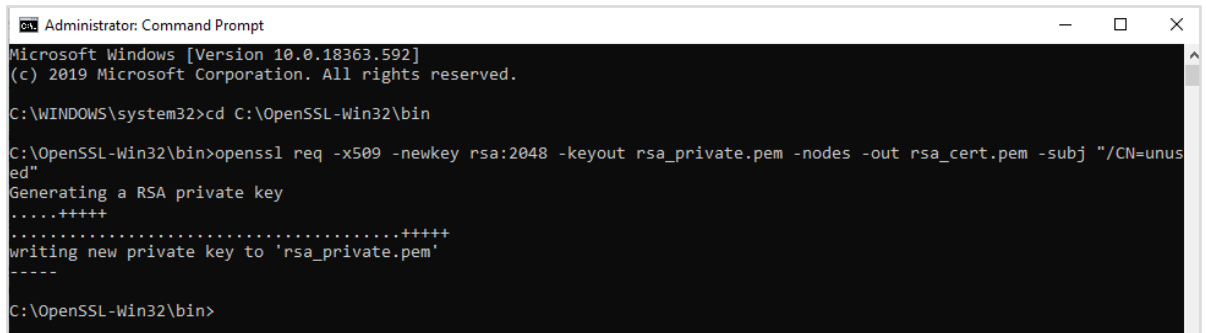
CANCEL CREATE TOPIC

1.3 Create Certificates

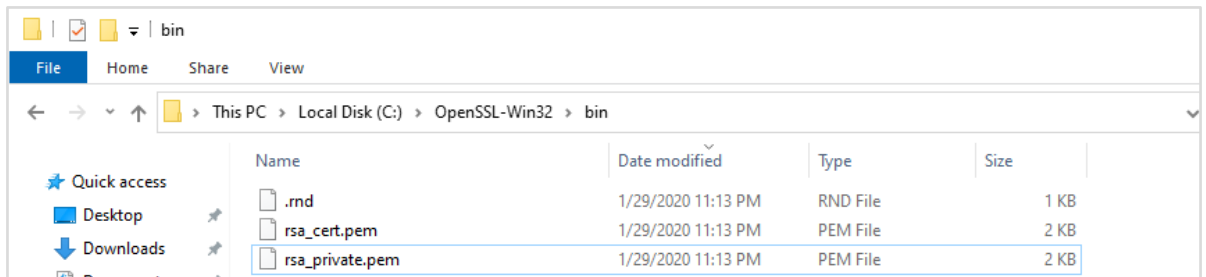
Certificates are required for both device and Java Web Token (JWT) authentication. To create self-signed certificates, download and install OpenSSL: <https://www.openssl.org/>

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"
```

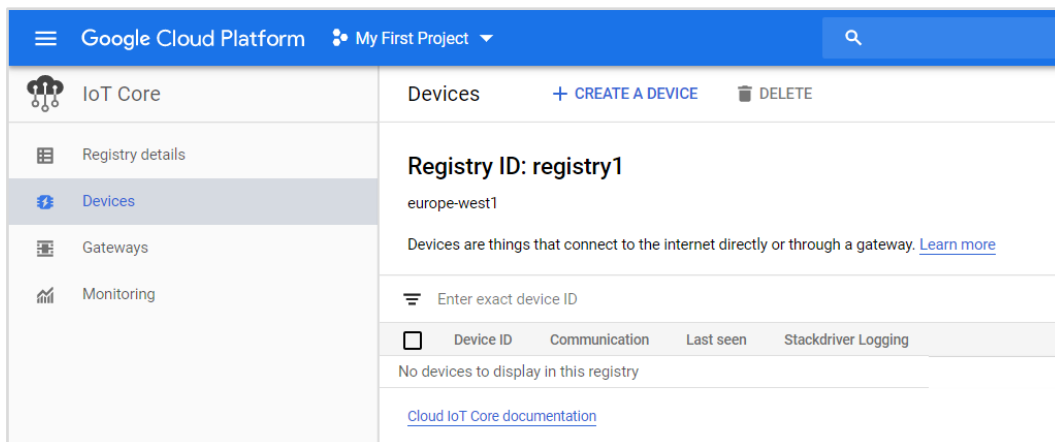


Navigate to ...**OpenSSL-Win32\bin** and open **rsa_cert.pem** with a text editor. Copy the contents to use during Device creation.

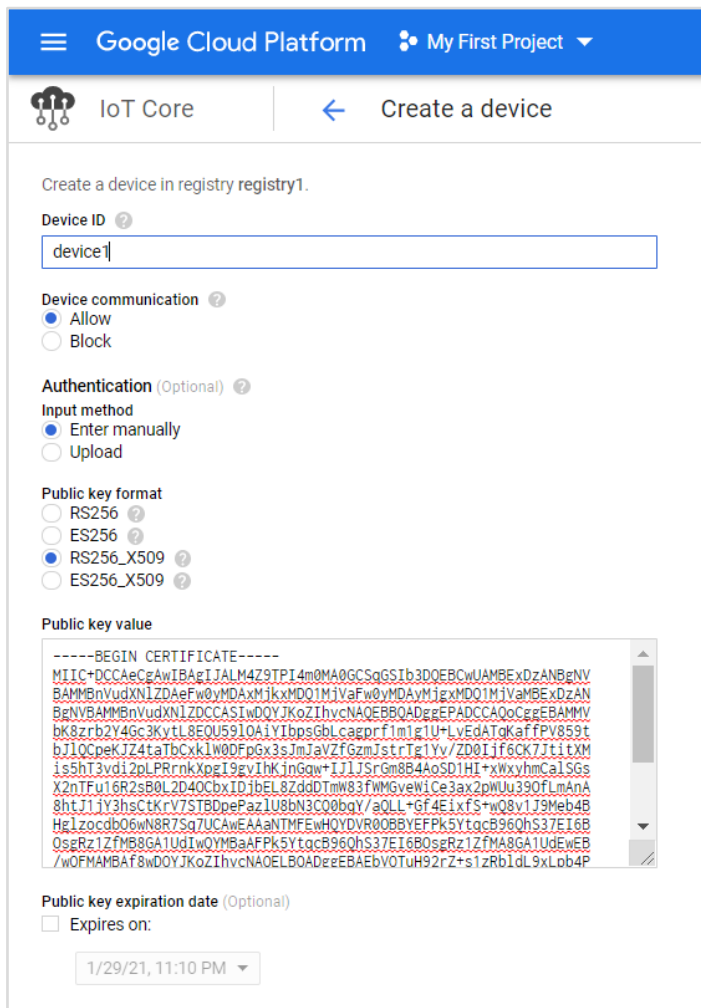


1.4 Create Device

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



Enter a Device ID, and select **RS256_X509** as the public key format. Do not adjust the default device communication and authentication settings. Paste the certificate and private key text in the **Public key value** field, then click **CREATE**.



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>

2.2 Configure MQTT Client

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

In KEPServerEX, click **Add Agent...** under the **IoT Gateway** plugin.

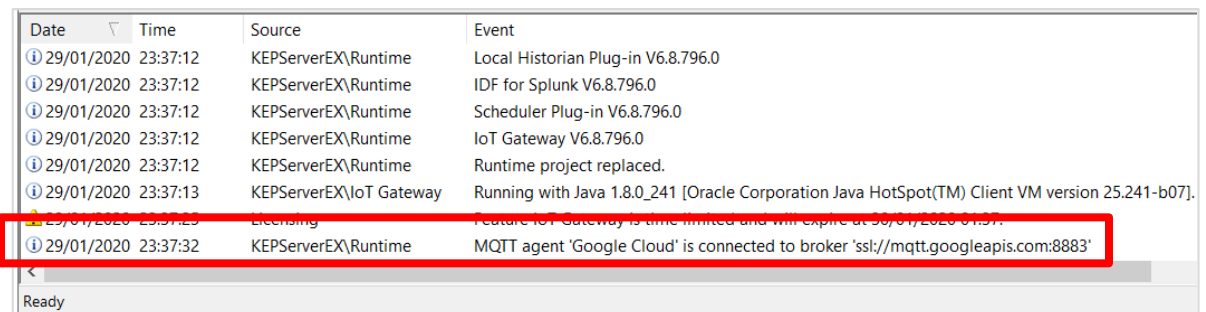
Enter a name and select **MQTT Client** as the agent type.

Continue through the configuration wizard to 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.



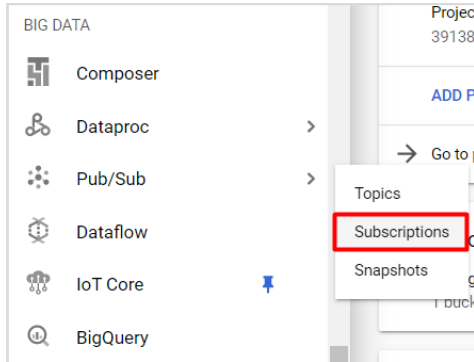
The screenshot shows the Windows Event Viewer with a list of events. The event at 23:37:32 is highlighted with a red box. The event details are as follows:

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	23:37:23	Licensing	Feature IoT Gateway is time limited and will expire at 29/01/2020 01:37.
29/01/2020	23:37:32	KEPServerEX\Runtime	MQTT agent 'Google Cloud' is connected to broker 'ssl://mqtt.googleapis.com:8883'

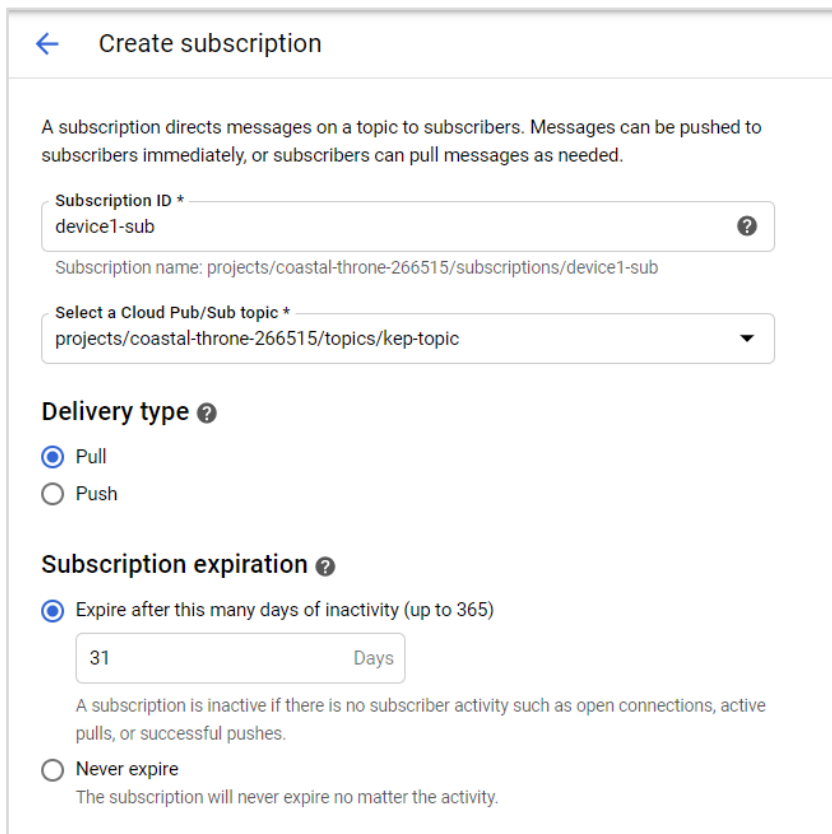
3. Review Data Flow on Google Cloud

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 has 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 and options: 1. 'Subscription ID *' with the value 'device1-sub' and a help icon. Below it, the 'Subscription name' is shown as 'projects/coastal-throne-266515/subscriptions/device1-sub'. 2. 'Select a Cloud Pub/Sub topic *' with a dropdown menu showing 'projects/coastal-throne-266515/topics/kep-topic'. 3. 'Delivery type ?' with two radio buttons: 'Pull' (selected) and 'Push'. 4. 'Subscription expiration ?' with two radio buttons: '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. Below this, there is a note: 'A subscription is inactive if there is no subscriber activity such as open connections, active pulls, or successful pushes.' 5. The 'Never expire' option has a note: 'The subscription will never expire no matter the activity.'

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.

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to coastal-throne-266515.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
ttooun@cloudshell:~ (coastal-throne-266515) $ gcloud pubsub subscriptions pull --auto-ack projects/coastal-throne-266515/subscriptions/device1-sub
```

DATA	MESSAGE_ID	ATTRIBUTES
[{"timestamp":1580374936574,"values":[{"id":"Simulation Examples.Functions.Ramp1","v":35,"q":true,"t":1580374935705}]]	949918307208407	deviceId=device1 deviceNumId=2892641596402375 deviceRegistryId=registry1 deviceRegistryLocation=europe-west1 projectId=coastal-throne-266515 subFolder=

```
ttooun@cloudshell:~ (coastal-throne-266515) $
```