Application Note

IMPINJ R700

CONNECTING TO AWS IOT

Version 1.0

www.impinj.com

TABLE OF CONTENTS

OVERVIEW

As of version 7.3 of the Impinj R700 Firmware, the Impinj R700 RAIN RFID reader has been able to send RAIN RFID data directly to an AWS IoT Service endpoint via MQTT. This document outlines the steps to configure this connection using Impinj firmware v8.1.

PREREQUISITES

Connecting the R700 to AWS IoT requires the following items:

- An Impinj R700 RAIN RFID reader running at least version 8.1 of the Impinj firmware.
- The 'curl' client or other similar utility installed on a host machine with network access to the reader.
- The OpenSSL binaries installed on a host machine with network access to the reader.
- An AWS IoT Account with a pre-created IoT "thing" device representing the reader. (**Hint**: Be sure to define the thing's name using only the characters a-z A-Z or 0-9. The name must match the reader's MQTT Client ID, which as of this writing, only accepts these characters.)
- The AWS IoT thing's device certificate and device private key in PEM format
- The AWS Certificate Authority (CA) certificate in PEM format

Note: the "Thing" certificate requires a policy document like the following to connect via MQTT successfully:

Figure 1: Sample reader 'Policy' document

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "arn:aws:iot:us-east-1:000000000000:*"
    }
  ]
}
```

Replace 'us-east-1' with your preferred region and '000000000000' with your AWS account number.

CONFIGURING THE IMPINJ R700 FOR AWS IOT CORE USAGE

Configuring the Impinj R700 RAIN RFID reader to send RFID data to the AWS Core requires copying the AWS IoT thing's device certificates, device private key, and Certificate Authority certificates to the reader and configuring MQTT event settings.

Both the reader configuration REST API and the Web UI can be used to manage the reader's configuration. Any configuration management done using the reader's Web UI can also be performed using the available REST API. However, not all REST API configuration options are available through the Web UI. This document uses the Web UI whenever possible; otherwise, it uses the REST API. In addition, although this document uses 'HTTPS' as its preferred protocol when accessing the REST API, either the HTTP or HTTPS protocols may be used. Please refer to this page for instructions on how to enable either protocol on the reader.

Copying over the AWS CA Certificate

The reader must trust the AWS IoT service before it sends its MQTT messages to the service. Do this by installing the AWS CA certificate on the reader using the following curl command:

curl -X POST https://<READER IP or HOSTNAME>/api/v1/system/certificates/ca/certs -H ‘Content-Type: multipart/form-data’ -H ‘Accept: application/json’ -u root:impinj -k -F ‘certFile=@"<PATH TO CERT FILE>"’

Figure 2: Installing the AWS CA Certificate

```
$ curl -X POST https://impinj-14-01-f1/api/v1/system/certificates/ca/certs -H
'Content-Type: multipart/form-data' -H 'Accept: application/json' -u root:impinj -k -F
'certFile=@"AmazonRootCA1.pem"'
[1]
```

Copying over the Thing’s Certificate and Private Key

Before uploading the thing’s certificate and private key to the reader, first combine them into a PFX file using the following OpenSSL command:

openssl.exe pkcs12 -export -out <PFX FILE NAME> -inkey <PATH TO PRIVATE KEY> -in <PATH TO CERTIFICATE>

Figure 3: Creating a PFX File

```
$ openssl.exe pkcs12 -export -out reader-certificate.p12 -inkey
421a06595863f58d65c40d00000000-private.pem.key -in 421a06595863f58d65c40d000000000-
certificate.pem.crt
```

Install the PFX file on the reader using the reader’s Install TLS certificate method:

curl -X POST https://<READER IP or HOSTNAME>/api/v1/system/certificates/tls/certs -H 'Content-Type: multipart/form-data' -H 'Accept: application/json' -u root:password -k -F 'certFile=@"<PATH TO PFX FILE>"'

Figure 4: Installing the Thing’s Certificate

```
$ curl -X POST https://impinj-14-01-f1/api/v1/system/certificates/tls/certs -H
'Content-Type: multipart/form-data' -H 'Accept: application/json'-u root:impinj -k -F
'certFile=@"impinj1401f1-certificate.p12"'
[2]
```

Note the Certificate ID returned by the Installation command surrounded by square brackets. Use the Certificate Services command to designate the thing’s certificate for use with the MQTT client using the Certificate ID mentioned above:

curl -X PUT https://<READER IP or HOSTNAME>/api/v1/system/certificates/tls/services/mqtt-client -H 'Content-Type: application/json' -H 'Accept: application/json' -u root:impinj -k --data-raw '{ "certId": <CERTIFICATE ID> }'

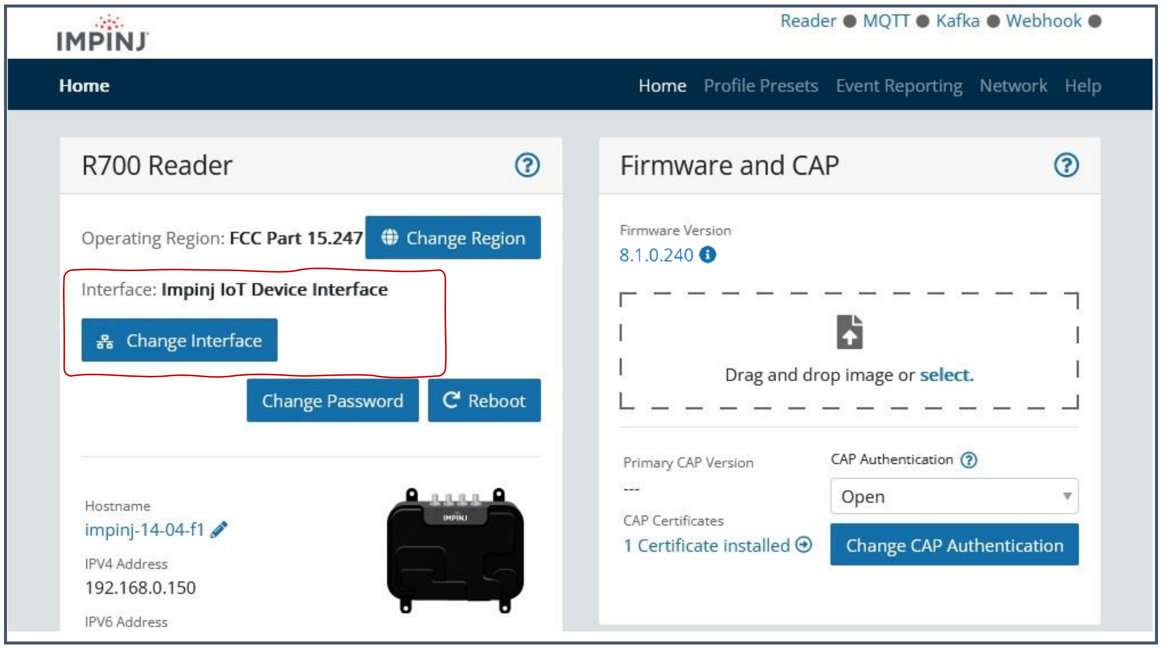
Figure 5: Updating the Thing's Certificate for MQTT use

```
$ curl -X PUT https://impinj-14-01-f1/api/v1/system/certificates/tls/services/mqtt-
client -H 'Content-Type: application/json' -H 'Accept: application/json' -u
root:impinj -k --data-raw '{ "certId": 2 }'
```

Enable the IoT Device Interface

The reader's MQTT event configuration can only be accessed if the **Impinj IoT Device Interface** is selected. To verify the Interface via the Web UI, first, open a web browser and navigate to the reader's IP address or hostname:

Figure 6: Impinj IoT Device Interface – Homepage


Reader ● MQTT ● Kafka ● Webhook ●
Home
Home Profile Presets Event Reporting Network Help
R700 Reader
Change Region
Change Interface
Change Password
Reboot
Firmware and CAP
Drag and drop image or select.
Change CAP Authentication

Operating Region: **FCC Part 15.247**

Interface: **Impinj IoT Device Interface**

Hostname
impinj-14-04-f1

IPV4 Address
192.168.0.150

IPV6 Address

Firmware Version
8.1.0.240

Primary CAP Version

CAP Authentication
Open

CAP Certificates
1 Certificate installed

If the **LLRP Interface** is selected, click the 'Change Interface' button to toggle to the **Impinj IoT Device Interface**.

Configuring MQTT Event Reporting

Most of the MQTT configuration can be done from the Web UI interface. However, AWS requires TLS enabled when connecting to the AWS IoT Service, which cannot be done from the Web UI. Run the following curl command to enable MQTT TLS on the reader:

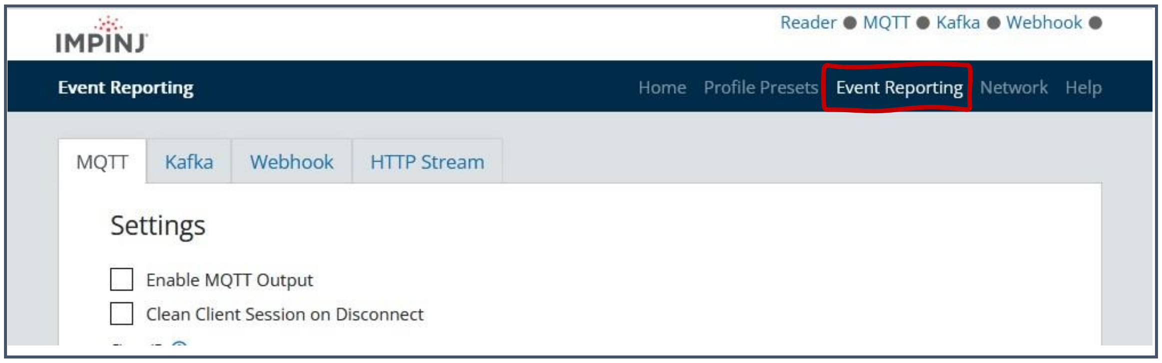
curl -X PUT https://<READER IP or HOSTNAME>/api/v1/mqtt -k -u root:impinj -H 'Content-Type: application/json' -H 'Accept: application/json' --data-raw '{"brokerHostname": "mqtt.example.com", "clientId": "mqtt", "eventTopic": "/", "tlsEnabled": true }'

Figure 7: Enable TLS in MQTT Event Reporting

```
$ curl -X PUT https://impinj-14-04-f1/api/v1/mqtt -k -u root:impinj -H 'Content-Type:
application/json' -H 'Accept: application/json' --data-raw '{"brokerHostname":
"mqtt.example.com", "clientId": "mqtt", "eventTopic": "/", "tlsEnabled": true }'
```

Once TLS is enabled, open the reader's Web UI in a local browser and click the 'Event Reporting' link at the top right of the page. This opens the configuration page for MQTT, Kafka, Webhook, and HTTP Stream event reporting, as shown below:

Figure 8: Impinj IoT Device Interface – Homepage

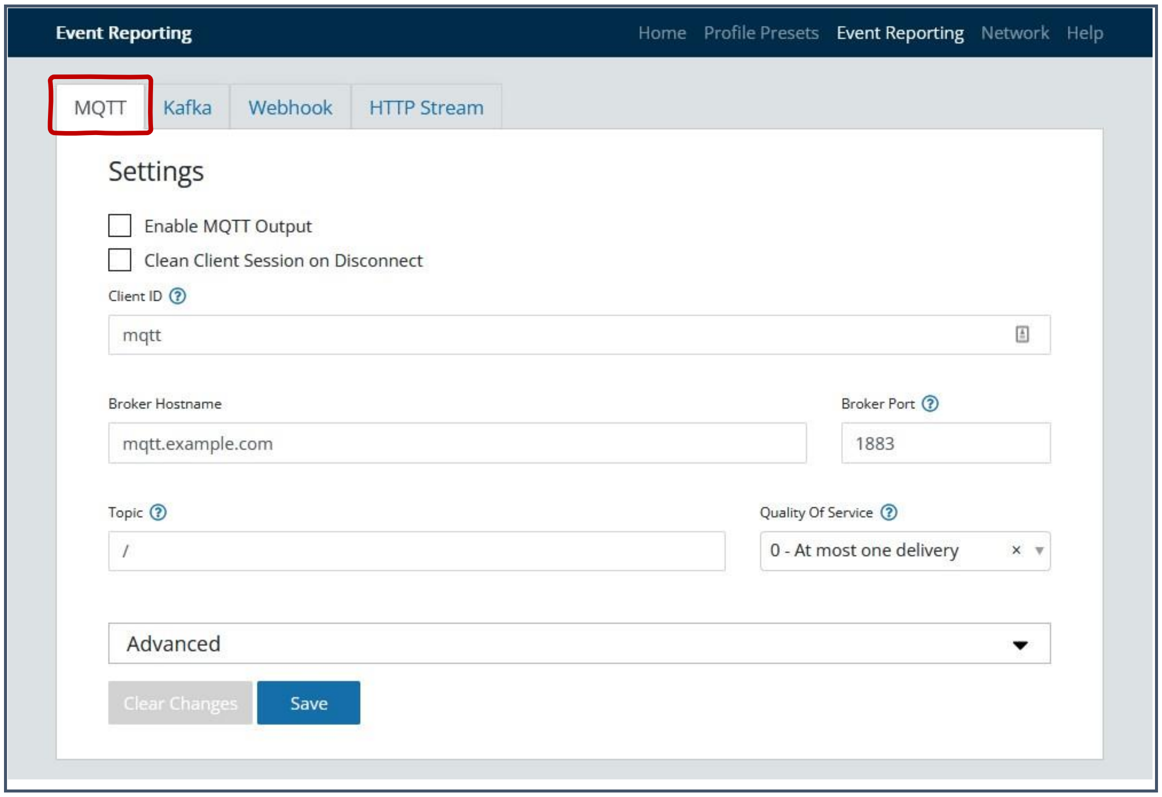


Settings

☐ Enable MQTT Output
☐ Clean Client Session on Disconnect

Selecting the MQTT tab will reveal the following MQTT settings:

Figure 9: Impinj IoT Device Interface – MQTT Event Configuration



Settings

☐ Enable MQTT Output
☐ Clean Client Session on Disconnect

Client ID: mqtt

Broker Hostname: mqtt.example.com

Broker Port: 1883

Topic: /

Quality Of Service: 0 - At most one delivery

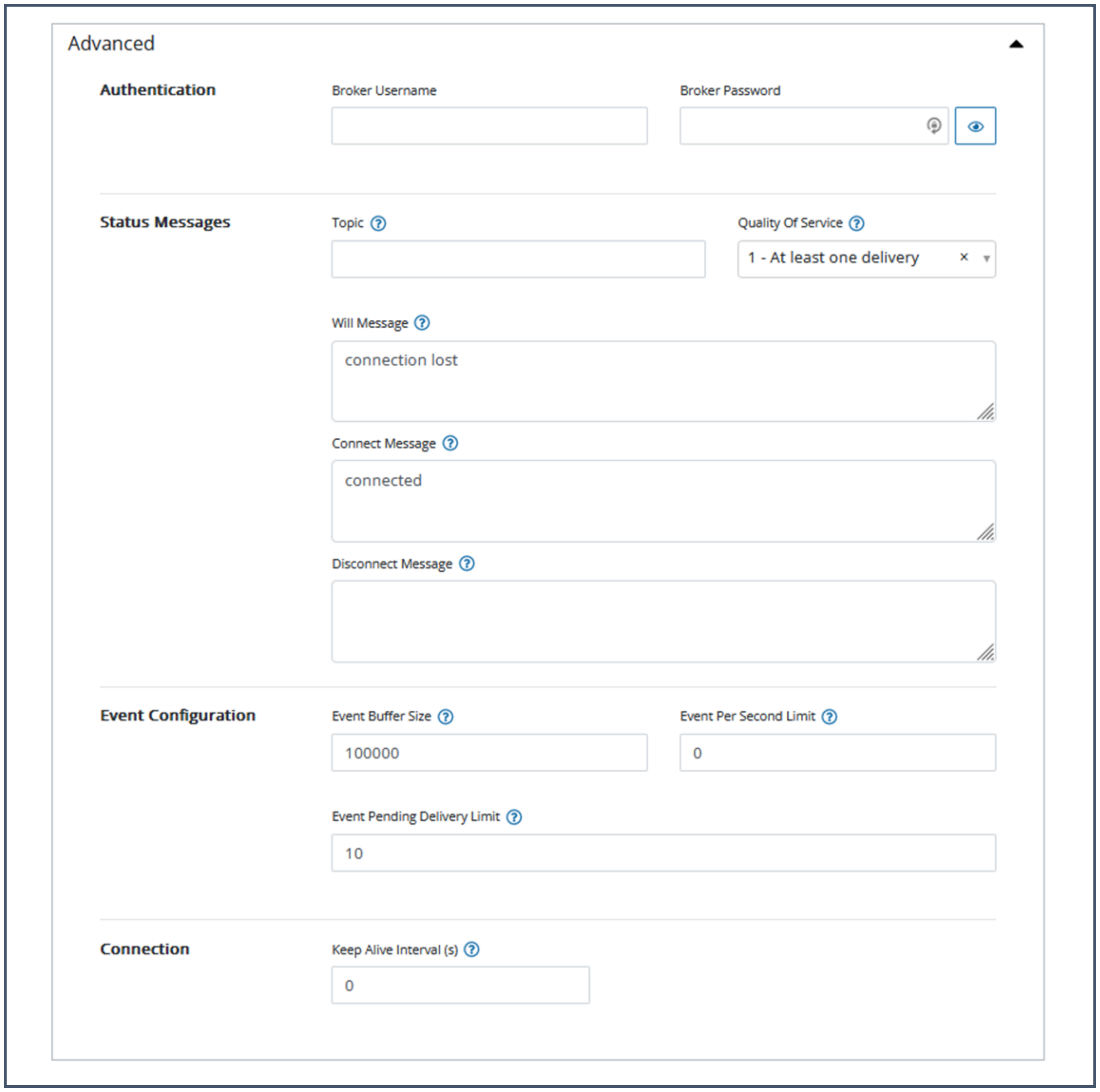
For a basic configuration, the following options are required:

Option	Description	Setting
Client ID	A string used to uniquely identify this device to the MQTT broker.	A string matching the AWS IoT Thing name.
Broker Hostname	The hostname or IO address of the target MQTT broker.	The AWS IoT Device Data Endpoint URL specified in AWS IoT Settings.
Broker Port	The port to use when communicating to the broker host.	8883
Topic	The base topic to which events will be published.	An arbitrary string. It is recommended that the format of this string include salient information about the reader such as installation site name and reader location (e.g. /Impinj/HQ/Test_Lab)
Quality of Service	The Quality of Service (QoS) level for sent messages.	A minimum of QoS 1 is recommended.

Advanced MQTT Configuration Options

In addition to the basic MQTT settings, the Impinj R700 also provides some extended options that can be used to help prevent overloading the MQTT broker and ensure the delivery of messages when network outages occur. These are accessible by expanding the “Advanced” section in the MQTT settings tab to reveal the options shown below:

Figure 10: Impinj IoT Device Interface – MQTT Advanced Configuration



Advanced

Authentication

Broker Username:

Broker Password:

Status Messages

Topic:

Quality Of Service: 1 - At least one delivery

Will Message: connection lost

Connect Message: connected

Disconnect Message:

Event Configuration

Event Buffer Size: 100000

Event Per Second Limit: 0

Event Pending Delivery Limit: 10

Connection

Keep Alive Interval (s): 0

The following table outlines what these options are used for, and what advanced settings to use with AWS IoT:

Option	Description	Setting
Broker Username	The username to use when authenticating with the broker.	Not required

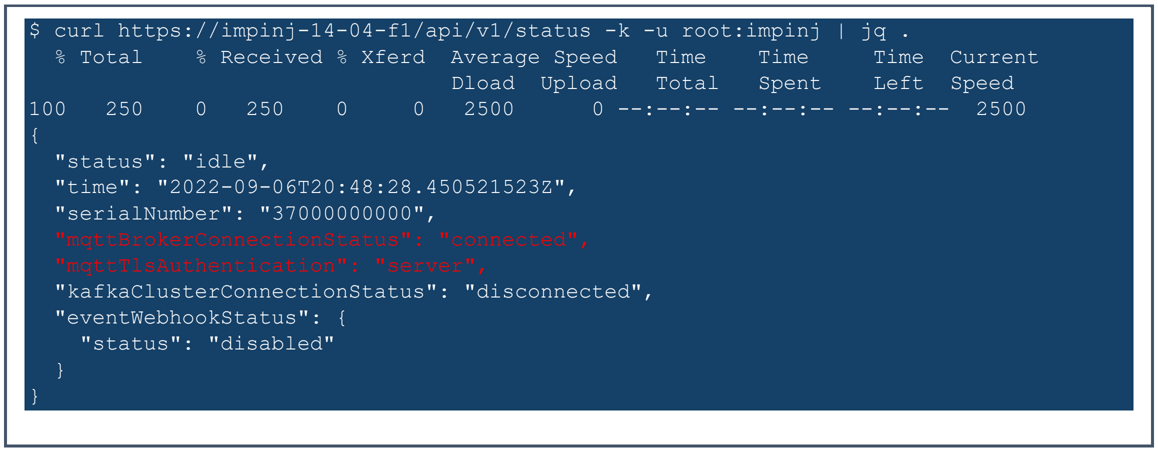
Broker Password	The password to use when authenticating with the broker	Not required
Topic	The topic where the "Last Will and Testament", "Connect" and "Disconnect" messages will be published.	Optional
Quality of Service	The Quality of Service (QoS) level for "Will Message", "Connect Message", and "Disconnect Message".	Optional
Will Message	The status message to be sent when the reader disconnects ungracefully.	Optional
Connect Message	The status message to be sent when the reader connects or reconnects.	Optional
Disconnect Message	The status message to be sent when the reader disconnects from the broker gracefully.	Optional
Event Buffer Size	The number of events that can be stored on the reader waiting for delivery to the MQTT broker.	Optional
Event Per Second Limit	The maximum number of events that may be delivered per second.	Optional

Verifying Successful Connection

Although not available on the Web UI, the R700 RESTful API provides a /status endpoint that can be checked to verify when the MQTT client has successfully connected to the MQTT broker. The endpoint returns a JSON packet, such as the following that indicates whether an MQTT connection is successful:

curl https://<READER IP or HOSTNAME>/api/v1/status -k -u root:impinj

Figure 11: Verifying a Successful Connection



```
$ curl https://impinj-14-04-f1/api/v1/status -k -u root:impinj | jq .
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   250    0   250    0     0   2500      0 --:--:-- --:--:-- --:--:--  2500
{
  "status": "idle",
  "time": "2022-09-06T20:48:28.450521523Z",
  "serialNumber": "37000000000",
  "mqttBrokerConnectionStatus": "connected",
  "mqttTlsAuthentication": "server",
  "kafkaClusterConnectionStatus": "disconnected",
  "eventWebhookStatus": {
    "status": "disabled"
  }
}
```

Note: the **‘ | jq.’** portion of the command above is optional and only used for formatting JSON output.

NOTICES