Cassia Networks Headquarters:
97 East Brokaw Road, Suite 130A/B
San Jose, CA 95112
support@cassianetworks.com| 408-444--9150

Cassia MQTT User Guide

Contents

Cassia Networks Headquarters:
97 East Brokaw Road, Suite 130A/B
San Jose, CA 95112
support@cassianetworks.com | 408-444--9150

1. Introduction

MQTT (MQ Telemetry Transport) is described on the mqtt.org site as a machine-to-machine (M2M) / IoT connectivity protocol. It is a publish/subscribe messaging transport protocol, designed for constrained devices and low-bandwidth, high-latency or unreliable networks. The design principles are to minimize network bandwidth and device resource requirements whilst attempting to ensure reliability and some degree of assurance of delivery. These principles turn out to make the protocol ideal for the emerging "machine-to-machine" (M2M) or "Internet of Things" world of connected devices, and for mobile applications where bandwidth and battery power are at a premium.[1]

1.1. Client and Broker

Cassia router has implemented an MQTT client which will publish advertise messages it receives from the BLE sensors to an MQTT server/broker. Other clients can subscribe to the topics from the MQTT-Broker. See Figure 1 for a data flow diagram[2].

Note: Cassia router can only serve as a publisher.

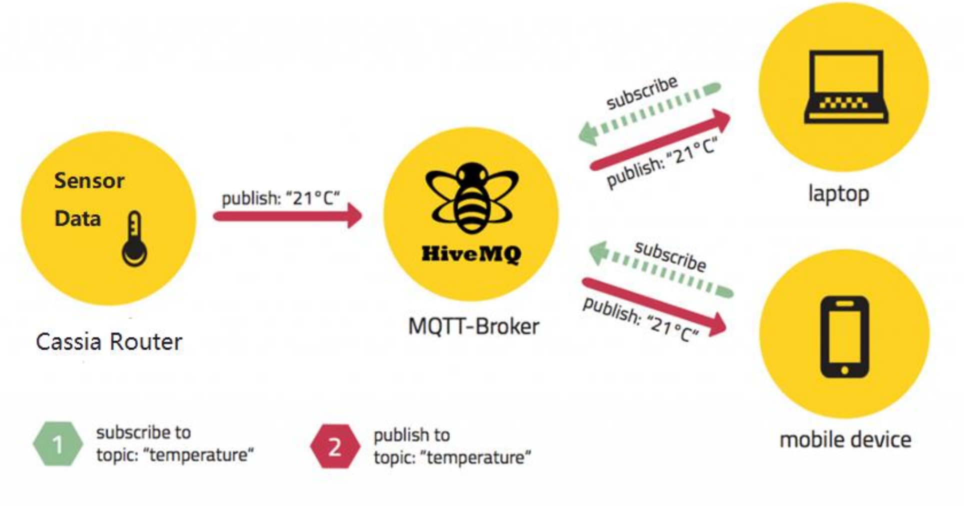


Figure 1 MQTT data flow diagram

1.2. MQTT Connection

The MQTT protocol is based on top of TCP/IP and both client and broker need to have a TCP/IP stack. The MQTT connection itself is always between one client and the broker, no client is connected to another client directly. The connection is initiated through a client sending a CONNECT message to the broker. The broker responses with a CONNACK and a status code. See Figure 2 for MQTT connection

[1] http://mqtt.org/faq
[2] https://www.hivemq.com/blog/mqtt-essentials-part2-publish-subscribe

diagram. Once the connection is established, the broker will keep it open as long as the client doesn't send a disconnect command or it loses the connection.

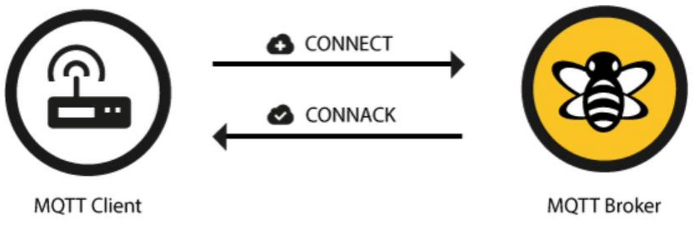


Figure 2 MQTT connection diagram[3]

Since the MQTT client always initiate the connection, there is no problem if client and broker are located in two different networks and the client is behind a NAT. A typical deployment is that you place Cassia routers inside a private network behind a firewall and the MQTT Broker in the cloud.

Note: Please make sure to open the TCP port that your MQTT service is using. By default, MQTT port is TCP 1883, and 8883 for MQTT over SSL. You can change them if needed.

1.3. Cassia MQTT Bypass Mode

Cassia MQTT is supported on bypass traffic, which means the BLE sensor data will be sent directly from the router to the MQTT broker, but the control messages are still sent to Cassia AC under CAPWAP, such as configuration changes, firmware update, etc. See Figure 3 for Cassia MQTT Bypass Architecture.

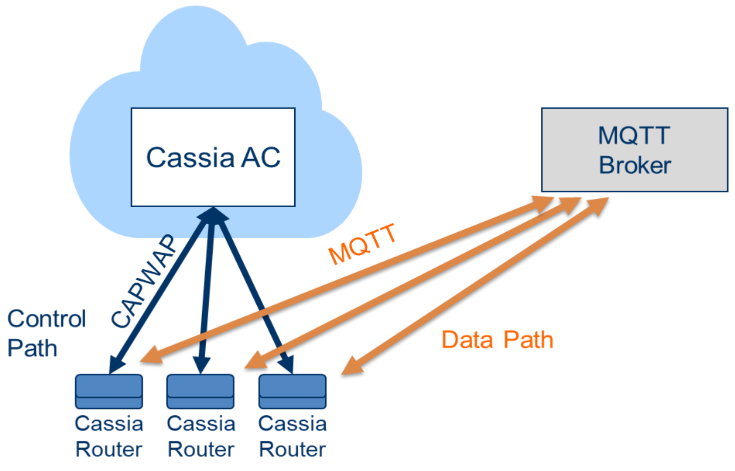


Figure 3 Cassia MQTT Bypass Architecture

[3] https://www.hivemq.com/blog/mqtt-essentials-part-3-client-broker-connection-establishment

2. Settings on MQTT Server/Broker

Setup your own MQTT Server/broker. You can reference online resources to install Mosquitto or other open source to achieve your goal. Details are beyond the scope of this document.

Note down your MQTT Server's SSH username and password. For example, if your MQTT Server's IP address is 1.1.1.1, username is root and password is cassia. Please use the following methods to login onto this server:

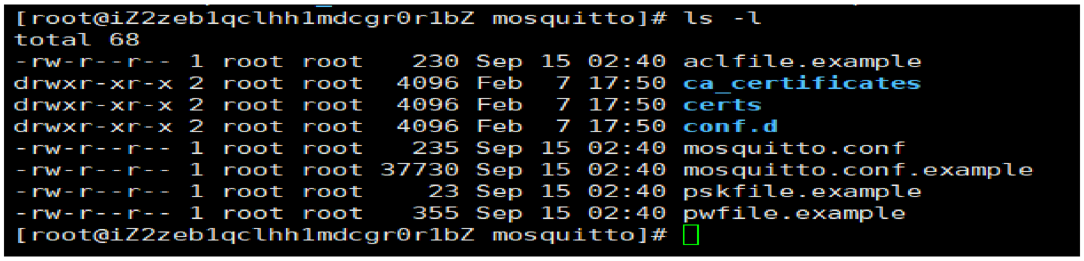
- Under Linux or MAC command line, type "ssh root@1.1.1.1", then enter its password "cassia".
- Under Windows OS, open an SSH client like SecureCRT, enter 1.1.1.1 into Hostname, then fill in username and password to connect.

2.1. Configuration Files

Change your current directory to MQTT configuration files

```
# cd /etc/mosquitto/
# ls -l
```

You will see a list of files in this directory.



```
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]# ls -l
total 68
-rw-r--r-- 1 root root   230 Sep 15 02:40 aclfile.example
drwxr-xr-x 2 root root  4096 Feb  7 17:50 ca_certificates
drwxr-xr-x 2 root root  4096 Feb  7 17:50 certs
drwxr-xr-x 2 root root  4096 Feb  7 17:50 conf.d
-rw-r--r-- 1 root root   235 Sep 15 02:40 mosquitto.conf
-rw-r--r-- 1 root root 37730 Sep 15 02:40 mosquitto.conf.example
-rw-r--r-- 1 root root    23 Sep 15 02:40 pskfile.example
-rw-r--r-- 1 root root   355 Sep 15 02:40 pwfile.example
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]#
```

Below are the ones you might be using:

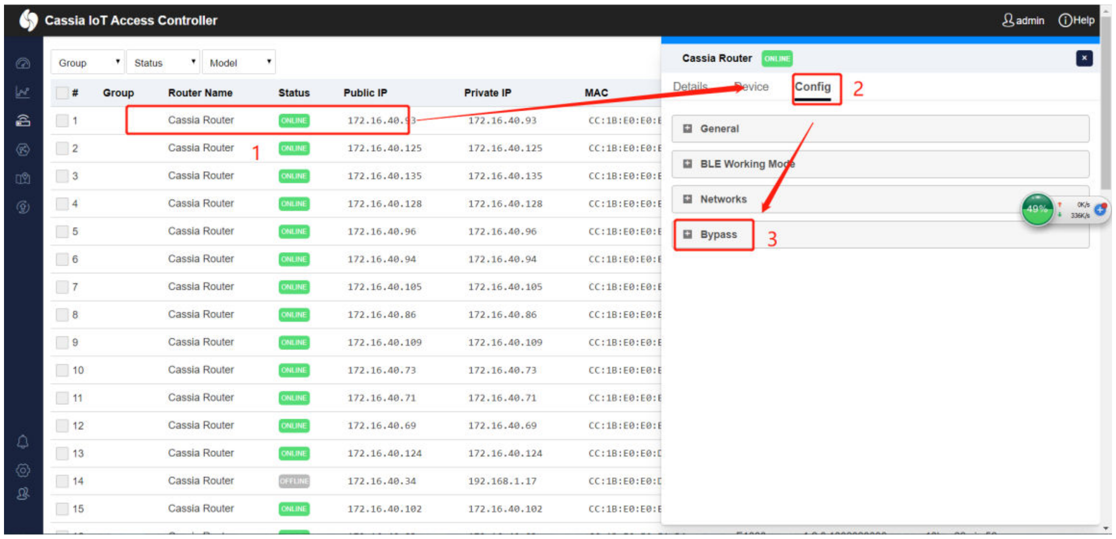
- mosquitto.conf: this is the main configuration file. You can use mosquitto.conf.example as a template to add/delete parameters you need.

  Important: In order for changes in the mosquitto.conf file to become effective you must restart the service after making the changes.
- pwfile: user account file. You can check pwfile.example for an example.
- pskfile: config file for pre-shared-key. You can check pskfile.example for an example.
- ca_certificates: a directory that stores the certificates and keys.

Because the MQTT server and MQTT client work in pair, you will have to modify configurations on both sides before them can take effect. For example: If you want to use PSK for encryption, first you need modify the configuration file on the server to support PSK encryption and restart the service. Once that completes, you need set the correct PSK parameters on the Cassia router, then you will be able to use this function.

3. Client Settings on Cassia Routers

Open your Access Controller, navigate to Routers page, select a router and click Edit, then Config tab -> Bypass. See Figure 4.



#	Group	Router Name	Status	Public IP	Private IP	MAC
1		Cassia Router	ONLINE	172.16.40.93	172.16.40.93	CC:1B:E0:E0:E
2		Cassia Router	ONLINE	172.16.40.125	172.16.40.125	CC:1B:E0:E0:E
3		Cassia Router	ONLINE	172.16.40.135	172.16.40.135	CC:1B:E0:E0:E
4		Cassia Router	ONLINE	172.16.40.128	172.16.40.128	CC:1B:E0:E0:E
5		Cassia Router	ONLINE	172.16.40.96	172.16.40.96	CC:1B:E0:E0:E
6		Cassia Router	ONLINE	172.16.40.94	172.16.40.94	CC:1B:E0:E0:E
7		Cassia Router	ONLINE	172.16.40.105	172.16.40.105	CC:1B:E0:E0:E
8		Cassia Router	ONLINE	172.16.40.86	172.16.40.86	CC:1B:E0:E0:E
9		Cassia Router	ONLINE	172.16.40.109	172.16.40.109	CC:1B:E0:E0:E
10		Cassia Router	ONLINE	172.16.40.73	172.16.40.73	CC:1B:E0:E0:E
11		Cassia Router	ONLINE	172.16.40.71	172.16.40.71	CC:1B:E0:E0:E
12		Cassia Router	ONLINE	172.16.40.69	172.16.40.69	CC:1B:E0:E0:E
13		Cassia Router	ONLINE	172.16.40.124	172.16.40.124	CC:1B:E0:E0:D
14		Cassia Router	OFFLINE	172.16.40.34	192.168.1.17	CC:1B:E0:E0:D
15		Cassia Router	ONLINE	172.16.40.102	172.16.40.102	CC:1B:E0:E0:E

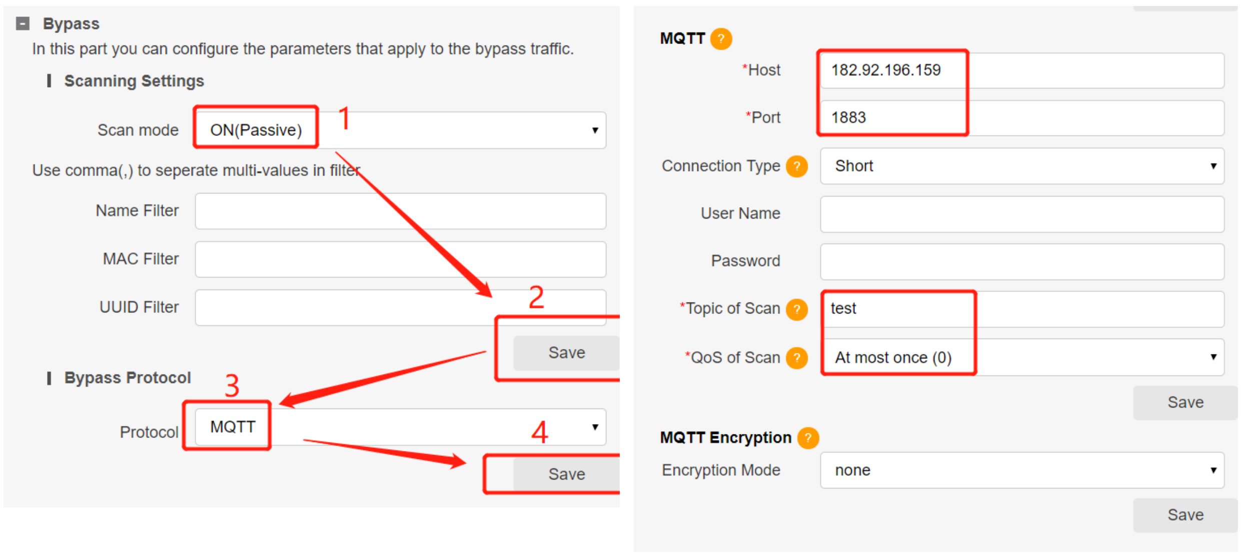
Cassia Router ONLINE

Details | Device | Config

- General
- BLE Working Mode
- Networks
- Bypass

Figure 4 Cassia Router Bypass Configuration

Turn on Scan mode, either passive or active, save it. Then select MQTT as the Bypass protocol and save. See Figure 5 Cassia Router MQTT Configuration.



Bypass

In this part you can configure the parameters that apply to the bypass traffic.

Scanning Settings

Scan mode	ON(Passive)

Use comma(,) to seperate multi-values in filter

Name Filter	
MAC Filter	
UUID Filter	

Bypass Protocol

Protocol	MQTT

MQTT

*Host	182.92.196.159
*Port	1883
Connection Type	Short
User Name	
Password	
*Topic of Scan	test
*QoS of Scan	At most once (0)

MQTT Encryption

Encryption Mode	none

Figure 5 Cassia Router MQTT Configuration

The parameters on MQTT client are:

- Host: the MQTT Broker domain name or IP address
- Port: fill in the MQTT port number configured in mosquitto.conf on the server (default is 1883)
- Connection type: long or short. Default is long.
  - Short: MQTT Client disconnects from the Broker right after it publishes the message.
  - Long: MQTT Client keeps its connection with the Broker.

  Note: If you send messages frequently, or you use certificate to encrypt messages, suggest to use long connection to reduce the overhead. Otherwise short connection is preferred.

- Username and Password: need to be consistent with what have been configured in Server main config file mosquitto.conf
- Topic of Scan: A topic is a UTF-8 string, which is used by the broker to filter messages for each connected client. A topic consists of one or more topic levels. Each topic level is separated by a forward slash (topic level separator). For example: Cassia/Heartbeat
- QoS of Scan: Quality of service (QoS) levels determine how each MQTT message is delivered and must be specified for every message sent through MQTT. Available options are at most once (0), at least once (1), or exactly once (2).
- Encryption Mode: none, pre-shared-key or certificate.

4. Use Scenarios

You can apply user authentication and data encryption in MQTT service. There are four different settings which match four scenarios.

4.1. No encryption, no authentication

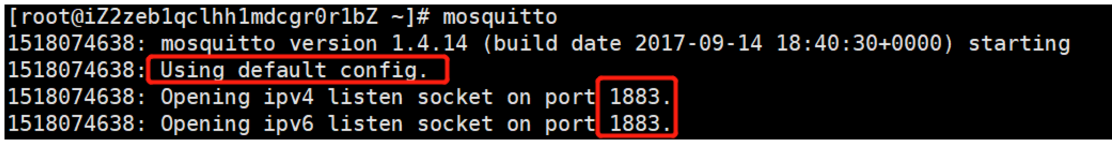
Server

The MQTT service can be started directly with the default configuration. This is the simplest way to use MQTT. However, the MQTT service has only the most basic functions. It doesn't authenticate users and nor does it support encryption. It runs on the default port 1883. Of course, you can change it later.

To start from default configuration, execute the following command.

```
#mosquitto
```

From screenshot below you can see that MQTT service is started successfully and listens to port 1883.



```
[root@iZ2zeb1qclhh1mdcgr0r1bZ ~]# mosquitto
1518074638: mosquitto version 1.4.14 (build date 2017-09-14 18:40:30+0000) starting
1518074638: Using default config.
1518074638: Opening ipv4 listen socket on port 1883.
1518074638: Opening ipv6 listen socket on port 1883.
```

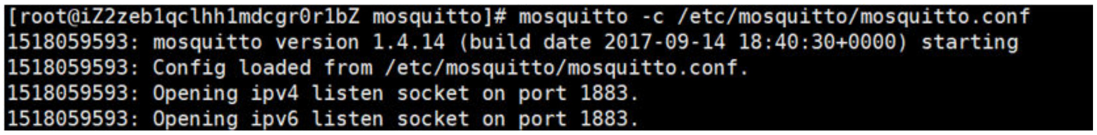
If for any reason you don't want to use the default configuration, you can execute on the main configuration file. Please follow the following steps.

a) Open mosquitto.conf, change "allow_anonymous false" to "allow_anonymous true"
b) Comment "#" out the rest of configurations
c) Change the port number if needed
d) Restart the service:

   # killall mosquitto

   # mosquitto -c /etc/mosquitto/mosquitto.conf -v

If everything works as expected, you will see below screenshot.

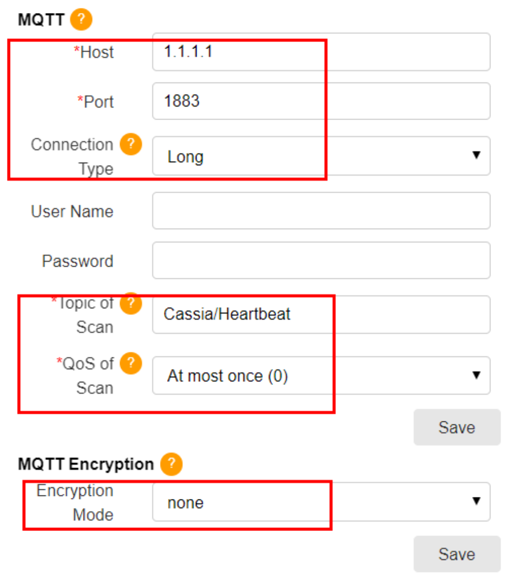


```
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]# mosquitto -c /etc/mosquitto/mosquitto.conf
1518059593: mosquitto version 1.4.14 (build date 2017-09-14 18:40:30+0000) starting
1518059593: Config loaded from /etc/mosquitto/mosquitto.conf.
1518059593: Opening ipv4 listen socket on port 1883.
1518059593: Opening ipv6 listen socket on port 1883.
```

Router

- Connection type: suggest to select long, as Cassia MQTT bypass is sending scanning data constantly, maintain a long connection will reduce protocol overhead.
- Username and Password: can be anything or leave them empty
- Encryption Mode: Select "none" for not using encryption.

See Figure 6 below for an example.


MQTT

*Host	1.1.1.1
*Port	1883
Connection Type	Long
User Name	
Password	
*Topic of Scan	Cassia/Heartbeat
*QoS of Scan	At most once (0)

Save

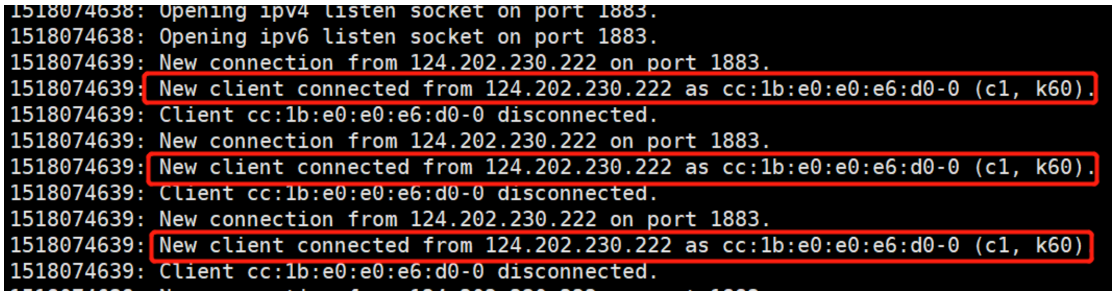
MQTT Encryption

Encryption Mode	none

Save

Figure 6 Cassia Router MQTT Config – no authentication, no encryption

If everything works as expected, Sever will receive MQTT messages sent from the router. See below screenshot.


```
1518074638: Opening ipv4 listen socket on port 1883.
1518074638: Opening ipv6 listen socket on port 1883.
1518074639: New connection from 124.202.230.222 on port 1883.
1518074639: New client connected from 124.202.230.222 as cc:1b:e0:e0:e6:d0-0 (c1, k60).
1518074639: Client cc:1b:e0:e0:e6:d0-0 disconnected.
1518074639: New connection from 124.202.230.222 on port 1883.
1518074639: New client connected from 124.202.230.222 as cc:1b:e0:e0:e6:d0-0 (c1, k60).
1518074639: Client cc:1b:e0:e0:e6:d0-0 disconnected.
1518074639: New connection from 124.202.230.222 on port 1883.
1518074639: New client connected from 124.202.230.222 as cc:1b:e0:e0:e6:d0-0 (c1, k60)
1518074639: Client cc:1b:e0:e0:e6:d0-0 disconnected.
```

4.2. No Encryption, Use Authentication

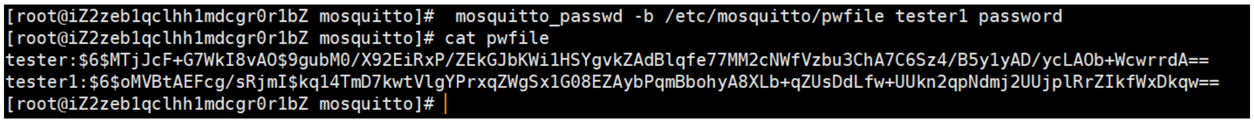
The MQTT service we started above is simple, but not secure. This section we will add user authentication which means that our MQTT server only receives data sent by authenticated users. Similar to the login function, only when the username and password are fully verified, the Server will accept the user's connection request, which can protect our MQTT server to some extent.

If you want to configure user authentication, first we should generate users on the MQTT server and then configure it on our Bluetooth router.

Server

a) First, create a user on the server. For example, you can create a user “tester1” with password “password”.

```
# mosquitto_passwd -b /etc/mosquitto/pwfile tester1 password
```



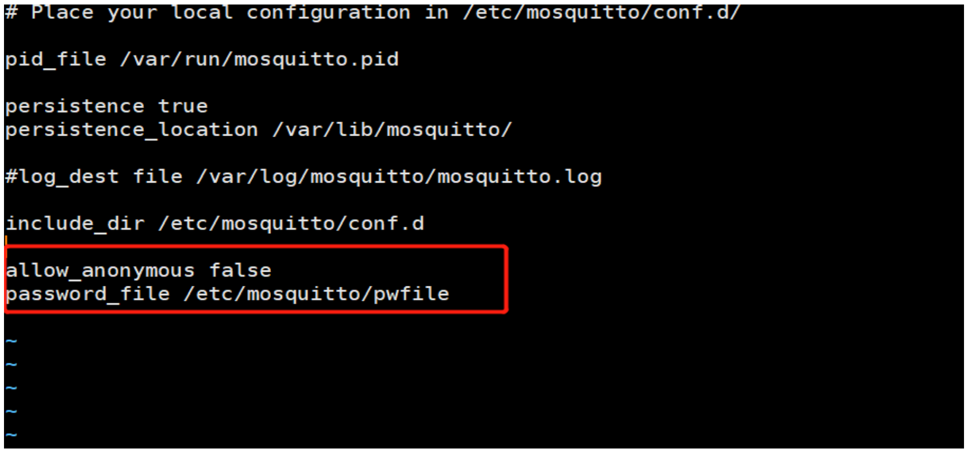
```
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]#  mosquitto_passwd -b /etc/mosquitto/pwfile tester1 password
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]# cat pwfile
tester:$6$MTjJcF+G7WkI8vAO$9gubM0/X92EiRxP/ZEkGJbKWi1HSYgvkZAdBlqfe77MM2cNWfVzbu3ChA7C6Sz4/B5y1yAD/ycLAOb+WcwrrdA==
tester1:$6$oMVBtAEFcg/sRjmI$kq14TmD7kwtVlgYPrxqZWgSx1G08EZAybPqmBbohyA8XLb+qZUsDdLfw+UUkn2qpNdmj2UUjplRrZIkfWxDkqw==
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]#
```

For the same user you can use this command to change its password

b) Second, to enable user authentication you need open mosquitto.conf, add the following settings into it.

   allow_anonymous false

   password_file /etc/mosquitto/pwfile



```
# Place your local configuration in /etc/mosquitto/conf.d/

pid_file /var/run/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

#log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

allow_anonymous false
password_file /etc/mosquitto/pwfile

~
~
~
~
~
```

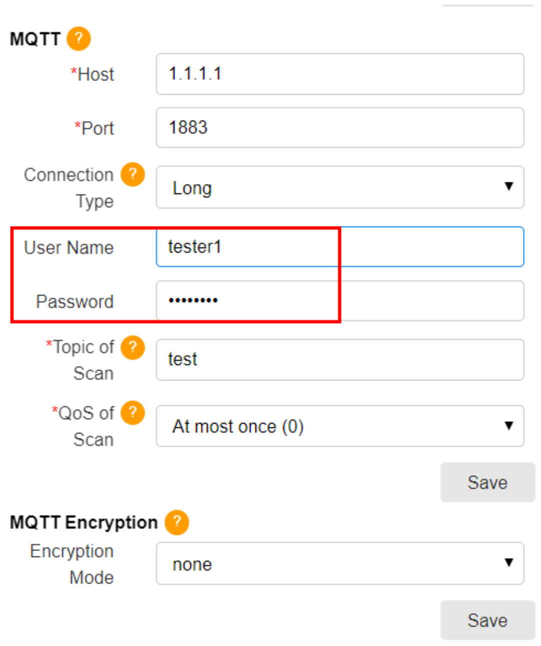
c) Restart the service:

   # killall mosquito

   # mosquitto -c /etc/mosquitto/mosquitto.conf -v

Router

Fill in the Username and Password exactly the same as the Server. The rest of settings are the same as scenario 4.1. See Figure 7.

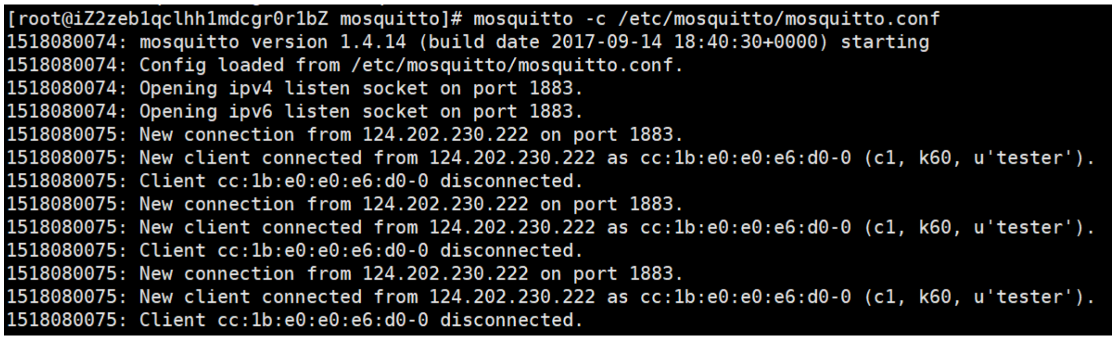


MQTT
*Host: 1.1.1.1
*Port: 1883
Connection Type: Long
User Name: tester1
Password: ••••••••
*Topic of Scan: test
*QoS of Scan: At most once (0)

MQTT Encryption
Encryption Mode: none

Figure 7 Configure MQTT User Authentication on Cassia Router

If everything works as expected, you will see response messages on the Server like this (the example below is using short connection. For long connection you won't see the client disconnect so frequently.):



```
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]# mosquitto -c /etc/mosquitto/mosquitto.conf
1518080074: mosquitto version 1.4.14 (build date 2017-09-14 18:40:30+0000) starting
1518080074: Config loaded from /etc/mosquitto/mosquitto.conf.
1518080074: Opening ipv4 listen socket on port 1883.
1518080074: Opening ipv6 listen socket on port 1883.
1518080075: New connection from 124.202.230.222 on port 1883.
1518080075: New client connected from 124.202.230.222 as cc:1b:e0:e0:e6:d0-0 (c1, k60, u'tester').
1518080075: Client cc:1b:e0:e0:e6:d0-0 disconnected.
1518080075: New connection from 124.202.230.222 on port 1883.
1518080075: New client connected from 124.202.230.222 as cc:1b:e0:e0:e6:d0-0 (c1, k60, u'tester').
1518080075: Client cc:1b:e0:e0:e6:d0-0 disconnected.
1518080075: New connection from 124.202.230.222 on port 1883.
1518080075: New client connected from 124.202.230.222 as cc:1b:e0:e0:e6:d0-0 (c1, k60, u'tester').
1518080075: Client cc:1b:e0:e0:e6:d0-0 disconnected.
```

4.3. PSK Encryption, User Authentication

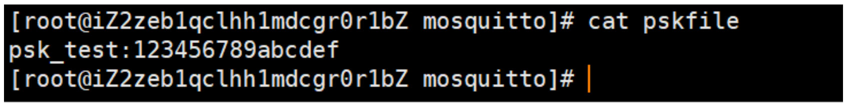
In the previous section we talked about the user's authentication function, but this is still not secure enough. Our communication data is transmitted in plaintext. This section we will turn on PSK key encryption on the MQTT server to further improve our security. Encryption and user authentication can be enabled at the same time.

Server

Similar to user authentication, if we want to enable PSK encryption, we need to configure the psk file on the MQTT server:

a) First create a file named pskfile and fill in the PSK key pair. Note that the key pair must have the format identity:key, and the key value must be a hexadecimal number without a prefix.

   Example: identity=psk_test，key=123456789abcdef

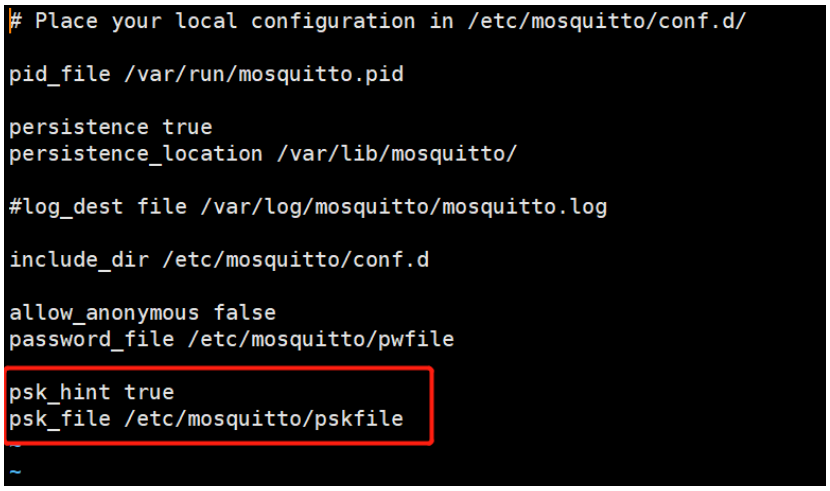


```
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]# cat pskfile
psk_test:123456789abcdef
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]#
```

b) Open mosquitto.conf, add the following settings into the file.

   psk_hint true

   psk_file /etc/mosquitto/pskfile



```
# Place your local configuration in /etc/mosquitto/conf.d/

pid_file /var/run/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

#log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

allow_anonymous false
password_file /etc/mosquitto/pwfile

psk_hint true
psk_file /etc/mosquitto/pskfile
~
~
```

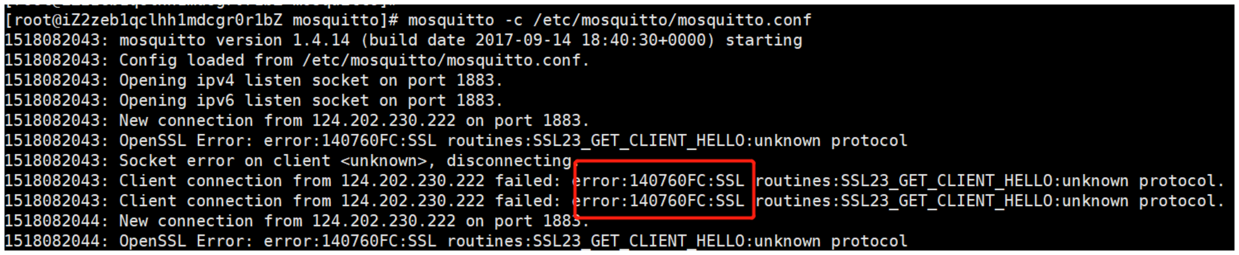
c) Kill the service:

   # killall mosquito

d) Restart the service:

   # mosquitto -c /etc/mosquitto/mosquitto.conf -v

Since we haven't changed the configuration on the router yet, the server will throw an SSL error. This proves that our PSK configuration on the server has taken effect. Just need to finish the configuration on the router side.



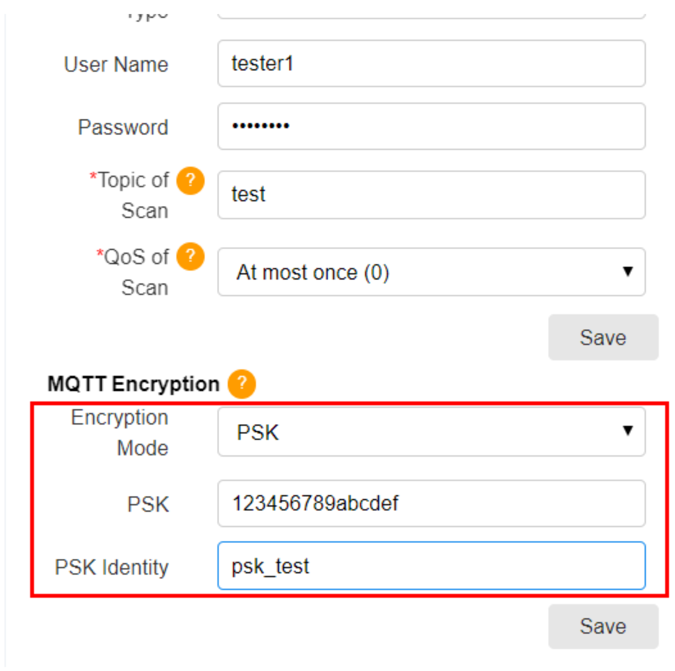
```
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]# mosquitto -c /etc/mosquitto/mosquitto.conf
1518082043: mosquitto version 1.4.14 (build date 2017-09-14 18:40:30+0000) starting
1518082043: Config loaded from /etc/mosquitto/mosquitto.conf.
1518082043: Opening ipv4 listen socket on port 1883.
1518082043: Opening ipv6 listen socket on port 1883.
1518082043: New connection from 124.202.230.222 on port 1883.
1518082043: OpenSSL Error: error:140760FC:SSL routines:SSL23_GET_CLIENT_HELLO:unknown protocol
1518082043: Socket error on client <unknown>, disconnecting.
1518082043: Client connection from 124.202.230.222 failed: error:140760FC:SSL routines:SSL23_GET_CLIENT_HELLO:unknown protocol.
1518082043: Client connection from 124.202.230.222 failed: error:140760FC:SSL routines:SSL23_GET_CLIENT_HELLO:unknown protocol.
1518082044: New connection from 124.202.230.222 on port 1883.
1518082044: OpenSSL Error: error:140760FC:SSL routines:SSL23_GET_CLIENT_HELLO:unknown protocol
```

Router

For this scenario you need change MQTT Encryption related parameters:

- Encryption Mode: Select "PSK" for using pre-shared-key encryption.
- PSK: fill in the key configured in the pskfile on your MQTT server. It is in hexadecimal (no leading 0x).
- PSK Identity: fill in the client identify string configured in the pskfile on your MQTT server.

See Figure 8 for an example.



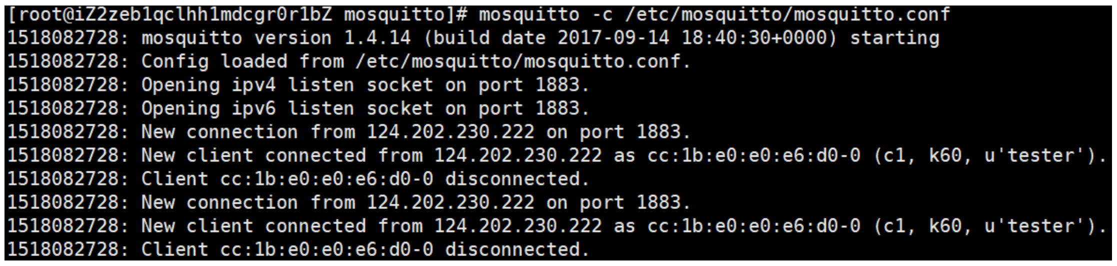
User Name	tester1
Password	••••••••
*Topic of Scan	test
*QoS of Scan	At most once (0)

MQTT Encryption

Encryption Mode	PSK
PSK	123456789abcdef
PSK Identity	psk_test

Figure 8 Configure MQTT Pre-Shared-Key Encryption on Cassia Router

Now go back to your MQTT server to check if everything works as expected. You should see the following messages on the server.



```
[root@iZ2zeb1qclhh1mdcgr0r1bZ mosquitto]# mosquitto -c /etc/mosquitto/mosquitto.conf
1518082728: mosquitto version 1.4.14 (build date 2017-09-14 18:40:30+0000) starting
1518082728: Config loaded from /etc/mosquitto/mosquitto.conf.
1518082728: Opening ipv4 listen socket on port 1883.
1518082728: Opening ipv6 listen socket on port 1883.
1518082728: New connection from 124.202.230.222 on port 1883.
1518082728: New client connected from 124.202.230.222 as cc:1b:e0:e0:e6:d0-0 (c1, k60, u'tester').
1518082728: Client cc:1b:e0:e0:e6:d0-0 disconnected.
1518082728: New connection from 124.202.230.222 on port 1883.
1518082728: New client connected from 124.202.230.222 as cc:1b:e0:e0:e6:d0-0 (c1, k60, u'tester').
1518082728: Client cc:1b:e0:e0:e6:d0-0 disconnected.
```

4.4. Certificate Encryption, User Authentication

In addition to PSK encryption, you also have the option to use digital certificate to encrypt. This option is more secure than PSK encryption, but also more cumbersome to configure and can affect certain performance.

Note: Certificate Encryption and PSK Encryption cannot be supported at the same time and must be selected one by one.

If you want to configure a certificate, you must first go to the CA to apply for a certificate for your company or generate a certificate by self-signing. This is a prerequisite for you to enable the certificate encryption!

Server

a) Place your certificate files under /etc/mosquitto/certs/

ca.crt # Authority certificates that have signed your server certificate
server.crt # the PEM encoded server certificate.
server.key # the PEM encoded keyfile.

b) Comment out all psk-related configurations (including psk_hint true and psk_file /etc/mosquitto/pskfile), because certificate and psk encryption you can only enable one at a time.

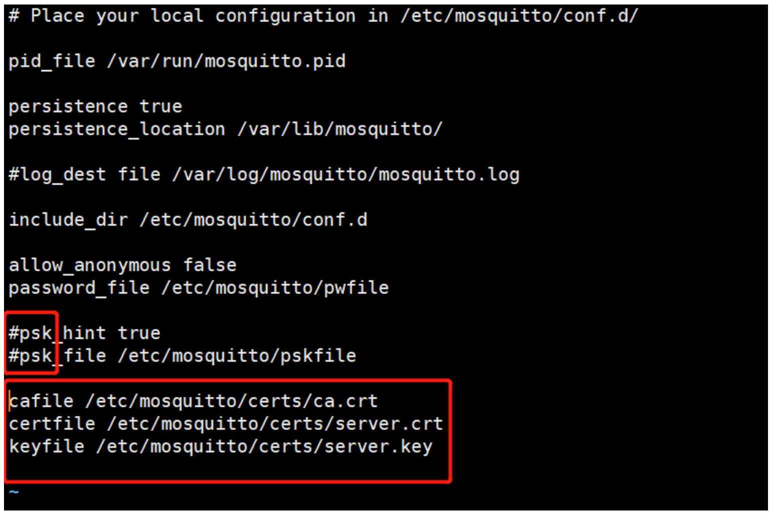
c) Add the following in the mosquitto.conf

cafile /etc/mosquitto/certs/ca.crt

certfile /etc/mosquitto/certs/server.crt

keyfile /etc/mosquitto/certs/server.key

See below screenshot for an example.



```
# Place your local configuration in /etc/mosquitto/conf.d/

pid_file /var/run/mosquitto.pid

persistence true
persistence_location /var/lib/mosquitto/

#log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d

allow_anonymous false
password_file /etc/mosquitto/pwfile

#psk_hint true
#psk_file /etc/mosquitto/pskfile

cafile /etc/mosquitto/certs/ca.crt
certfile /etc/mosquitto/certs/server.crt
keyfile /etc/mosquitto/certs/server.key
~
```

d) Restart the service

```
# killall mosquito
# mosquitto -c /etc/mosquitto/mosquitto.conf -v
```

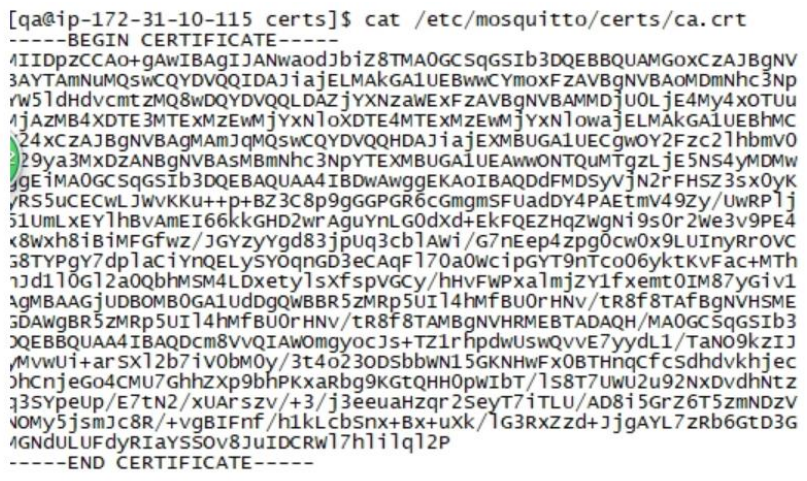
Note: Under SSL the default MQTT port is 8883.

Router

For this scenario you need change MQTT Encryption related parameters:

- Encryption Mode: Select "Certificate".
- CA certificate: copy and paste the server certificate here.
- MQTT Require Client Certificate: select No to only certify the client to the server. Otherwise you will have to upload your client certificate.

From your MQTT server, execute "cat /etc/mosquitto/certs/ca.crt", the content of the certificate will display. See below screenshot for an example.



```
[qa@ip-172-31-10-115 certs]$ cat /etc/mosquitto/certs/ca.crt
-----BEGIN CERTIFICATE-----
MIIDpzCCAo+gAwIBAgIJANwaodJbiZ8TMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAmNuMQswCQYDVQQIDAJiajELMAkGA1UEBwwCYmoxFzAVBgNVBAoMDmNhc3Np
YW5ldHdvcmtzMQ8wDQYDVQQLDAZjYXNzaWExFzAVBgNVBAMMDjU0LjE4My4xOTUu
MjAzMB4XDTE3MTExMzEwMjYxNloXDTE4MTExMzEwMjYxNlowajELMAkGA1UEBhMC
Y24xCzAJBgNVBAgMAmJqMQswCQYDVQQHDAJiajEXMBUGA1UECgwOY2Fzc2lhbmV0
d29ya3MxDzANBgNVBAsMBmNhc3NpYTEXMBUGA1UEAwwONTQuMTgzLjE5NS4yMDMw
ggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQDdFMDSyVjN2rFHSZ3sx0yK
yRS5uCECwLJWvKKu++p+BZ3C8p9gGGPGR6cGmgmSFUadDY4PAEtmV49Zy/UwRPlj
51UmLxEYlhBvAmEI66kkGHD2wrAguYnLG0dXd+EkFQEZHqZWgNi9s0r2We3v9PE4
k8Wxh8iBiMFGfwz/JGYzyYgd83jpUq3cblAWi/G7nEep4zpg0cw0x9LUInyRrOVC
G8TYPgY7dplaCiYnQELySYOqnGD3eCAqFl70a0WcipGYT9nTco06yktKvFac+MTh
mJd1l0Gl2a0QbhMSM4LDxetylsXfspVGCy/hHvFWPxalmjZY1fxemt0IM87yGiv1
AgMBAAGjUDBOMB0GA1UdDgQWBBR5zMRp5UIl4hMfBU0rHNv/tR8f8TAfBgNVHSME
GDAWgBR5zMRp5UIl4hMfBU0rHNv/tR8f8TAMBgNVHRMEBTADAQH/MA0GCSqGSIb3
DQEBBQUAA4IBAQDcm8VvQIAWOmgyocJs+TZ1rhpdwUswQvvE7yydL1/TaNO9kzIJ
yMvwUi+arSXl2b7iV0bM0y/3t4o23ODSbbWN15GKNHwFx0BTHnqCfcSdhdvkhjec
DhCnjeGo4CMU7GhhZXp9bhPKxaRbg9KGtQHH0pWIbT/lS8T7UWU2u92NxDvdhNtz
q3SYpeUp/E7tN2/xUArszv/+3/j3eeuaHzqr2SeyT7iTLU/AD8i5GrZ6T5zmNDzV
NOMy5jsmJc8R/+vgBIFnf/h1kLcbSnx+Bx+uXk/lG3RxZzd+JjgAYL7zRb6GtD3G
MGNdULUFdyRIaYSSOv8JuIDCRWl7hlilql2P
-----END CERTIFICATE-----
```

Copy the certificate on the server and paste it to the router configuration page under CA Certificate. See Figure 9 for an example.

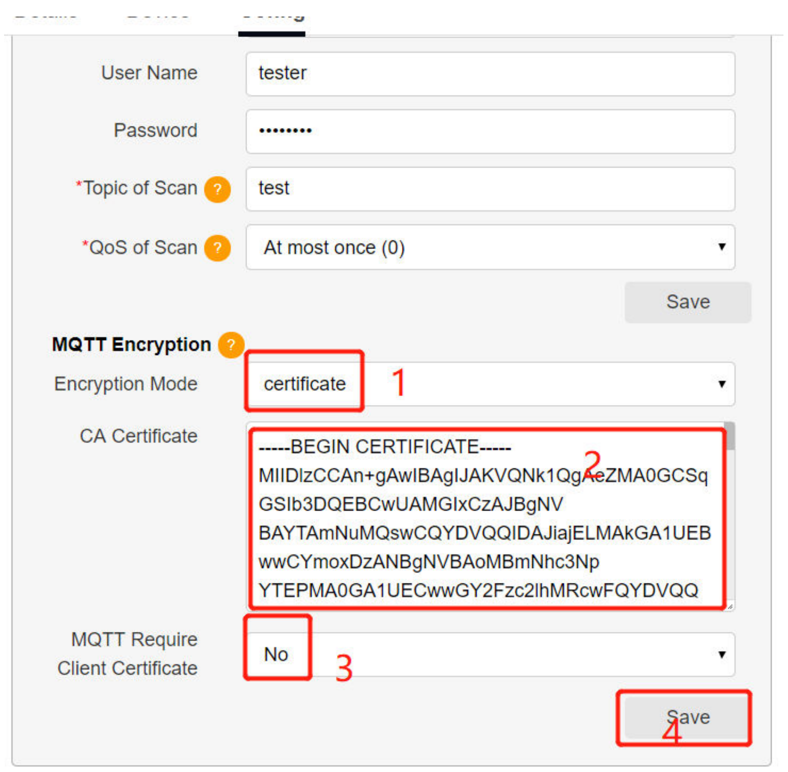

User Name: tester
Password: ••••••••
*Topic of Scan: test
*QoS of Scan: At most once (0)
Save
MQTT Encryption
Encryption Mode: certificate
CA Certificate:
```
-----BEGIN CERTIFICATE-----
MIIDlzCCAn+gAwIBAgIJAKVQNk1QgAeZMA0GCSq
GSIb3DQEBCwUAMGIxCzAJBgNV
BAYTAmNuMQswCQYDVQQIDAJiajELMAkGA1UEB
wwCYmoxDzANBgNVBAoMBmNhc3Np
YTEPMA0GA1UECwwGY2Fzc2lhMRcwFQYDVQQ
```
MQTT Require Client Certificate: No
Save

Figure 9 Configure MQTT Certificate Encryption on Cassia Router