

Weather Station Raspberry Pi Security Setup

For Pro Version of the MyWX Client

V2.3.3

By David W. Enström

Contents

Weather Station Raspberry Pi Security Setup	1
MyWX Client	2
Install MySQL (Maria Database)	2
Configuring WeeWX	2
Configuring PKI Security	3
Setup the WeeWX Database for Remote Access	3
Define PKI Certificates and Keys on the Pi	4
Point the Maria DB to the Certificates and Keys	14
Installing CA Certificate on your Android Phone	16
Define Port Forwarding to WeeWX	17
List of Figures	18
Reference Documentation	18

My personal weather station uses the SparkFun weather sensors (Argent Data Systems), and the BC Robotics add-on hardware (HAT) to interface these sensors to the Raspberry Pi.

See: <https://www.sparkfun.com/products/8942> and
<https://www.bc-robotics.com/tutorials/raspberry-pi-weather-station-part-1/>

The weather station monitors (and optionally shares the data with Wunderground):

- **Wind Speed** – in km/h
- **Wind Direction** – in degrees
- **Rainfall** – in mm
- **Rain Rate** – in mm/hr
- **Temperature** – in degrees C
- **Air Pressure** – in hecto Pascals
- **Humidity** – in percent

Instructions for constructing this weather station and all the required components are in the “Weather Station Raspberry Pi construction.pdf” PDF file. Another document, “Weather Station Raspberry Pi Software Setup for the MyWX app.pdf” describes how to install and configure the required software on the Pi, including WeeWX. You can find these documents here:

<https://github.com/David-Enst/WeeWX-BCRobotics/tree/version3>



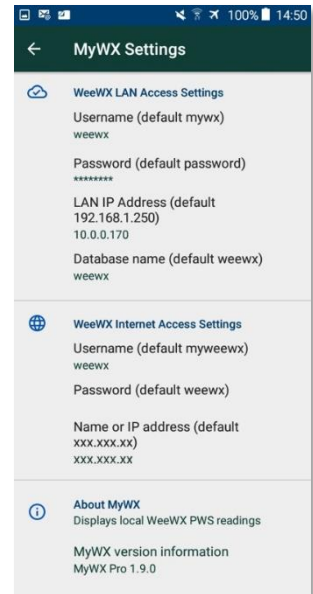
MyWX Client

The MyWX Pro android app permits the definition of two types of access and logins to the WeeWX system.

1. A simple username and password access identical to that used in the free version of the app. This login is associated with the local LAN IP address, since it is not very secure; use it only for local LAN access to the device.
2. A very secure PKI (SSL / TLS) based access process. The login is associated with a URL or Internet address. It allows for access to your WeeWX system over the Internet.

The MyWX Pro app first determines if the LAN address is accessible. If it is, then it attempts to login to the WeeWX database via this LAN-based access method. If this address is not reachable, or this login fails, then the app tries to login via the PKI-based access method.

A view of the settings screen for the MyWX Pro app is on the right. Note the two accounts defined to connect to your PWS, one for local LAN connections and one for access via the Internet. Define either or both.



Install MySQL (Maria Database)

This section sets up the mandatory MySQL database for MyWX Pro, instead of the default SQLite database. Using the MySQL database is required for multi-user applications (i.e., sharing the data with another client over your local LAN or the Internet).

First, run these commands to upgrade the current packages to the latest version.

```
$ sudo apt update
$ sudo apt upgrade
```

Verify that the MySQLdb python package is installed:

```
$ python -c "import MySQLdb"
```

If this results in an import error:

```
ImportError: No module named MySQLdb
```

Then install the MySQLdb package. For example, on Debian systems:

```
$ sudo apt-get install mariadb-client // installs the Maria SQL database
$ sudo apt-get install mariadb-server // installs the Maria SQL server
$ sudo apt-get install python3-mysqldb // installs the Python SQL software
```

Configuring WeeWX

As noted, the WeeWX configuration *must* specify MySQL instead of SQLite. In the WeeWX configuration file, change the `[[wx_binding]]` section to point to the MySQL database, `archive_mysql`, instead of the SQLite database `archive_sqlite`.

If you are switching an existing WeeWX setup, follow the process described on the WeeWX web site, see:

http://weewx.com/docs/usersguide.htm#configuring_mysql

<https://github.com/weewx/weewx/wiki/Transfer-from-sqlite-to-MySQL>

Configuring PKI Security

In order to provide secure remote (from the Internet) access to WeeWX several definitions and configurations need doing.

The process of setting up the required Certificate Authority (CA), server certificates, client certificates, and public keys is rather involved.

A summary of the steps as a preview:

- Define a specific Maria DB username that requires “X.509”
e.g. GRANT ALL ON weewx.* TO 'myweewx'@'%' REQUIRE X509;
- Define PKI certificates and keys for the Maria DB server;
- Enable SSL/TSL in the Maria DB and pointing to the defined certs and keys;
- Copy the PKI certs to your MyWX Pro client device (e.g., a phone, see below).
- Define port forwarding to your Pi on your Internet router / modem.

Setup the WeeWX Database for Remote Access

Follow these procedures to allow remote access to the MySQL server on your Pi. Note that by default the Maria Data Base only allows local access (i.e., to localhost 127.0.0.1).

In the example below, the local LAN to which the Pi is connected (and your MyWX app and device) has IP addresses in the range 192.168.1.0 to 192.168.1.256. Adjust as required.

1. In the `/etc/mysql/mariadb.conf.d/50-server.cnf` file change the line as follows to allow remote TCP/IP connections (also make sure that it is not commented out (i.e., #):

```
bind-address = *
```

2. Create a new MySQL user with required privileges for remote access:

Start the MariaDB app, and enter these commands (the user account “weewx” is created when weewx is setup and started). Note that the “%” character is a wildcard. Adjust the IP address to match your LAN setup.

```
$ sudo mysql -u root -p      // Start the interactive access to the database
Enter password:              // your root password

// Definition of the LAN access account (adjust IP address as required):

MariaDB [(none)]> CREATE USER 'myweewx'@'10.0.0.%' IDENTIFIED BY 'password';
MariaDB [(none)]> GRANT ALL ON weewx.* TO 'myweewx'@'10.0.0.%';

// Definition of the WAN (Internet) access account:

MariaDB [(none)]> CREATE USER 'secweewx'@'%' IDENTIFIED BY 'password';
MariaDB [(none)]> GRANT ALL ON weewx.* TO 'secweewx'@'%' REQUIRE X509;
MariaDB [(none)]> FLUSH PRIVILEGES;    // ensures the accesses are updated
MariaDB [(none)]> exit;
```

3. To change a password (if needed):

```
> SET PASSWORD FOR 'weewx'@'10.0.0.%' = PASSWORD('newpass');    or
> ALTER USER 'weewx'@'localhost' IDENTIFIED BY 'NEWPASSWORD';
```

NOTE: There is a difference between `REQUIRE X509` vs. `REQUIRE SSL`:

- ❖ **SSL** – a user can only connect over secured transport (SSL), but still can still authenticate with *username* and *password*
- ❖ **X509** – a user can only connect over secured transport (SSL) *and* the client requires a certificate for authentication.

Define PKI Certificates and Keys on the Pi

Defining certificates and keys for secure access to your Pi is required. One can purchase these from a CA, but for our simple application, we will use self-generated and self-signed certificates, since you control both ends of the remote connection. The one proviso is that we use mutual (2-way) authentication to ensure that both the client (the MyWX Pro app) and the server (the WeeWX/Maria DB) authenticate.

The most convenient way to setup the required PKI information on your Raspberry Pi is to install a CA Manager. The following assumes that `TinyCA2` is used for this purpose. It simply provides a nice user interface to `OpenSSL`, which typically comes with every Raspberry Pi setup.

Install `TinyCA2` on your Pi:

```
$ sudo apt-get update -y
$ sudo apt-get install -y tinyca
```

Now run `TinyCA2` on your Pi:

```
$ tinyca2
```

The first step is to define your **ROOT CA**, from which all other Sub CAs and keys derive. When `TinyCA2` is first run, it opens a window where you define the **ROOT CA**, as illustrated in Figure 1 below.

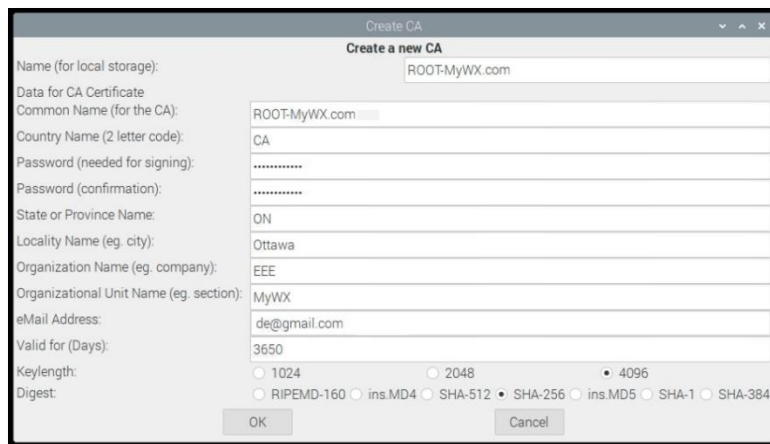


Figure 1 – Create ROOT CA

In this example, note that:

- The *Name* and *Data for CA Certificate Common Name (for the CA)* should be the same;
- The Password should be random, long, and difficult to guess. You need this password when creating new Sub CAs.

After clicking `OK`, the CA Configuration window will appear, as shown in Figure 2. This screen defines how the CA will operate. At this point, we make some important decisions.

Select whether this CA is `critical` or not `critical`, it is recommended to select `critical`.

Change the Netscape Certificate Type to “SSL CA, S/MIME CA, Object Signing CA.”

The two revocation URL fields need to be set. This is the address of a web server where the certificates revocation list is accessible to client PCs. To determine if a certificate is valid, this list is consulted to check if a certificate has been revoked. This cannot change after the creation of the CA so you need to have a location setup now. It is recommended that this be hosted on a server separate to the CA itself.

The two policy URL fields relate to a webpage where people can go to read about the policy of the issuing CA, and any certificate use policy you may have. You can make the actual document later, but you must set the web address now.

See the example in Figure 2.

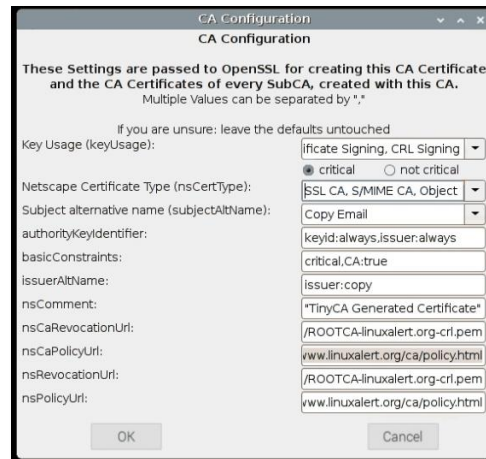


Figure 2 – CA Configuration

Click OK to accept the settings. After creating the CA, which may take some time, the main TinyCA window, showing your root CA, opens as shown in Figure 3.

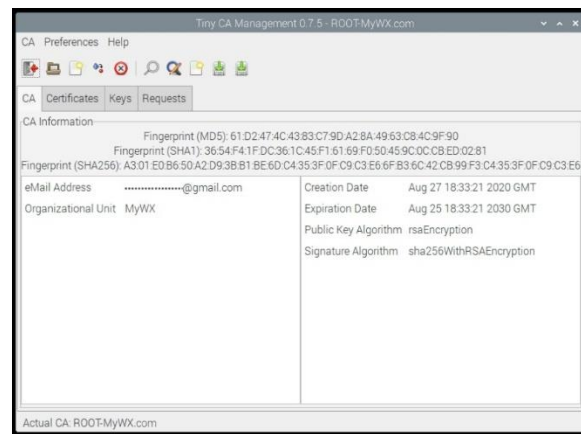
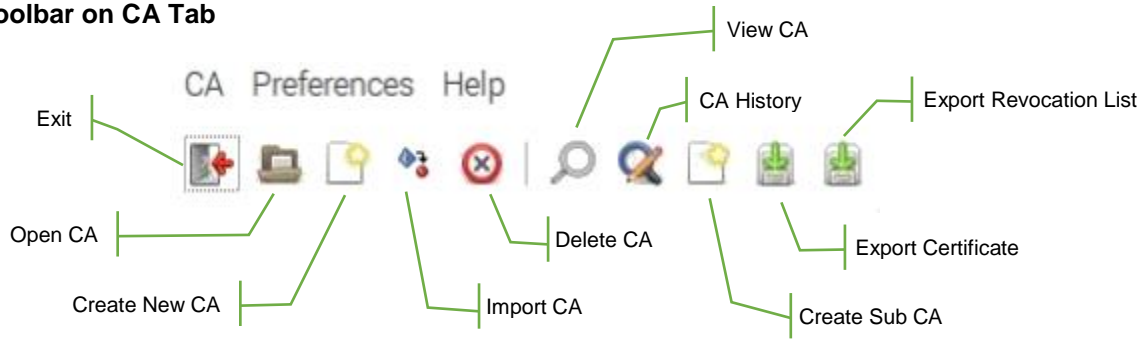


Figure 3 – CA Management Showing Root Certificate

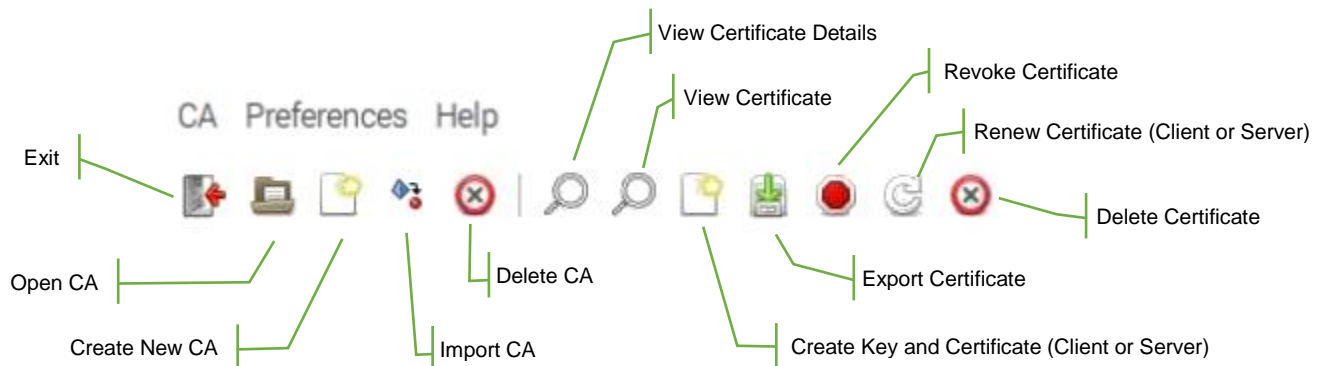
Now that we have the ROOT CA defined, we can create the Subordinate CA, which is what we use within the Pi / Maria DB, and the MyWX Pro app on your Android device. You may use this root CA, however it is highly recommended to define a Sub CA (more flexible, among other reasons).

One annoying thing you will notice with TinyCA2 is that none of the buttons on the toolbar has captions or tooltips. Very annoying, however it is very safe to click each one to see what it does since you can cancel every screen to get back. A summary of these Icons and what they do for each tab:

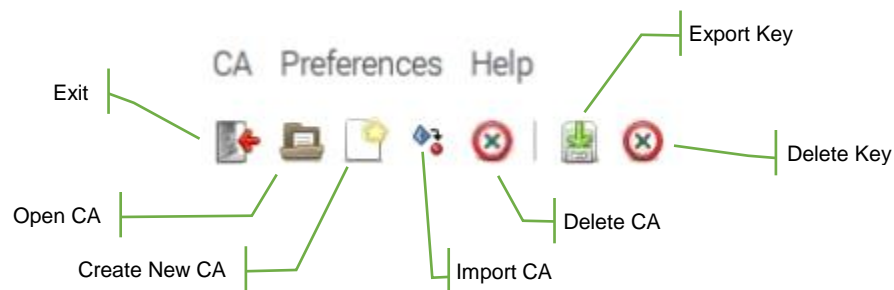
TinyCA – Toolbar on CA Tab



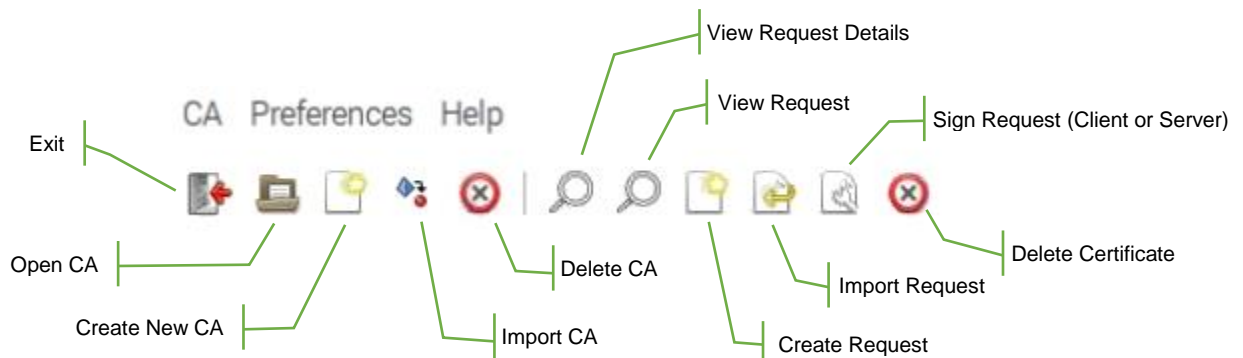
TinyCA – Toolbar on Certificates Tab



TinyCA – Toolbar on Keys Tab



TinyCA – Toolbar on Requests Tab



To create the Sub CA:

1. Make sure you have TinyCA2 open with your Root CA displayed (see Figure 3). On the action icons, click the “Create a new Sub CA” button, which is the 3rd button from the right.
2. Fill in the details for the Sub CA (see Figure 4). The CA Password field is the password of the root CA (Or in the case of multiple Sub CA levels, the password of the parent CA). The second password asked for is a password for this Sub CA, which needs to be different from the root-CA password. This password is used for signing client certificates.
3. Click OK when satisfied. The Sub CA is created and you are returned to the main application window with the Sub CA opened (see Figure 5).

Here is an example definition:

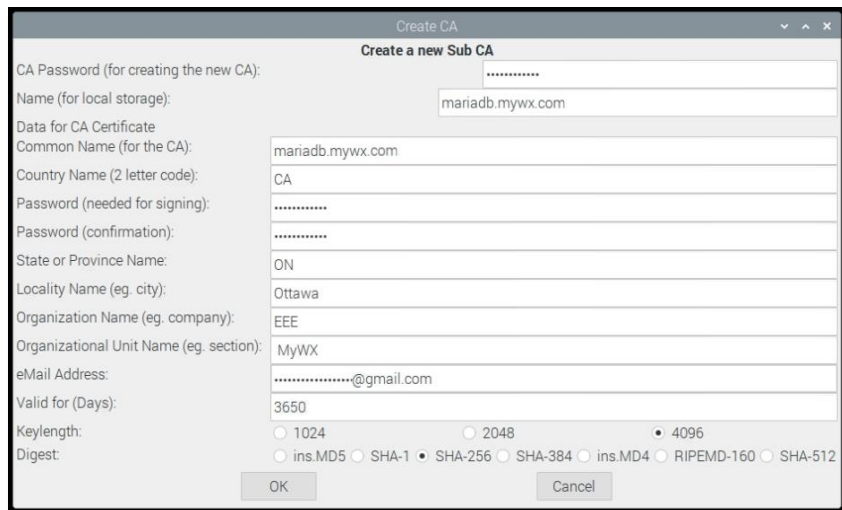


Figure 4 – Create a Sub CA

Important: Make sure that you keep track of all these passwords; they are needed in order to export and make use of these Certificates and Keys!

An example main application window with the Sub CA opened:

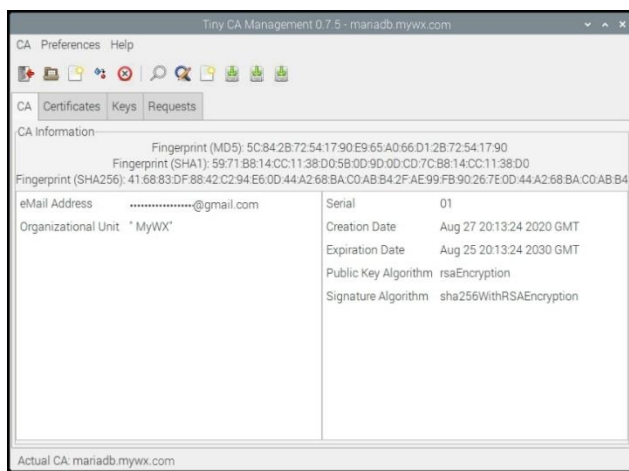


Figure 5 – Sub CA Window

Now, we use the Sub CA definition to create certificates and keys and then export them for use with the Maria DB and the MyWX Pro app. For this new CA hierarchy to be of any use, the required trust by the devices that use its services is defined.

Create Server Certificate and Key

Now its time to create the certificate for the Maria database *server*. Ensure that you have the Sub CA (e.g., mariadb.mywx.ca) open, as shown in Figure 5. Select the Requests tab. Right click and select “New Request”, or click the New Request icon. You get a new window called Create Request as shown in Figure 6. The common name must be the same name as the one used to access your server, in my case myweewx.ca. Once again, you have to enter a password and once again, it must *not* be the same as the previous passwords. Fill in the form, similar to the example below, and press OK.

Figure 6 – Create Sub CA Server Certificate Request

Common Name	eMail Address	Organizational Unit	Organization	Location	State	Country
myweewx.com	de@gmail.com	* MyWX*	EEE	Ottawa	ON	CA

Figure 7 – Sub CA Request Tab

The new certificate Request is listed in the Requests tab, as shown in Figure 7. Right click on the Request and select “Sign Request”. You are prompted to define if it is a *server* or *client* request. In this case, select *server*. You get a new window asking for the CA password (i.e., the Sub CA password). It is now that you, as a CA,

approve the request and confirm that the certificate holder is the correct owner of the common name in the certificate and the rest of the information in the certificate is correct. At the same time, you decide for how long time the certificate is valid. Enter your Sub CA's password and press OK. You have now signed your certificate, which creates the required server Certificate and Key.

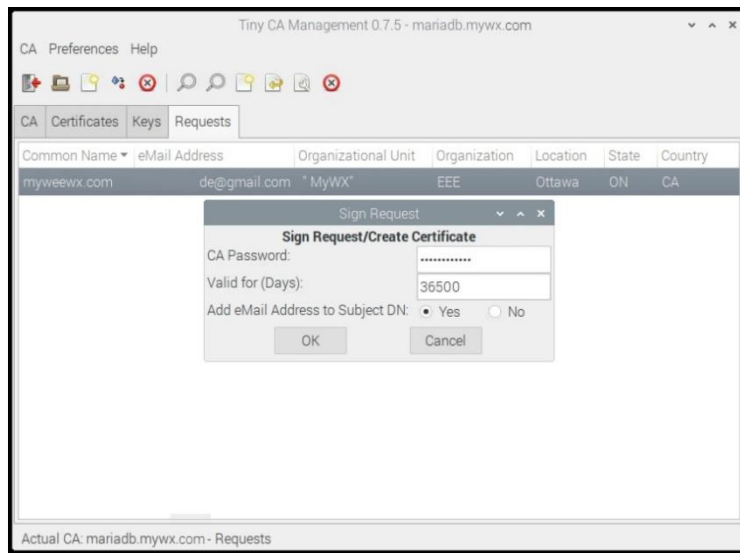


Figure 8 – Sub CA, Sign Server Request

Now we will:

- Export the Certificate and Key for the Maria DB server (two **.PEM** files)
- Export the Certificate and Key for use with the MyWX Pro Android app (a single **.P12** file)
- Export the Sub CA certificate chain (a **.PEM** file)

Return to TinyCA2 and verify that you have your Sub CA opened. Select the **Certificates** tab, right click on the server's certificate and select **Export Certificate**. You may change the file name, then click **Save**. I use the filename **server-cert.pem**. The Maria DB uses this **PEM** file.

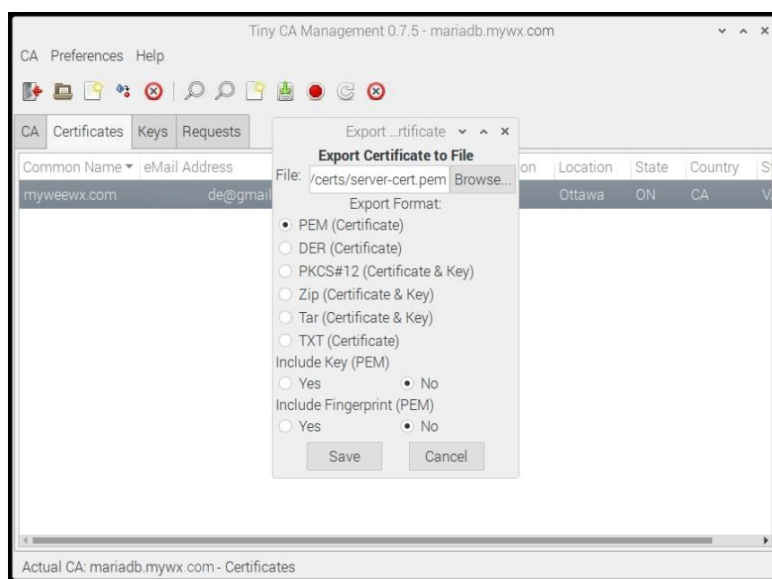


Figure 9 – Export Server Certificate (PEM)

Now we will export the **server** Certificate and Key for use with the MyWX Pro Android app. Select the Certificates tab, right click on the cert and select Export Certificate. In this case, the **file name must be** server-cert.p12. Note that a **P12** format file includes both the Certificate information *and* the Key. The Android app uses this **P12** file. See Figure 10.

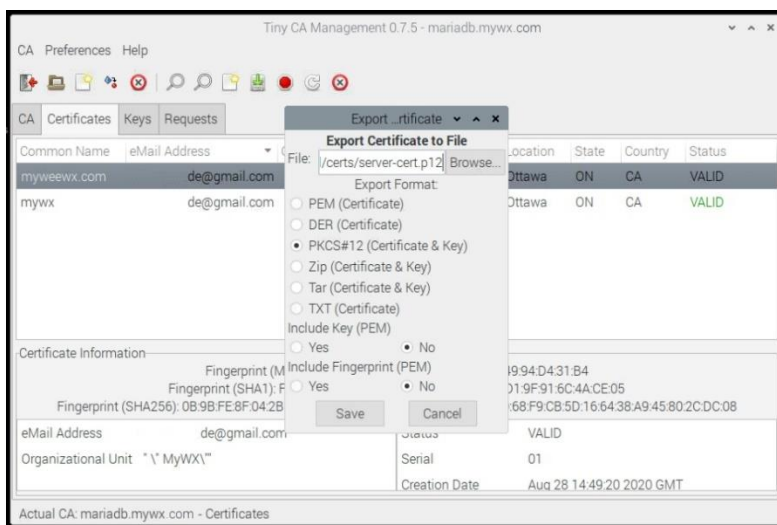


Figure 10 – Export Cert and Key for MyWX Pro (P12)

Now do the same with the Key. Select the Keys tab, right click on the key and select Export Key. You may change the file name. I use the name server-cert.pem as filename. Also, set “Without Passphrase” to Yes. This lets you start the web server without enter the certificates password. This lets you start the web server without enter the certificates password. Then click Save. TinyCA2 will ask you for the password of the certificate for the web server certificate / key. The Maria DB uses this **PEM** file.

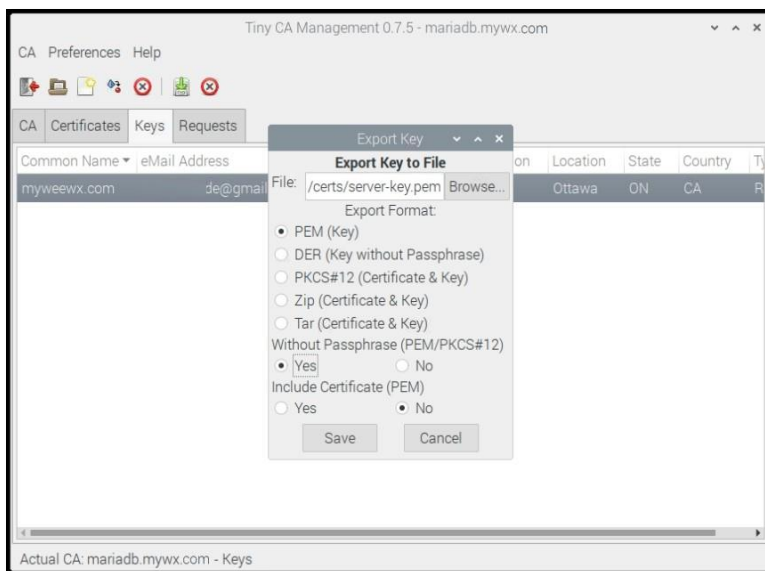


Figure 11 – Export Server Key (PEM)

We now export the certificate chain for use with the Maria DB. Go to the CA tab and select the right-most icon in the tool bar and save the file. Select a location and name (e.g., /etc/mysql/certs/ca-chain.pem), and click Save. The Maria DB uses this **PEM** file.

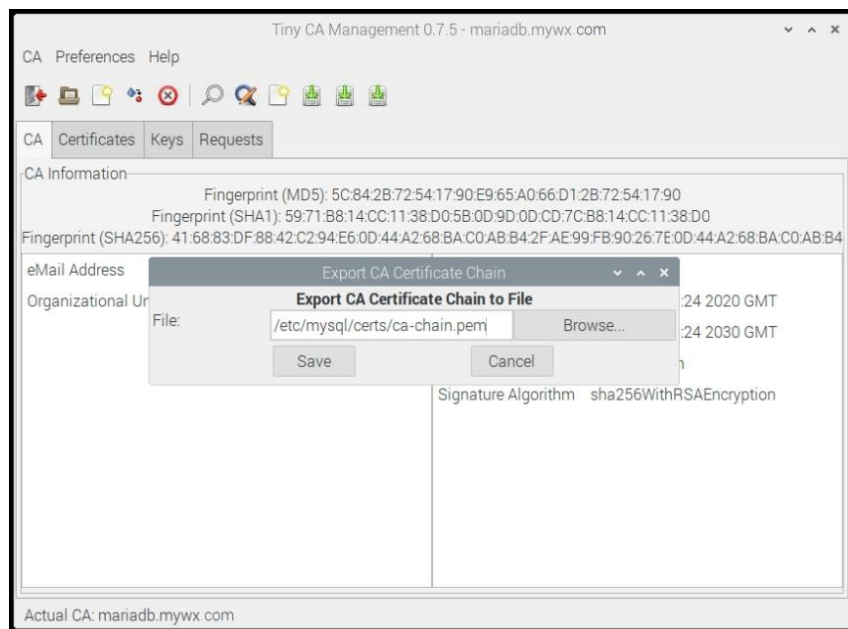


Figure 12 – Export Sub CA Certificate Chain (PEM)

Create Client Certificate and Key

Now we create the certificate for the Maria database *clients*. Ensure that you have the Sub CA (e.g., mariadb.mywx.ca) open, as shown in Figure 5. Select the **Requests** tab. Right click and select “New Request”, or click the New Request icon. You get a new window called **Create Request**. The common name should be the same name as the one used to access your server, in my case MyWX. Once again, you have to enter a password and once again, it must *not* be the same as the previous passwords. Fill in the form, similar to the example below, and press OK.

Figure 13 – Create Client Certificate Request

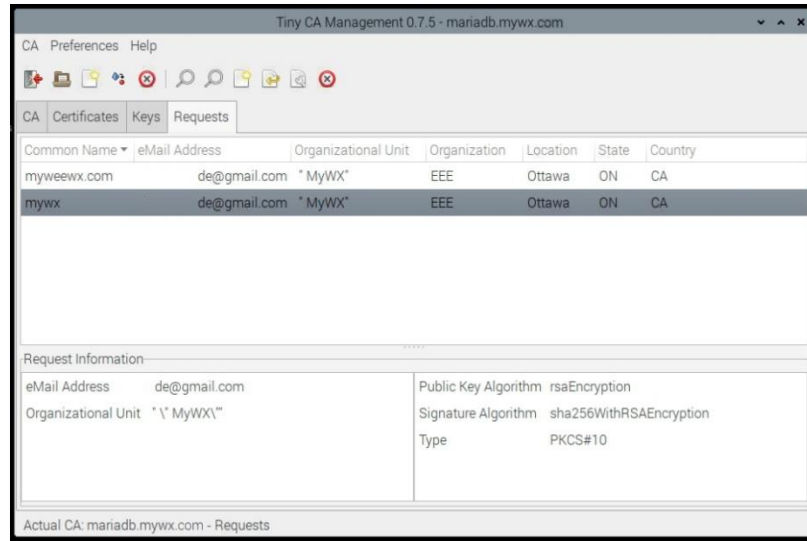


Figure 14 –Client (and Server) Certificate Requests

The new certificate request is listed in the `Requests` tab, as shown in Figure 14. Right click on the Request and select “Sign Request”. You are prompted to define if it is a *server* or *client* request. In this case, select *client*. You get a new window asking for the CA password (i.e., the Sub CA password). It is now that you, as a CA, approve the request and confirm that the certificate holder is the correct owner of the common name in the certificate and the rest of the information in the certificate is correct. At the same time, you decide for how long the certificate is valid. Enter your Sub CA’s password and press OK. You have now signed your certificate and created the required client certificate and key.

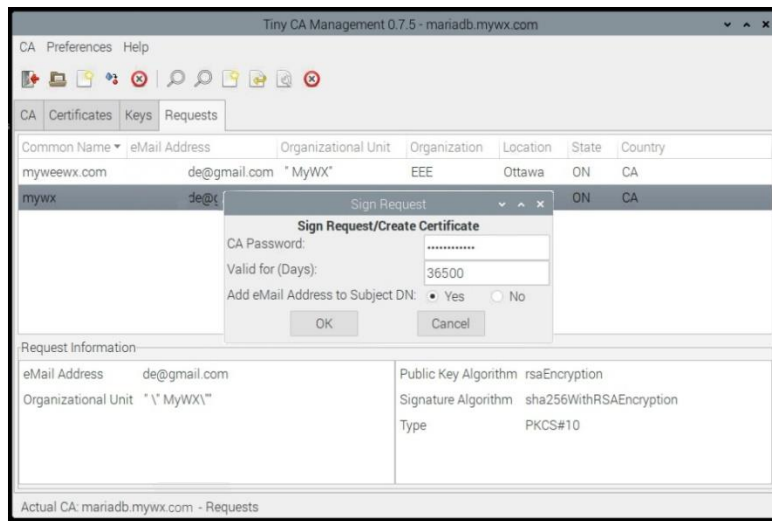


Figure 15 – Sign Client Certificate Request

Now we will:

- Export the Certificate and Key for the Maria DB clients (two **.PEM** files)
- Export the Certificate and Key for use with the MyWX Pro Android app (a single **.P12** file)

Return to TinyCA2 and verify that you have your Sub CA opened. Select the **Certificates** tab, right click on the client's certificate and select **Export Certificate**. You may change the file name, then click **Save**, as shown in Figure 16. I use the filename `client-cert.pem`. The Maria DB uses this **PEM** file.

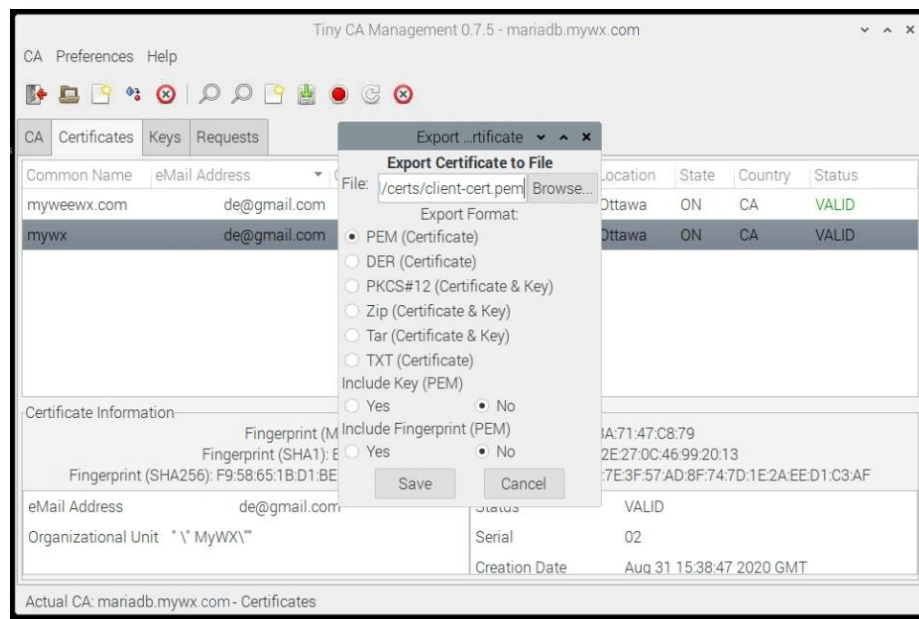


Figure 16 – Export Client Certificate (PEM)

Now we will export the *client* Certificate and Key for use with the MyWX Pro Android app. Select the **Key** tab, right click on the key and select **Export Key**. In this case, the *file name must be* `client-cert.p12`. Note that a **P12** format file includes both the Certificate information *and* the Key; see Figure 17.

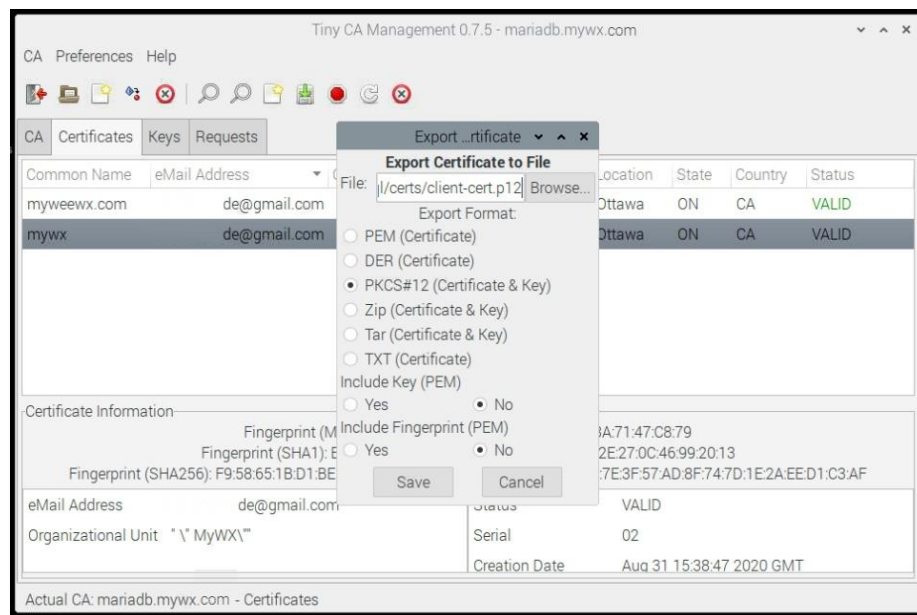


Figure 17 – Export Client Certificate and Key (P12)

Now do the same with the Key. Select the **Keys** tab, right click on the key and select **Export Key**. You may change the file name. I use the name `client-key.pem` as the filename. Also, set “Without Passphrase” to

Yes. This lets you start the web server without enter the certificate password. Then click Save. TinyCA2 will ask you for the password of the certificate for the web client certificate / key. The Maria DB uses this **PEM** file, see Figure 17Figure 18.

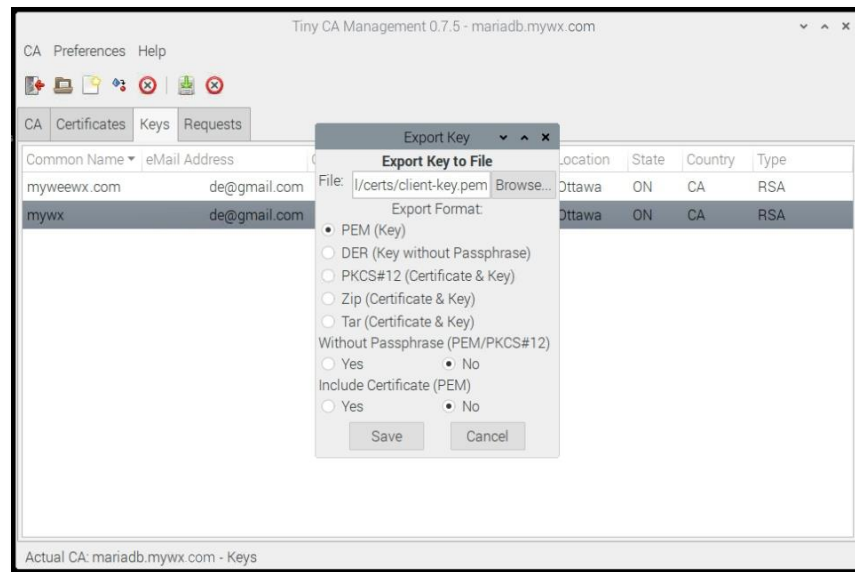


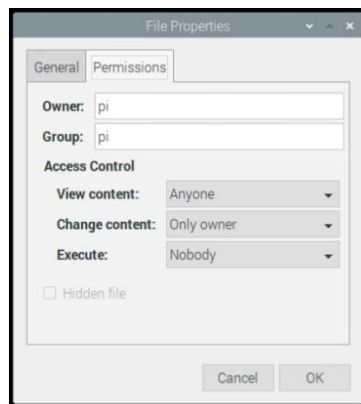
Figure 18 – Export Client Key (PEM)

Point the Maria DB to the Certificates and Keys

The required files were saved to the certs folder on your Raspberry Pi, as described.

The other file created, the `server-cert.p12` file, we will copy to your Android device.

Note: You **must** ensure that the permissions on these generated files are properly defined on your Raspberry Pi; see below.



Server Definitions

The three files used by the Maria DB server, to provide TLS security for the remote access to the WeeWX database, are:

`/etc/mysql/certs/ca-chain.pem` – defines the base trust

`/etc/mysql/certs/server-cert.pem` – defines the certificate for the server (public key)
`/etc/mysql/certs/server-key.pem` – defines the key for the server (private key)

These three server certificate and key files, used by the Maria DB server, are referenced in a configuration file, typically:

`/etc/mysql/mariadb.conf.d/50-server.cnf`

Modify this section of file as shown (uncomment lines and adjust filenames):

```
# * Security Features
#
# Read the manual, too, if you want chroot!
#chroot = /var/lib/mysql/
#
# Use the GUI tool "tinyca" for creating TLS certs and keys.
#
ssl-ca = /etc/mysql/ca-chain.pem
ssl-cert = /etc/mysql/server-cert.pem
ssl-key = /etc/mysql/server-key.pem
#
# Accept only connections using secure TLS protocols
# ...when MariaDB is compiled with OpenSSL:
#
#ssl-cipher = TLSv1.2
#
# ...when MariaDB is compiled with YaSSL (default in Debian):
#
ssl = on
```

Note that the last line requires TLS on all remote accesses!

Client Definitions

The two files, used by the Maria DB clients, are referenced in a configuration file, typically:

`/etc/mysql/mariadb.conf.d/50-client.cnf`

Modify this section of file as shown (uncomment lines and adjust filenames):

```
# Example of client certificate usage
ssl-cert = /etc/mysql/certs/client-cert.pem
ssl-key = /etc/mysql/certs/client-key.pem
#
# Allow only TLS encrypted connections
#ssl-verify-server-cert=on
```

Secure the Maria DB Installation

One final step is to define a secure installation for the Maria DB. Execute the following script and answer the prompts. It enables you to improve the security of your Maria DB installation in the following ways:

- You can set a password for root accounts;
- You can remove root accounts that are accessible from outside the local host;
- You can remove anonymous-user accounts;
- You can remove the test database, which by default can be accessed by anonymous users.

Important: run this script in order to avoid issues with your ISP!

```
$ sudo mysql_secure_installation
```

Now you can enable security by re-starting MySQL:

```
$ sudo service mysql restart
```

Now check the setup:

```
$ sudo mysql -u root -p
MariaDB [(none)]> SHOW VARIABLES LIKE '%ssl%';
```

Variable_name	Value
have_openssl	NO
have_ssl	YES
ssl_ca	/etc/mysql/certs/ca-chain.pem
ssl_capath	
ssl_cert	/etc/mysql/certs/server-cert.pem
ssl_cipher	
ssl_crl	
ssl_crlpath	
ssl_key	/etc/mysql/certs/server-key.pem
version_ssl_library	YaSSL 2.4.4

Or perhaps this result:

Variable_name	Value
have_openssl	Yes
have_ssl	YES
ssl_ca	/etc/mysql/certs/ca-chain.pem
ssl_capath	
ssl_cert	/etc/mysql/certs/server-cert.pem
ssl_cipher	TLSv1.2
ssl_crl	
ssl_crlpath	
ssl_key	/etc/mysql/certs/server-key.pem
version_ssl_library	OpenSSL 1.1.1n 15 Mar 2022

Note: If the Maridb was compiled with OpenSSL then you need to update the 50-server.cnf file (i.e., uncomment the line #ssl-cipher = TLSv1.2)

As you can see, have_ssl says YES, therefore PKI security is now enabled.

Installing CA Certificate on your Android Phone

The MyWX Pro app and Maria DB (WeeWX) app are setup to authenticate each other, ensuring that a man-in-the-middle attack is thwarted. To do this you must copy the CA certificates (i.e., the client-cert.p12 file and the server-cert.p12 file) to your phone (or other Android devices) on which the MyWX Pro app is installed.

- Copy the `client-cert.p12` file and the `server-cert.p12` to your Android device. They **must** be copied to the `Android\data/com.unified.mywx.pro/files` folder and be named as shown.

This will allow your device to authenticate to the WeeWX device and Maria DB server.

Define Port Forwarding to WeeWX

A final step to make your WeeWX PWS accessible from the Internet is forwarding Port 3306 (i.e., the default port that Maria DB uses) to your Pi. Two things are required:

1. A fixed IP address for your Raspberry Pi;
2. A port forwarding definition in your Internet (ISP) router / modem. It specifies that port 3306 be forwarded to the Pi's fixed IP address.

How to define a port forward is very hardware specific. Consult the documentation for your ISP's router / modem.

List of Figures

FIGURE 1 – CREATE ROOT CA	4
FIGURE 2 – CA CONFIGURATION	5
FIGURE 3 – CA MANAGEMENT SHOWING ROOT CERTIFICATE.....	5
FIGURE 4 – CREATE A SUB CA.....	7
FIGURE 5 – SUB CA WINDOW	7
FIGURE 6 – CREATE SUB CA SERVER CERTIFICATE REQUEST	8
FIGURE 7 – SUB CA REQUEST TAB	8
FIGURE 8 – SUB CA, SIGN SERVER REQUEST	9
FIGURE 9 – EXPORT SERVER CERTIFICATE (PEM).....	9
FIGURE 10 – EXPORT CERT AND KEY FOR MYWX PRO (P12).....	10
FIGURE 11 – EXPORT SERVER KEY (PEM)	10
FIGURE 12 – EXPORT SUB CA CERTIFICATE CHAIN (PEM).....	11
FIGURE 13 – CREATE CLIENT CERTIFICATE REQUEST	11
FIGURE 14 –CLIENT (AND SERVER) CERTIFICATE REQUESTS	12
FIGURE 15 – SIGN CLIENT CERTIFICATE REQUEST	12
FIGURE 16 – EXPORT CLIENT CERTIFICATE (PEM)	13
FIGURE 17 – EXPORT CLIENT CERTIFICATE AND KEY (P12)	13
FIGURE 18 – EXPORT CLIENT KEY (PEM)	14

Reference Documentation

Follow these links:

<https://github.com/David-Enst/WeeWX-BCRobotics/tree/version3>
<http://www.steves-internet-guide.com/ssl-certificates-explained/>
<https://www.cyberciti.biz/faq/how-to-setup-mariadb-ssl-and-secure-connections-from-clients/>
<https://stackoverflow.com/questions/4461360/how-to-install-trusted-ca-certificate-on-android-device>
<https://www.lastbreach.com/blog/importing-private-ca-certificates-in-android>
<https://mariadb.com/kb/en/using-tls-ssl-with-mariadb-java-connector/>
<https://mariadb.com/kb/en/about-mariadb-connector-j/>
<https://domoticproject.com/accessing-raspberry-ddns/> (Accessing from the Internet)