Setup the software for the intercom server

Requirements:

The server can run on any hardware but in this guide we will use a Raspberry Pi 3.

Download the latest Raspbian Stretch Lite image from
https://www.raspberrypi.org/downloads/raspbian/

This guide assumes that you have a basic understanding of how linux computers work and that you have access to a Windows computer (both Mac and Linux can be used but you have to find the tools needed). The setup is tested with Raspbian Stretch Lite image built on the 2018-06-27. When copying code from the blue fields remember to check that there are no whitespace characters (invisible spaces) at the end of the lines because that can break the code.

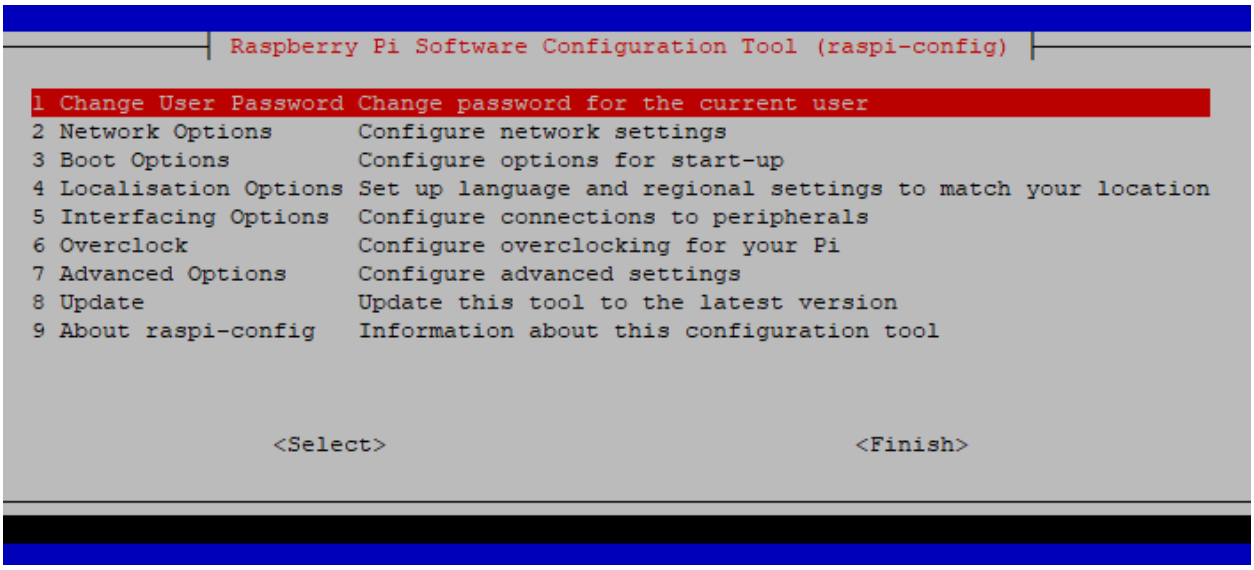
Do NOT use the placeholder passwords or IP numbers in the guide, use safe passwords that you remember and IP numbers you are allowed to use!

Write the image to a SD card

Etcher https://etcher.io/ is recommended to write the image to the sd card. Insert the card into the Raspberry Pi, connect a keyboard and monitor and power up, after a few seconds the Raspberry will resize the partition to fill the card used and reboot. After the reboot you will see a login screen prompting for a username, use pi as username and raspberry as password. The first task is to setup the Raspberry Pi as needed. Enter the command below and press enter

```
sudo raspi-config
```

You will now see a menu like this



1. Configure the Raspberry

1 Change User Password (to something you remember)
2 Network Options
    N1 Hostname (icserver)
4 Localisation Options
    I2 Timezone
    I3 Change Keyboard-Layout
5 Interfacing options
    P2 SSH Enable

Reboot on exit

2. Set a static IP address

```
sudo pico /etc/dhcpcd.conf
```

Uncomment the example static IP configuration and use a IP that you are allowed to use

```
interface eth0
static ip_address=192.168.10.125/24
static routers=192.168.10.254
static domain_name_servers=8.8.8.8
```

Save file (CTRL+X) and reboot

```
sudo reboot
```

You should now have a IP address (look for the line My IP address is:)

You can now decide if you prefer to work with the keyboard and screen or continue the setup over SSH (using for example putty)

3. Update the system

```
sudo apt update
sudo apt dist-upgrade
```

4. Install Mumble and Mosquitto

```
sudo apt install mumble-server
sudo apt install mosquitto mosquitto-clients
```

5. Edit Mumble settings

Mumble is the voice chat server we use as the base for the intercom

```
sudo pico /etc/mumble-server.ini
```

```
welcometext="<br /><b>YOUR INTERCOM.<br />RESTRICTED USE ONLY!</b><br />"
serverpassword=R3plac3M3
bandwidth=150000
registerName=Your Intercom

autobanAttempts = 0
autobanTimeframe = 0
autobanTime = 300
```

Save file (CTRL+x)

```
sudo dpkg-reconfigure mumble-server
```

```
Autostart -> Yes
Higher Priority -> Yes
Set SuperUser password -> Sup3R_USER
```

```
sudo service mumble-server restart
```

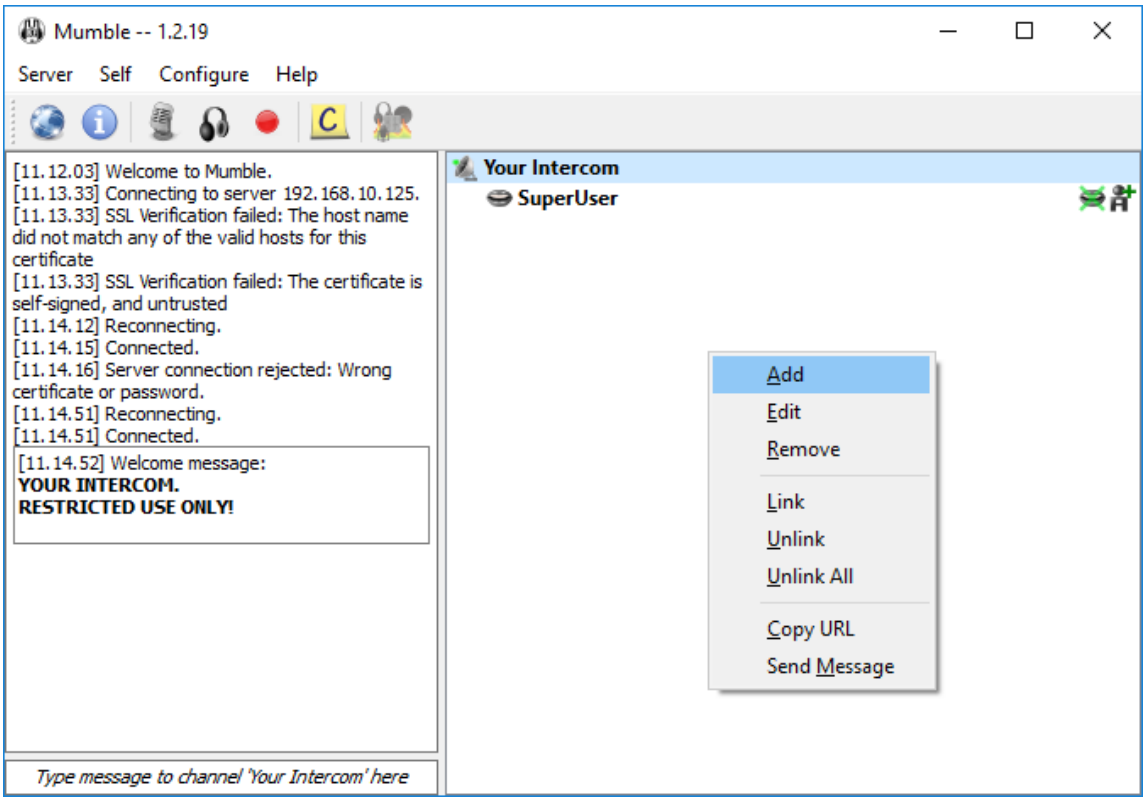
6. Setup the intercom channels

The next steps require a computer with a graphical interface (Windows 10 in this example). Download and install the Mumble client (https://www.mumble.com/mumble-download.php). Start Mumble, you can close all messages (except the create certificate) that might pop up as we will not be using this client to talk to users.

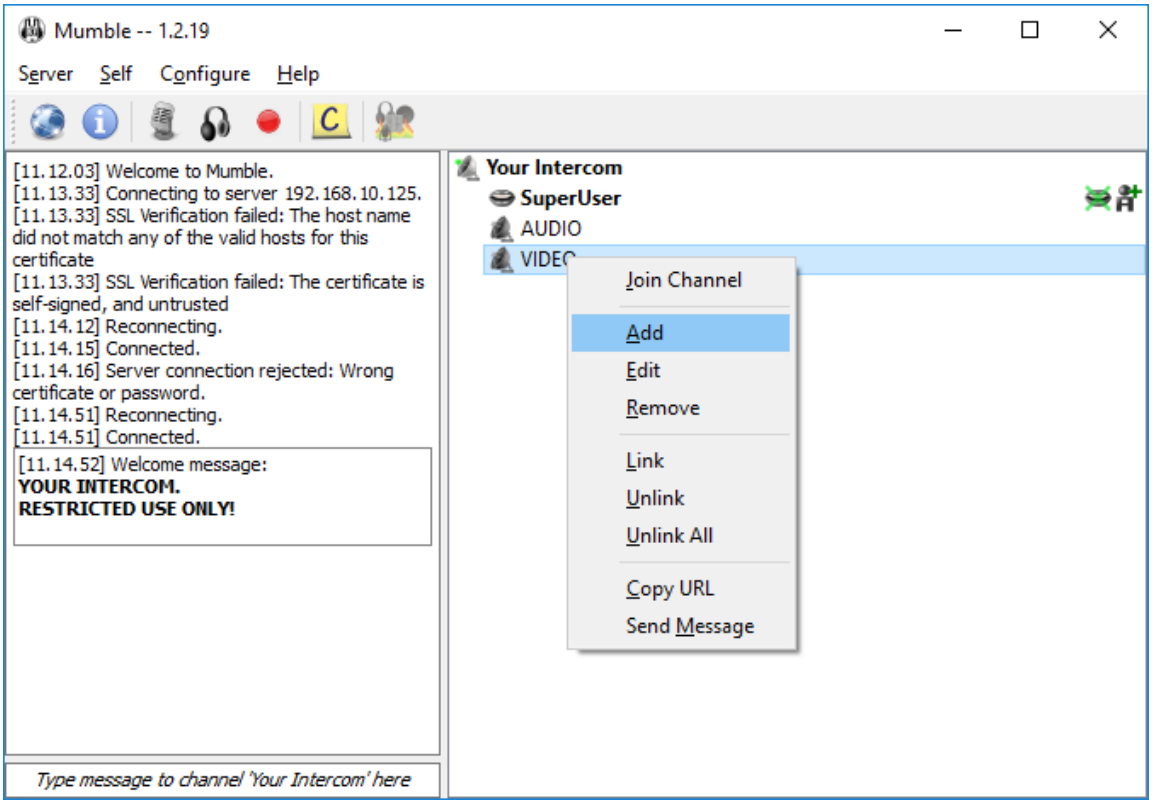
```
Server -> Connect -> Add new
Label: SuperUser .125
Address: 192.168.10.125
Port: 64738
Username: SuperUser
```

Press OK and connect using the profile you just created, accept the self-signed certificate (Yes) and use the SuperUser password created earlier (Sup3R_USER in the example).

Right click to the right on the white background and choose Add to add a new channel, in this example we will make two channels AUDIO and VIDEO



Right click on the Video channel and choose Add to make a sub channel (CAMS) under VIDEO.

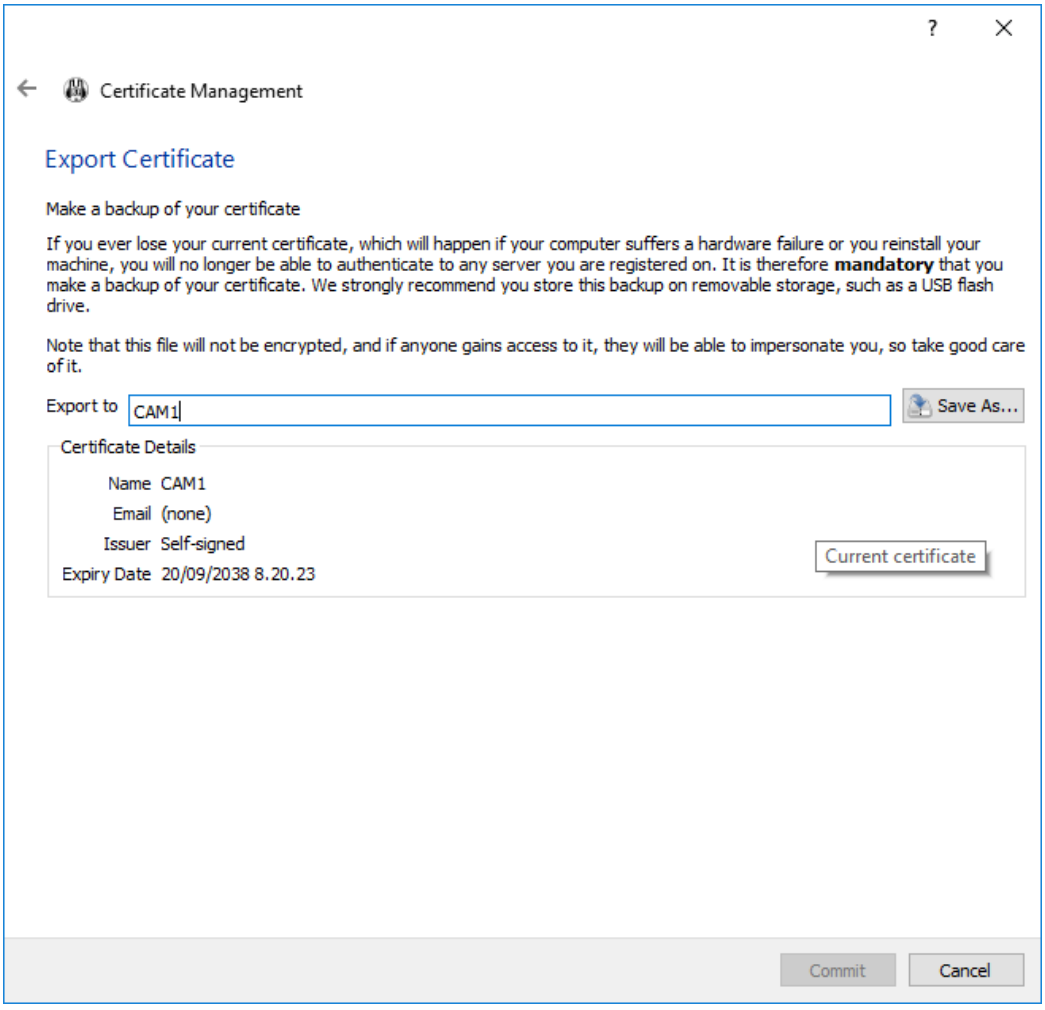


Repeat the last step but this time we add FM (floor manager) under VIDEO, then disconnect from the server from the menu, Server -> Disconnect

7. Generate intercom (Mumble) certificates

Every intercom unit needs a certificate and here is how it is done. In the guide we will only make certificates for CAM1, CAM2 and DIR. In the Mumble client, Configure -> Certificate Wizard -> Create a new certificate

Name: CAM1 (No email needed) -> Next
Replace Certificate -> Next
Export Certificate, Export to CAM1 -> Save As (make a folder called certs and save to it)
Click Commit -> Finish



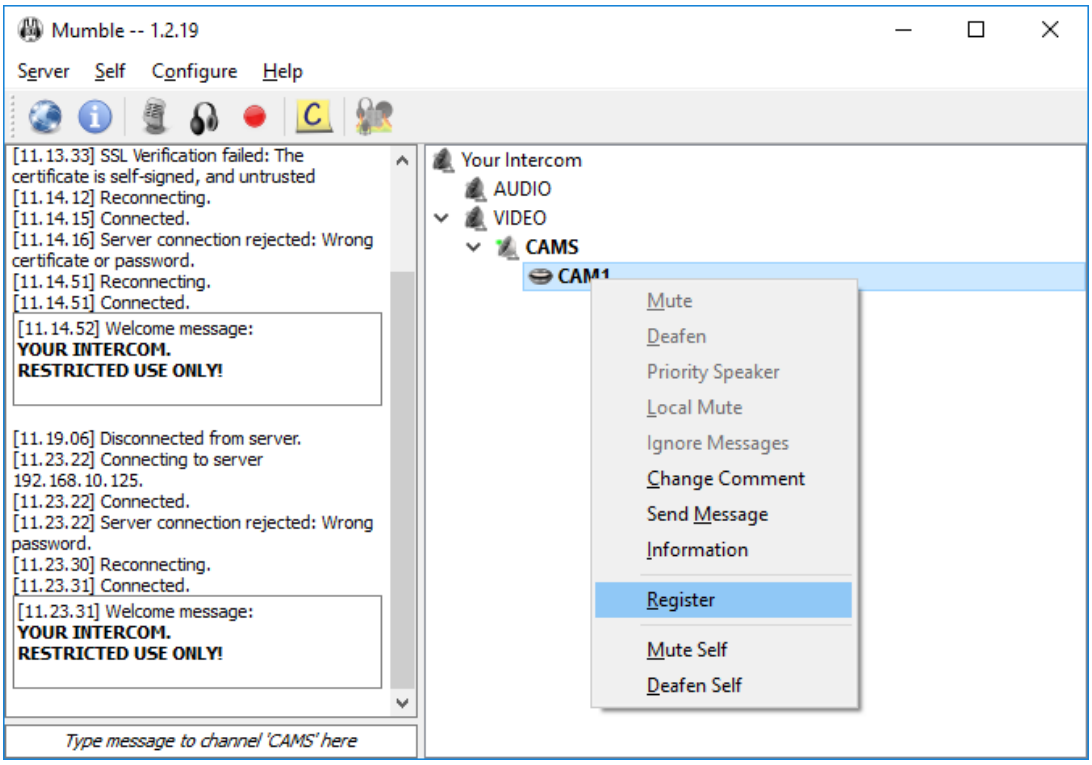
Now we need to connect with the new certificate to the Mumble server.

Server -> Connect -> Add New

Label: User
Address: 192.168.10.125 (use your server IP)

Port: 64738
Username: CAM1 (Use the same username as in the certificate name)

Connect with User profile just created -> Use password: R3plac3M3
Drag CAM1 into CAMS channel (VIDEO -> CAMS)
Right click CAM1 -> Register -> Yes

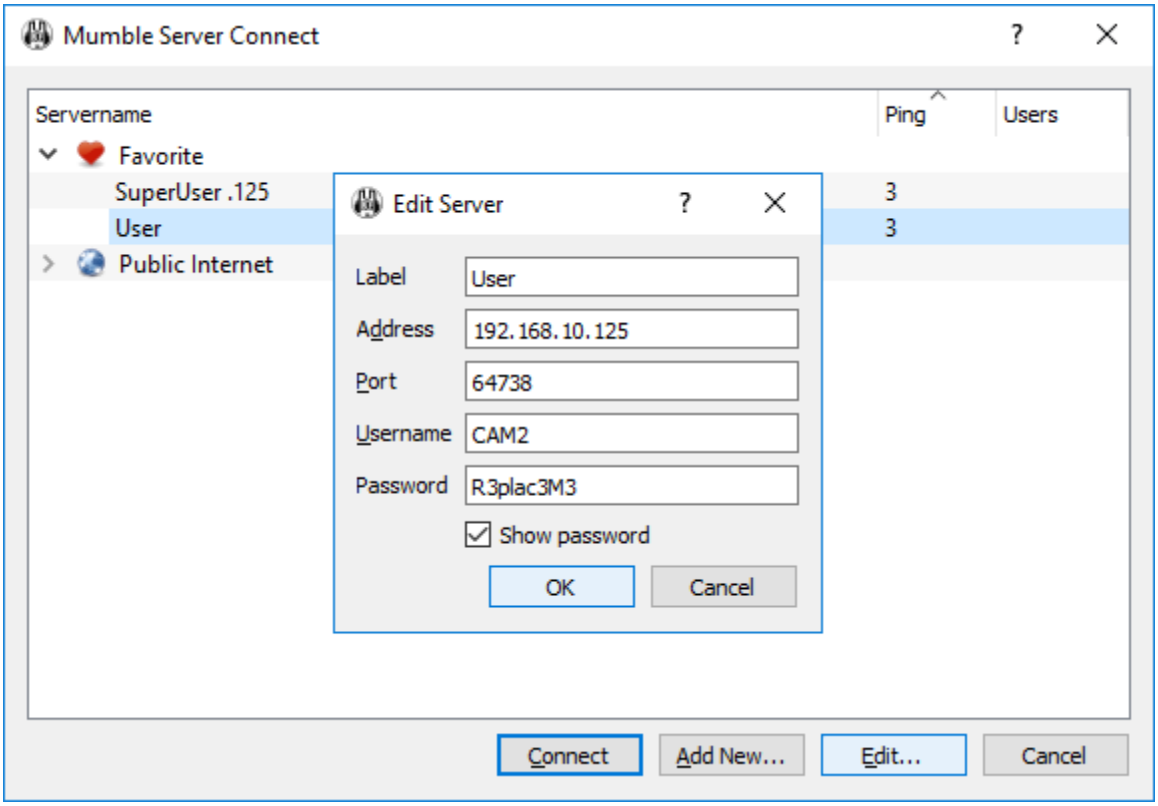


To create a certificate for CAM2, disconnect from the server and repeat the earlier steps

Configure -> Certificate Wizard -> Create a new certificate

Name: CAM2 (No email needed) -> Next
Replace Certificate -> Next
Export Certificate, Export to CAM2 -> Save As
Click Commit -> Finish

Server -> Connect -> Choose User from list (that you made earlier) -> Edit (Change only Username)



Connect with User profile (You are now known as CAM2 because of the changed certificate)
Drag CAM2 to wanted channel (CAM2 -> VIDEO -> CAMS)
Then right click CAM2 -> Register -> Yes

Repeat the steps above for DIR (Director), and drag DIR to the VIDEO channel.

If later needed to move a user from one channel to another, just log in as that user and drag to the wanted channel. This can also be done by the SuperUser but the user must be logged in to be visible on the server.

8. Setup Mosquitto

Now we need to get back to the Raspberry Pi command line to generate SSL certificates for Mosquitto that is used for delivering messages between all the intercom devices. Mosquitto is a MQTT broker that we use for communication between the intercom devices, the MQTT protocol is a machine to machine / IoT connectivity protocol.

```
mkdir ~/certs
cd ~/certs
```

Next we will use a ready-made script to generate a self-signed certificate.

```
wget https://raw.githubusercontent.com/owntracks/tools/master/TLS/generate-CA.sh
```

We need to change IP and hostname in the script

```
pico generate-CA.sh
```

Uncomment and change the following lines, use your server IP and hostname

```
IPLIST="192.168.10.125"
HOSTLIST="icserver"
```

Save the file (CTRL+x), add execution rights to the script and run it.

```
chmod +x generate-CA.sh
./generate-CA.sh
```

Copy the SSL certificates to the right place for Mosquitto (if you used another server name (not icserver), as in this guide the files will have other names)

```
sudo cp ca.crt /etc/mosquitto/certs/
sudo cp icserver.crt /etc/mosquitto/certs/
sudo cp icserver.key /etc/mosquitto/certs/
```

Make the Mosquitto config file

```
sudo pico /etc/mosquitto/conf.d/icserver.conf
```

```
user mosquitto

queue_qos0_messages false
persistent_client_expiration 2h
allow_duplicate_messages false

protocol mqtt
listener 8883
cafile /etc/mosquitto/certs/ca.crt
certfile /etc/mosquitto/certs/icserver.crt
keyfile /etc/mosquitto/certs/icserver.key
tls_version tlsv1

autosave_interval 1800
allow_anonymous true
```

Save the file (CTRL+x) and restart the Mosquitto server

```
sudo service mosquitto restart
```

Check that the Mosquitto server is running

```
ps aux | grep mosquito
```

Should output something like this if the server is running

mosquit+ 1094 0.0 0.4 8232 4620 ? S 15:44 0:00 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

pi 1102 0.0 0.0 4372 560 pts/0 S+ 15:45 0:00 grep --color=auto mosquito

Now try to publish something to the server to check that it works (everything must be on the same line)

```
mosquitto_pub -h 127.0.0.1 --cafile ~/certs/ca.crt -p 8883 -t "test" -m "Hello World" -r --
tls-version tlsv1
```

No output means that it works, now try to subscribe to the message we sent.

```
mosquitto_sub -h 127.0.0.1 --cafile ~/certs/ca.crt -p 8883 -t "test" --tls-version tlsv1
```

The output show now be “Hello World”, (CTRL+c) to stop. Next we need to setup username and password for the mosquito user.

```
sudo pico /etc/mosquitto/conf.d/icserver.conf
```

Change allow anonymous to false and add the password and acl lines

```
allow_anonymous false
password_file /etc/mosquitto/pwfile
acl_file /etc/mosquitto/acl
```

Make the user file

```
sudo pico /etc/mosquitto/acl
```

```
user intercom
topic readwrite media/intercom/#
```

Generate a password for the intercom user (NOTE! Only use –c if the pwfile does NOT exist)

```
sudo mosquitto_passwd -c /etc/mosquitto/pwfile intercom
```

Here we use _Interc0M_ as password but use your own and take note as you will need it later, restart

```
sudo service mosquitto restart
```

Try to publish again with the username/password created (on one line again)

```
mosquitto_pub -h 127.0.0.1 --cafile ~/certs/ca.crt -p 8883 -t "test" -m "Hello 2!" -r  -u
intercom -P _Interc0M_ --tls-version tlsv1
```

If everything works you should not get any output, subscribe to check the result.

```
mosquitto_sub -h 127.0.0.1 --cafile ~/certs/ca.crt -p 8883 -t "test" -u intercom -P _Interc0M_
--tls-version tlsv1
```

You should now see “Hello 2!”, CTRL+c to quit

9. Install Node.js

```
cd ~/
sudo apt-get install curl
curl -sL https://deb.nodesource.com/setup_8.x | sudo -E bash -
sudo apt-get install nodejs
```

Test that node installed correctly

```
nodejs –v
```

Outputs the installed version (v8.12.0 in my case)

10. Install node.js app

```
mkdir ~/nodeapps
mkdir ~/nodeapps/intercom
cd ~/nodeapps/intercom
```

Go to your Windows computer and copy the node app files (node_intercom.zip) with for example the WinSCP software to /home/pi/nodeapps/intercom/

Download WinSCP (https://winscp.net/) on your Windows computer, install and start.

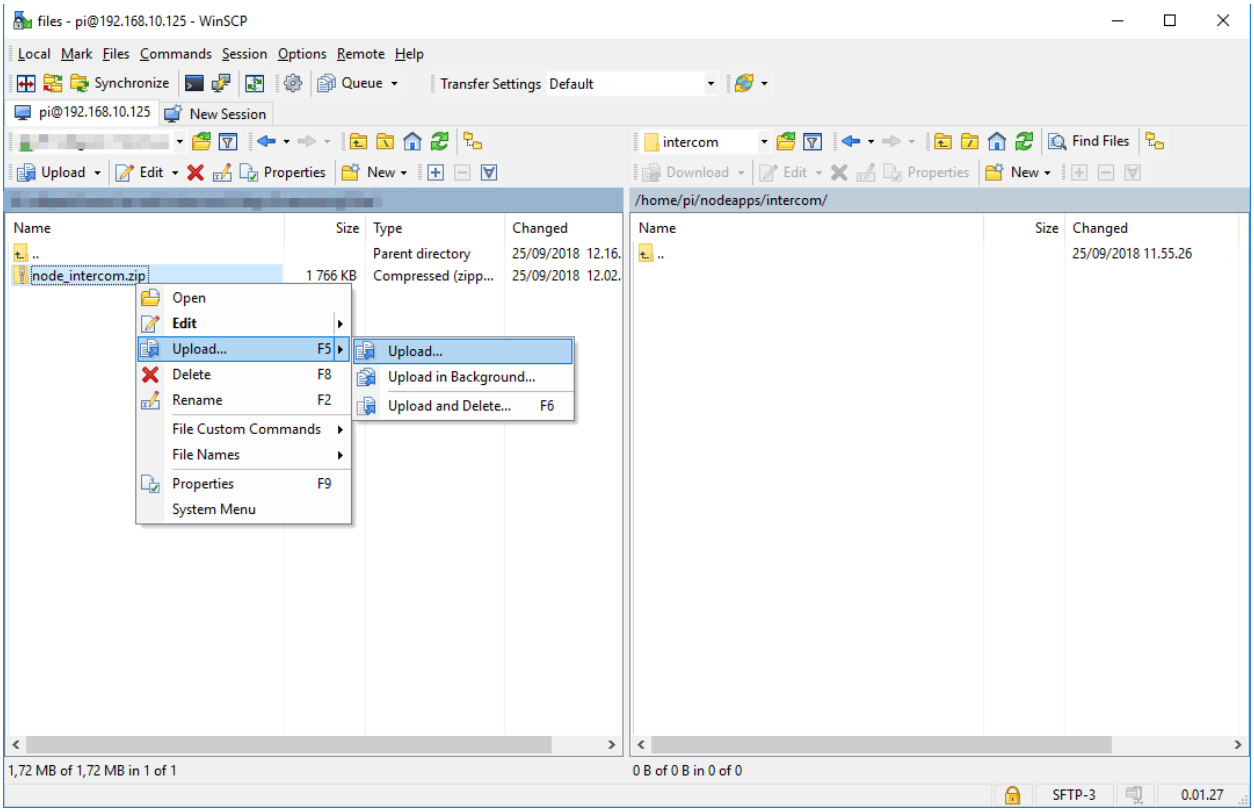
Under session choose SCP host name is your server IP,

User name: pi, Password: (set in step 1)

When you are logged into the server, on the left side you go to the folder where you downloaded the intercom files

On the right side you have your server, double click on nodeapps and then intercom

Select the node_intercom.zip on the left side, right click and select Upload and OK



Go back to the Raspberry Pi command line and unzip the files just uploaded

```
unzip node_intercom.zip
```

Enter the intercom folder and install needed modules (based on the package.json file)

```
cd ~/nodeapps/intercom
npm install
```

Edit the intercom.js file under connectOptions to change the host (your server IP), username and password. This is also the file you need to change if you want more devices and roles in the future.

```
pico ~/nodeapps/intercom/intercom.js
```

CTRL+x to save and try to start the intercom app

```
node intercom.js
```

The output should be “listening on *:3000 and Connected to MQTT server”, CTRL+c to stop. If everything works, we now need to start the intercom app on boot.

```
pico ~/start_node.sh
```

```
#!/bin/bash
/usr/bin/node /home/pi/nodeapps/intercom/intercom.js &
```

CTRL+x to save and then we need to give execute rights to the file

```
chmod +x ~/start_node.sh
```

Add script to crontab to start on boot (if it is the first time you use crontab, choose nano)

```
crontab -e
```

Add after last line

```
@reboot /home/pi/start_node.sh
```

CTRL+x to save and reboot

```
sudo reboot
```

When the server has rebooted, log in and check that the intercom app is running

```
ps aux | grep node
```

The output should be similar to this

pi 971 14.7 3.6 120600 34404 pts/0 Sl 13:55 0:01 /usr/bin/node /home/pi/nodeapps/intercom/intercom.js
pi 983 0.0 0.0 4372 572 pts/0 R+ 13:55 0:00 grep --color=auto node

Congratulations, the server is setup.