

USER GUIDE

HOME AUTOMATION USING CoT

Ghazi Tounsi, Mohamed Karaa
ghazi.tounsi@supcom.tn, mohamed.karaa@supcom.tn

1 Introduction

This user guide serves as a step-by-step guide containing instructions on installing, using, and troubleshooting the hardware and software parts of this project. The document is divided into 5 sections, each covers the steps used to get a service/component to work.

2 Domaine name

In this first section, we detail the steps needed to get a free domaine name to use it as our project domaine **homeautomationcot.me**.

1. *Github Student Pack*: First step was to get a Github Student Pack, that gives us the opportunity to use various premium services and tools for free as long as we still students. There is no much else, you just need to be a student and apply at: Github student pack
2. *Namecheap*: After being approved as a student and getting your pack, you need to get Namecheap benefits. Namecheap is an Affordable registration, hosting, and domain management service and gives us 1 year domain name registration on the .me TLD.
You need to access namecheap.com from github and follow the steps to register you domain name.
3. *Add CAA record*: After getting a domaine name, it is a good thing to setup a CAA record. Figure 6 shows the settings that needs to be set at the Advanced DNS tab.

<input type="checkbox"/>	CAA Record	@	0 issue "letsencrypt.org"	Automatic	
<input type="checkbox"/>	CAA Record	@	0 issuewild "letsencrypt.org"	Automatic	
<input type="checkbox"/>	CAA Record	@	0 iodef "mailto:ghazi.tounsi@supcom.tn"	Automatic	

Figure 1: DNS TXT record

4. *Congratulations*: You now have a .me domaine name that points to Github Pages. You need to remove those records as we will see how to point it to our virtual machines and create subdomaines later at Section 6.

3 TLS Certificate

In this section, a set of commands and steps to generate the TLS certificate using the free certification tool Certbot.

1. *Virtual Machine*: For this step we need a virtual machine running Ubuntu (20.04 LTS in our case) to generate the certificate. The virtual machine can be from any cloud provider, but in our case, we used our benefits from a 300\$ credit on Google Cloud Platform to get our VMs. This VMs can later be used as a middleware or MQTT broker.

2. *Install Certbot*: Certbot is a free, open source software tool for automatically using Let's Encrypt certificates on manually-administrated websites to enable HTTPS. Certbot can easily installed using the following commands:

```
$ sudo snap install --classic certbot
$ sudo ln -s /snap/bin/certbot /usr/bin/certbot
```

3. *Generate wildcard certificate*: As we are going to use subdomains for the middleware API and the MQTT broker and possibly others, we had to generate a wildcard certificate. The certificate can be then generated using:

```
$ certbot certonly --manual -d *.$DOMAIN -d $DOMAIN --agree-tos --
  ↪ manual-public-ip-logging-ok --preferred-challenges dns-01 --
  ↪ server https://acme-v02.api.letsencrypt.org/directory --register
  ↪ -unsafely-without-email --key-type ecdsa --elliptic-curve 384
```

This command can be broken into 3 main parts:

- Using certbot manually to only generate a certificate for the desired domain. Note the "*" before the domain name, this is the key for wildcard certificate.
- setting the preferred challenge to *dns-01* and this is because we don't have access for cPanel in namecheap. Thus, we need to use the dns challenge to later verify with a TXT record.
- Setting the type of the key to *elliptic curve*

This command will output something similar to:

```
-----
  ↪
Please deploy a DNS TXT record under the name
_acme-challenge.example.me with the following value:

qqiR_lsa2AjMfoVR16mH4UDb0xy_E0210K1CNyz1RdI

Before continuing, verify the record is deployed.
-----
  ↪
Press Enter to Continue
```

You need to use that DNS TXT record to verify that you own the domain name. You need to carefully pay attention to the name of the TXT record as namecheap automatically add the *.example.me* to the name. Figure 3 illustrate the process. After doing that, you will be asked to enter an other TXT record



Figure 2: DNS TXT record

the same as the first. You just need to add it into Namecheap Advanced DNS records and wait for about 30 seconds to press enter again.

4. *Congratulations*: You now have a valid certificate to use with all subdomaines. You need to download that resulting files (privkey.pem and cert.pem) and save them somewhere secure. We will need them later at Sections 4 and 6.

4 MQTT Broker

In this section, we list the steps needed to create an MQTT broker using Eclipse Mosquitto.

1. *Get a Virtual Machine:* The first step is to get a virtual machine. That virtual machine can be the one created at for generating the certificate at Section 3. That really depends on your measures, budget and preferences.
2. *Install Mosquitto:* To proceed with the installation, type the following commands, through install. **mosquitto** and **mosquitto-clients** will be installed:

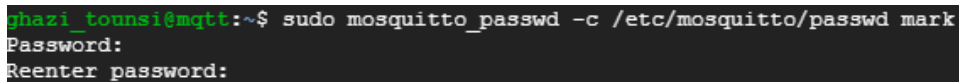
```
$ sudo apt update -y && sudo apt install mosquitto mosquitto-clients -  
↪ y
```

3. *Start Mosquitto:* Check if mosquitto isn't already running, if so we start the service and enable it to automatically start at each system restart.

```
$ sudo systemctl start mosquitto  
$ sudo systemctl enable mosquitto
```

4. *Configuring MQTT Password:* At this moment anyone who knows even just the Server IP address with your Broker could send messages. To deal with this security problem, indicate a set of users, with username and password, enabled to send messages. (Mainly this will be set at the middleware configuration, so only the middleware can send messages). Mosquitto includes a password encryption tool for the users you want to add. Assuming you want to add "middleware" user, the command to execute is the following:

```
$ sudo mosquitto_passwd -c /etc/mosquitto/passwd middleware
```



```
ghazi_tounsi@mqtt:~$ sudo mosquitto_passwd -c /etc/mosquitto/passwd mark  
Password:  
Reenter password:
```

Figure 3: MQTT password set

5. *Configuring MQTT settings:* Now, you have an MQTT broker up and running, but we still need to activate it through websockets and secure it using our certificate. You need to open the default Mosquitto configuration file and add the content illustrated at 3. Please note that you need to enter the path for your certificate and key :

```
$ sudo nano /etc/mosquitto/conf.d/default.conf  
allow_anonymous false  
password_file /etc/mosquitto/passwd  
  
listener 8883  
certfile /etc/ca-certificates/fullchain.pem  
cafile /etc/ca-certificates/fullchain.pem  
keyfile /etc/ca-certificates/privkey.pem  
  
listener 8083  
protocol websockets  
certfile /etc/ca-certificates/fullchain.pem  
cafile /etc/ca-certificates/fullchain.pem  
keyfile /etc/ca-certificates/privkey.pem
```

6. *Restart the broker:* Now you need to restart the broker using the following command:

```
$ sudo systemctl restart mosquitto
```

If everything went okay, you need to get the active status when cheking the mosquitto service status:

```
ghazi_tounsi@mqtt:~$ sudo systemctl status mosquitto
mosquitto.service - Mosquitto MQTT v3.1/v3.1.1 Broker
Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled;
       ↪ vendor preset: enabled)
Active: active (running) since Fri 2022-01-07 21:36:06 UTC; 2min 27s
       ↪ ago
Docs: man:mosquitto.conf(5)
      man:mosquitto(8)
Main PID: 53365 (mosquitto)
Tasks: 3 (limit: 2367)
Memory: 1.2M
CGroup: /system.slice/mosquitto.service
        53365 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf
```

7. *Link the VM to a subdomain:* Now, the MQTT broker is setup and running, the final step is to link the VM to a subdomaine. The process is straightforward and easy, you just need to add the VM public IP address to the DNS records at Namecheap as an A record with the host as **mqtt** for example.

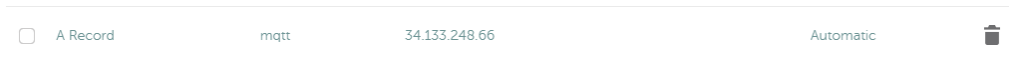


Figure 4: mqtt subdomaine setting

8. *Congratulations:* The broker is now running and you can use the broker via *mqtt.example.me*.

5 IoT sensors

This section illustrates the schema of the IoT sensors. you can follow the wiring and replicate the same ecosystem as ours. The used sensors are the following:

- **Gas sensor MQ-2:** MQ2 is one of the commonly used gas sensors in the MQ sensor series for sensing gas.
- **Motion sensor PIR:** PIR sensors allow to sense motion, almost always used to detect whether a human has moved in or out of the sensors range. They are small and inexpensive.
- **CO sensor MQ135:** The MQ135 is one of the commonly used gas sensors in the MQ sensor series for sensing air quality and CO percentage.
- **Temperature sensor DHT11:** The DHT11 is a basic, ultra low-cost digital temperature and humidity sensor.
- **Photo-resistors:** Enable the measurement of the luminosity in a room.

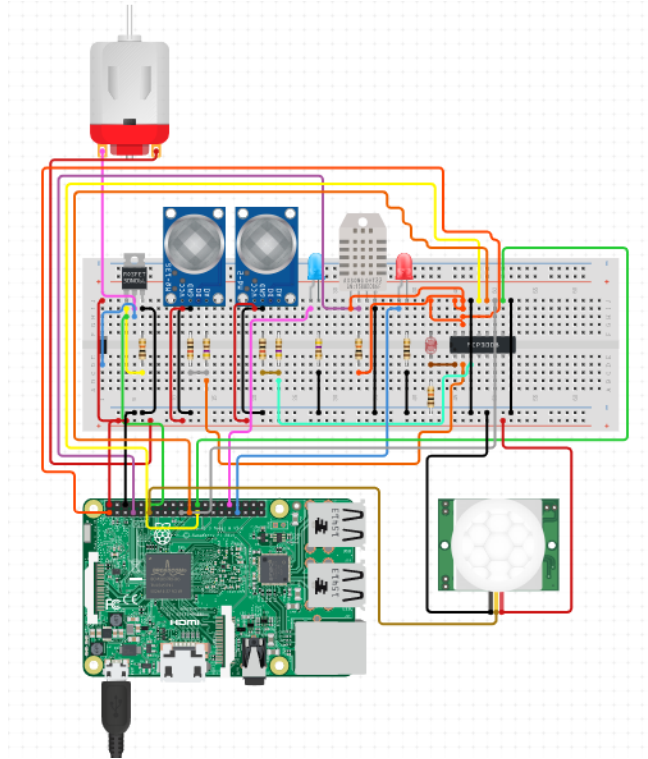


Figure 5: IoT architecture

6 Deployment

The last section in the project is the deployment phase, in which we move our code from the development environment to the production development. A set of steps are followed to ensure CI/CD flow.

1. *Get a VM:* This step consists of getting a virtual machine to run the server on. You can the previously generated VM but it is preferably to create a new separate VM.
2. *MongoDB:* To ensure the security of our database, we need to set the connection settings of MongoDB Atlas, so we can only connect to the database from a predefined IP address. Figure [?] shows the steps to only activate the VM IP address.

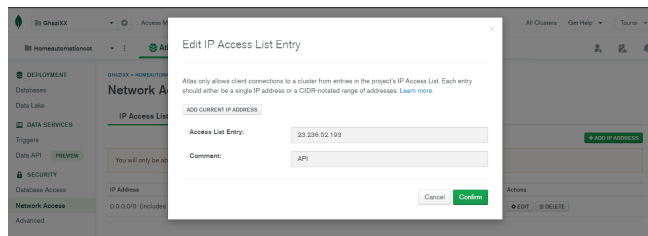


Figure 6: Database IP address restriction

3. *Prepare a .env file:* This file holds all the settings of the production environment like the database link, the MQTT username and password, the path to the certificates etc. you can find a template of the structure of the file in the repo. This file should be kept at the virtual machine an not to be shared.
4. *Install dependancies:* As the server is based on Node.js, You need to make sure that you have installed node and npm in your VM using the following commands:

```
$ sudo curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.39.1/  
  ↪ install.sh | bash  
$ sudo restart  
$ sudo nvm install 16.13.1
```

5. *Create a cron job:* We now prepared our environment, we need to create a Cron job that runs every day at midnight to check the repo for changes, clone it and run the server again. A *deploy.sh* and *cronjob.conf* files are available on the repo, you just need to change the necessary on the *deploy.sh* file, put it on your VM, create the cronjob and leave the rest to be automatically done
6. *Link the VM to a subdomain:* Now, the API is setup and running, the final step is to link the VM to a subdomain. The process is straightforward and easy, the same as for the MQTT broker setup, you just need to add the VM public IP address to the DNS records at Namecheap as an A record with the host as **api** for example.



Figure 7: api subdomain setting

7. *Congratulations:* The api is now running and you can use it via *api.example.me*.