Tunisian Republic
Ministry of Communication Technologies and the Digital Transformation
Ministry of Higher Education and Scientific Research
University of Carthage
Higher School of Communications of Tunis

USER GUIDE

SMART PARKING BASED ON COT

Realised by

AZZOUZ IMEN

TOUMI MOHAMED AMINE

Supervised by

Dr. Kaâniche Mohamed-Bécha

Academic year: 2022 - 2023

1 Introduction

This user guide is a helps you how to install and manage a smart parking project. This guide is designed to provide a step-by-step walkthrough of the entire process, from initial planning and hardware selection to final installation and ongoing maintenance while considering the knowledge and tools you needed to successfully install and manage the smart parking project.

2 Domain name

a domain name is necessary because it provides a user-friendly and memorable way for users to access a website or application, and it helps to establish credibility and professionalism for the application that why we created the domain name **smart-parking.me** by follwing these steps.

1. **Github Student Pack:**
   Getting started with the Github Student Pack is easy. The first step is to ensure that you are a currently enrolled student. Once you have confirmed your student status, you can apply for the pack by visiting the Github Student Pack website.

2. **Namecheap:**
   It offers a wide variety of domain extensions at competitive prices, and their hosting plans include features such as website builders and one-click installs. It also provides free domain names and some services thanks to the github student pack

3 TLS Certificate

We have received a 400$ credit from Google Cloud to create three virtual machines, each with a specific purpose. The first virtual machine will be used as the middleware, which will handle the communication and data transfer between the different systems. The second virtual machine will be used for the Mosquitto broker. The last virtual machine will be used for the dashboard, which will monitor and managing the system. By utilizing this credit, we will be able to set up a secure and efficient infrastructure for our IoT project at a free cost.
To generate a certificate follow these commands :

```
$ sudo snap install --classic certbot
```

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

Since we have 3 different subdomains we will generate different certificates using the following command . You should replace the DMAIN with your DOMAIN name .In our case we have smart-parking.me

```
$ 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 will generate a TXT record that is necessary to your domain's DNS settings. Once you have added the record to the namecheap as in figure 1, the service will check for it in the domain's DNS records and if it finds it, it will confirm that you are the owner of the domain.

Your namecheap dashboard should look like the figure1:

Figure 1: Namecheap dashboard

Once you have generated successfully your certificate.Make sure you keep the privkey.pem and cert.pem in safe directory we need them. You will find these two files in /etc/letsencrypt/live/YOUR_DOMAIN_NAME/

4 MQTT Broker

In order to handle the transmission from Iot devices to middleware and vice versa. First, we need to create a VM to deploy the broker .
Then, type the following commands to install MQTT on your VM:

```
$ sudo apt update -y
```

```
$ sudo apt install mosquitto mosquitto -clients -y
```

```
$ sudo systemctl start mosquitto
```

```
$ sudo systemctl enable mosquitto
```

Now, we have to create a password for our broker in order to disable sending messages to our broker so we tap the following command

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

Then, you have to add the certificated and mqtt ports to the configuration file that we find under the /etc/mosquitto/conf.d/default.conf directory:

```
$ 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
```

At a final stage, we are left only with restarting the mosquito broker using this command :

```
$ sudo systemctl restart mosquitto
```

if everything is working as expected, you will get like this

```
mohamedamine_toumi@broker:~$ sudo systemctl status mosquitto
● mosquitto.service - Mosquitto MQTT Broker
     Loaded: loaded (/lib/systemd/system/mosquitto.service; enabled; vendor preset: enabled)
     Active: active (running) since Tue 2022-12-27 11:13:34 UTC; 3 weeks 3 days ago
       Docs: man:mosquitto.conf(5)
             man:mosquitto(8)
   Main PID: 519 (mosquitto)
      Tasks: 1 (limit: 2353)
     Memory: 3.2M
        CPU: 19min 42.792s
     CGroup: /system.slice/mosquitto.service
             └─519 /usr/sbin/mosquitto -c /etc/mosquitto/mosquitto.conf

Dec 27 11:13:34 broker systemd[1]: Starting Mosquitto MQTT Broker...
Dec 27 11:13:34 broker mosquitto[519]: 1672139614: Loading config file /etc/mosquitto/conf.d/default.conf
Dec 27 11:13:34 broker systemd[1]: Started Mosquitto MQTT Broker.
Jan 09 00:00:39 broker systemd[1]: Reloading Mosquitto MQTT Broker.
Jan 09 00:00:39 broker systemd[1]: Reloaded Mosquitto MQTT Broker.
Jan 20 00:00:00 broker systemd[1]: Reloading Mosquitto MQTT Broker.
Jan 20 00:00:00 broker systemd[1]: Reloaded Mosquitto MQTT Broker.
mohamedamine_toumi@broker:~$
```

Figure 2: Mosquitto broker status

5 Wildfly setup and security

- **Install JDK 17** The first step is about installing JDK17 in order to

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

- **Install Wildfly-preview 26.1.2 Final** Check WildFly Downloads page for latest releases before downloading the file. Here we will download WildFly 26.1.2 Final

```
$ wget https://github.com/wildfly/wildfly/releases/download/${
    ↪ WILDFLY_RELEASE}/wildfly-${WILDFLY_RELEASE}.tar.gz
$ tar xvf wildfly-\${WILDFLY_RELEASE}.tar.gz
$ sudo mv wildfly-\${WILDFLY_RELEASE} /opt/wildfly
```

- **Configure Systemd for WildFly** The first step is about installing JDK17 in order to

```
$ sudo groupadd --system wildfly
$ sudo useradd -s /sbin/nologin --system -d /opt/wildfly -g wildfly
    ↪  wildfly
$ sudo mkdir /etc/wildfly
```

Copy WildFly systemd service, configuration file and start scripts templates from the /opt/wildfly/docs/contrib/scripts/systemd/ directory.

```
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.conf /
    ↪ etc/wildfly/
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.service
    ↪  /etc/systemd/system/
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/launch.sh /opt/
    ↪ wildfly/bin/
$ sudo chmod +x /opt/wildfly/bin/launch.sh
```

Set /opt/wildfly permissions.

```
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.conf /
    ↪ etc/wildfly/
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.service
    ↪  /etc/systemd/system/
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/launch.sh /opt/
    ↪ wildfly/bin/
$ sudo chmod +x /opt/wildfly/bin/launch.sh
```

Set /opt/wildfly permissions.

```
$ sudo chown -R wildfly:wildfly /opt/wildfly
```

Reload systemd service.

```
$ sudo systemctl daemon-reload
```

Start and enable WildFly service:

```
$ sudo systemctl start wildfly
$ sudo systemctl enable wildfly
```

Confirm WildFly Application Server status.

```
$ sudo systemctl status wildfly
```

Add wildfly users:

```
$ sudo /opt/wildfly/bin/add-user.sh
```

- **Wildfly TLS configuration** Obtain or generate a certificate and private key. This can be done using a certificate authority or by generating a self-signed certificate. Then we'll convert the certificates from .pem format to .jks format.

```
$ sudo openssl pkcs12 -export -out cert.pfx -inkey /etc/letsencrypt
    ↪ /live/ api.smart-parking.me/privkey.pem -in /etc/letsencrypt/
    ↪ live/ api.smart-parking.me/cert.pem -certfile /etc/
    ↪ letsencrypt/live/ api.smart-parking.me/chain.pem
```

6 Mongo DB

To use MongoDB as the database for the project, it must be installed on the virtual machine. The following are the steps to install it:

```
$ sudo apt install -y mongodb
```

Create a list file for MongoDB:

```
$ echo "deb␣[␣arch=amd64␣]␣https://repo.mongodb.org/apt/ubuntu␣bionic/
    ↪ mongodb-org/4.4␣multiverse" | sudo tee /etc/apt/sources.list.d/
    ↪ mongodb-org-4.4.list
```

Update the package list:

```
$ sudo apt update
```

Install the MongoDB package:

```
$ sudo apt install -y mongodb-org
```

Start MongoDB:

```
$ sudo service mongod start
```

Verify that MongoDB has started successfully by checking the status of the service:

```
$ sudo service mongod status
```

7 IoT sensors

This section describes the structure of the IoT sensors.

- **IR sensor** (Infrared sensor) is a device that detects and measures infrared radiation.
- **Motion sensor PIR:** It detects the presence of a moving object by sensing changes in infrared (IR) radiation levels.
- **Servo motor SG-90:** It is a position control type servo motor, which means that it can rotate to a specific position based on the control signal it receives.
- **PiCam** It is a position control type servo motor, which means that it can rotate to a specific position based on the control signal it receives.