Faculté des Sciences & Techniques — Université de Limoges

Master 2nde année

TMC

TP n°2

ESP8266 & WiFi

ESP8266 & MicroPython : protocole sécurisé de communication

Pour la documentation des commandes, vous vous reporterez à l'URL suivante :
`https://docs.micropython.org/en/latest/`

1 – a. En vous mettant en binôme, écrivez un programme en micropython permettant :
- ▷ à l'ESP8266 A de se mettre en mode AP ;
- ▷ à l'ESP8266 B de trouver le point d'accès de A, puis de se connecter à lui en mode client et d'échanger un message avec lui en TCP.

b. Peux-t-on essayer de mettre les deux ESP8266 en mode AP/Client et tenter de le faire simultanèment ?

2 – On voudrait réaliser un protocole **d'échange sécurisé** utilisant uniquement le **ESSID** diffusé par le mode AP de l'ESP8266 :
- ▷ l'ESP8266 d'Alice choisi un ESSID contenant un message obtenu cryptographiquement ;
- ▷ l'ESP8266 de Bob écoute les ESSID diffusés et pense trouver celui d'Alice, il peut alors vérifier que cet ESSID appartient bien à Alice ;
- ▷ si c'est celui d'Alice il se connecte à cet ESSID et fait un « *challenge/response* » basé TCP pour authentifier Alice ;

a. De quels algorithmes cryptographiques l'ESP8266 dispose sur micropython ?
Vous évaluerez leur application à l'authentification.
Une liste de bibliothèques est disponible à `https://awesome-micropython.com`

b. Que doit-on partager entre les interlocuteurs ?
Vous indiquerez les éléments cryptographiques et évaluerez leur qualité.

c. Proposez une méthode de construction du ESSID :
- ◇ que peut-il contenir ?
- ◇ quels algorithmes cryptographique disponible sur l'ESP8266 pouvez vous utiliser ?

Vous proposerez deux solutions :
- ◇ la première basée sur l'utilisation d'un réseau **WiFi ouvert** ;
- ◇ la seconde basée sur un réseau **WiFi sécurisé**.

d. Est-il possible de faire de la « *diffusion* » au travers du ESSID, c-à-d envoyer un message en clair dont la provenance est **prouvable** ?

Vous pourrez analyser le programme suivant :
`https://learn.adafruit.com/circuitpython-totp-otp-2fa-authy-authenticator-friend/software`

Échange avec l'hôte par le port série

Pour l'installation de PySerial :

```
xterm
$ python3 -m pip install pyserial
```

Soient les deux programmes :

le fichier `chat.py` :

```
#!/usr/bin/python3

import serial

s = serial.Serial('/dev/ttyUSB0',115200)

print('start')
while 1:
    saisie = input(">")
    s.write(bytes(saisie,encoding='utf8')+b'\r\n')
    # lire l'echo de la ligne entrée
    ligne = s.readline()
    # lire la ligne réponse
    ligne = s.readline()
    if not ligne:
        continue
    print(ligne)
```

le fichier `mchat.py` :

```
print('micropython: start')

while 1:
    ligne = input()
    if not ligne:
        continue
    if ligne == "stop":
        break
    print('Echo :'+ligne)
```

Pour lancer le programme sur l'ESP8266 en libérant le port série :

```
xterm
ampy --port /dev/ttyUSB0 run --no-output mchat.py
```

et sur le linux :

```
xterm
python3 chat.py
```

Vous vérifierez que le texte que vous saisissez dans votre terminal est bien renvoyé par l'ESP8266 avec ne préfixe, la chaîne `'ECHO :'`

3 – a. Écrivez un programme micropython pouvant communiquer avec hôte par le port série et qui réalise des commandes simples : passage en caractères majuscules, opérations arithmétiques.

b. En reprenant l'idée de l'exercice 2), pouvez vous écrire un programme micropython communiquant par ESSID et dont les messages sont fournis par l'hôte ?

Comment doit-on alterner les « *communications* » pour que cela marche ?
Que doit-on partager entre les interlocuteurs ?
Comment se « *synchroniser* » ?