

Responsable du document : Arnaud HINCELIN

1. INTRODUCTION

a. RAPPELS DU PROJET

A grande échelle, le projet SCOUTBOT met en scène 2 parties que sont le robot et la télécommande, chacune de ces parties sont dirigées par un STM32MP157C-DK2. Ce modèle de MPU provenant de chez STMicroelectronics donne la possibilité de faire des communication en Wi-Fi normé 802.11b/g/n.

Afin de faire communiquer nos 2 cartes MP1, nous pouvons les connecter au même routeur Wi-Fi, mais auquel cas cela oblige à disposer d'un routeur de disponible. La 2nd solution que nous avons choisi est de configurer une des deux cartes MP1 en mode point d'accès (« HotSpot »), ce qui permet de générer un signal Wi-Fi auquel d'autres appareils peuvent se connecter. Notre carte MP1 de la télécommande va alors se connecter à la carte MP1 du robot configurée en Hotspot.

b. GENERALITES

La carte MP1 possède plusieurs interfaces internet. Ces dernières sont visibles avec la commande suivante.

```
$ ifconfig
```

```
usb0      Link encap:Ethernet  HWaddr 92:57:7A:FE:BC:06
          inet addr:192.168.7.1  Bcast:192.168.7.255  Mask:255.255.255.0
          inet6 addr: fe80::9057:7aff:fefe:bc06/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:968 errors:0 dropped:0 overruns:0 frame:0
          TX packets:246 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:106155 (103.6 KiB)  TX bytes:60354 (58.9 KiB)

wlan0     Link encap:Ethernet  HWaddr 00:9D:6B:92:4F:0D
          inet addr:192.168.72.1  Bcast:192.168.72.255  Mask:255.255.255.0
          inet6 addr: fe80::29d:6bff:fe92:4f0d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1360 errors:0 dropped:0 overruns:0 frame:0
          TX packets:400 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:145892 (142.4 KiB)  TX bytes:71641 (69.9 KiB)
```

Pour rappel, afin de que l'utilisateur puisse piloter la carte MP1 depuis un PC, il y a 2 choix ;

- En UART avec ST-Link

Via le ST-Link de la carte, s'assurer que le câble micro USB est bien connecté.

```
$ minicom -D /dev/ttyACM0
```

```
arnaud@arnaud-VirtualBox: ~
Fichier  Édition  Affichage  Rechercher  Terminal  Aide

Bienvenue avec minicom 2.7.1

OPTIONS: I18n
Compilé le Dec 23 2019, 02:06:26.
Port /dev/ttyACM0, 12:15:17

Tapez CTRL-A Z pour voir l'aide concernant les touches spéciales

root@stm32mp1:~#
```

- En SSH avec internet

Via la prise USB OTG de la carte, avec l'adresse IP 192.168.7.1 (pas en Wi-Fi à l'adresse 192.168.72.1)

```
$ ssh root@192.168.7.1
```

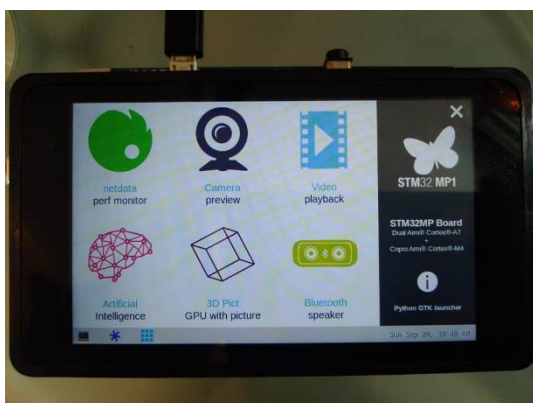
```
arnaud@arnaud-VirtualBox: ~
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
arnaud@arnaud-VirtualBox:~$ ssh root@192.168.7.1
root@stm32mp1:~#
```

2. STM32MP1 EN MODE POINT D'ACCES – PARTIE SERVEUR

a. ACTIVER LE MODE POINT D'ACCES MANUELLEMENT AFIN DE LE CONFIGURER

Démarrer la carte MP1 à que l'on souhaite configurer en mode point d'accès.

Puis une fois arrivé au menu, cliquer sur l'icône netdata perf monitor, et activer le switch « netdata over wifi ». On a maintenant activé le mode point d'accès, il faut le configurer.



b. CONFIGURER LE MODE POINT D'ACCES

Etape 1 : Se connecter à la carte depuis le PC avec le ST-Link :

```
PC $ minicom -D /dev/ttyACM0
```

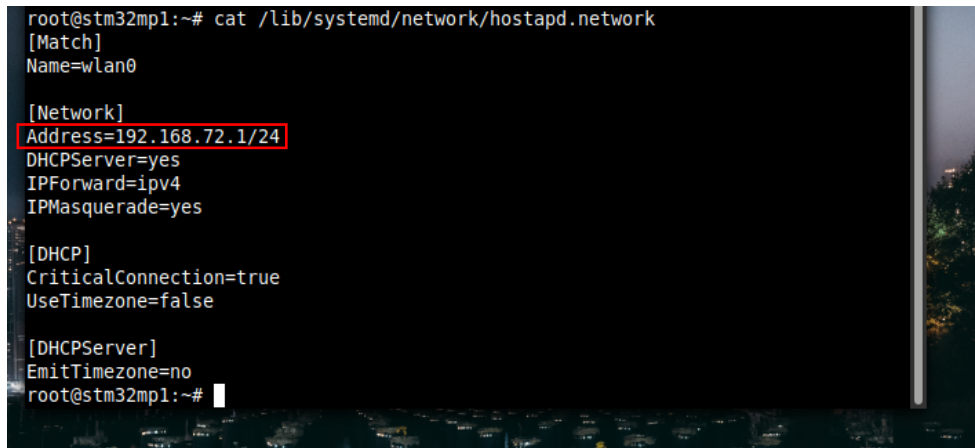
Etape 2 : Configurer l'adresse IP de notre point d'accès mobile :

Ajouter l'adresse IP dans le fichier `/lib/systemd/network/hostpad.network`. Avec l'éditeur de texte vi, on ajoute la ligne « `Address=192.168.72.1/24` » en dessous de la ligne « `[Network]` ».

```
MP1 $ vi /lib/systemd/network/hostpad.network
```

Et on vérifie

```
MP1 $ cat /lib/systemd/network/hostpad.network
```



```
root@stm32mp1:~# cat /lib/systemd/network/hostpad.network
[Match]
Name=wlan0

[Network]
Address=192.168.72.1/24
DHCPv4=yes
IPForward=ipv4
IPMasquerade=yes

[DHCP]
CriticalConnection=true
UseTimezone=false

[DHCPv4]
EmitTimezone=no
root@stm32mp1:~#
```

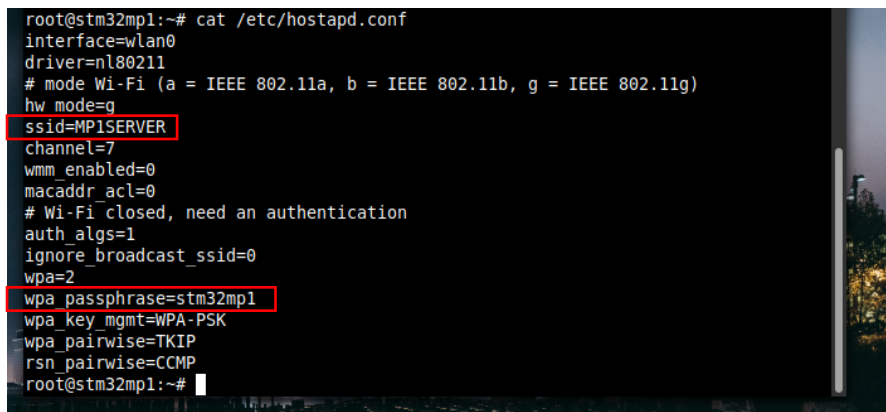
Etape 3 : Configurer le point d'accès avec un SSID et un mot de passe :

Ajouter le SSID et le mot de passe dans le fichier de configuration du point d'accès /etc/hostapd.conf

Ajouter le nom du ssid « ssid=MP1SERVER » et le mot de passe « wpa_passphrase=stm32mp1 » avec l'éditeur de texte vi.

```
MP1 $ vi /etc/hostapd.conf
```

```
MP1 $ cat /etc/hostapd.conf
```



```
root@stm32mp1:~# cat /etc/hostapd.conf
interface=wlan0
driver=nl80211
# mode Wi-Fi (a = IEEE 802.11a, b = IEEE 802.11b, g = IEEE 802.11g)
hw mode=g
ssid=MP1SERVER
channel=7
wmm_enabled=0
macaddr_acl=0
# Wi-Fi closed, need an authentication
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=stm32mp1
wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
root@stm32mp1:~#
```

Etape 4 : Démarrer automatiquement le routeur au démarrage de la carte

Ajouter la ligne « ExecStartPre=/sbin/ip link set wlan0 up » dans le fichier /lib/systemd/system/hostpad.service avec l'éditeur de texte vi.

```
MP1 $ vi /lib/systemd/system/hostpad.service
```

```
MP1 $ cat /lib/systemd/system/hostpad.service
```

```
root@stm32mp1:~# cat /lib/systemd/system/hostapd.service
[Unit]
Description=Hostapd IEEE 802.11 AP, IEEE 802.1X/WPA/WPA2/EAP/RADIUS Authenticatr
After=network.target

[Service]
Type=forking
PIDFile=/run/hostapd.pid
ExecStartPre=/sbin/ip link set wlan0 up
ExecStart=/usr/sbin/hostapd /etc/hostapd.conf -P /run/hostapd.pid -B

[Install]
WantedBy=multi-user.target
root@stm32mp1:~#
```

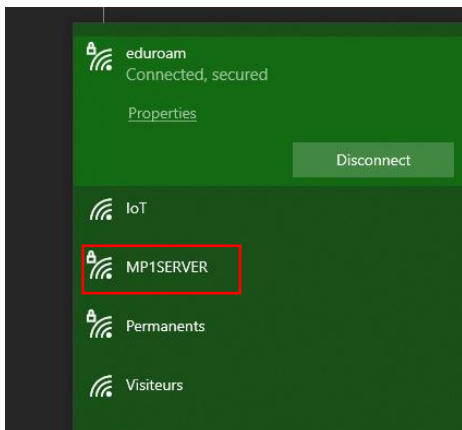
Etape 5 : Activer le point d'accès avec le service systemd

```
MP1 $ systemctl enable hostapd
```

Etape 6 : Faire un reboot de la carte MP1

```
MP1 $ reboot
```

Une fois démarré, le MP1 a démarré automatiquement le point d'accès et propose un réseau avec comme nom visible (ou SSID) « MP1SERVER ». Pour se connecter à ce réseau, il suffit de se connecter avec le mot de passe du réseau, « stm32mp1 » dans notre cas.



Remarque : Le SSID visible par les autres appareils est « MP1SERVER », mais le SSID indiqué par le « netdata perf monitor » est différent.

Plus de détails : [How to configure a wlan interface on hotspot mode - stm32mpu](#)

3. STM3MP1 CONNECTE A UN POINT D'ACCES – PARTIE CLIENT

a. INITIALISER ET CONFIGURER L'INTERFACE WLAN

Etape 1 : Visualiser l'interface WLAN

On visualise l'ensemble des interfaces disponibles sur le MP1, on doit y trouver l'interface du Wi-Fi wlan0. Mettre l'option « -a » avec la commande ifconfig afin de visualiser aussi les interfaces qui ne sont pas activées.

```
MP1 $ ifconfig -a
```

```
wlan0    Link encap:Ethernet  HWaddr 00:9D:6B:92:A0:B5
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@stm32mp1:/usr/local#
```

Etape 2 : Configurer l'interface WLAN

Associer à l'interface wlan0 une adresse IP, une diffusion, et un masque.

```
MP1 $ ifconfig wlan0 192.168.43.135 broadcast 192.168.43.255 netmask 255.255.255.0
```

Etape 3 : Activer l'interface WLAN

On active l'interface wlan0, puis on affiche afin de valider l'activation. On doit alors voir « UP » qui indique que l'interface est correctement activée. De plus, on doit visualiser la configuration de l'interface.

```
MP1 $ ifconfig wlan0 up
```

```
MP1 $ ifconfig wlan0
```

```
wlan0    Link encap:Ethernet  HWaddr 00:9D:6B:92:A0:B5
          inet addr:192.168.43.135  Bcast:192.168.43.255  Mask:255.255.255.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

root@stm32mp1:~#
```

b. Scanner les points d'accès disponibles

Etape 1 : Scanner les réseaux disponibles

Il faut effectuer un scan des réseaux disponibles, afin de vérifier que l'on voit bien le second MP1 configuré en mode point d'accès. Pour cela, on doit y voir le SSID « MP1SERVER ».

```
MP1 $ iw dev wlan0 scan | grep SSID
```

```
root@stm32mp1:~# iw dev wlan0 scan | grep SSID
SSID: MP1SERVER
SSID: Permanents
SSID: IoT
SSID: IoT
SSID: Visiteurs
SSID: eduroam
SSID: Permanents
SSID: IoT
SSID: IoT
SSID: IoT
SSID: Visiteurs

root@stm32mp1:~#
```

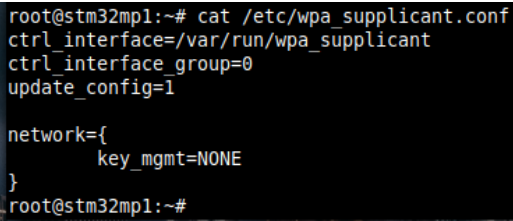
Donc le réseau du MP1SERVER est bien disponible.

c. Configurer la connexion Wi-Fi

Pour configurer la connexion, on utilise l'outil « wpa_supplicant ». L'idée va être d'ajouter le réseau du MP1SERVER dans les réseaux connus, en inscrivant le SSID et le mot de passe du réseau dans le fichier de configuration.

Etape 1 : Visualiser la configuration du Wi-Fi

```
MP1 $ cat /etc/wpa_supplicant.conf
```



```
root@stm32mp1:~# cat /etc/wpa_supplicant.conf
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

network={
    key_mgmt=NONE
}
```

On voit que aucun réseau n'est connu actuellement, on ne peut donc pas se connecter au réseau MP1SERVER.

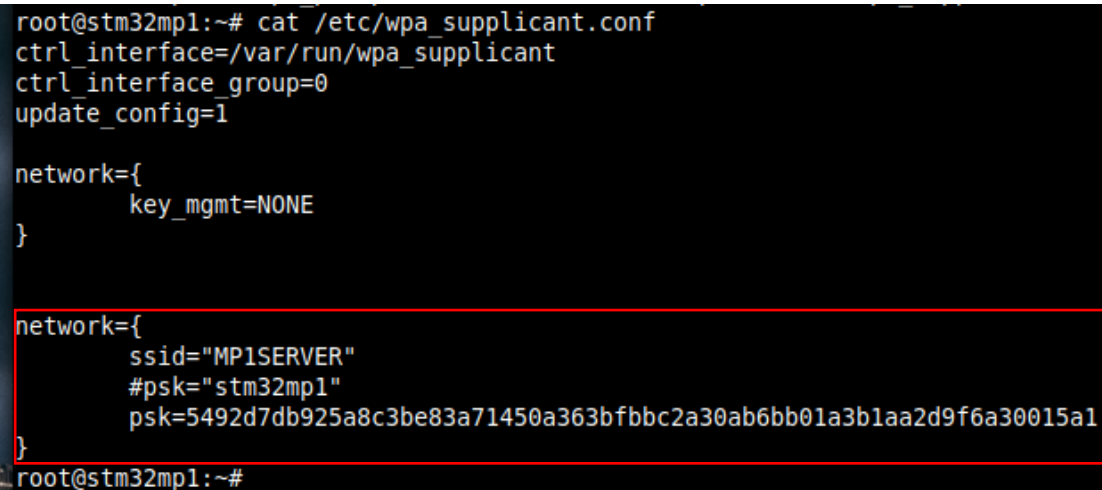
Etape 2 : Ajouter le réseau Wi-Fi

On ajoute notre réseau MP1SERVER, en connaissant le SSID « MP1SERVER » et le mot de passe « stm32mp1 ». On écrit dans le fichier de configuration du Wi-Fi.

```
MP1 $ wpa_passphrase MP1SERVER stm32mp1 >> /etc/wpa_supplicant.conf
```

Puis on visualise afin de voir si le réseau a bien été ajouté.

```
MP1 $ cat /etc/wpa_supplicant.conf
```



```
root@stm32mp1:~# cat /etc/wpa_supplicant.conf
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

network={
    key_mgmt=NONE
}

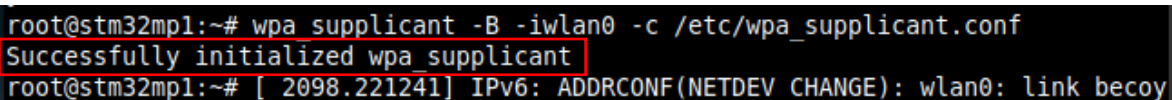
network={
    ssid="MP1SERVER"
    #psk="stm32mp1"
    psk=5492d7db925a8c3be83a71450a363bfbbc2a30ab6bb01a3b1aa2d9f6a30015a1
}
```

Le réseau connu avec le SSID « MP1SERVER » est maintenant connu.

Etape 3 : Initialiser le wpa_supplicant avec la nouvelle configuration

On établit la nouvelle configuration, on doit avoir un message qui confirme le succès de l'initialisation.

```
MP1 $ wpa_supplicant -B -iwlan0 -c /etc/wpa_supplicant.conf
```



```
root@stm32mp1:~# wpa supplicant -B -iwlan0 -c /etc/wpa_supplicant.conf
Successfully initialized wpa supplicant
root@stm32mp1:~# [ 2098.221241] IPv6: ADDRCONF(NETDEV_CHANGE): wlan0: link becoy
```

Etape 4 : Etablir le lien de l'interface wlan0 avec le réseau

```
MP1 $ iw wlan0 link
```



```

root@stm32mp1:~# iw wlan0 link
Connected to 00:9d:6b:92:4f:0d (on wlan0)
    SSID: MP1SERVER
    freq: 2442
    RX: 1234 bytes (7 packets)
    TX: 6765 bytes (50 packets)
    signal: -53 dBm
    rx bitrate: 65.0 MBit/s
    tx bitrate: 24.0 MBit/s

    bss flags:        short-slot-time
    dtim period:      2
    beacon int:       100
root@stm32mp1:~#

```

On est bien connecté au réseau identifié par le SSID « MP1SERVER », notre 2nd MP1 en mode point d'accès.

d. AUTOMATISER L'ADRESSAGE ET TESTER LA CONNEXION

Etape 1 : Utiliser le DHCP (Dynamic Host Configuration Protocol)

Ce protocole de management de réseau est utilisée afin d'automatiser l'assignement d'adresses IP, ce qui évite de configurer manuellement l'adressage IP de connexion.

On utilise ce protocole sur notre interface Wi-Fi wlan0. Rien ne doit s'afficher.

```
MP1 $ dhclient wlan0
```

Puis, on visualise cette automatisation. Pour confirmer l'opération, on doit y voir une adresse IP de type 192.168.72.XXX correspondant à l'adresse IP assignée par le protocole DHCP, qui va communiquer avec le serveur MP1SERVER.

```
MP1 $ ip addr show wlan0
```

```

root@stm32mp1:~# dhclient wlan0
RTNETLINK answers: File exists
root@stm32mp1:~# ip addr show wlan0
4: wlan0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group0
    link/ether 00:9d:6b:92:a0:b5 brd ff:ff:ff:ff:ff:ff
    inet 192.168.43.135/24 brd 192.168.43.255 scope global wlan0
        valid lft forever preferred lft forever
    inet 192.168.72.39/24 brd 192.168.72.255 scope global wlan0
        valid lft forever preferred lft forever
    inet6 fe80::29d:6bff:fe92:a0b5/64 scope link
        valid lft forever preferred_lft forever
root@stm32mp1:~#

```

Ici, notre adresse IP est 192.168.72.39, il est possible que n'est ne soit pas la même dans votre cas.

Etape 2 : Tester la connexion

Afin de tester la connexion, on utilise la commande « ping » vers l'adresse du réseau MP1SERVER (192.168.72.1). Si la connexion fonctionne, on voit que des paquets de données sont échangés dans le terminal, stopper avec un CTRL-C ou CTRL-Z.

```
MP1 $ ping 192.172.72.1
```

```

root@stm32mp1:~# ping 192.168.72.1
PING 192.168.72.1 (192.168.72.1) 56(84) bytes of data.
64 bytes from 192.168.72.1: icmp_seq=1 ttl=64 time=24.3 ms
64 bytes from 192.168.72.1: icmp_seq=2 ttl=64 time=32.1 ms
64 bytes from 192.168.72.1: icmp_seq=3 ttl=64 time=30.9 ms
64 bytes from 192.168.72.1: icmp_seq=4 ttl=64 time=40.2 ms
^Z[1]+  Stopped                  ping 192.168.72.1
root@stm32mp1:~#

```

La connexion est maintenant réussie. On peut alors parler avec des scripts basés sur des sockets.

e. AUTOMATISER LE WI-FI AVEC LA CONFIGURATION ACTUELLE AU DEMARRAGE

Afin de ne pas avoir à refaire toutes les commandes de configuration du Wi-Fi, il est utile de les automatiser.

Etape 1 : Visualiser le statut de chaque interface réseau

On utilise le service « networkctl » de l'outil « systemd-networkd » afin de voir les interfaces réseaux. Nous en avons 4, avec « lo » pour le localHost, « eth0 » pour le réseau du port ethernet de la carte, « usb0 » pour le réseau internet via le câble USB-C (OTG) entre la carte et le PC, et enfin « wlan0 » qui correspond à l'interface Wi-Fi.

L'interface "wlan0" est « routable » et « unmanaged » ce qui signifie qu'elle est utilisée mais non gérée par l'outil networkd.

```
MP1 $ networkctl --no-pager
```

```
root@stm32mp1:~# networkctl --no-pager
IDX LINK   TYPE       OPERATIONAL SETUP
  1 lo      loopback   carrier    unmanaged
  2 eth0    ether      no-carrier configuring
  3 usb0    gadget     routable   configured
  4 wlan0   wlan       routable   unmanaged

4 links listed.
root@stm32mp1:~#
```

Etape 2 :

```
root@stm32mp1:~# cat /lib/systemd/network/51-wireless.network
[Match]
Name=wlan0
[Network]
DHCP=ipv4
```

Etape 3 :

```
root@stm32mp1:~# iw dev wlan0 scan |grep SSID
SSID: MP1SERVER
SSID: Visiteurs
SSID: eduroam
SSID: Permanents
SSID: IoT
SSID: TachotWIFI
    * Multiple BSSID
SSID: Visiteurs
SSID: eduroam
SSID: Permanents
SSID: IoT
SSID: eduroam
SSID: Permanents
root@stm32mp1:~#
```


Etape 4 :

```
root@stm32mp1:~# cat /etc/wpa_supplicant/wpa_supplicant-wlan0.conf
ctrl_interface=/var/run/wpa_supplicant
eapol_version=1
ap_scan=1
fast_reauth=1

network={
    ssid="MP1SERVER"
    #psk="stm32mp1"
    psk=5492d7db925a8c3be83a71450a363bfbbc2a30ab6bb01a3b1aa2d9f6a30015a1
}
```

4. Communication avec sockets

Tuer le processus de wpa_supplicant : `$ pkill wpa_supplicant`