

# APPLICATION NOTE

## Linux 系统\_ATBM\_WIFI 相关\_FAQ

**ATBM603X**1x1 802.11b/g/n  
Wi-Fi 芯片

## Table of contents

<b>1</b>	<b>STA MODE.....</b>
1.1	内核需要打开如下宏用于使能 iwconfig, iwpriv, iwlist 命令 ....
1.2	启动流程.....
(1)	加载 cfg80211.ko.....
(2)	加载 atbm_wifi.ko .....
(3)	启动 wpa_supplicant .....
(4)	启动 dhcpc.....
1.3	连接路由方法.....
(1)	通过 wpa_cli 连接 .....
(2)	通过修改 wpa_supplicant.conf 连接【常用】 .....
(3)	通过 iwconfig 连接不加密路由器 .....
(4)	ssid 带有中文字符的连接方法.....
1.4	扫描隐藏 ssid 的 AP 的方法.....
(1)	通过 wpa_cli 扫描 .....
(2)	通过 iwlist 扫描 .....
1.5	WEP 连接方法配置说明 .....
(1)	路由器 WEP 配置说明 .....
(2)	不同的路由配置, 对应的 wpa_supplicant.conf 的参数设置
1.6	连接 WPA3 加密 AP 方法.....
(1)	驱动配置.....
(2)	wpa_supplicant 配置 .....
1.7	连接管理帧加密的路由器的配置.....
(1)	确认路由器为管理帧加密 .....
(2)	wpa_supplicant 配置修改 .....
1.8	问题分析.....
(1)	连接不上路由器.....

AN9310

Doc Rev: 3.3

Released:2023-01-30

(2)	Ifconfig wlan0 up 不起来.....	20
(3)	Wpa_supplicant 使用 WEXT 接口, Wpa_cli 扫描执行一直失败 .....	22
(4)	Wpa_cli 执行一直报 Failed to connect to non-golbal ctrl_ifname:(nil) error:No such file or directly .....	23
<b>2</b>	<b>AP MODE .....</b>	<b>24</b>
2.1	内核需要打开如下宏用于使能 iwconfig, iwpriv, iwlist 命令 .....	24
2.2	启动流程.....	24
(1)	加载 cfg80211.ko 见 STA MODE 部分 .....	24
(2)	加载 atbm_wifi.ko 见 STA MODE 部分 .....	24
(3)	启动 hostapd .....	24
(4)	启动 dnsmasq/dhcpd 二选一 .....	25
2.3	WEP 加密方式配置 .....	26
(1)	WPA/HT 相关的全部拿掉.....	26
(2)	WEP 加密配置方式 .....	26
(3)	WEP 配置说明 .....	27
2.4	WPA3 加密方式配置 .....	30
2.5	注意事项.....	30
(1)	hostapd 启动失败 .....	30
(2)	客户端连接不上热点 .....	31
(3)	设置 n 模式, HT40 .....	32
(4)	客户端获取不到 IP .....	33
(5)	nl80211:set key fail error = -22 .....	34
(6)	hostapd.conf 设置隐藏 ssid 导致驱动挂掉 .....	34
(7)	执行了命令, hostapd 重载配置无效 .....	35
(8)	hostapd 运行起来, 手机连接上了但是没法建立 TCP 连接 .....	36
(9)	绑定组播地址失败 .....	37
<b>3</b>	<b>BRIDGE MDOE.....</b>	<b>40</b>
3.1	内核需要打开如下宏用于使能 iwconfig, iwpriv, iwlist 命令 .....	40
3.2	功能说明.....	40
3.3	启动流程.....	40
(1)	加载 cfg80211.ko 见 STA MODE 部分 .....	40
(2)	加载 atbm_wifi.ko 见 STA MODE 部分 .....	40
(3)	加载 llc.ko .....	41
(4)	加载 stp.ko .....	41
(5)	加载 bridge.ko .....	41
(6)	启动 wlan0 接口 .....	41
(7)	启动 wlan1/p2p0 接口 .....	41
(8)	设置网桥接口 br0 .....	41
(9)	启动网桥接口 br0 .....	42
(10)	启动 wpa_supplicant .....	42
(11)	启动 dhcpc .....	42
(12)	启动 hostapd .....	42
3.4	注意事项.....	43

---

(1)	网桥接口 br0 起不来 .....	43
(2)	使用 brctl 没法添加接口 br0 .....	43
(3)	使用桥接以后发现获取 IP 地址很慢 .....	44
(4)	将 p2p0 加入桥接以后想修改网口模式，但是修改失败 .....	44
<b>4</b>	<b>MONITOR MODE .....</b>	<b>46</b>
4.1	使用 iwconfig 配置 .....	46
(1)	内核需要打开如下宏用于使能 iwconfig, iwpriv, iwlist 命令 .....	46
(2)	启动 monitor 接口 .....	46
(3)	切换监听信道 .....	46
(4)	关闭 monitor 接口 .....	46
4.2	使用 iw 配置 .....	46
(1)	启动 monitor 模式 .....	47
(2)	切换监听信道 .....	47
(3)	退出 monitor 模式 .....	47
<b>5</b>	<b>一个 SOC 运行两个 ALTOBEAM WIFI 的方法 .....</b>	<b>47</b>
5.1	usb id 相同的问题 .....	47
(1)	修改内核的 usb 驱动，枚举的时候修改掉内核保存的 usb id .....	47
(2)	驱动修改 .....	48
5.2	修改完成，使用方法 .....	48
(1)	加载两个驱动 .....	48
(2)	Ifconfig -a 可以看到多个网口 .....	49
(3)	使用 wpa_supplicant 连接 ap .....	49
<b>6</b>	<b>应用接收 WPA_SUPPLICANT &amp; HOSTAPD EVENT 处理方法 .....</b>	<b>51</b>
6.1	wpa_supplicant .....	51
(1)	wpa_supplicant 运行参数 .....	51
(2)	新增参数说明 & unix domain socket 路径说明 .....	51
6.2	hostapd .....	51
(1)	hostapd 运行参数 .....	51
(2)	unix domain socket 路径说明 .....	51

Author	version	change	日期
yuzhihuang	Init v1.0	Init	
Yuzhihuang	v1.1	Add monitor func	
yuzhihuang	V1.2	增加桥接模式下获取 ip 地址慢的解决方法	
Yuzhihuang	V1.3	增加 hostapd 重载配置的方法	2019-12-31
Yuzhihuang	V1.4	增加需要打开内核相关的宏的说明	2020-03-03
Yuzhihuang	V1.5	增加 hostapd 运行异常说明与解决方法	2020-03-20
Yuzhihuang	V1.6	模板	2020-04-29
Yuzhihuang	V1.7	内核宏没开完全导致的通信问题	2020-05-08
Yuzhihuang	V1.8	有些平台内核开支持 iwpriv 宏需要驱动支持的方法	2021-01-11
Yuzhihuang	V1.9	增加 wpa 连接配置说明	2021-05-26
Yuzhihuang	V2.0	增加一个 soc 如何配置跑两个 altobeam wifi 的方法	2021-09-03
Yuzhihuang	V2.1	增加连接 WPA3 路由器的方法	2021-10-14
Yuzhihuang	V2.2	ap 模式下绑定组播地址失败的问题	2021-10-19
Yuzhihuang	V2.3	增加 Sta 模式下连接管理帧加密的路由器配置方法	2021-12-31
Yuzhihuang	V2.4	增加 hostapd wpa 加密方式配置	2022-01-06
Yuzhihuang	V2.5	增加 udhcpd.conf 配置内容	2022-01-25
Yuzhihuang	V2.6	增加 wpa_cli 无法进入命令行交互的解决方法	2022-03-10
Yuzhihuang	V2.7	增加扫描隐藏 ssid 热点的方法	2022-08-15
Yuzhihuang	V2.8	增加接收 hostapd & wpa_supplicant EVENT 方法	2022-09-06
Yuzhihuang	V2.9	Bridge 配置宏说明修改	2022-10-25
Yuzhihuang	V3.0	增加通过 iwconfig 连接不加密路由器的方法	2022-12-13

Yuzhihuang	V3.1	增加通过 iw 命令设置 monitor 方式	2022-12-15
Yuzhihuang	V3.2	增加连接带有中文字符的 ssid 的路由器的方法	2023-01-11
Yuzhihuang	V3.3	增加 dnsmasq 启动参数, 解决 iphone 连接速度慢的问题, 参照 redmine 任务 3359	2023-01-30

# 1 STA MODE

## 1.1 内核需要打开如下宏用于使能 iwconfig, iwpriv, iwlist

### 命令

```
CONFIG_CFG80211_WEXT=y
CONFIG_WIRELESS=y
CONFIG_WIRELESS_EXT=y
CONFIG_WEXT_CORE=y
CONFIG_WEXT_PROC=y
CONFIG_WEXT_PRIV=y
```

某些内核部分宏需要有模块使用才可以打开，所以需要在驱动里面添加以下信息：

```
menuconfig ATBM_WIRELESS
    tristate "Atbm Wireless Lan"
    default m

if ATBM_WIRELESS
    config ATBM_APOLLO
        tristate "ATBM_APOLLO WLAN support"
        select CRYPTO
        select CRYPTO_ARC4
        select CRYPTO_AES
        select CRC32
        select AVERAGE
        depends on !ATBM_MENUCONFIG
        help

        This is an experimental driver for the ATBM_APOLLO chip-set.
        Enabling this option enables the generic driver without
        any platform support.

        Please select the appropriate platform below.

    config ATBM_WEXT
        tristate "support wireless wext"
        select WIRELESS_EXT
        select WEXT_PRIV
        ---help---
        only, select WEIRELESS_EXT and WEXT_PRIV

if ATBM_MENUCONFIG || ATBM_APOLLO
    choice
        prompt "select which atbm Wi-Fi product will be used:ATBM601x,ATBM602x,default:ATBM602x"
        default ATBM602x
        depends on ATBM_MENUCONFIG || ATBM_APOLLO
        help
        Here,you must make sure which atbm Wi-Fi product you will want to use,ATBM601x,ATBM602x
    config ATBM601x
        bool "ATBM601x chip"
        depends on ATBM_MENUCONFIG || ATBM_APOLLO
        help
```

内核执行 `make menuconfig` 能够遍历到该配置，需要在 `drivers/net/wireless/Kconfig` 里面添加 `wifi` 驱动目录的路径：

需要注意该路径是相对路径。

```

menuconfig WLAN
bool "Wireless LAN"
depends on !S390
depends on NET
select WIRELESS
default y
---help---
This section contains all the pre 802.11 and 802.11 wireless
device drivers. For a complete list of drivers and documentation
on them refer to the wireless wiki:

http://wireless.kernel.org/en/users/Drivers

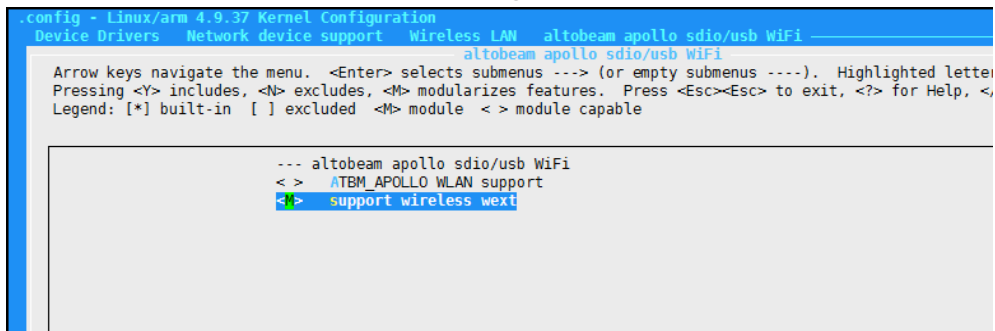
if WLAN

source "drivers/net/wireless/admtek/Kconfig"
source "drivers/net/wireless/ath/Kconfig"
source "drivers/net/wireless/atmel/Kconfig"
source "drivers/net/wireless/broadcom/Kconfig"
source "drivers/net/wireless/cisco/Kconfig"
source "drivers/net/wireless/intel/Kconfig"
source "drivers/net/wireless/intersil/Kconfig"
source "drivers/net/wireless/marvell/Kconfig"
source "drivers/net/wireless/mediatek/Kconfig"
source "drivers/net/wireless/ralink/Kconfig"
source "drivers/net/wireless/realtek/Kconfig"
source "drivers/net/wireless/rsi/Kconfig"
source "drivers/net/wireless/st/Kconfig"
source "drivers/net/wireless/ti/Kconfig"
source "drivers/net/wireless/zydas/Kconfig"
source "drivers/net/wireless/atbm_wifi_SVN1149_LMAC7111_20190528/Kconfig"

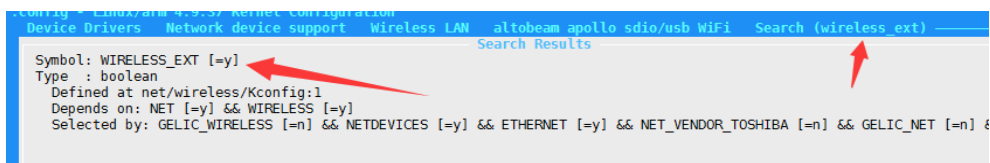
config PCMCIA_RAYCS

```

这时候需要在内核再执行下 make menuconfig 去将配置打开:



打开配置以后，再去确认相关的宏打开了:



## 1.2 启动流程

### (1) 加载 cfg80211.ko

```
insmod cfg80211.ko
```

```

[/ext/demo]## insmod /lib/modules/cfg80211.ko
cfg80211: Calling CRDA to update world regulatory domain
[/ext/demo]##

```

### (2) 加载 atbm\_wifi.ko

```
insmod atbm_wifi.ko
```

```
[/ext/demo]## insmod /ext/lib/modules/atbm_wifi.ko
atbm_init_firmware
xxxx minstrel ht init
xxxx minstrel ht init
SVN_VER=901,DPLL_CLOCK=1,BUILD_TIME=[===USB-APOLLO===
[no platform]:platform set power [1]
Probe called
self->tx_hwchanid 0
Allocated hw_priv @ c6353300
atbm_before_load_firmware++
+++++1.0v+++++
atbm_start_load_firmware++
atbm_start_load_firmware: used firmware.h=
atbm_start_load_firmware: START DOWNLOAD ICM=====
atbm_load_firmware_generic: addr 0: len 20000
atbm_start_load_firmware: START DOWNLOAD DCM=====
atbm_load_firmware_generic: addr 800000: len 8000
atbm_after_load_firmware++
mdelay wait wsm_startup_done !!
wsm_caps.firmwarecap 1 Firmware used old-rate policy
apollo wifi wsm init done.
Input buffers: 32 x 1632 bytes
Hardware: 7.21
WSM firmware [MODEM=RF-ATHENA_B 2GHz Jul 27 2018 11:46:49], ver: 5050, build: 2029, api: 1060, cap: 0x0001 Config[30004] expection 900b8b4, ep0 cmd addr 901d1a0
#####ERROR: hwmac cmd not use ep0! hwmac cmd use ep0!#####
atbm_firmware_init_check send data (null)
apollo wifi : can't open /data/.mac.info
efuse data is [0x1,0x32,0x8,0xa,0x9,0x1b,0x0,0x0,0x0:0x0:0x0:0x0]
ELOG_INIT len 64
[atbm_wtd]:set wtd_probe = 1
usbcore: Registered new interface driver atbm_wlan
[wtd] register
[/ext/demo]##
```

### (3) 启动 wpa\_supplicant

要注意 wpa\_supplicant.conf 文件的配置

一般 wpa\_supplicant.conf 内容为:

ctrl\_interface 参数要根据实际去修改

注意:

/ext/demo/run/wpa\_supplicant 这个目录一定要存在, 不存在手动创建一个。

ctrl\_interface=/ext/demo/run/wpa\_supplicant

该参数代表如果配置内容有变化会更新

update\_config=1

启动的方式为:

```
wpa_supplicant -Dnl80211 -i wlan0 -c /ext/demo/run/wpa_supplicant.conf &
```

如果内核支持 wext 也可以是:

```
wpa_supplicant -Dwext -i wlan0 -c /ext/demo/run/wpa_supplicant.conf &
```

启动以后要确认几件事情: 右下图说明一直在扫描周围的热点

1、wpa\_supplicant 正常运行

2、ifconfig 可以看到 wlan0 接口

```

104 root      0 SW      [usb_atbm_bh]
106 root      2816 S      wpa_supplicant -Dwext -i wlan0 -c /ext/demo/run/wpa_
112 root      2200 R      ps
[/ext/demo]## atbm_hw_scan:if_id(0)
ifconfig wlan0 up
hw_priv->bstartTx 0
hw_priv->scan.status 0
atbm_scan_work:end(0)
nfig
eth0      Link encap:Ethernet  Hwaddr 08:00:20:49:32:60
          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)
          Interrupt:90 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  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:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

wlan0     Link encap:Ethernet  Hwaddr 00:12:34:68:3E:8B
          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)

[/ext/demo]##
```

### (4) 启动 dhcpcd

dhcpcd -i wlan0 &



## 1.3 连接路由方法

### (1) 通过 wpa\_cli 连接

a) 看 wpa\_supplicant.conf 中:

```
ctrl_interface=/var/run/wpa_supplicant
```

b) `wpa_cli -p/var/run/wpa_supplicant -iwlan0`

//如果 wpa\_supplicant.conf 没有保存 network 的话就为 0, 有 2 个 network 的话就为 2

```
>add_network 0 // 添加一个网络号
```

```
>set_network 0 ssid '"TP-LINK_64F8"' // 设置要连接的 ssid
```

OK

```
>set_network 0 priority 9 // 设置优先级
```

OK

```
>set_network 0 key_mgmt WPA_PSK // 设置要连接的路由器加密方式
```

OK

```
>set_network 0 psk '"12345678"' // 设置路由器密码
```

OK

```
>select_network 0 // 选中这个 id
```

OK

```
>enable_network 0 // 使能网络, 设置了这一步以后会自动去连接路由器了
```

OK

### (2) 通过修改 wpa\_supplicant.conf 连接【常用】

常用配置说明:

```
# 请不要修改下面这一行内容, 否则将不能正常工作
ctrl_interface=/var/run/wpa_supplicant

# 确保只有 root 用户能读取 WPA 的配置
ctrl_interface_group=0

# 使用 wpa_supplicant 来扫描和选择 AP
ap_scan=1

# 简单的情形: WPA-PSK 密码验证方式, PSK 是 ASCII 密码短语, 所有合法的加密方式都
允许连接
network={
    ssid="simple"
    psk="very secret passphrase"
    # 优先级越高, 就能越早匹配到。
    priority=5
}
```

```
# 与前面的设置相同，但要求对特定的 SSID 进行扫描（针对那些拒绝广播 SSID 的 AP 也就我们常说的隐藏 ap）
network={
    ssid="second ssid"
    scan_ssid=1
    psk="very secret passphrase"
    priority=2
}

#连接特定 bssid 的 ap
network={
    ssid="second ssid"
    psk="very secret passphrase"
    bssid="AA:BB:CC:DD:EE:FF"
    priority=2
}

# 仅使用 WPA-PSK 方式。允许使用任何合法的加密方式的组合
network={
    ssid="example"
    proto=WPA
    key_mgmt=WPA-PSK
    pairwise=CCMP TKIP
    group=CCMP TKIP WEP104 WEP40
    psk=06b4be19da289f475aa46a33cb793029d4ab3db7a23ee92382eb0106c72ac7bb
    priority=2
}

# 明文连接方式（不使用 WPA 和 IEEE802.1X）
network={
    ssid="plaintext-test"
    key_mgmt=NONE
}

# 共享 WEP 密钥连接方式（不使用 WPA 和 IEEE802.1X）
network={
    ssid="static-wep-test"
    key_mgmt=NONE
    # 引号包含的密钥是 ASCII 密钥
    wep_key0="abcde"
    # 没有引号包含的密钥是十六进制密钥
    wep_key1=0102030405
```

```

wep_key2="1234567890123"
wep_tx_keyidx=0 #关键参数代表使用的哪一组 wep_key
priority=5
}

# 共享 WEP 密钥连接方式（无 WPA 和 IEEE802.1X），使用共享密钥 IEEE802.11 验证方式
network={
    ssid="static-wep-test2"
    key_mgmt=NONE
    wep_key0="abcde"
    wep_key1=0102030405
    wep_key2="1234567890123"
    wep_tx_keyidx=0
    priority=5
    auth_alg=SHARED
}

# 在 IBSS/ad-hoc 网络中使用 WPA-None/TKIP
network={
    ssid="test adhoc"
    mode=1
    proto=WPA
    key_mgmt=WPA-NONE
    pairwise=NONE
    group=TKIP
    psk="secret passphrase"
}

```

默认的 wpa\_supplicant.conf 的内容为:

```

1 ctrl_interface=/ext/demo/run/wpa_supplicant
2 update_config=1

```

修改后的 wpa\_supplicant.conf 的内容为:

```

ctrl_interface=/ext/demo/run/wpa_supplicant
update_config=1

network={
    ssid="TP-LINK_64F8"
    psk="12345678"
}

```

修改完配置需要重启 wpa\_supplicant 应用:

```

killall wpa_supplicant
wpa_supplicant -Dnl80211 -i wlan0 -c
/ext/demo/run/wpa_supplicant.conf &

```

下图是已经连接上路由器但是没有去获取 IP

```
[/ext/demo]##
[/ext/demo]## wpa_supplicant -Dwext -i wlan0 -c /ext/demo/run/wpa_supplicant.conf &
[/ext/demo]## successfully initialized wpa_supplicant
rfkill: cannot oobm_add_interface: vif->type 2, vif->p2p 0, addr 00:12:34:6b:3e:8b
pen RFKILL control device
[STA] slot time :20 us.
atbm_hw_scan:if_id(0)
hw_priv->bstartTx 0
hw_priv->scan_status 0
atbm_scan_work:end(0)
wlan0: Trying to ieee80211_mgd_auth:(34:96:72:f4:8a:8d)
associate with ieee80211_work_work:start work ch(6)
34:96:72:f4:8a:8d[WSM] issue join command.
d (SSID="altobeaatbm_join_work:ch(6),chtype(0),if_id(0)
MSZ_2.4g" freq=2437 MHz)
ioctl[SIOCSIWFREQ]: Device or resource busy
wlan0: Association request atbm_join_work:enable combination mode
atbm_join_work:end
t to the driver failed
ieee80211_start_connecting_work:bssid(34:96:72:f4:8a:8d)
minstrel_ht_update_caps,739:ieee80211_ht_cap_sup_width_20_40
atbm_config: push queue chatype_change_work_pending,joinstatus(3),chatype(2)
[STA] arp ip filter enable: 0
atbm_channel_type_change_work:chatype(2),channelNumber(6)
atbm_channel_type_change_work 426
atbm_channel_type_change_work 430
atbm_bss_info_changed:priv->htcap(1)
[STA] slot time :9 us.
wlan0: Associated with 34:96:72:f4:8a:8d
down_eap_rate
ieee80211_work_work:reset work ch(6)
ieee80211_work_work:start work ch(6)
ieee80211_wk_connecting:connecting,tries(1)
down_eap_rate
wlan0: WPA: key negotiation completed with 34:96:72:f4:8a:8d [PTK=CCMP GTK=TKIP]
wlan0: CTRL-EVENT-CONNECTED - Connection to 34:96:72:f4:8a:8d completed [id=3 id_str=]
ieee80211_wk_connecting:connecting,tries(2)
ieee80211_wk_connecting:connecting,tries(3)
ieee80211_wk_connecting:connecting,tries(4)
ieee80211_wk_connecting:connecting,tries(5)
ieee80211_wk_connecting:connecting,tries(6)
ieee80211_wk_connecting:time out
ieee80211_work_work:reset work ch(6)
ieee80211_connecting_work_done
■
```

如果发现获取不到 IP 地址需要重启 udhcpc

```
killall udhcpc
```

```
udhcpc -i wlan0 &
```

```
[/ext/demo]##
[/ext/demo]##
[/ext/demo]## udhcpc -i wlan0
udhcpc (v1.24.1) started
Setting IP address 0.0.0.0 on wlan0
Sending discover...
down_bootp_rate
<WIFI> rx ampdu ++
cancel dhcp_retry_work
Sending select for 192.168.1.120cancel dhcp_retry_work
...
down_bootp_rate
cancel dhcp_retry_work
Lease of 192.168.1.120 obtained, lease time 7200
Setting IP address 192.168.1.120 on wlan0
[STA] arp ip filter enable: 3
wsm_set_pm_indication
ieee80211_ifa_changed(wlan0):IPv4 enable,end_time(97210)
ieee80211_cancel_connecting_work,(34:96:72:f4:8a:8d)
Deleting routers
route: SIOCDELRT: No such process
Adding router 192.168.1.1
Recreating /data/resolv.conf
Adding DNS server 192.168.1.1
Adding DNS server 192.168.1.1
[/ext/demo]## ■
```

```

[/ext/demo]##
[/ext/demo]##
[/ext/demo]## ifconfig
eth0      Link encap:Ethernet  Hwaddr 08:00:20:49:32:60
          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)
          Interrupt:90 Base address:0x2000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:65536  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:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

wlan0     Link encap:Ethernet  Hwaddr 00:12:34:68:3E:8B
          inet addr:192.168.1.120 Bcast:192.168.1.255 Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:586 errors:0 dropped:149 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:60118 (58.7 KiB)  TX bytes:1070 (1.0 KiB)

[/ext/demo]## ping 192.168.1.1
PING 192.168.1.1wsm_handle_tx_data:WSM_HT_TX_WIDTH_40M
(192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: seq=0 ttl=64 time=8.466 ms
64 bytes from 192.168.1.1: seq=1 ttl=64 time=1.987 ms
^C
192.168.1.1: 100% packet transmit

```

### (3) 通过 iwconfig 连接不加密路由器

无法连接 ssid 带有中文字符的路由器，似乎是 iwconfig 内部解析的问题。

命令格式：

```
iwconfig wlan0 essid "WIFI_SSID"
```

```

[/tmp]## iwconfig wlan0 essid "yzh_test"
[atbm_log]:ieee80211_check_country_limit_scan_2_4G_chan : scan_n_channels = 1
+++++++
[atbm_log]:ieee80211_check_country_limit_scan_2_4G_chan : scan_n_channels = 1
-----
[atbm_log]:atbm_hw_scan:if_id(0)
[atbm_log]:atbm_hw_scan:scan, delay suspend
[atbm_log]:scan start band(0),(14)
[/tmp]## [atbm_log]:hw_priv->scan.status 0
[atbm_log]:atbm_scan_work:end(0)
[atbm_log]:wlan0:free authen bss ++
[atbm_log]:authen:(cc:08:fb:92:97:27),ssid(yzh_test)
[atbm_log]:wlan0: authenticated
[atbm_log]:wlan0:free authen bss ++
[atbm_log]:wlan0:free authen bss --
[atbm_log]:wlan0: associated
[atbm_log]:[cc:08:fb:92:97:27]:20M channel
[atbm_log]:ieee80211_recalc_ps:work busy

```

### (4) ssid 带有中文字符的连接方法

wpa\_supplicant 支持通过 16 进制方式去连接。

路由器的 ssid 为：



扫描出来的十六进制为：

```
[atbm_log]:ssid:[F7D2.4]ssid_len:[0] 463944322e34
[atbm_log]:
[atbm_log]:ssid:[yzh_test]ssid_len:[11] 797a685f74657374efbfa5
[atbm_log]:
[atbm_log]:ssid:[yzh_test]ssid_len:[8] 7a636c5f74657374
```

其中 yzh\_test 十六进制为: 797a685f74657374

丕的十六进制为: efbfa5

那么 wpa\_supplicant.conf 这么写:

```
[/ext/demo/wifi]##
[/ext/demo/wifi]## cat wpa_supplicant.conf
ctrl_interface=/tmp/wpa_supplicant
ap_scan=1

network={
    ssid=797A685F74657374efbfa5
    psk="12345678"
}
```

wpa\_cli 连接方式:

```
[/ext/demo/wifi]##
[/ext/demo/wifi]## wpa_cli -p /tmp/wpa_supplicant/
wpa_cli v_ATBM_2.6_0.1
Copyright (c) 2004-2016, Jouni Malinen <j@w1.fi> and contributors

This software may be distributed under the terms of the BSD license.
See README for more details.

selected interface 'wlan0'

Interactive mode
> add_network 0
0
> set_network 0 ssid 797a685f74657374efbfa5
OK
> set_network 0 psk "12345678"
OK
>
> list_network
network id / ssid / bssid / flags
0 yzh_test\xef\xbf\xa5 any [DISABLED]
> select_network 0
[atbm_log]:ieee80211_check_country_limit_scan_2_4G_chan : scan_n_channels = 14 ++++++
OK
> [atbm_log]:ieee80211_check_country_limit_scan_2_4G_chan : scan_n_channels = 14 -----
[atbm_log]:atbm_hw_scan:if_id(0)
[atbm_log]:atbm_hw_scan:scan, delay suspend
[atbm_log]:scan start band(0),(14)
<3>CTRL-Event-SCAN-STARTED
> [atbm_log]:hw_priv->scan.status 0
[atbm_log]:atbm_scan_work:end(0)
<3>CTRL-Event-SCAN-RESULTS
<3>WPS-A[atbm_log]:wlan0:free authen bss ++
P-AVAILABLE
<3>SME: Trying to authenticate with cc:08:fb:92:97:27 (SSID='yzh_test\xef\xbf\xa5' freq=2412 MHz)
> [atbm_log]:authen:(cc:08:fb:92:97:27),ssid(yzh_test丕 [atbm_log]:wlan0: authenticated

<3>Trying to[atbm_log]:wlan0:free authen bss ++
associate with cc:08:fb:92:97:27 (SSID='yzh_test\xef\xbf\xa5' freq=2412 MHz)
> [atbm_log]:wlan0:free authen bss --
[atbm_log]:wlan0: associated
[atbm_log]:[cc:08:fb:92:97:27]:20M channel
[atbm_log]:ieee80211_recalc_ps:work busy
<3>Associated with cc:08:fb:92:97:27
<3>CTRL-Event-SUBNET-STATUS-UPDATE status=0
> [atbm_log]:ieee80211_recalc_ps:work busy
<3>WPA: Key negotiation completed with cc:08:fb:92:97:27 [PTK=CCMP GTK=CCMP]
<3>CTRL-Event-CONNECTED - Connection to cc:08:fb:92:97:27 completed [id=0 id_str=]
> [atbm_log]:ieee80211_wk_connecting: time out
```

## 1.4 扫描隐藏 ssid 的 AP 的方法

扫描隐藏 ssid 的 AP 需要知道这个 AP 的 ssid 才行, 有如下两种方法。

## (1) 通过 wpa\_cli 扫描

```
# 添加一个网络连接, 会返回<network id>
wpa_cli -i wlan0 add_network
# ssid名称
wpa_cli -i wlan0 set_network <network id> ssid '"name"'
# psk密码
wpa_cli -i wlan0 set_network <network id> psk '"psk"'
# 可以扫描隐藏的AP
wpa_cli -i wlan0 set_network <network id> scan_ssid 1
# 优先级
wpa_cli -i wlan0 set_network <network id> priority 1
```

## (2) 通过 iwlist 扫描

评论一个很老的帖子，但刚刚遇到这个问题，我不相信 `iwlist` 扫描隐藏的AP。请记住，AP名称根本不播放，因此 `iwlist` 无法神奇地找到名称。

相反，您必须在扫描期间传递AP的名称。即你必须积极探索那个隐藏的网络。

所以你的命令应该是这个样子：`iwlist <my_wireless_interface> scan essid <my_fancy_essid>`

虽然我不知道如何在单次扫描扫描多个隐藏 `essid` 的，这应该发现一个隐藏的SSID。

我不得不提这个 - 隐藏的SSID并没有增加安全性。

## 1.5 WEP 连接方法配置说明

### (1) 路由器 WEP 配置说明

WEP

认证类型: 共享密钥 ▼

WEP密钥格式: 十六进制 ▼

密钥选择: WEP密钥

密钥选择	WEP密钥	密钥类型
密钥 1: <input checked="" type="radio"/>	1234567890	64位 ▼
密钥 2: <input type="radio"/>		禁用 ▼
密钥 3: <input type="radio"/>		禁用 ▼
密钥 4: <input type="radio"/>		禁用 ▼

注意: 您选择的WEP加密经常在老的无线网卡上使用, 新的802.11n不支持此加密方式。所以, 如果您选择了此加密方式, 路由器可能工作在较低的传输速率上。建议使用WPA2-PSK等级的AES加密。

保存 帮助

其中

认证类型:

共享密钥&开放认证, 该参数可以不用配置

对应 wpa\_supplicant 配置:

共享密钥: auth\_alg=SHARED

开放认证: auth\_alg=OPEN

WEP 密钥格式:

十六进制&ASCII 码

密钥长度: 64 位, 128 位, 156 位

密钥格式为十六进制, 对应的密钥长度:

64 位: 密码长度有 10 位

128 位: 密码长度有 26 位

156 位: 密码长度有 32 位

密钥格式为 ASCII 码, 对应的密钥长度:

64 位: 密码长度有 5 位

128 位: 密码长度有 13 位

156 位: 密码长度有 16 位

**注意:**

十六进制密钥格式在 wpa\_supplicant.conf 里面不需要加双引号  
ASCII 码密钥格式需要加双引号。

**密码长度是固定的**

### (2) 不同的路由配置, 对应的 wpa\_supplicant.conf 的参数设置



- 1、共享密钥类型，十六进制格式密钥类型 64 位的密码，  
使用的密钥 1，密钥为：1234567890

The screenshot shows the WEP configuration window. The 'Authentication Type' is set to 'Shared Key' and the 'WEP Key Format' is set to 'Hexadecimal'. Under 'Key Selection', 'Key 1' is selected with the value '1234567890' and a '64位' (64-bit) key type. Other keys are either empty or have different values and key types.

对应的 wpa\_supplicant.conf 配置为：

```
[/ext/demo/wifi]## cat wpa.conf
ctrl_interface=/ext/demo/run/wpa_supplicant
ap_scan=1
#network={
#    ssid="123"
#    psk="12345678"
#    ssid="yzh_test_1"
#    psk="12345678"
#}

network={
    ssid="TEST#1"
    key_mgmt=NONE
    wep_key0=1234567890
    wep_tx_keyidx=0
    auth_alg=SHARED
}
```

**注意：**

**wep\_tx\_keyidx=0** 这个参数，该参数代表的是使用哪一组密钥。  
该参数的取值范围：0~3 对应的是 密钥 1~4

- 2、共享密钥类型，十六进制格式密钥类型 152 位的密码，  
使用的密钥 2，密钥为：12345678900987654321123456789009

The screenshot shows the WEP configuration window. The 'Authentication Type' is set to 'Shared Key' and the 'WEP Key Format' is set to 'Hexadecimal'. Under 'Key Selection', 'Key 2' is selected with the value '12345678900987654321123456789009' and a '152位' (152-bit) key type. Other keys are either empty or have different values and key types.

对应的 wpa\_supplicant.conf 配置为：

```

[/ext/demo/wifi]## cat wpa.conf
ctrl_interface=/ext/demo/run/wpa_supplicant
ap_scan=1
#network={
#    ssid="123"
#    psk="12345678"
#    ssid="yzh_test_1"
#    psk="12345678"
#}

network={
    ssid="TEST#1"
    key_mgmt=NONE
    wep_key1=12345678900987654321123456789009
    wep_tx_keyidx=1
    auth_alg=SHARED
}

```

**注意：**

其中 `wep_tx_keyidx` 值发生了变化，此时为 1

- 3、开放认证类型，十六进制格式密钥类型 152 位的密码，  
使用的密钥 2，密钥为：12345678900987654321123456789009

WEP configuration interface showing the following settings:

- 认证类型: 开放系统
- WEP密钥格式: 十六进制
- 密钥选择: WEP密钥
- 密钥 1: 1234567890 (64位)
- 密钥 2: 12345678900987654321123456789009 (152位)
- 密钥 3: 1234567890098 (128位)
- 密钥 4: (禁用)

对应的 `wpa_supplicant.conf` 配置为:

```

[/ext/demo/wifi]## cat wpa.conf
ctrl_interface=/ext/demo/run/wpa_supplicant
ap_scan=1
#network={
#    ssid="123"
#    psk="12345678"
#    ssid="yzh_test_1"
#    psk="12345678"
#}

network={
    ssid="TEST#1"
    key_mgmt=NONE
    wep_key1=12345678900987654321123456789009
    wep_tx_keyidx=1
    auth_alg=OPEN
}

```

注意 `auth_alg` 参数发生了改变

- 4、共享密钥类型，ASCII 码格式密钥类型 152 位的密码，  
使用的密钥 3，密钥为：1234567890123456

WEP configuration interface showing the following settings:

- 认证类型: 共享密钥
- WEP密钥格式: ASCII码
- 密钥选择: WEP密钥
- 密钥 1: 12345 (64位)
- 密钥 2: 1234567890123 (128位)
- 密钥 3: 1234567890123456 (152位)
- 密钥 4: (禁用)

对应的 wpa\_supplicant.conf 配置为:

```
[/ext/demo/wifi]## cat wpa.conf
ctrl_interface=/ext/demo/run/wpa_supplicant
ap_scan=1
#network={
#    ssid="123"
#    psk="12345678"
#    ssid="yzh_test_1"
#    psk="12345678"
#}

network={
    ssid="TEST#1"
    key_mgmt=NONE
    wep_key2="1234567890123456"
    wep_tx_keyidx=2
    auth_alg=SHARED
}
```

注意其中 wpa\_key 那边使用的是添加了双引号的密钥  
同时 wpa\_tx\_keyidx=2

## 1.6 连接 WPA3 加密 AP 方法

### (1) 驱动配置

```
[*] Enable loader driver fast function
[ ] Enable iwpriv some priv func
[*] Enable WPA3 support, but make sure the kernel support sae before enabled
```

### (2) wpa\_supplicant 配置

wpa\_supplicant 需要 2.7 版本以上, 目前使用的是 2.9 版本

需要编译 libnl-3.2.5, openssl-1.1.1, wpa\_supplicant-2.9, 编译以及使用方式见文档《ATBM WIFI 连接 WPA3 加密路由器的使用方法.pdf》

## 1.7 连接管理帧加密的路由器的配置

### (1) 确认路由器为管理帧加密

如果路由器设置为管理帧加密扫描到的加密方式为【WPA2-PSK-SHA256-CCMP】如下:

```
Selected interface wlan0
open file:/tmp/wpa_supplicant_test/wlan0 , client_socket_dir:(null)
bssid / frequency / signal level / flags / ssid
cc:08:fb:92:97:27 2432 -27 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS] \xe5\xb0\x8f\xe4\xbd\x99AP
22:3a:7c:be:6c:4b 2437 -29 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS] altobeam_2.4g
22:3a:7c:be:6c:4d 2437 -32 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS] \x00\x00\x00\x00\x00\x00\x00\x00
58:41:20:b9:e4:d3 2462 -32 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS] AXBIO-C233
62:41:20:b9:e4:d3 2462 -32 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS]
00:0e:1f:87:5d:4b 2472 -35 [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS] tcl_test
80:ea:07:d8:96:34 2437 -45 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS] hm_test
50:fa:84:06:99:cb 2462 -46 [WPA2-PSK-CCMP][ESS] TEST1
7c:b5:9b:f0:7f:b2 2412 -57 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS] TP-LINK_2.4G_7FB2
00:9a:cd:89:d4:5c 2437 -61 [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS] HUAWEI-VK7LNJ
28:3b:82:84:3c:9b 2462 -72 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS] test_ap1
36:f7:16:53:13:ca 2472 -42 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS]
34:f7:16:c3:13:ca 2472 -41 [WPA-PSK-CCMP][WPA2-PSK-CCMP][ESS] eason_wifi
40:b0:76:ca:34:80 2442 -41 [WPA2-PSK-SHA256-CCMP][ESS] test110
cc:2d:21:25:30:b1 2462 -47 [WPA-PSK-CCMP][ESS] \x00\x00
e0:e0:fc:f6:71:d9 2412 -57 [WPA2-PSK-CCMP][ESS] A310KJW_Wi-Fi5
e0:e0:fc:b6:71:d4 2412 -56 [WPA2-PSK-CCMP][ESS] A310KJW
4a:5f:99:2b:6f:77 2437 -60 [WPA2-PSK-CCMP][ESS][P2P] DIRECT-77-HP M104 LaserJet
28:d1:27:64:6c:aa 2462 -58 [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS] fdn_plus
28:6c:07:61:89:63 2417 -55 [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS] B1
94:83:c4:06:11:2c 2462 -61 [WPA2-PSK-CCMP][ESS] 750s
40:31:3c:d3:f8:7b 2452 -59 [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS] A306
```

## (2) wpa\_supplicant 配置修改

- 1、编译 wpa\_supplicant 需要支持 ieee80211w

```
# IEEE 802.11w (management frame protection), also known as PMF
# Driver support is also needed for IEEE 802.11w.
CONFIG_IEEE80211W=y
```

- 2、在 wpa\_supplicant.conf 里面增加配置

```
/customer # cat wpa_supplicant.conf
ctrl_interface=/tmp/wpa_supplicant_test
update_config=1
network={
    ssid="test110"
    psk="12345678"
    scan_ssid=1
    key_mgmt=WPA-PSK WPA-PSK-SHA256
    ieee80211w=1
}
/customer #
/customer #
```

## 1.8 问题分析

### (1) 连接不上路由器

- a) 获取不到 IP

- i. 判断 wpa\_supplicant 是否已经运行
- ii. 判断 WIFI 是否已经正常连接上, 如上图 LOG
  1. 如果没有正常连接上重启 wpa\_supplicant 加上 -d 参数

```
wpa_supplicant -Dnl80211 -i wlan0 -c
/ext/demo/run/wpa_supplicant.conf -d&
```

- iii. 判断 udhcpc 是否已经运行

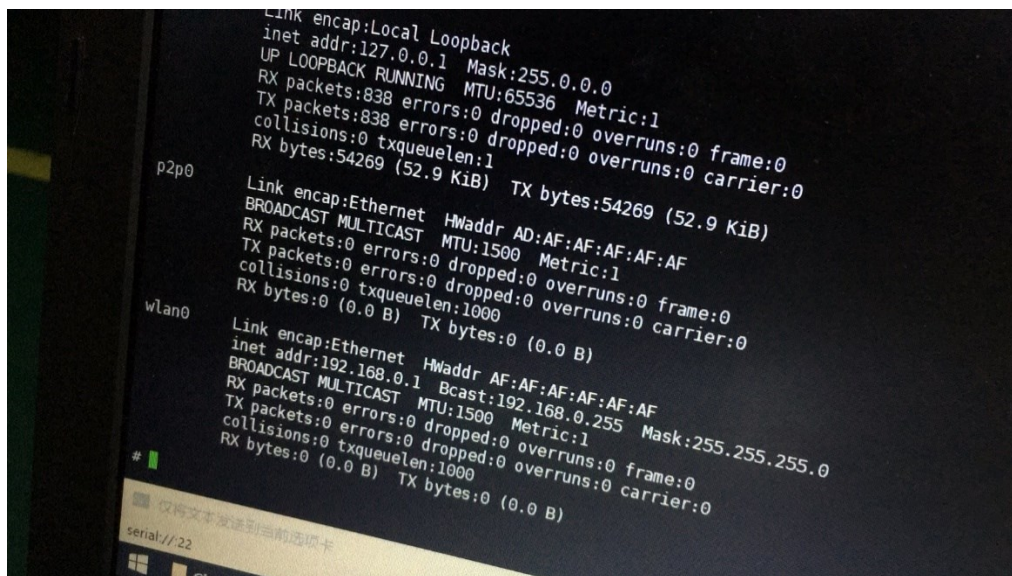
- b) 连接不上 SSID 是中文的路由器, 联系 FAE 拿 patch, 需要修改 wpa\_supplicant 源码

### (2) Ifconfig wlan0 up 不起来

执行 ifconfig wlan0 up 会报错 ifconfig: SIOCSIFHWADDR: Cannot assign requested address

执行 ifconfig -a:





此时的 mac 地址为组播地址。

IP 地址分为三类：广播，组播和单播。

**广播**就是：FF:FF:FF:FF:FF:FF。

**组播**：第一字节最后一位是 1，如 47:72:65:65:6e:00，47 的最后一位是 1。

**单播**：第一字节最后一位是 0，如 48:72:65:65:6e:00。

把上面的地址改为 48 就不会出现这个问题了。

问题深入学习：

IEEE 802 定义 MAC 地址为

```
|<----- 24 bit ----->|<----- 24 bit ----->|
----->|
| ccccccug cccccccc cccccccc | xxxxxxxx xxxxxxxx xxxxxxxx |
```

由 ug 控制 mac 地址类型：

u:

0: 由 IEEE 指定 ID 统一管理

1: 本地管理

g:

0: 单播

1: 多播

也就是 12 位 MAC 地址分为四类，由其中第二位决定

第二位为

0 | 4 | 8 | C : (00) 统一管理的单播 MAC

1 | 5 | 9 | D : (01) 统一管理的多播 MAC

2 | 6 | A | E : (10) 本地管理的单播 MAC

## 3 | 7 | B | F: (11) 本地管理的多播 MAC

```
=====
=====
```

由于针对 ADSL 路由等这样的网络终端，一般使用的都是 统一管理的单播 MAC 所以会判断 02:10:18:01:00:01 或者 (11:01:18:00:00:30) 为无效 MAC，导致无线等功能失效，或者网络连接失败等现象。

而对于 00:25:5E:08:DE:43 这样的 MAC 就被认为是有效的

## (3) Wpa\_supplicant 使用 WEXT 接口，Wpa\_cli 扫描执行一直失败

通过执行 wpa\_cli scan 会打印：

```
ioctl[SIOCSIWSCAN]: Invalid argument
```

并且在初始化 wpa\_supplicant 也发现如下打印：

```
WEXT: SIOCSIWAUTH(param 7 value 0x1) failed: Invalid argument)
```

分析：

由于 ATBM wifi 依赖 cfg80211 模块，wpa\_supplicant 扫描命令会先下发给到 cfg80211，cfg80211 再传给驱动进行扫描。

经过检查发现 wpa\_supplicant 正常绑定 socket ioctl 到 cfg80211，打印如下：

```
8 74 2f 09 09 t/
9 CTRL_IFACE GLOBAL INTERFACE_ADD 'wlan0' wext /var/tmp/wpa_supplicant/
0 Initializing interface 'wlan0' conf 'N/A' driver 'wext' ctrl_interface '/var/tmp/wpa_supplicant/' bridge 'N/A'
1 WEXT: cfg80211-based driver detected
2 WEXT: interface wlan0 phy: phy0
3 rfkill: Cannot open RFKILL control device
4 WEXT: RFKILL status not available
5 f08:00:10:820 info task VSP HS 650 HStask.c:1874!wait eth0 dhcn ok 9
```

但是通过 ioctl 传输参数不正常，所以就怀疑是否是内核的宏没有开完全？

```
CONFIG_WIRELESS=y
CONFIG_WIRELESS_EXT=y
CONFIG_WEXT_CORE=y
CONFIG_WEXT_PROC=y
CONFIG_WEXT_PRIV=y
CONFIG_CFG80211=m
# CONFIG_NL80211_TESTMODE is not set
# CONFIG_CFG80211_DEVELOPER_WARNINGS is not set
# CONFIG_CFG80211_CERTIFICATION_ONUS is not set
CONFIG_CFG80211_DEFAULT_PS=y
# CONFIG_CFG80211_INTERNAL_REGDB is not set
CONFIG_CFG80211_CRDA_SUPPORT=y
# CONFIG_CFG80211_WEXT is not set
# CONFIG_LIB80211 is not set
CONFIG_MAC80211=m
CONFIG_MAC80211_HAS_RC=y
CONFIG_MAC80211_RC_MINSTREL=y
CONFIG_MAC80211_RC_MINSTREL_HT=y
# CONFIG_MAC80211_RC_MINSTREL_VHT is not set
CONFIG_MAC80211_RC_DEFAULT_MINSTREL=y
CONFIG_MAC80211_RC_DEFAULT="minstrel_ht"
# CONFIG_MAC80211_MESH is not set
/WEXT
```

经过检查内核 CONFIG\_CFG80211\_WEXT 宏没开，打开宏以后重新编译 cfg80211 以及 ATBM 驱动就可以正常使用了。

**(4) Wpa\_cli 执行一直报 Failed to connect to non-global ctrl\_ifname:(nil)  
error:No such file or directly**

原因 wpa\_cli 工具找不到可以使用的网口，参数加上 -i wlan0 就好了  
Wpa\_supplicant.conf 内容为：

```
ctrl_interface=/ext/demo/run/wpa_supplicant
update_config=1

network={
    ssid="TP-LINK_64F8"
    psk="12345678"
}
```

wpa\_supplicant 运行命令为：

```
wpa_supplicant -Dnl0211 -iwlan0 -c wpa_supplicant.conf -B
```

wpa\_cli 完整的命令为：

```
wpa_cli -p /ext/demo/run/wpa_supplicant -i wlan0
```

## 2 AP MODE

### 2.1 内核需要打开如下宏用于使能 iwconfig, iwpriv, iwlist 命令

```
CONFIG_CFG80211_WEXT=y
CONFIG_WIRELESS=y
CONFIG_WIRELESS_EXT=y
CONFIG_WEXT_CORE=y
CONFIG_WEXT_PROC=y
CONFIG_WEXT_PRIV=y
```

### 2.2 启动流程

- (1) 加载 `cfg80211.ko` 见 STA MODE 部分
- (2) 加载 `atbm_wifi.ko` 见 STA MODE 部分
- (3) 启动 `hostapd`

一般 `hostapd.conf` 为如下配置: `ssid : atbm6022 password:12345678`

```
interface=p2p0
#bridge=br0
ctrl_interface=/ext/demo/run/hostapd/
ctrl_interface_group=0
driver=nl80211
ieee80211n=1
ssid=ATBM_WIFI_AP
hw_mode=g
channel=9
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=12345678
wpa_key_mgmt=WPA-PSK
wpa_pairwise=CCMP
rsn_pairwise=CCMP
#ht_capab=[HT40-]
#beacon_int=200
wpa_group_rekey=0
max_num_sta=4
```

设置为 HT40 参数配置如下:

```
ht_capab=[HT40-][HT40+]
```

HT40 配置说明:



chan/freq	ht_capab
1~4	[HT40+]
5~7	[HT40+]或者[HT40-]或者[HT40+][HT40-]
7~13	[HT40-]
2380	[HT40-]
2504	[HT40+]

需要设置为不加密修改 `wpa=0` 即可

运行 `hostapd`:

```
hostapd -B /ext/demo/run/hostapd.conf &
```

需要调试加上 `-d`: debug 的意思

```
hostapd -B /ext/demo/run/hostapd.conf -d&
```

#### (4) 启动 `dnsmasq/dhcpd` 二选一

##### 2.2.4.1 `dnsmasq` 方式

```
dnsmasq interface=wlan0 --no-daemon --no-resolv --no-poll  
--dhcp-range=192.168.100.2,192.168.100.254,12h  
--dhcp-authoritative --no-ping &
```

新增的两个参数说明:

```
--dhcp-authoritative 不同同一网段, dnsmasq 直接发送 NACK, 连接速度快  
--no-ping 在发送 dhcp offer 不会发送 ICMP 报文, 连接速度加快
```

#### 【注意】

这么配置不需要使用 `dnsmasq.conf`, 在系统中找到 `dnsmasq.conf` 直接删除。

##### 2.2.4.2 `udhcpd` 方式

```
ifconfig wlan0 172.14.10.1  
udhcpd -S udhcpd.conf
```

`udhcpd.conf`:

```
option domain local  
option lease 864000  
max_leases 52  
start 172.14.10.2  
end 172.14.10.254  
interface wlan0  
opt dns 0.0.0.0  
option subnet 255.255.255.0  
opt router 172.14.10.1
```

如果需要让手机连接上 **wifi** 同时还使用手机的 **4G** 网络:

udhcpd.conf:

```
option domain local
option lease 864000
max_leases 52
start 172.14.10.2
end 172.14.10.254
interface wlan0
```

## 2.3 WEP 加密方式配置

### (1) WPA/HT 相关的全部拿掉

```
#ieee80211n=1

#wpa=2

#wpa_passphrase=12345678

#wpa_key_mgmt=WPA-PSK WPA-PSK-SHA256

#wpa_pairwise=TKIP CCMP

#rsn_pairwise=CCMP

#ieee80211w=2

#group_mgmt_cipher=AES-128-CMAC

#peerkey=1

#ht_capab=[HT40- ]

#wpa_group_rekey=0
```

### (2) WEP 加密配置方式

Hostapd.conf:

```
interface=p2p0

ctrl_interface=/tmp/hostapd

ctrl_interface_group=0

driver=nl80211

ssid=IPC_123456

hw_mode=g
```

```
channel=1
macaddr_acl=0
ignore_broadcast_ssid=0
#wep config start
auth_algs=1
wep_default_key=0
wep_key0="12345"
#wep config end
```

### (3) WEP 配置说明

**auth\_algs = 1 opensyskey 开放认证方式**

**auth\_algs = 2 shared key 共享秘钥方式**

wep\_default\_key 取值范围为: 0~3

对应下面四组 key:

```
#wep_key0=123456789a
#wep_key1="vwxyz"
#wep_key2=0102030405060708090a0b0c0d
#wep_key3=".2.4.6.8.0.23"
```

如果 wep\_default\_key=0, 代表使用 wep\_key0

如果 wep\_default\_key=1, 代表使用 wep\_key1

如果 wep\_default\_key=2, 代表使用 wep\_key2

如果 wep\_default\_key=3, 代表使用 wep\_key3

wep\_key 可以配置的长度有限制:

如果配置为 16 进制那么长度有: 10 , 26 , 32 三个长度

如果配置为字符串那么长度有: 5 , 13 , 16 三个长度

#### 2.3.3.1 举例说明

wep\_default\_key=0 代表使用 wep\_key0 有效

**wep\_key0 配置为 16 进制 10 位密码:**

**wep\_key0=123456789a**

**sta 端的配置:**

**wpa\_supplicant.conf:**

```
network={  
    ssid="IPC_123456"  
    ket_mgmt=NONE  
    auth_algs=OPEN # OPEN 开放认证方式 : SHARED 共享秘钥方式  
    wep_tx_keyidx=0  
    wep_key0=123456789a  
}
```

**wep\_key0 配置为 16 进制 26 位密码:**

**wep\_key0=1234567890123456789abcdef**

**sta 端的配置:**

**wpa\_supplicant.conf:**

```
network={  
    ssid="IPC_123456"  
    ket_mgmt=NONE  
    auth_algs=OPEN # OPEN 开放认证方式 : SHARED 共享秘钥方式  
    wep_tx_keyidx=0  
    wep_key0=1234567890123456789abcdef  
}
```

**wep\_key0 配置为 16 进制 32 位密码:**

**wep\_key0=123456789012345678901234567890123456**

**sta 端的配置:**

**wpa\_supplicant.conf:**

```
network={  
    ssid="IPC_123456"  
    ket_mgmt=NONE
```

```
auth_algs=OPEN # OPEN 开放认证方式 : SHARED 共享秘钥方式

wep_tx_keyidx=0

wep_key0=123456789012345678901234567890123456

}
```

**wep\_key0 配置为字符串 5 位密码:**

```
wep_key0="12345"
```

**sta 端的配置:**

**wpa\_supplicant.conf:**

```
network={

    ssid="IPC_123456"

    ket_mgmt=NONE

    auth_algs=OPEN # OPEN 开放认证方式 : SHARED 共享秘钥方式

    wep_tx_keyidx=0

    wep_key0="12345"

}
```

**wep\_key0 配置为字符串 10 位密码:**

```
wep_key0="1234567890"
```

**sta 端的配置:**

**wpa\_supplicant.conf:**

```
network={

    ssid="IPC_123456"

    ket_mgmt=NONE

    auth_algs=OPEN # OPEN 开放认证方式 : SHARED 共享秘钥方式

    wep_tx_keyidx=0

    wep_key0="1234567890"

}
```

**wep\_key0 配置为字符串 16 位密码:**

```
wep_key0="123456789abcdef"
```

**sta 端的配置:**

```
wpa_supplicant.conf:

network={

    ssid="IPC_123456"

    ket_mgmt=NONE

    auth_algs=OPEN # OPEN 开放认证方式 : SHARED 共享秘钥方式

    wep_tx_keyidx=0

    wep_key0="123456789abcdef"

}
```

## 2.4 WPA3 加密方式配置

hostapd 需要 2.7 版本以上，目前使用的是 2.9 版本

需要编译 libnl-3.2.5,openssl-1.1.1,hostapd-2.9,编译以及使用方式见文档  
《ATBM WIFI 连接 WPA3 加密路由器的使用方法.pdf》

## 2.5 注意事项

### (1) hostapd 启动失败

#### a) 没法设置 11 以上的信道

无法设置 11 信道是因为 cfg80211 做了限制导致，需要修改 cfg80211。

修改方式如下：

在 kernel/net/wireless/reg.c 237 行左右的位置

注释掉第一个箭头指向的枚举变量， 可以将 hostapd 的信道设置为 12/13

注释掉第二个箭头指向的枚举变量，可以将 hostapd 的信道设置为 14  
框框里面需要注意都为 40，代表 cfg80211 支持 HT40

```

/* We keep a static world regulatory domain in case of the absence of CRDA */
static const struct ieee80211_regdomain world_regdom = {
    .n_reg_rules = 5,
    .alpha2 = "00",
    .reg_rules = {
        /* IEEE 802.11b/g, channels 1..11 */
        REG_RULE(2412-10, 2462+10, 40, 6, 20, 0),
        /* IEEE 802.11b/g, channels 12..13 */
        REG_RULE(2467-10, 2472+10, 40, 6, 20,
            #if 0
                NL80211_RRF_NO_IR
            #endif
            0),
        /* IEEE 802.11 channel 14 - Only JP enables
         * this and for 802.11b only */
        REG_RULE(2484-10, 2484+10, 20, 6, 20,
            #if 0
                NL80211_RRF_NO_IR |
                NL80211_RRF_NO_OFDM
            #endif
            0),
        /* IEEE 802.11a, channel 36..48 */
        REG_RULE(5180-10, 5240+10, 160, 6, 20,
            NL80211_RRF_NO_IR),
        /* IEEE 802.11a, channel 52..64 - DFS required */
        REG_RULE(5260-10, 5320+10, 160, 6, 20,
            NL80211_RRF_NO_IR |
            NL80211_RRF_DFS),
        /* IEEE 802.11a, channel 100..144 - DFS required */
        REG_RULE(5500-10, 5720+10, 160, 6, 20,
            NL80211_RRF_NO_IR |
            NL80211_RRF_DFS),
    }
};

```

b) Hostapd 为 14 信道只能运行 g 模式

需要修改 hostapd 源码，对应源码位置如下：

hostapd/src/ap/ hw\_features.c 按照下图注释修改

```

int hostapd_select_hw_mode(struct hostapd_iface *iface)
{
    int i;

    if (iface->num_hw_features < 1)
        return -1;

    if ((iface->conf->hw_mode == HOSTAPD_MODE_IEEE80211G ||
        iface->conf->ieee80211n || iface->conf->ieee80211ac) &&
        iface->conf->channel == 14) {
        wpa_printf(MSG_INFO, "Disable OFDM/HT/VHT on channel 14");
        //iface->conf->hw_mode = HOSTAPD_MODE_IEEE80211B;
        //iface->conf->ieee80211n = 0;
        iface->conf->ieee80211ac = 0;
    }

    iface->current_mode = NULL;
    for (i = 0; i < iface->num_hw_features; i++) {
        struct hostapd_hw_modes *mode = &iface->hw_features[i];
        if (mode->mode == iface->conf->hw_mode) {
            iface->current_mode = mode;
            break;
        }
    }
}

```

## (2) 客户端连接不上热点

出现如下的错误：

```

103 dnsmasq: failed to load names from /etc/hosts: No such file or directory
104 Using interface p2p0 with hwaddr 00:12:34:4d:24:99 and ssid "atbm_6022"
105 random: Cannot read from /dev/random: Resource temporarily unavailable
106 random: Only 0/20 bytes of strong random data available from /dev/random
107 random: Not enough entropy pool available for secure operations
108 WPA: Not enough entropy in random pool for secure operations - update keys later when the first station connects
109 atbm_upload_beacon:atbm_clear_wpas_p2p_40M_ie
110 atbm_clear_wpas_p2p_40M_ie:filter index(0)
111 atbm_clear_wpas_p2p_40M_ie:htcap_ie == NULL,len(85),pkg_len(121)
112 [BH] wakeup.
113 [BH] wakeup.
114 [BH] wakeup.
115 atbm_start_ap:start.channel_type,(1),channelNumber(1)
116 [BH] wakeup.
117 [BH] wakeup.

```

说明是获取随机数失败:

具体分析过程请看:《hostapd 启动后有时候密码对了还是会连接不上\_解决方案.docx》

解决方法有两种:

1.在运行 hostapd /wpa\_supplicant 直接执行如下命令

```
rm -f /dev/random
```

```
ln -s /dev/urandom /dev/random
```

2.修改 hostapd 源码:

src/crypto/random.c 下图位置

将

```
fd = open("/dev/random",O_RDONLY|O_NONBLOCK)
```

修改为

```
fd = open("/dev/urandom",O_RDONLY|O_NONBLOCK)
```

```

225:     if (dummy_key_avail == sizeof(dummy_key))
226:         return 1; /* Already initialized - good to continue */
227:
228:     /*
229:      * Try to fetch some more data from the kernel high quality
230:      * /dev/random. There may not be enough data available at this point,
231:      * so use non-blocking read to avoid blocking the application
232:      * completely.
233:      */
234:     fd = open("/dev/random", O_RDONLY | O_NONBLOCK);
235:     //fd = open("/dev/urandom", O_RDONLY | O_NONBLOCK);
236:     if (fd < 0) {
237:         wpa_printf(MSG_ERROR, "random: Cannot open /dev/random: %s",
238:             strerror(errno));
239:         return -1;
240:     }
241:
242:     res = read(fd, dummy_key + dummy_key_avail,
243:         sizeof(dummy_key) - dummy_key_avail);
244:     if (res < 0) {
245:         wpa_printf(MSG_ERROR, "random: Cannot read from /dev/random: "
246:             "%s", strerror(errno));
247:         res = 0;
248:     }
249:     wpa_printf(MSG_DEBUG, "random: Got %u/%u bytes from "
250:         "/dev/random", (unsigned) res,
251:         (unsigned) (sizeof(dummy_key) - dummy_key_avail));
252:     dummy_key_avail += res;
253:     close(fd);
254:
255:     if (dummy_key_avail == sizeof(dummy_key)) {
256:         if (own_pool_ready < MIN_READY_MARK)
257:             own_pool_ready = MIN_READY_MARK;
258:         random_write_entropy();

```

### (3) 设置 n 模式, HT40

第一步,在编译源码的时候要确认 hostapd 支持高速率 CONFIG\_IEEE80211N=y

第二步,配置 hostapd.conf 增加如下参数:

```
ieee80211n=1
```

```
ht_capab=[HT40-]
```

参数说明如下:



```
##### IEEE 802.11n related configuration #####
# ieee80211n: Whether IEEE 802.11n (HT) is enabled
# 0 = disabled (default)
# 1 = enabled
# Note: You will also need to enable WMM for full HT functionality.
# Note: hw_mode=g (2.4 GHz) and hw_mode=a (5 GHz) is used to specify the band.
#ieee80211n=1

# ht_capab: HT capabilities (list of flags)
# LDPC coding capability: [LDPC] = supported
# Supported channel width set: [HT40-] = both 20 MHz and 40 MHz with secondary
#   channel below the primary channel; [HT40+] = both 20 MHz and 40 MHz
#   with secondary channel above the primary channel
#   (20 MHz only if neither is set)
# Note: There are limits on which channels can be used with HT40- and
# HT40+. Following table shows the channels that may be available for
# HT40- and HT40+ use per IEEE 802.11n Annex J:
# freq      HT40-    HT40+
# 2.4 GHz   5-13     1-7 (1-9 in Europe/Japan)
# 5 GHz     40,48,56,64 36,44,52,60
# (depending on the location, not all of these channels may be available
# for use)
# Please note that 40 MHz channels may switch their primary and secondary
# channels if needed or creation of 40 MHz channel maybe rejected based
# on overlapping BSSes. These changes are done automatically when hostapd
# is setting up the 40 MHz channel.
# Spatial Multiplexing (SM) Power Save: [SMPS-STATIC] or [SMPS-DYNAMIC]
#   (SMPS disabled if neither is set)
# HT-greenfield: [GF] (disabled if not set)
# Short GI for 20 MHz: [SHORT-GI-20] (disabled if not set)
# Short GI for 40 MHz: [SHORT-GI-40] (disabled if not set)
# Tx STBC: [TX-STBC] (disabled if not set)
# Rx STBC: [RX-STBC1] (one spatial stream), [RX-STBC12] (one or two spatial
#   streams), or [RX-STBC123] (one, two, or three spatial streams); Rx STBC
#   disabled if none of these set
# HT-delayed Block Ack: [DELAYED-BA] (disabled if not set)
# Maximum A-MSDU length: [MAX-AMSDU-7935] for 7935 octets (3839 octets if not
#   set)
# DSSS/CCK Mode in 40 MHz: [DSSS-CCK-40] = allowed (not allowed if not set)
# 40 MHz intolerant [40-INTOLERANT] (not advertised if not set)
# L-SIG TXOP protection support: [LSIG-TXOP-PROT] (disabled if not set)
#ht_capab=[HT40-][SHORT-GI-20][SHORT-GI-40]
```

第三步，这一步看情况执行，如果没法运行起来 HT40，在修改 hostapd 源码，Hostapd/src/ap/hw\_features.c 增加如下内容

```
static void ieee80211n_check_scan(struct hostapd_iface *iface)
{
    struct wpa_scan_results *scan_res;
    int oper40;
    int res;
    int fix40 = 1;

    /* Check list of neighboring BSSes (from scan) to see whether 40 MHz is
     * allowed per IEEE Std 802.11-2012, 10.15.3.2 */

    iface->scan_cb = NULL;

    scan_res = hostapd_driver_get_scan_results(iface->bss[0]);
    if (scan_res == NULL) {
        hostapd_setup_interface_complete(iface, 1);
        return;
    } else if (fix40) {
        wpa_scan_results_free(scan_res);
        hostapd_setup_interface_complete(iface, 0);
        return;
    }

    if (iface->current_mode->mode == HOSTAPD_MODE_IEEE80211A)
        oper40 = ieee80211n_check_40mhz_5g(iface, scan_res);
    else
        oper40 = ieee80211n_check_40mhz_2g4(iface, scan_res);
}
```

#### (4) 客户端获取不到 IP

第一步，确认 dnsmasq/udhcpd 绑定的 IP 地址是否和 ifconfig wifi 接口的 IP 地址在同一个网段

第二步，如果使用的 /etc/dnsmasq.conf 【 /etc/udhcpd.conf 】，注意 dnsmasq.conf 【udhcpd.conf】 内容不要和 dnsmasq 【udhcpd】 运行参数重复，特别是 IP 网段设置那一块。

### (5) nl80211:set key fail error = -22

```

250 16:37:29:424 Get randomness: len=16 entropy=0
251 16:37:29:286 random from os_get_random - hexdump(len=16): [REMOVED]
252 16:37:29:291 random mix_pool - hexdump(len=20): [REMOVED]
253 16:37:29:343 random from internal pool - hexdump(len=16): [REMOVED]
254 16:37:29:343 mixed random - hexdump(len=16): [REMOVED]
255 16:37:29:343 GTK - hexdump(len=32): [REMOVED]
256 16:37:29:343 WPA: group state machine entering state SETKEYSDONE (VLAN-ID 0)
257 16:37:29:343 wpa_driver_nl80211_set_key: ifindex=7 (p2p0) alg=2 addr=0x4b760 key_idx=1 set_tx=1 seq_len=0 key_len=32
258 16:37:29:343 nl80211: KEY_DATA - hexdump(len=32): [REMOVED]
259 16:37:29:343 broadcast key
260 16:37:29:343 nl80211: set_key failed; err=-22 Invalid argument)
261 16:37:29:344 WPA: group state machine entering state FATAL_FAILURE
262 16:37:29:344 p2p0: Flushing old station entries
263 16:37:29:344 nl80211: flush -> DEL_STATION p2p0 (all)
264 16:37:29:344 p2p0: Deauthenticate all stations
265 16:37:29:344 nl80211: send_mframe - da= ff:ff:ff:ff:ff:ff noack=0 freq=0 no_cck=0 offchanok=0 wait_time=0 fc=0xc0 (WLAN_FC:
266 16:37:29:344 nl80211: send_mframe -> send_frame
267 16:37:29:346 nl80211: send_frame - Use bss->freq=2437
268 16:37:29:357 nl80211: send_frame(freq=2437 bss->freq=2437) -> send_monitor
269 16:37:29:365 wpa_driver_nl80211_set_key: ifindex=7 (p2p0) alg=0 addr=(nil) key_idx=0 set_tx=0 seq_len=0 key_len=0
270 16:37:29:380 wpa_driver_nl80211_set_key: ifindex=7 (p2p0) alg=0 addr=(nil) key_idx=1 set_tx=0 seq_len=0 key_len=0
271 16:37:29:381 wpa_driver_nl80211_set_key: ifindex=7 (p2p0) alg=0 addr=(nil) key_idx=2 set_tx=0 seq_len=0 key_len=0
272 16:37:29:395 wpa_driver_nl80211_set_key: ifindex=7 (p2p0) alg=0 addr=(nil) key_idx=3 set_tx=0 seq_len=0 key_len=0
273 16:37:29:396 hostapd_free_hapd_data(p2p0)

```

解决方法:

打开内核的.config 配置的宏: CONFIG\_CRYPTO\_ARC4=y

> Are you sure your kernel has CONFIG\_CRYPTO\_ARC4=y?

Indeed, the arc4 module is enabled as a module in the kernel but not deployed in my filesystem.

I added it and my error message is gone. I will test on several WiFi routers tonight.

Question:

How could I have know that this module is needed? I see not dependencies to the arc4 crypto. What did I miss?

I now needed to get deep into the source code to find the origin.

Are there any more crypto's I need for other routers?

--

Met vriendelijke groet / With kind regards,

Richard Knoop ([richard.knoop@ibb.nl](mailto:richard.knoop@ibb.nl)), Ingenieursburo Balvers BV

Tel +31 72 576 2552

Newtonstraat 27, 1704SB Heerhugowaard, Netherlands

### (6) hostapd.conf 设置隐藏 ssid 导致驱动挂掉

挂掉 log 如下:

```

random from internal pool - hexdump(len=16): [REMOVED]
mixed random - hexdump(len=32): [REMOVED]
GMK - hexdump(len=32): [REMOVED]
Get randomness[TEL add_start
atbm_upload_beacon: hidden ssid with ssid len 0 not supported
-----[ cut here ]-----
: len=32 entropyWARNING: CPU: 0 PID: 102 at /usr/lchome/yuzhihuang/Mstar/IPC_I
Modules linked in: atbm602x_wifi_usb(0) cfg80211 usb_storage ehci_hcd usbcore
CPU: 0 PID: 102 Comm: hostapd Tainted: G      O 3.18.30 #27
[<c000faad>] (unwind_backtrace) from [<c000e2e3>] (show_stack+0xb/0xc)
[<c000e2e3>] (show_stack) from [<c0015c71>] (warn_slowpath_common+0x45/0x60)
[<c0015c71>] (warn_slowpath_common) from [<c0015ce3>] (warn_slowpath_null+0xf/0x10)
random from [<c0015ce3>] (warn_slowpath_null) from [<bf939f45>] (atbm_bss_info
os_get_random - [<bf939f45>] (atbm_bss_info_changed [atbm602x_wifi_usb]) from
hexdump(len=32): [<bf911e01>] (ieee80211_bss_info_change_notify [atbm602x_wifi_
[REMOVED]
rand[<bf9206b9>] (ieee80211_config_beacon [atbm602x_wifi_usb]) from [<bf921711:
om_mlx_pool - he[<bf921711>] (ieee80211_start_ap [atbm602x_wifi_usb]) from [<b
[<bf8f88fb>] (nl80211_start_ap [cfg80211]) from [<c019a88f>] (genl_rcv_msg+0x1
[<c019a88f>] (genl_rcv_msg) from [<c019a179>] (netlink_rcv_skb+0x2f/0x64)
[<c019a179>] (netlink_rcv_skb) from [<c019a6ff>] (genl_rcv+0x15/0x26)
[<c019a6ff>] (genl_rcv) from [<c019a453>] (netlink_unlink+0x1b/0x15a)

```

原因:

```

# Send empty SSID in beacons and ignore probe request frames that do not
# specify full SSID, i.e., require stations to know SSID.
# default: disabled (0)
# 1 = send empty (length=0) SSID in beacon and ignore probe request for
# broadcast SSID
# 2 = clear SSID (ASCII 0), but keep the original length (this may be required
# with some clients that do not support empty SSID) and ignore probe
# requests for broadcast SSID
ignore_broadcast_ssid=0

```

设置隐藏 ssid 的参数是 ignore\_broadcast\_ssid。

上图是 hostapd.conf 对参数的说明, 设置成 1, 传给驱动的参数 ssid 长度直接为 0, 而 ATBM 驱动如果遇到传入的 ssid\_length=0, 会导致驱动判断为出错。

所以需要将这个参数配置为 2, 只是将 beacon 包中的 ssid 字段写 0, ssid\_length 还是不变的, 这样子驱动才能正常工作。

## (7) 执行了命令, hostapd 重载配置无效

Hostapd 重定义了信号的执行函数:

```

...
#ifdef CONFIG_NATIVE_WINDOWS
...
eloop_register_signal(SIGHUP, handle_reload, interfaces);
eloop_register_signal(SIGUSR1, handle_dump_state, interfaces);
#endif /* CONFIG_NATIVE_WINDOWS */
eloop_register_signal_terminate(handle_term, interfaces);

#ifdef CONFIG_NATIVE_WINDOWS
openlog("hostapd", 0, LOG_DAEMON);
#endif /* CONFIG_NATIVE_WINDOWS */
...

```

重载配置的命令为:

```
Kill(hostapd_pid, SIGHUP);
```

Hostapd\_pid : hostapd 运行起来的进程号

修改 src/ap/hostapd.c 文件的 hostapd\_reload\_config 函数:

```

175:         return -1;
176:
177:     hostapd_clear_old(iface);
178:
179:     oldconf = hapd->iconf;
180:     iface->conf = newconf;
181:
182:     for (j = 0; j < iface->num_bss; j++) {
183:         hapd = iface->bss[j];
184:         hapd->iconf = newconf;
185:
186:         /*
187:          * Copy old configuration to new configuration
188:          */
189:         hapd->iconf->channel = oldconf->channel;
190:         hapd->iconf->acs = oldconf->acs;
191:         hapd->iconf->secondary_channel = oldconf->secondary_channel;
192:         hapd->iconf->ieee80211n = oldconf->ieee80211n;
193:         hapd->iconf->ieee80211ac = oldconf->ieee80211ac;
194:         hapd->iconf->ht_capab = oldconf->ht_capab;
195:         hapd->iconf->vht_capab = oldconf->vht_capab;
196:         hapd->iconf->vht_oper_chwidth = oldconf->vht_oper_chwidth;
197:         hapd->iconf->vht_oper_central_freq_seg0_idx =
198:             oldconf->vht_oper_central_freq_seg0_idx;
199:         hapd->iconf->vht_oper_central_freq_seg1_idx =
200:             oldconf->vht_oper_central_freq_seg1_idx;
201:
202:         /*
203:          * Copy new configuration to new configuration
204:          */
205:         hapd->iconf->channel = newconf->channel;
206:         hapd->iconf->acs = newconf->acs;
207:         hapd->iconf->secondary_channel = newconf->secondary_channel;
208:         hapd->iconf->ieee80211n = newconf->ieee80211n;
209:         hapd->iconf->ieee80211ac = newconf->ieee80211ac;
210:         hapd->iconf->ht_capab = newconf->ht_capab;
211:         hapd->iconf->vht_capab = newconf->vht_capab;
212:         hapd->iconf->vht_oper_chwidth = newconf->vht_oper_chwidth;
213:         hapd->iconf->vht_oper_central_freq_seg0_idx =
214:             newconf->vht_oper_central_freq_seg0_idx;
215:         hapd->iconf->vht_oper_central_freq_seg1_idx =
216:             newconf->vht_oper_central_freq_seg1_idx;
217:
218:         hapd->conf = newconf->bss[j];
219:         hostapd_reload_bss(hapd);
220:     }
221:     /* end for j=0;j<iface->num_bss;... */
222:
223:     hostapd_config_free(oldconf);
224: }

```

## (8) hostapd 运行起来，手机连接上了但是没法建立 TCP 连接

并且出现如下错误：

```

11  * listen_interval=10
12  * flags set=0x6 mask=0x6
13  p2p0: STA b0:e1:7e:13:af:a7 IEEE 802.11: binding station to interface 'p2p0'
14  nl80211: p2p0[4]: set_sta_vlan(b0:e1:7e:13:af:a7, ifname=p2p0[4], vlan_id=0)
15  nl80211: Set STA flags - ifname=p2p0 addx=b0:e1:7e:13:af:a7 total_flags=0x5 flags_or=0x5 flags_and=0xffffffff5 authorize
16  p2p0: STA b0:e1:7e:13:af:a7 RADIUS: starting accounting session 03708603-00000000
17  IEEE 802.1X: Ignore STA - 802.1X not enabled or forced for WPA
18  hostapd_new_assoc_sta: reschedule ap_handle_timer timeout for b0:e1:7e:13:af:a7 (1 seconds - ap_max_inactivity)
19  nl80211: Event message available
20  nl80211: RX Event 19 (NL80211_CMD_NEW_STATION) received for p2p0
21  nl80211: New station b0:e1:7e:13:af:a7
22  ap_handle_timer: b0:e1:7e:13:af:a7 flags=0x80a3 timeout_next=0
23  p2p0: Station b0:e1:7e:13:af:a7 has been active 0s ago
24  ap_handle_timer: register ap_handle_timer timeout for b0:e1:7e:13:af:a7 (4 seconds)
25  ieee80211 phy0: Multicast delivery timeout.
26  dnsmasq dnsmasq: DHCPDISCOVER(p2p0) b0:e1:7e:13:af:a7
27  dnsmasq dnsmasq: DHCPDISCOVER(p2p0) 192.168.0.131 b0:e1:7e:13:af:a7
28  dnsmasq dnsmasq: DHCPDISCOVER(p2p0) b0:e1:7e:13:af:a7
29  dnsmasq dnsmasq: DHCPDISCOVER(p2p0) 192.168.0.131 b0:e1:7e:13:af:a7
30  dnsmasq dnsmasq: DHCPDISCOVER(p2p0) b0:e1:7e:13:af:a7
31  dnsmasq dnsmasq: DHCPDISCOVER(p2p0) 192.168.0.131 b0:e1:7e:13:af:a7
32  dnsmasq dnsmasq: DHCPREQUEST(p2p0) 192.168.0.131 b0:e1:7e:13:af:a7
33  dnsmasq dnsmasq: DHCPREQUEST(p2p0) 192.168.0.131 b0:e1:7e:13:af:a7
34  dnsmasq dnsmasq: DHCPACK(p2p0) 192.168.0.131 b0:e1:7e:13:af:a7 HUWEI_Mate_10_Pro-513400
35  ap_handle_timer: b0:e1:7e:13:af:a7 flags=0x80a3 timeout_next=0
36  p2p0: Station b0:e1:7e:13:af:a7 has been active 0s ago
37  ap_handle_timer: register ap_handle_timer timeout for b0:e1:7e:13:af:a7 (7 seconds)
38  ap_handle_timer: b0:e1:7e:13:af:a7 flags=0x80a3 timeout_next=0
39  p2p0: Station b0:e1:7e:13:af:a7 has been inactive too long: 1 sec, max allowed: 1
40  Polling STA
41  nl80211: send_mlm - da= b0:e1:7e:13:af:a7 noack=0 freq=0 no_ock=0 offchanack=0 wait_time=0 fc=0x248 (WLAN_FC_STYPE_NULL)

```

查看了 hostapd.conf 配置如下：

```

interface=p2p0
ctrl_interface=/var/run/hostapd
ctrl_interface_group=0
driver=nl80211
ssid=lyx
hw_mode=g
channel=11
macaddr_acl=0
auth_algs=1
ignore_broadcast_ssid=0

```

```
rsn_pairwise=CCMP
ap_max_inactivity=1
```

该配置缺少，加密类型以及密码就配置上加密算法。

正常配置如下：

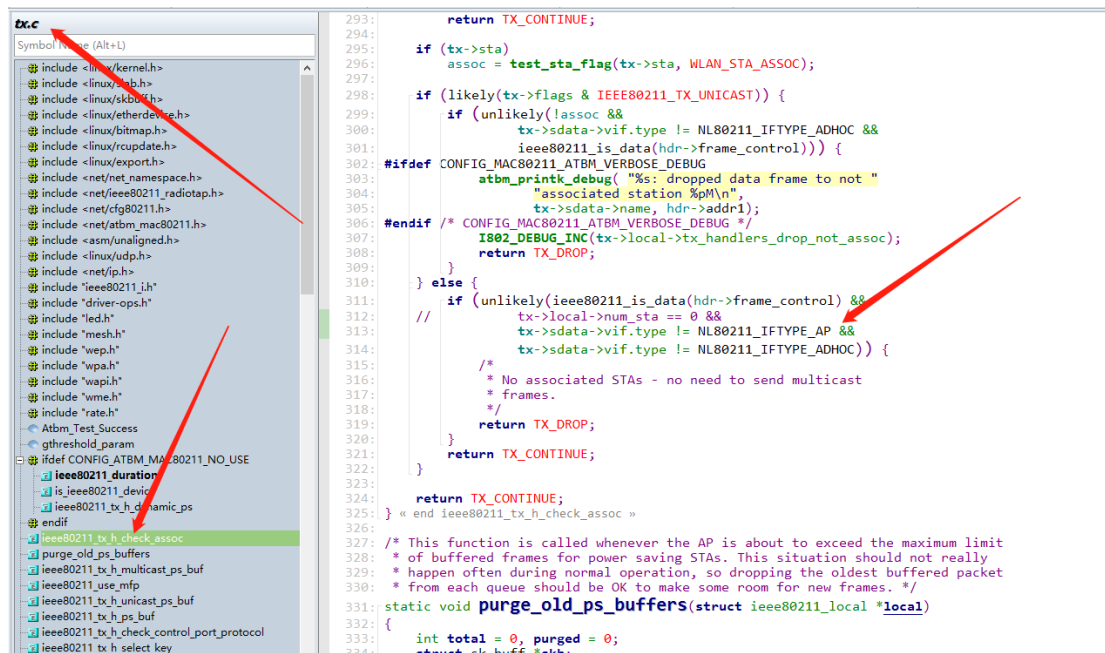
```
interface=wlan0
#bridge=br0
ctrl_interface=/tmp/hostapd/
ctrl_interface_group=0
driver=nl80211
ieee80211n=1

#ht_capab=[HT40+]
#beacon_int=200
ssid=Sstar_ATBM_TEST
hw_mode=g
channel=1
macaddr_acl=0
auth_algs=1
wpa_group_rekey=0
ignore_broadcast_ssid=0
wpa=2
wpa_passphrase=12345678
wpa_key_mgmt=WPA-PSK
wpa_pairwise=CCMP
rsn_pairwise=CCMP
```

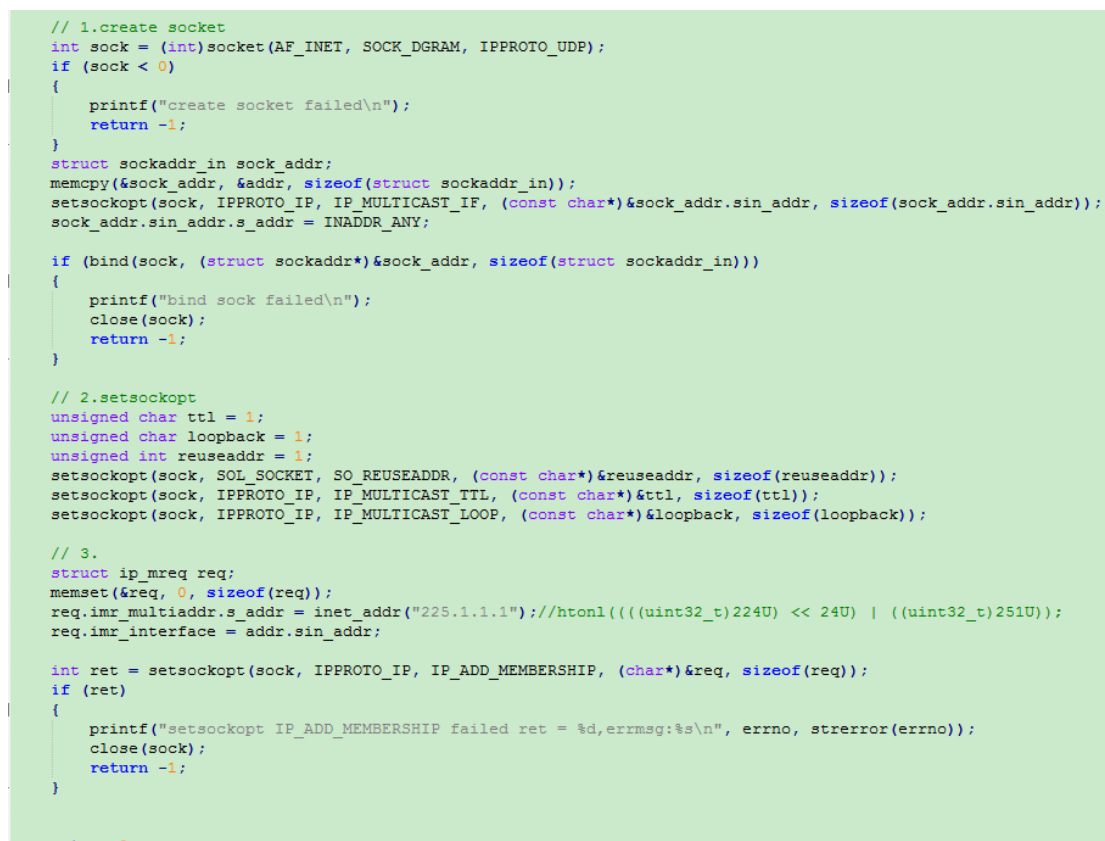
### (9) 绑定组播地址失败

旧版本的驱动，ap 模式下不支持广播，所以需要先确认下驱动是否已经修改过了。

在 hal\_apollo/mac80211/tx.c 里面，修改为如下：



如果确认已经修改好使了，在接着往下看。



一直在 IP\_ADD\_MEMBERSHIP 这一步设置失败：

setsockopt IP\_ADD\_MEMBERSHIP failed ret = 19,errmsg:No such device

```
[root@anyka /etc/config]$
[root@anyka /etc/config]$ /tmp/boardcast
setsockopt IP_ADD_MEMBERSHIP failed ret = 19,errmsg:No such device
```

经过检查发现没有 route 网关：

```
[root@anyka /etc/config]$ route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.40.0 0.0.0.0 255.255.255.0 U 0 0 0 wlan0
[root@anyka /etc/config]$ ifconfig wlan0 192.168.40.1
```

给设置上路由网关:

```
route add default gw 192.168.40.1
```

```
[root@anyka /etc/config]$
[root@anyka /etc/config]$ route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
0.0.0.0 192.168.40.1 0.0.0.0 UG 0 0 0 wlan0
192.168.40.0 0.0.0.0 255.255.255.0 U 0 0 0 wlan0
[root@anyka /etc/config]$
[root@anyka /etc/config]$
```

在执行就好使了:

```
[root@anyka /etc/config]$
[root@anyka /etc/config]$
[root@anyka /etc/config]$ /tmp/boardcast
[root@anyka /etc/config]$
[root@anyka /etc/config]$
```

### 3 BRIDGE MDOE

#### 3.1 内核需要打开如下宏用于使能 iwconfig, iwpriv, iwlist 命令

```
CONFIG_CFG80211_WEXT=y
CONFIG_WIRELESS=y
CONFIG_WIRELESS_EXT=y
CONFIG_WEXT_CORE=y
CONFIG_WEXT_PROC=y
CONFIG_WEXT_PRIV=y
```

#### 3.2 功能说明

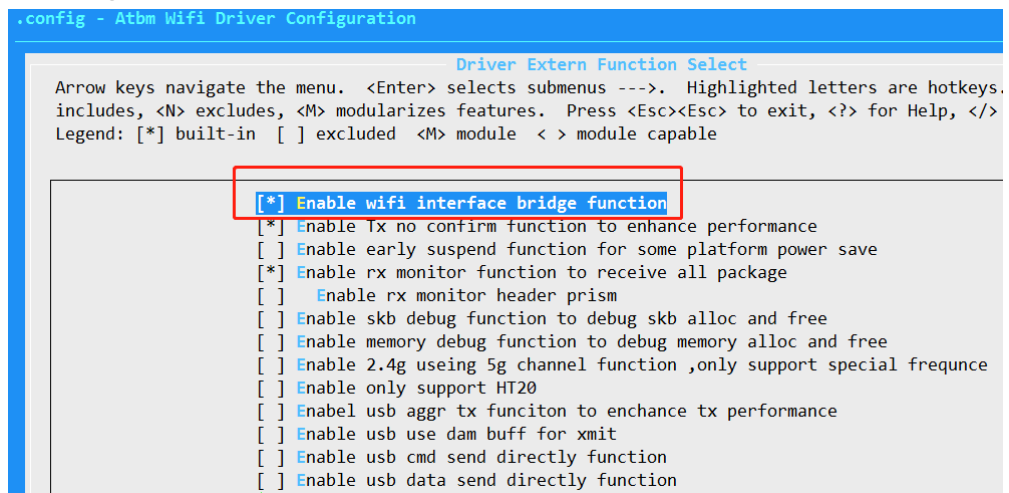
- a) Altobeam6022 是单天线 WIFI, 所以必须保证 wpa\_supplicant 连接的路由器所在的信道和 hostapd 是在同一个信道。
- b) 桥接启动成功, wpa\_supplicant 连接上路由器, 路由器分配的 IP 地址是作用在 br0 接口上的, 此时启动 hostapd, 手机连接上 hostapd, 手机分配的 IP 地址也是路由器分配的。如果路由器可以上网, 手机连接上 hostapd 后, 正常也是可以上网的, 如果不能上网就有问题。

#### 3.3 启动流程

(1) 加载 cfg80211.ko 见 STA MODE 部分

(2) 加载 atbm\_wifi.ko 见 STA MODE 部分

编译驱动的时候驱动要支持 bridge, 需要在驱动根目录下执行 make menuconfig 将 bridge 配置选上





如果这个宏没有打开，在设置的时候会出现如下问题：

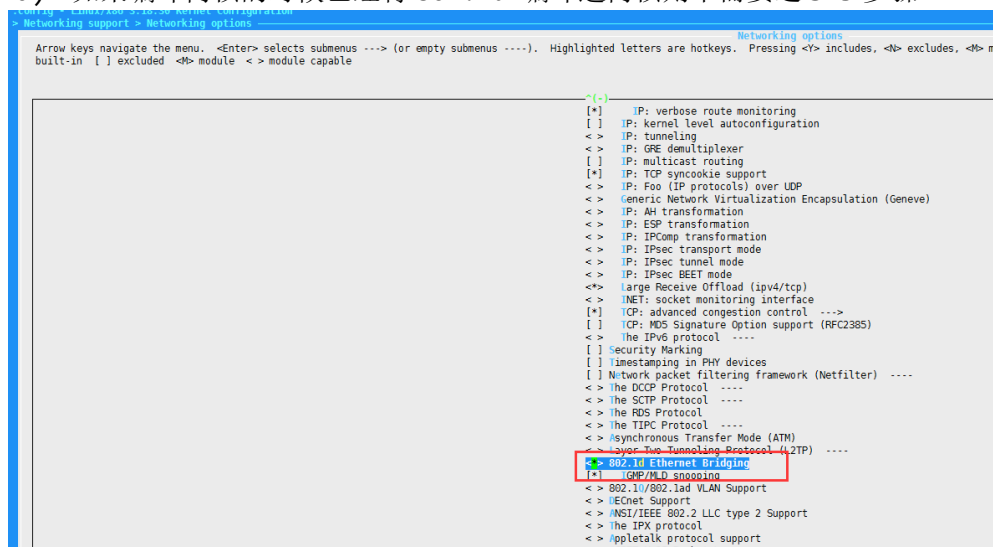
```
[/ext/demo]##
[/ext/demo]##
[/ext/demo]## ./brctl addif br0 wlan0
can't add wlan0 to bridge br0: Operation not supported
[/ext/demo]##
```

(3) 加载 llc.ko

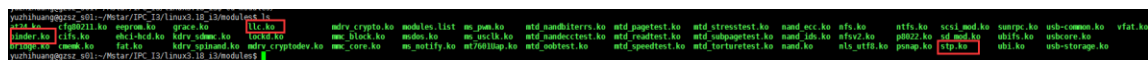
(4) 加载 stp.ko

(5) 加载 bridge.ko

a) 如果编译内核的时候已经将 802.1d 编译进内核则不需要这 3~5 步骤



b) 如果编译内核的时候将该选项选择为 M，编译为模块就需要 3~5 步骤  
选择为“M”编译内核，最终生成的 llc.ko、stp.ko、bridge.ko 会在内核根目录的 modules 目录里面



(6) 启动 wlan0 接口

```
ifconfig wlan0 up
```

(7) 启动 wlan1/p2p0 接口

```
ifconfig [ wlan1 | p2p0 ] up
```

(8) 设置网桥接口 br0

```
ifconfig wlan0 0.0.0.0
ifconfig [ wlan1 | p2p0 ] 0.0.0.0
brctl addbr br0
brctl addif br0 wlan0
```

```
brctl addif br0 [ wlan1 | p2p0 ]
```

```
brctl setfd br0 0
```

正常 log 应如下图所示:

```
msc313:/opt/conf#brctl addbr br0
msc313:/opt/conf#brctl addif br0 wlan0
RTW: rtw_ndev_notifier_call(wlan0) state:20
RTW: rtw_ndev_notifier_call(wlan0) state:21
device wlan0 entered promiscuous mode
msc313:/opt/conf#brctl addif br0 wlan1
RTW: rtw_ndev_notifier_call(wlan1) state:20
RTW: rtw_ndev_notifier_call(wlan1) state:21
device wlan1 entered promiscuous mode
msc313:/opt/conf#brctl setfd br0 0
brctl: invalid argument 'setfd' to 'brctl'
msc313:/opt/conf#
msc313:/opt/conf#
msc313:/opt/conf#ifconfig br0 up
br0: port 2(wlan1) entered forwarding state
br0: port 2(wlan1) entered forwarding state
br0: port 1(wlan0) entered forwarding state
br0: port 1(wlan0) entered forwarding state
```

### (9) 启动网桥接口 br0

```
ifconfig br0 up
```

### (10) 启动 wpa\_supplicant

wpa\_supplicant.conf 的内容如下:

```
[/ext/demo]## cat run/wpa_supplicant.conf
ctrl_interface=/ext/demo/run/wpa_supplicant
update_config=1

#network={
#    ssid="ASUS"
#    psk="12345678"
#}

network={
    ssid="abc"
    psk="12345678"
}

#network={
#    ssid="tp123456"
#    psk="12345678"
#}

network={
    ssid="altobeamSZ_2.4g"
    psk="6270181130"
}
# ... / ...
```

wpa\_supplicant 执行命令为:

调试时候加上 -d 参数

```
wpa_supplicant -Dnl80211 -i wlan0 -c
/ext/demo/run/wpa_supplicant.conf -bbr0 &
```

### (11) 启动 dhcpc

```
udhcpc -i br0
```

### (12) 启动 hostapd

hostapd.conf 内容如下:

```
[/ext/demo]## cat run/hostapd_brige.conf
interface=p2p0
bridge=br0
ctrl_interface_group=0

driver=nl80211

ssid=mt7601

hw_mode=g
channel=11

macaddr_acl=0

auth_algs=1

ignore_broadcast_ssid=0

wpa=3

wpa_passphrase=12345678

wpa_key_mgmt=WPA-PSK
wpa_pairwise=TKIP
rsn_pairwise=CCMP
[/ext/demo]## █
```

Hostapd 执行命令为:

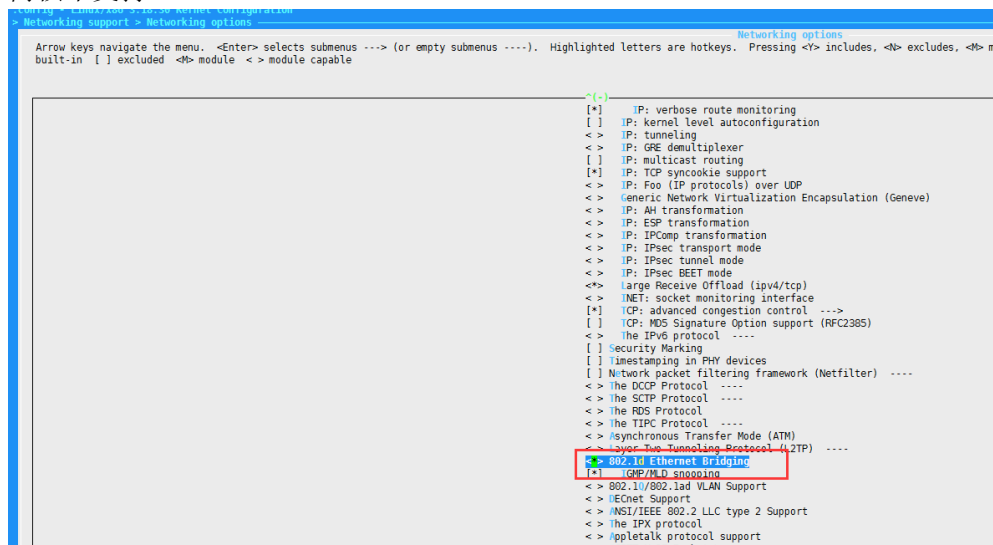
调试时候加上 `-d` 参数

```
hostapd -B /ext/demo/run/hostapd_brige.conf &
```

### 3.4 注意事项

#### (1) 网桥接口 `br0` 起不来

内核不支持 802.1d



#### (2) 使用 `brctl` 没法添加接口道 `br0`

错误如下:

```
[/ext/demo]##
[/ext/demo]## ./brctl addif br0 wlan0
can't add wlan0 to bridge br0: Operation not supported
[/ext/demo]##
```

altobeam 驱动中没有将 DCONFIG\_MAC80211\_BRIDGE 宏打开

```

76: ccflags-y += -DWIFI_FW_DOWNLOAD
77: # Extra IE for probe response from upper layer is needed in P2P GO
78: # For offloading probe response to FW, the extra IE must be included
79: # in the probe response template
80: ccflags-y += -DATBM_PROBE_RESP_EXTRA_IE
81: ccflags-y += -DCONFIG_ATBM_APOLLO_DEBUG
82: #ccflags-y += -DCONFIG_MAC80211_BRIDGE
83: #ccflags-y += -DIPV6_FILTERING
84: #ccflags-y += -DCONFIG_ATBM_APOLLO_BH_DEBUG
85: #ccflags-y += -DCONFIG_ATBM_APOLLO_WSM_DEBUG
86: #ccflags-y += -DCONFIG_ATBM_APOLLO_WSM_DUMPS
87: #ccflags-y += -DCONFIG_ATBM_APOLLO_WSM_DUMPS_SHORT
88: #ccflags-y += -DCONFIG_ATBM_APOLLO_TXRX_DEBUG
89: #ccflags-y += -DCONFIG_ATBM_APOLLO_TX_POLICY_DEBUG
90: #ccflags-y += -DCONFIG_ATBM_APOLLO_STA_DEBUG
91: #ccflags-y += -DCONFIG_ATBM_APOLLO_DUMP_ON_ERROR
92: #ccflags-y += -DCONFIG_ATBM_APOLLO_ITP
93: ccflags-y += -DCONFIG_ATBM_APOLLO_TESTMODE
94: # use the mac addr in file :"/data/.mac.info"
95: ccflags-y += -DCUSTOM_FEATURE_MAC
96: #ccflags-y += -DTEST_RF_POWER

```

### (3) 使用桥接以后发现获取 IP 地址很慢

通过执行 `brctl show` 查看正常和非正常获取 IP 地址的区别:

正常获取 IP 地址:

```

bridge name      bridge id      STP enabled      interfaces
br0              8000.6095ce60f39b  no              p2p0
                wlan0

```

获取 IP 地址时间很长:

```

root@jabsco:/tmp# brctl show
bridge name      bridge id      STP enabled      interfaces
br0              8000.dc2919007645  yes              p2p0
                wlan0

```

上网查了资料:

如何处理STP开启时终端Ping网关不通或获取IP地址慢的问题?

终端设备(如服务器、网管等)不支持运行STP协议,但是由于接入交换机的STP功能是开启状态,那么接口状态在Up/Down上不停变换时,接口在30秒时间内才能进入转发状态,导致终端Ping网关不通或者获取IP地址慢。

为了解决上述问题,需要在连接终端的接口上配置边缘端口或关闭STP功能。

从可用性和安全性考虑,建议把端口指定为边缘端口,因为在连接的终端设备出现环路时,边缘端口可以自动切换为非边缘端口,自动启动该接口STP破环功能。

解决方法:

```
brctl stp br0 off
```

### (4) 将 p2p0 加入桥接以后想修改网口模式, 但是修改失败

```

~ #
~ # /home/iwconfig p2p0 mode manager
[atbm_log]:ieee80211_netdev_ioctl:cmd err
Error for wireless request "Set Mode" (8B06) :
      SET failed on device p2p0 ; Invalid argument.
~ #

```

P2p0 启动了 ap 模式，将 hostapd kill 掉以后查询 p2p0 也是处于 ap 模式，必须要将 p2p0 移出桥接才能够修改模式成功。

```

~ #
~ # iw dev
phy#0
    Interface p2p0
        ifindex 4
        type AP
    Interface eth3
        ifindex 3
        type managed

~ # brctl show
bridge name      bridge id        STP enabled      interfaces
eth2             8000.dc2919c69af3  no              eth3
                                                         p2p0

~ #
~ #
~ # iw p2p0 set type managed
command failed: Device or resource busy (-16)
~ #
~ # ifconfig eth2 down
eth2: port 2(p2p0) entered disabled state
~ # brctl delbr eth2
device p2p0 left promiscuous mode
eth2: port 2(p2p0) entered disabled state
device eth3 left promiscuous mode
eth2: port 1(eth3) entered disabled state
~ #
~ # iw p2p0 set type managed
[atbm_log]:p2p0: is not sta mode
[atbm_log]:br0_netdev_open()~1116: dev_get_by_name(br0) failed2!~ #
~ #
~ # iw dev
phy#0
    Interface p2p0
        ifindex 4
        type managed
    Interface eth3
        ifindex 3
        type managed

```

原因是桥接模式下内核限制了不让网口修改模式：

```

return -EOPNOTSUPP;

/* if it's part of a bridge, reject changing type to station/ibss */
if ((dev->priv_flags & IFF_BRIDGE_PORT) &&
    (ntype == NL80211_IFTYPE_ADHOC ||
     ntype == NL80211_IFTYPE_STATION ||
     ntype == NL80211_IFTYPE_P2P_CLIENT))
return -EBUSY;

```

## 4 Monitor mode

### 4.1 使用 iwconfig 配置

(1) 内核需要打开如下宏用于使能 **iwconfig**, **iwpriv**, **iwlist** 命令

```
CONFIG_CFG80211_WEXT=y  
CONFIG_WIRELESS=y  
CONFIG_WIRELESS_EXT=y  
CONFIG_WEXT_CORE=y  
CONFIG_WEXT_PROC=y  
CONFIG_WEXT_PRIV=y
```

(2) 启动 **monitor** 接口

- 1、**ifconfig wlan0 down**
- 2、**iwconfig wlan0 mode monitor**
- 3、**ifconfig wlan0 up**
- 4、**ifconfig wlan0 0.0.0.0**
- 5、**iwconfig wlan0 channel [1,13] // 选择一个信道监听**

(3) 切换监听信道

```
iwconfig wlan0 channel [1,13] // 选择一个信道
```

(4) 关闭 **monitor** 接口

- 1、**ifconfig wlan0 down**
- 2、**iwconfig wlan0 mode manager**

### 4.2 使用 iw 配置

编译 **iw** 工具需要用到 **libnl** 库，平台需要支持 **libnl** 库

### (1) 启动 monitor 模式

```
ifconfig wlan0 down
iw dev wlan0 del
iw phy phy0 interface add wlan0 type monitor
ifconfig wlan0 up
```

### (2) 切换监听信道

```
iw dev wlan0 set channel 1
```

### (3) 退出 monitor 模式

```
ifconfig wlan0 down
iw dev wlan0 del
iw phy phy0 interface add wlan0 type manager
ifconfig wlan0 up
```

## 5 一个 SOC 运行两个 altobeam wifi 的方法

### 5.1 usb id 相同的问题

因为 altobeam wifi 驱动使用了大量的全局变量，所以一个驱动没法跑多个 device,所以只能一个驱动对应一个 device。

这里需要做如下的修改：

#### (1) 修改内核的 usb 驱动，枚举的时候修改掉内核保存的 usb id

在如下文件修改：Linux\_kernel/drivers/usb/core/hub.c

在 usb\_enumerate\_device 函数里面修改

```

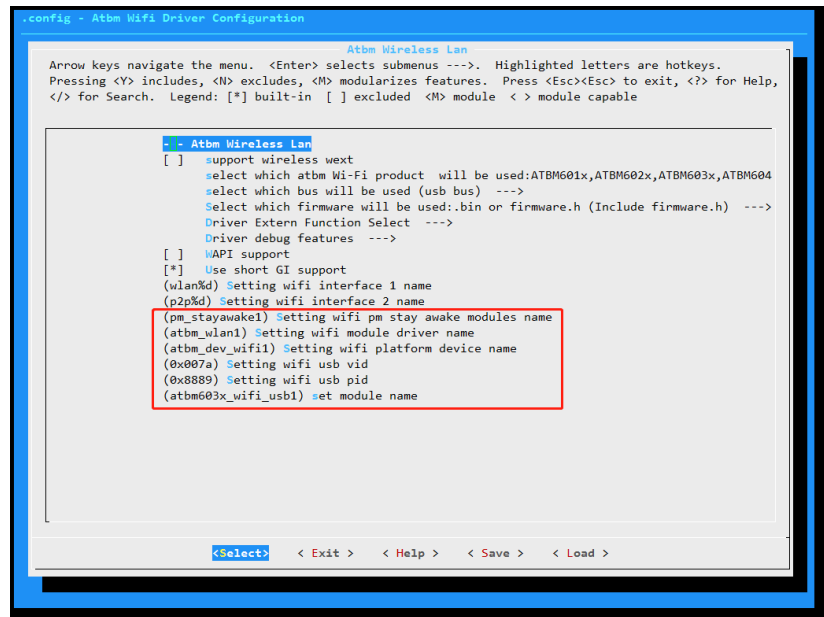
        }
    }
    return err;
}

/* read the standard strings and cache them if present */
udev->product = usb_cache_string(udev, udev->descriptor.iProduct);
udev->manufacturer = usb_cache_string(udev, udev->descriptor.iManufacturer);
udev->serial = usb_cache_string(udev, udev->descriptor.iSerialNumber);
//if(MP_USB_MSTAR == 1)
if((le16_to_cpu(udev->descriptor.idVendor) == 0x007a)
    && (le16_to_cpu(udev->descriptor.idProduct) == 0x8888)){
    if(flag == 0){
        flag = 1;
    }else{
        udev->descriptor.idProduct = 0x8889;
        printk("USB PID is : 0x%x \n",udev->descriptor.idProduct);
    }
}
}
//endif
err = usb_enumerate_device_otg(udev);
if (err < 0)
    return err;

```

## (2) 驱动修改

在驱动根目录执行 make menuconfig 的时候修改一些注册到内核的全局变量



其中 usb vid/pid 需要修改的与 hub.c 里面修改的对应。

## 5.2 修改完成，使用方法

### (1) 加载两个驱动

```

/tmp # lsmod
atbm603x_wifi_usb1 567170 0 - Live 0xbfdb0000 (0)
atbm603x_wifi_usb 568305 0 - Live 0xbfd11000 (0)
cfg80211 175553 2 atbm603x_wifi_usb1,atbm603x_wifi_usb, Live 0xbfc6000
drv_ms_cus_imx291_MIPi 6337 0 - Live 0xbfc32000 (0)
mi_shadow 39260 0 - Live 0xbfc23000 (0)

```



## (2) Ifconfig -a 可以看到多个网口

```

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1
RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

p2p0    Link encap:Ethernet  HWaddr 86:7A:B6:90:72:81
        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)

p2p1    Link encap:Ethernet  HWaddr DE:29:19:00:09:C9
        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)

wlan0   Link encap:Ethernet  HWaddr 84:7A:B6:90:72:81
        inet addr:192.168.40.129 Bcast:192.168.40.255 Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:1399 errors:0 dropped:538 overruns:0 frame:0
        TX packets:91 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:203020 (198.2 KiB) TX bytes:9454 (9.2 KiB)

wlan1   Link encap:Ethernet  HWaddr DC:29:19:00:09:C9
        inet addr:192.168.1.2 Bcast:192.168.1.255 Mask:255.255.255.0
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
        RX packets:297 errors:0 dropped:15 overruns:0 frame:0
        TX packets:97 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:1000
        RX bytes:41580 (40.6 KiB) TX bytes:7766 (7.5 KiB)

```

## (3) 使用 wpa\_supplicant 连接 ap

### 5.2.3.1 配文件内容

```

/tmp #
/tmp # cat /customer/wpa_supplicant.conf
ctrl_interface=/tmp/wpa_supplicant_test
update_config=1

network={
    ssid="yzh_test"
    psk=69073e0817eb53ec05be5176ceb23e14c8f8fd804ecba2d241903bf8bb5d9038
    #
    key_mgmt=WPA-PSK
    scan_freq=2412
}
/tmp #
/tmp # cat /customer/wpa_supplicant1.conf
ctrl_interface=/tmp/wpa_supplicant_test1
update_config=1

network={
    ssid="test_f"
    psk="12345678"
    key_mgmt=WPA-PSK
}
/tmp #

```

### 5.2.3.2 一个 wpa\_supplicant 运行两个 sta 连接两个 ap

```

wpa_supplicant -Dnl80211 -iwlan0 -c /customer/wpa_supplicant.conf -N -Dnl80211
-iwlan1 -c /customer/wpa_supplicant1.conf -B

```

```

/tmp # ./wpa_supplicant -Dnl80211 -iwlan0 -c /customer/wpa_supplicant.conf -N -D
nl80211 -iwlan1 -c /customer/wpa_supplicant1.conf -B
Successfully initialized wpa_supplicant
rfkill: Cannot open RFKILL control device
open file error/n[atbm_log]:br0_netdev_open()-1197: dev_get_by_name(br0)

12_packet_init : 11.sll_protocol = 0x8e88 ,protocol = 0x888e
atbm_get_bss_to_system open fail!
atbm_save_bss_to_system open /ext/etc/.wpa_bss_info fail!
rfkill: Cannot open RFKILL control device
[atbm_log]:br0_netdev_open()-1197: dev_get_by_name(br0)

12_packet_init : 11.sll_protocol = 0x8e88 ,protocol = 0x888e
atbm_get_bss_to_system open fail!
atbm_save_bss_to_system open /ext/etc/.wpa_bss_info fail!
/tmp # [atbm_log]:atbm_hw_scan:if id(0)
[atbm_log]:atbm_hw_scan:scan, delay suspend
[atbm_log]:scan start band(0),(1)
[atbm_log]:hw_priv->scan.status 0
[atbm_log]:atbm_scan_work:end(0)
[atbm_log]:wlan0:free authen bss ++
[atbm_log]:authen:(cc:08:fb:92:97:27),ssid(yzh_test)
[atbm_log]:wlan0: authenticated
[atbm_log]:wlan0:free authen bss ++
[atbm_log]:wlan0:free authen bss --
[atbm_log]:wlan0: associated
[atbm_log]:[cc:08:fb:92:97:27]:20M channel
[atbm_log]:ieee80211_recalc_ps:work busy
[atbm_log]:ieee80211_recalc_ps:work busy
[atbm_log]:atbm_hw_scan:if_id(0)
[atbm_log]:atbm_hw_scan:scan, delay suspend
[atbm_log]:scan start band(0),(14)
[atbm_log]:hw_priv->scan.status 0
[atbm_log]:atbm_scan_work:end(0)
[atbm_log]:wlan1:free authen bss ++
[atbm_log]:authen:(90:76:9f:41:d0:c8),ssid(test_f)
[atbm_log]:wlan1: authenticated
[atbm_log]:wlan1:free authen bss ++
[atbm_log]:wlan1:free authen bss --
[atbm_log]:wlan1: associated
[atbm_log]:[90:76:9f:41:d0:c8]:20M channel
[atbm_log]:ieee80211_recalc_ps:work busy
[atbm_log]:ieee80211_recalc_ps:work busy

```

wpa\_supplicant运行的参数

wlan0去连接wpa\_supplicant.conf配置的ap

wlan1 去连接wpa\_supplicant1.conf配置的ap

这么运行只有一个 wpa\_supplicant 进程在跑:

```

524 root      0:00 [cfg80211]
527 root      0:00 [phy0-atbm_wq]
528 root      0:00 [phy0-usb_atbm_b]
556 root      0:00 [phy1-atbm_wq]
557 root      0:00 [phy1-usb_atbm_b]
577 root      0:00 [kworker/0:1]
591 root      0:00 ./wpa_supplicant -Dnl80211 -iwlan0 -c /customer/wpa_suppl
600 root      0:00 [kworker/u2:3]
621 root      0:00 udhcpc -i wlan0
627 root      0:00 udhcpc -i wlan1
644 root      0:00 [kworker/u2:0]
647 root      0:00 ps

/tmp # ifconfig
wlan0      Link encap:Ethernet  HWaddr 84:7A:B6:90:72:81
           inet addr:192.168.40.129  Bcast:192.168.40.255  Mask:255.255.255.0
           UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
           RX packets:2673 errors:0 dropped:1041 overruns:0 frame:0
           TX packets:156 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:1000
           RX bytes:393124 (383.9 KiB)  TX bytes:15388 (15.0 KiB)

wlan1      Link encap:Ethernet  HWaddr DC:29:19:00:09:C9
           inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.255.0
           UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
           RX packets:466 errors:0 dropped:15 overruns:0 frame:0
           TX packets:158 errors:0 dropped:0 overruns:0 carrier:0
           collisions:0 txqueuelen:1000
           RX bytes:69537 (67.9 KiB)  TX bytes:11548 (11.2 KiB)

```

### 5.2.3.3 分别使用两个 wpa\_supplicant 去连接两个 ap

```
wpa_supplicant -Dnl80211 -iwlan0 -c /customer/wpa_supplicant.conf -B  
wpa_supplicant -Dnl80211 -iwlan1 -c /customer/wpa_supplicant1.conf -B
```

这种方式比较直观，运行起来有两个 wpa\_supplicant 进程在运行。

## 6 应用接收 wpa\_supplicant & hostapd EVENT 处理方法

### 6.1 wpa\_supplicant

#### (1) wpa\_supplicant 运行参数

```
wpa_supplicant -Dnl80211 -iwlan0 -c wpa_supplicant.conf -g /tmp/wpa_global
```

#### (2) 新增参数说明 & unix domain socket 路径说明

-g /tmp/wpa\_global 是 wpa\_supplicant 配置的 unix 本地 socket 路径用来接收接收 demo 消息的，用户可以自定义修改。

Wpa\_supplicant 专门有一套 ctrl\_interface\_global 的接口，-g 参数为初始化 wpa\_supplicant 本地的 unix domain socket 通信路径。

接收事件的 demo 见 get\_wpa\_event.c

### 6.2 hostapd

#### (1) hostapd 运行参数

```
hostapd -B hostap.conf -g /tmp/wpa_global_event
```

hostapd.conf 里面的参数：

```
interface=wlan0  
ctrl_interface=/tmp/hostapd
```

#### (2) unix domain socket 路径说明

Hostapd 创建的本地 unix domain socket 通信路径为 /tmp/hostapd/wlan0

消息类型参照 `hostapd_cli` 格式。

接收事件的 demo 见 `get_hostapd_event.c`



---

### **CONTACT INFORMATION**

AltoBeam (China) Inc.

Address: B808, Tsinghua Tongfang Hi-Tech Plaza, Haidian, Beijing, China 100083

Tel: (8610) 6270 1811

Fax: (8610) 6270 1830

Website: [www.altobeam.com](http://www.altobeam.com)

Email: [support@altobeam.com](mailto:support@altobeam.com)

### **DISCLAIMER**

Information in this document is provided in connection with AltoBeam products. No license, express or implied, by estoppels or otherwise, to any intellectual property rights is granted by this document. Except as provided in AltoBeam's terms and conditions of sale for such products, AltoBeam assumes no liability whatsoever, and AltoBeam disclaims any express or implied warranty, relating to sale and/or use of AltoBeam products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right.

AltoBeam may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." AltoBeam reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them.

Unauthorized use of information contained herein, disclosure or distribution to any third party without written permission of AltoBeam is prohibited.

AltoBeam™ is the trademark of AltoBeam. All other trademarks and product names are properties of their respective owners.

Copyright © 2007~2020 AltoBeam, all rights reserved