Realtek 802.11r Development guide

Date: 2014/12/25

Version: 1.0

CHANGE HISTORY

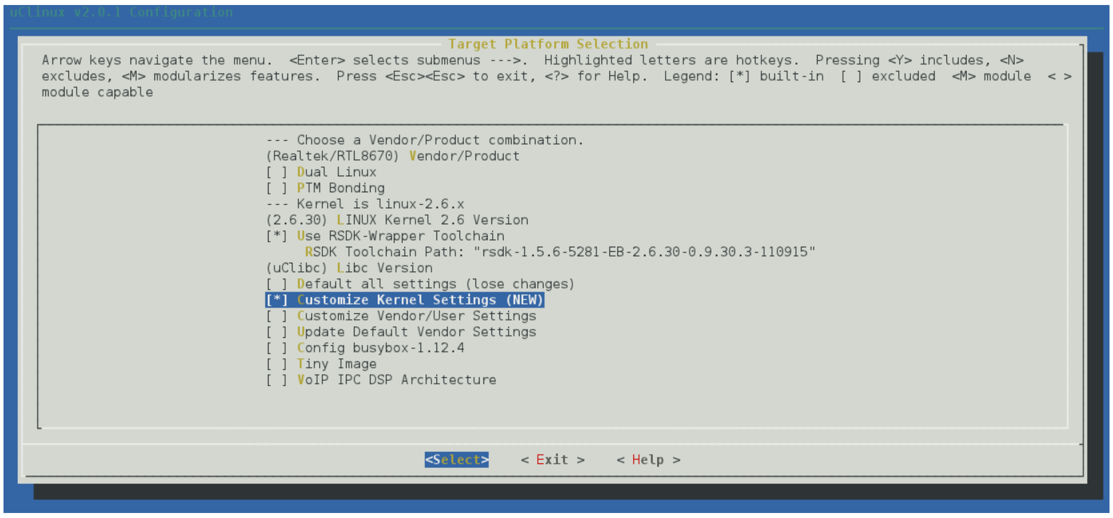
VERSION	DATE	REMARKS
1.0	2014/12/25	INITIAL RELEASE

TABLE OF CONTENTS

1. How to enable 802.11r function

To enable 802.11r support, select the “IEEE 802.11R Support“ in menuconfig, as following demonstrated.

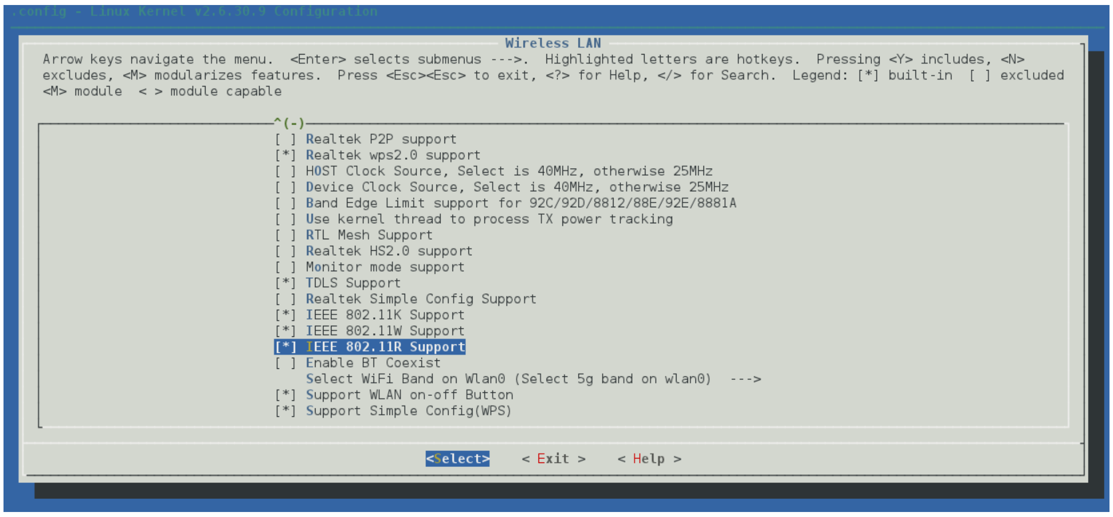
Select “Config kernel”



```
uClinux v2.0.1 Configuration
                              Target Platform Selection
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.  Pressing <Y> includes, <N>
  excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.  Legend: [*] built-in  [ ] excluded  <M> module  < >
  module capable

                    --- Choose a Vendor/Product combination.
                    (Realtek/RTL8670) Vendor/Product
                    [ ] Dual Linux
                    [ ] PTM Bonding
                    --- Kernel is linux-2.6.x
                    (2.6.30) LINUX Kernel 2.6 Version
                    [*] Use RSDK-Wrapper Toolchain
                         RSDK Toolchain Path: "rsdk-1.5.6-5281-EB-2.6.30-0.9.30.3-110915"
                    (uClibc) Libc Version
                    [ ] Default all settings (lose changes)
                    [*] Customize Kernel Settings (NEW)
                    [ ] Customize Vendor/User Settings
                    [ ] Update Default Vendor Settings
                    [ ] Config busybox-1.12.4
                    [ ] Tiny Image
                    [ ] VoIP IPC DSP Architecture

                          <Select>    < Exit >    < Help >
```

, and then exist.

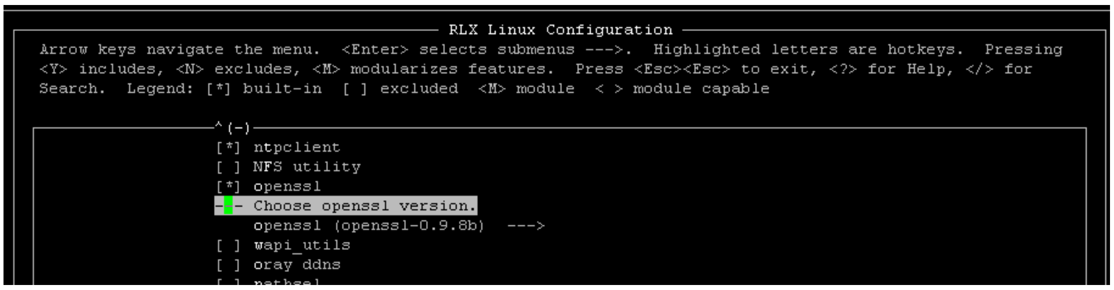
Select option of “Device Drivers”->“Network device support”->“Wireless LAN”, you would see bellow figure.



```
.config - Linux Kernel v2.6.30.9 Configuration
                                Wireless LAN
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.  Pressing <Y> includes, <N>
  excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ] excluded
  <M> module  < > module capable

                    ^(-)
                    [ ] Realtek P2P support
                    [*] Realtek wps2.0 support
                    [ ] HOST Clock Source, Select is 40MHz, otherwise 25MHz
                    [ ] Device Clock Source, Select is 40MHz, otherwise 25MHz
                    [ ] Band Edge Limit support for 92C/92D/8812/88E/92E/8881A
                    [ ] Use kernel thread to process TX power tracking
                    [ ] RTL Mesh Support
                    [ ] Realtek HS2.0 support
                    [ ] Monitor mode support
                    [*] TDLS Support
                    [ ] Realtek Simple Config Support
                    [*] IEEE 802.11K Support
                    [*] IEEE 802.11W Support
                    [*] IEEE 802.11R Support
                    [ ] Enable BT Coexist
                        Select WiFi Band on Wlan0 (Select 5g band on wlan0)  --->
                    [*] Support WLAN on-off Button
                    [*] Support Simple Config(WPS)

                          <Select>    < Exit >    < Help >
```

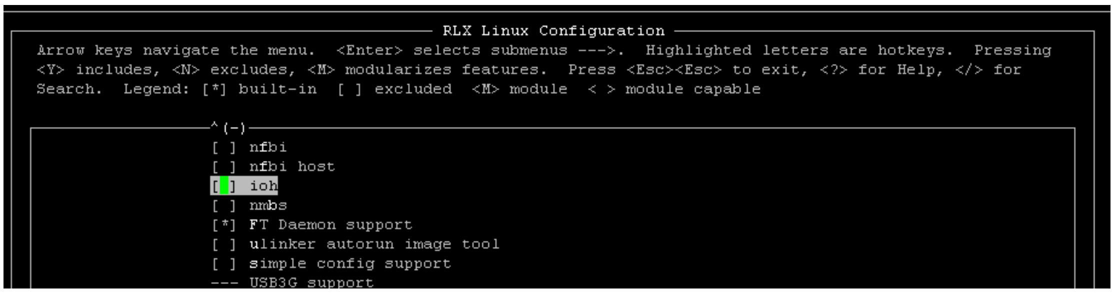
Make sure “IEEE 802.11R Support” is selected

Besides, select option of”Users”->”openssl”->”openssl-0.9.8b”



```
                           RLX Linux Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for
  Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

                  ^(-)
                  [*] ntpclient
                  [ ] NFS utility
                  [*] openssl
                  --- Choose openssl version.
                      openssl (openssl-0.9.8b)  --->
                  [ ] wapi_utils
                  [ ] oray ddns
```

and select option of”Users”->”FT Daemon support”



```
                       RLX Linux Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.  Pressing
  <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </> for
  Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

                    ^(-)
                    [ ] nfbi
                    [ ] nfbi host
                    [ ] ioh
                    [ ] nmbs
                    [*] FT Daemon support
                    [ ] ulinker autorun image tool
                    [ ] simple config support
                    --- USB3G support
```

Compile SDK to generate new image.

2. Related MIBs

Name	Meaning	Value	Default	Comment
ft_enable	Activate/Deactivate 802.11r feature	0 – activate, 1 - deactivate	1	
ft_mdid	Mobility Domain identifier	2-octet hex string	0000	This MIB is valid only when ft_enable is 1
ft_over_ds	Enable/Disable support over-the-DS FT protocol	0 – disable, 1 – enabl	1	This MIB is valid only when ft_enable is 1
ft_reasoc_timeout	Reassociation timeout in TUs	0 – disable, 1000 ~ 65535	0	This MIB is valid only when ft_enable is 1
ft_r0key_timeout	Key expiration timeout in minutes	0 – disable, 1 ~ 65535	0	This MIB is valid only when ft_enable is 1
ft_r0kh_id	R0 Key Holder identifier	“string value”, 1 ~ 48 characters	0	This MIB is valid only when ft_enable is 1
ft_push	Enable/Disable Key-Push	0 – disable, 1 - enabl	1	This MIB is valid only when ft_enable is 1

*The default value of MIB will be ‘0’ if it is not specified

Table 1

3. Configurations

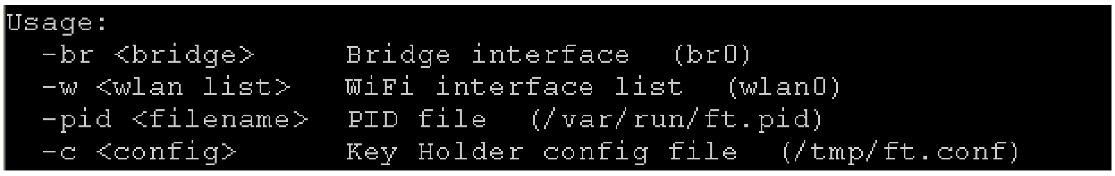
To support 802.11r feature in Auth daemon, the following field in Auth configuration file should change to new value.

Name	Description	Value
authentication	Set authentication mode.	1 – RSN, 2 – PSK, 3 – FT-RSN

Table 2

4. 802.11r Daemon

- Parameters for 802.11r daemon (ftd)



```
Usage:
  -br <bridge>      Bridge interface  (br0)
  -w <wlan list>    WiFi interface list  (wlan0)
  -pid <filename>   PID file  (/var/run/ft.pid)
  -c <config>       Key Holder config file  (/tmp/ft.conf)
```

- Turn on debug message

  Use –v <debug level: 0 ~ 4> to turn on debug message.

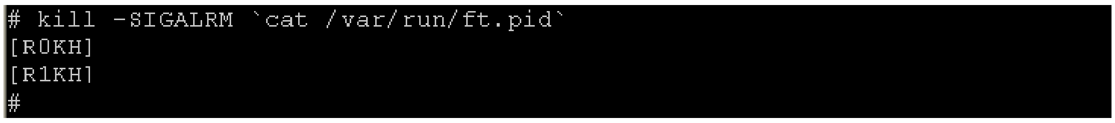
- Key Holder configuration file (ft.conf)

  List R0KHs and R1KHs in the mobility domain.

```
r0kh=<MAC address> <R0KH-ID> <128-bit key hex string/quoted passphrase > <interface>
r1kh=<MAC address> <R1KH-ID> <128-bit key hex string/quoted passphrase> <interface>
...
```

- Debug signal – SIGALRM

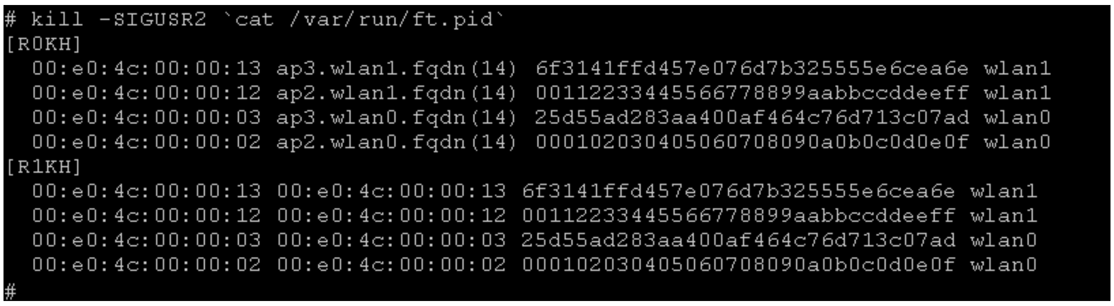
  Clear all Key Holder settings.



```
# kill -SIGALRM `cat /var/run/ft.pid`
[R0KH]
[R1KH]
#
```

- Debug signal – SIGUSR2

  Add Key Holder setting by reloading configuration file and dump settings.



```
# kill -SIGUSR2 `cat /var/run/ft.pid`
[R0KH]
  00:e0:4c:00:00:13 ap3.wlan1.fqdn(14) 6f3141ffd457e076d7b325555e6cea6e wlan1
  00:e0:4c:00:00:12 ap2.wlan1.fqdn(14) 00112233445566778899aabbccddeeff wlan1
  00:e0:4c:00:00:03 ap3.wlan0.fqdn(14) 25d55ad283aa400af464c76d713c07ad wlan0
  00:e0:4c:00:00:02 ap2.wlan0.fqdn(14) 000102030405060708090a0b0c0d0e0f wlan0
[R1KH]
  00:e0:4c:00:00:13 00:e0:4c:00:00:13 6f3141ffd457e076d7b325555e6cea6e wlan1
  00:e0:4c:00:00:12 00:e0:4c:00:00:12 00112233445566778899aabbccddeeff wlan1
  00:e0:4c:00:00:03 00:e0:4c:00:00:03 25d55ad283aa400af464c76d713c07ad wlan0
  00:e0:4c:00:00:02 00:e0:4c:00:00:02 000102030405060708090a0b0c0d0e0f wlan0
#
```

5. Activate/Deactivate 802.11r Example

- Below is an example to activate 802.11r function.

```
ifconfig wlan0 down
iwpriv wlan0 set_mib ft_enable =1
iwpriv wlan0 set_mib ft_mdid=a1b2
iwpriv wlan0 set_mib ft_over_ds=1
iwpriv wlan0 set_mib ft_reasoc_timeout=0
iwpriv wlan0 set_mib ft_r0key_timeout=0
iwpriv wlan0 set_mib ft_r0kh_id=ap1.r0kh-id.fqdn
iwpriv wlan0 set_mib ft_push=1
ifconfig wlan0 up
ftd –br br0 –w wlan0 –pid /var/run/ft.pid –c /tmp/ft.conf &
```

- Example of /tmp/ft.conf

```
r0kh=00:e0:4c:00:11:20 ap2.r0kh-id.fqdn 00112233445566778899aabbccddeeff wlan0

r1kh=00:e0:4c:00:11:20 00:e0:4c:00:11:20 00112233445566778899aabbccddeeff wlan0

r0kh=00:e0:4c:00:11:30 ap3.r0kh-id.fqdn "quoted passphrase" wlan0

r1kh=00:e0:4c:00:11:30 00:e0:4c:00:11:30 "quoted passphrase" wlan0
```

➢ Below is an example to deactivate 802.11r function.

```
ifconfig wlan0 down
iwpriv wlan0 set_mib ft_enable=0
ifconfig wlan0 up
kill `cat /var/run/ft.pid`
```

6. PROC

- Show current R0/R1 key information

```
# cat /proc/wlan0/ft_info
  Fast BSS Transition Info...
    R0KHs:
    ==================================
    + r0kh 0
      sta_mac:   4ce6766a5344
      pmk_r0:    86eed41cdac8189bb8f1d334817d9ef1c635ef3924cc5574a1de7875e5d40122
      pmk_r0_id: f3e561ef573ebac865f38756993c4992
      expire_to: 0

    R1KHs:
    ==================================
    + r1kh 0
      sta_mac:   4ce6766a5344
      r1kh_id:   00e04c115515
      r0kh_id:   8685.wlan1.root.fqdn
      pmk_r1:    21c5ce966b62fa245d154ffe4896f8cd867a08d6c3c7b863ce86e49f6b10f58e
      pmk_r1_id: 4ed99366284306f9cbbe5f792b5f2f84
      pmk_r0_id: f3e561ef573ebac865f38756993c4992
      pairwise:  16
```

7. IOCTL

Following are the IOCTL commands list for 802.11r daemon to get/set 802.11r resources and events from/to driver.

- **0x8BE0** – Register PID of 802.11r daemon to driver

Input: PID of 802.11r daemon

Output: none

Comment: none

- **0x8BE1** – Get event from driver

Input: A large buffer to receive data comes from driver. The first byte is event identifier which is set to DOT11_EVENT_FT_GET_EVENT

Output: Depends on the receiving event. First byte is the received event type, and the second byte indicates if there is more events queued in driver.

Comment:

typedef enum {

```
        …
        DOT11_EVENT_FT_GET_EVENT           = 122,
        DOT11_EVENT_FT_IMD_ASSOC_IND       = 123,
        DOT11_EVENT_FT_GET_KEY             = 124,
        DOT11_EVENT_FT_SET_KEY             = 125,
        DOT11_EVENT_FT_PULL_KEY_IND        = 126,
        DOT11_EVENT_FT_ASSOC_IND           = 127,
        DOT11_EVENT_FT_KEY_EXPIRE_IND      = 128,
        DOT11_EVENT_FT_ACTION_IND          = 129,
        DOT11_EVENT_FT_QUERY_INFO          = 130,
        DOT11_EVENT_FT_SET_INFO            = 131,
        DOT11_EVENT_FT_AUTH_INSERT_R0      = 132,
        DOT11_EVENT_FT_AUTH_INSERT_R1      = 133,
        DOT11_EVENT_FT_TRIGGER_EVENT       = 134,
        …
    } DOT11_EVENT;
```

Event Type	**Event Body**	**Action**
DOT11_EVENT_FT_IMD_ASSOC_IND	DOT11_FT_IMD_ASSOC_IND	If push enabled, distribute key to all R1KH. Otherwise, issue key expire message to all R1KH.
DOT11_EVENT_FT_PULL_KEY_IND	DOT11_FT_PULL_KEY_IND	Request key from driver and response to the orignator.
DOT11_EVENT_FT_ASSOC_IND	DOT11_FT_ASSOC_IND	Notify R1KHs on successful BSS transition of specified STA.
DOT11_EVENT_FT_KEY_EXPIRE_IND	DOT11_FT_KEY_EXPIRE_IND	Notify R1KHs about key expiration event.
DOT11_EVENT_FT_ACTION_IND	DOT11_FT_ASSOC_IND	Warp in RRB format and transmit to designated address

Table 3

- **0x8BE2 –** Get key from driver

Input: DOT11_FT_GET_KEY or DOT11_FT_PULL_KEY_IND

Output: DOT11_FT_GET_KEY_PUSH or DOT11_FT_GET_KEY_PULL

Comment:

```
typedef struct _DOT11_FT_GET_KEY{
        unsigned char EventId;
        unsigned char IsMoreEvent;
        unsigned char Type;
        unsigned int Length;
        unsigned char r1kh_id[MACADDRLEN];
        unsigned char s1kh_id[MACADDRLEN];
} __WLAN_ATTRIB_PACK__ DOT11_FT_GET_KEY;
typedef struct _DOT11_FT_PULL_KEY_IND{
        unsigned char EventId;
        unsigned char IsMoreEvent;
        unsigned char Type;
        unsigned char r0kh_id[MAX_R0KHID_LEN];
        unsigned int Length;
        unsigned char nonce[FT_R0KH_R1KH_PULL_NONCE_LEN];
        unsigned char pmk_r0_name[FT_PMKID_LEN];
        unsigned char r1kh_id[MACADDRLEN];
        unsigned char s1kh_id[MACADDRLEN];
} __WLAN_ATTRIB_PACK__ DOT11_FT_PULL_KEY_IND;
typedef struct _DOT11_FT_GET_KEY{
        unsigned char EventId;
        unsigned char IsMoreEvent;
        unsigned char Type;
        unsigned int Length;
        unsigned char r1kh_id[MACADDRLEN];
        unsigned char s1kh_id[MACADDRLEN];
} __WLAN_ATTRIB_PACK__ DOT11_FT_GET_KEY;
typedef struct _DOT11_FT_GET_KEY_PULL{
        unsigned char EventId;
        unsigned char IsMoreEvent;
        unsigned char Type;
        unsigned int Length;
        unsigned char nonce[FT_R0KH_R1KH_PULL_NONCE_LEN];
        unsigned char r1kh_id[MACADDRLEN];
        unsigned char s1kh_id[MACADDRLEN];
        unsigned char pmk_r1[FT_PMK_LEN];
        unsigned char pmk_r1_name[FT_PMKID_LEN];
        unsigned short pairwise;
} __WLAN_ATTRIB_PACK__ DOT11_FT_GET_KEY_PULL;
```

- **0x8BE3 –** Set key to driver

Input: DOT11_FT_SET_KEY_PUSH or DOT11_FT_SET_KEY_PULL

Output: none

Comment:

```
typedef struct _DOT11_FT_SET_KEY_PUSH{
        unsigned char EventId;
        unsigned char IsMoreEvent;
        unsigned char Type;
        unsigned int Length;
        unsigned int timestamp;
        unsigned char r1kh_id[MACADDRLEN];
        unsigned char s1kh_id[MACADDRLEN];
        unsigned char pmk_r0_name[FT_PMKID_LEN];
        unsigned char pmk_r1[FT_PMK_LEN];
        unsigned char pmk_r1_name[FT_PMKID_LEN];
        unsigned short pairwise;
} __WLAN_ATTRIB_PACK__ DOT11_FT_SET_KEY_PUSH;
typedef struct _DOT11_FT_SET_KEY_PULL{
        unsigned char EventId;
        unsigned char IsMoreEvent;
        unsigned char Type;
        unsigned int Length;
        unsigned char nonce[FT_R0KH_R1KH_PULL_NONCE_LEN];
        unsigned char r1kh_id[MACADDRLEN];
        unsigned char s1kh_id[MACADDRLEN];
        unsigned char pmk_r1[FT_PMK_LEN];
        unsigned char pmk_r1_name[FT_PMKID_LEN];
        unsigned short pairwise;
} __WLAN_ATTRIB_PACK__ DOT11_FT_SET_KEY_PULL;
```

- **0x8BE4 –** Notify driver on receiving special events from Ethernet

Input: DOT11_FT_ASSOC_IND or DOT11_FT_KEY_EXPIRE_IND

Output: none

Comment:

```
typedef struct _DOT11_FT_ASSOC_IND{
        unsigned char EventId;
        unsigned char IsMoreEvent;
        unsigned char MACAddr[MACADDRLEN];
} __WLAN_ATTRIB_PACK__ DOT11_FT_ASSOC_IND;
typedef struct _DOT11_FT_KEY_EXPIRE_IND{
```

```
            unsigned char EventId;
            unsigned char IsMoreEvent;
            unsigned char MACAddr[MACADDRLEN];
    } __WLAN_ATTRIB_PACK__ DOT11_FT_KEY_EXPIRE_IND;
```

- **0x8BE5 –** Notify driver on receiving FT Action frames

Input: DOT11_FT_ACTION

Output: none

Comment:

```
    typedef struct _DOT11_FT_ACTION{
            unsigned char EventId;
            unsigned char IsMoreEvent;
            unsigned char MACAddr[MACADDRLEN];
            unsigned char ActionCode;
            unsigned int packet_len;
            unsigned char packet[MAX_FTACTION_LEN];
    } __WLAN_ATTRIB_PACK__ DOT11_FT_ACTION;
```

8. Use wpa_supplicant to test 802.11r feature

To test 802.11r feature, we need to build customized wpa_supplicant binary. Get the source of wpa_supplicant and edit config file to set *CONFIG_IEEE80211R=y* and *CONFIG_IEEE80211N=y.* Then type make command to get the binary.
Use wpa_cli command to trigger wpa_supplicant to roam and use sniffer to check if FT protocol is

- **wpa_supplicant configuration file (80211r.conf)**

WPA2-PSK, WPA/WPA2-PSK Mixed mode:

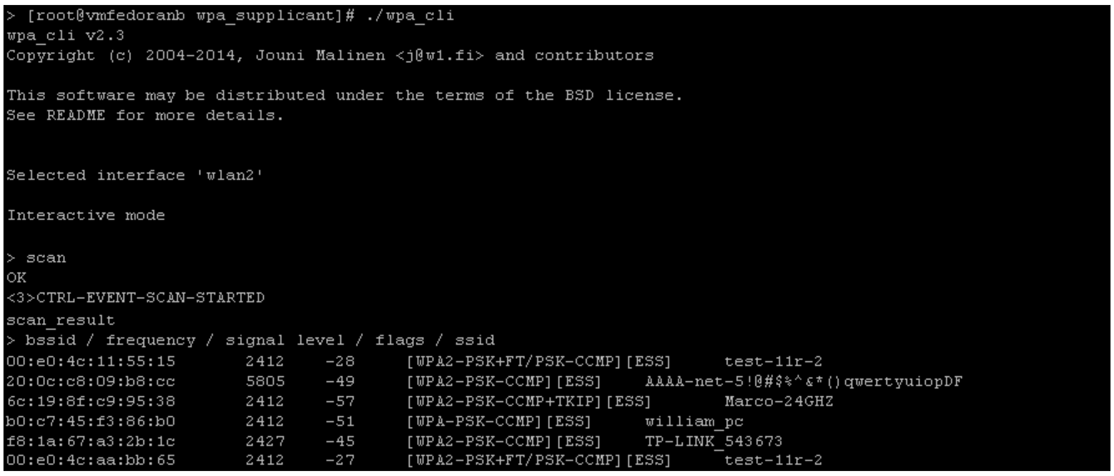
```
network={
        ssid="test-11r-2"
        key_mgmt=FT-PSK
        psk="12345678"
}
```

WPA2-EAP, WPA/WPA2-EAP Mixed mode:

```
network={
        ssid="test-11r-2"
        key_mgmt=FT-EAP
        eap=PEAP
        identity="wifitest"
        password="TestUser@123"
        ca_cert="wifitest.pem"
        phase1="peapver=0"
        phase2="MSCHAPV2"
}
```

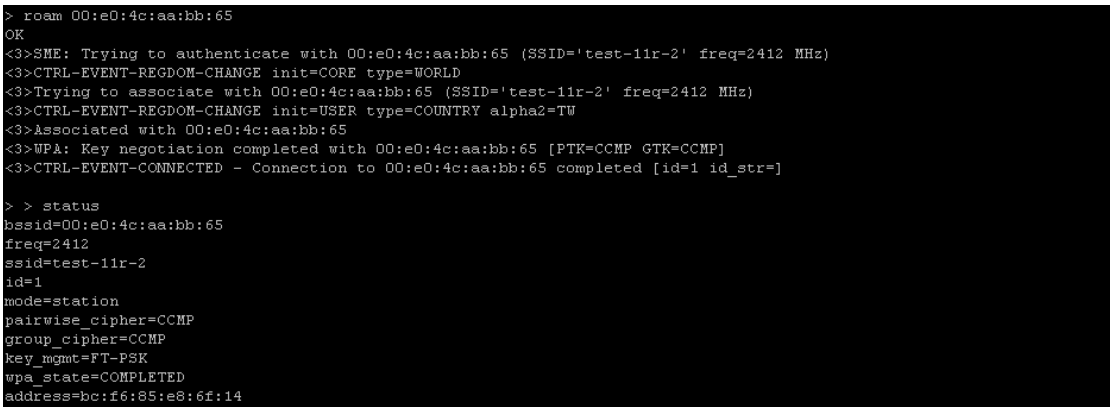
- **Start wpa_supplicant**
  wpa_supplicant –D nl80211 –i wlan0 –c 802.11r.conf –B
  wpa_supplicant –D nl80211 –i wlan0 –c 802.11r.conf –d

- **Issue scan request to wpa_supplicant and check scan result**
  wpa_cli scan
  wpa_cli scan_result



```
> [root@vmfedoranb wpa_supplicant]# ./wpa_cli
wpa_cli v2.3
Copyright (c) 2004-2014, 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 'wlan2'

Interactive mode

> scan
OK
<3>CTRL-EVENT-SCAN-STARTED
scan_result
> bssid / frequency / signal level / flags / ssid
00:e0:4c:11:55:15       2412    -28     [WPA2-PSK+FT/PSK-CCMP][ESS]     test-11r-2
20:0c:c8:09:b8:cc       5805    -49     [WPA2-PSK-CCMP][ESS]    AAAA-net-5!@#$%^&*()qwertyuiopDF
6c:19:8f:c9:95:38       2412    -57     [WPA2-PSK-CCMP+TKIP][ESS]       Marco-24GHZ
b0:c7:45:f3:86:b0       2412    -51     [WPA-PSK-CCMP][ESS]     william_pc
f8:1a:67:a3:2b:1c       2427    -45     [WPA2-PSK-CCMP][ESS]    TP-LINK_543673
00:e0:4c:aa:bb:65       2412    -27     [WPA2-PSK+FT/PSK-CCMP][ESS]     test-11r-2
```

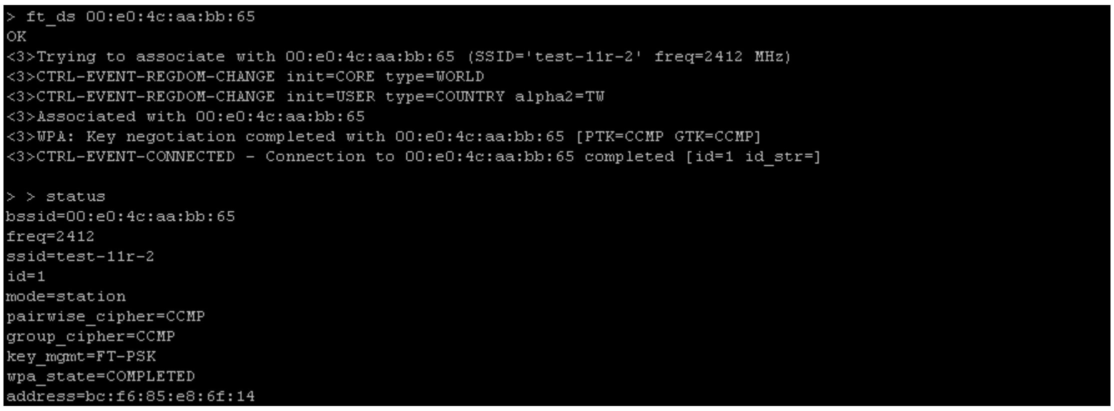
- **Issue over-the-air FT request to roam to specified BSSID in the scan result**
  wpa_cli roam 00:E0:4C:AA:BB:65



```
> roam 00:e0:4c:aa:bb:65
OK
<3>SME: Trying to authenticate with 00:e0:4c:aa:bb:65 (SSID='test-11r-2' freq=2412 MHz)
<3>CTRL-EVENT-REGDOM-CHANGE init=CORE type=WORLD
<3>Trying to associate with 00:e0:4c:aa:bb:65 (SSID='test-11r-2' freq=2412 MHz)
<3>CTRL-EVENT-REGDOM-CHANGE init=USER type=COUNTRY alpha2=TW
<3>Associated with 00:e0:4c:aa:bb:65
<3>WPA: Key negotiation completed with 00:e0:4c:aa:bb:65 [PTK=CCMP GTK=CCMP]
<3>CTRL-EVENT-CONNECTED - Connection to 00:e0:4c:aa:bb:65 completed [id=1 id_str=]

> > status
bssid=00:e0:4c:aa:bb:65
freq=2412
ssid=test-11r-2
id=1
mode=station
pairwise_cipher=CCMP
group_cipher=CCMP
key_mgmt=FT-PSK
wpa_state=COMPLETED
address=bc:f6:85:e8:6f:14
```

- **Issue over-the-DS FT request with specified BSSID in the scan result**

  wpa_cli ft_ds 00:E0:4C:AA:BB:65



```
> ft_ds 00:e0:4c:aa:bb:65
OK
<3>Trying to associate with 00:e0:4c:aa:bb:65 (SSID='test-11r-2' freq=2412 MHz)
<3>CTRL-EVENT-REGDOM-CHANGE init=CORE type=WORLD
<3>CTRL-EVENT-REGDOM-CHANGE init=USER type=COUNTRY alpha2=TW
<3>Associated with 00:e0:4c:aa:bb:65
<3>WPA: Key negotiation completed with 00:e0:4c:aa:bb:65 [PTK=CCMP GTK=CCMP]
<3>CTRL-EVENT-CONNECTED - Connection to 00:e0:4c:aa:bb:65 completed [id=1 id_str=]

> > status
bssid=00:e0:4c:aa:bb:65
freq=2412
ssid=test-11r-2
id=1
mode=station
pairwise_cipher=CCMP
group_cipher=CCMP
key_mgmt=FT-PSK
wpa_state=COMPLETED
address=bc:f6:85:e8:6f:14
```