REALTEK

Quick Guide for wpa_supplicant Wi-Fi P2P test

Date: 2020/12/18

Version: 1.2

(1) Release note

Document Version	Note
V1.0	1.　First release

V1.1	1. Add case "Connect in PBC and establish a persistent P2P group"
	2. Add case "Re-invoke a persistent P2P group"
V1.2	1. New (7) Improve the success rate of P2P nego to GO

(2) P2P description:

Wi-Fi Direct（P2P）is a technology developed by Wi-Fi Alliance. It is a solution for Wi-Fi device-to-device connectivity. And it is also backward compatible with existing Wi-Fi Certified devices.

In usual, there are 3 stages in the Wi-Fi Direct scenario.
1. "Device Discovery"
2. "Group Formation" + "Provisioning" (WPS)
3. "Device connection" (DHCP)

The following picture will provide the overall concept for Wi-Fi Direct functionality and it will also contain these 3 stages described above.
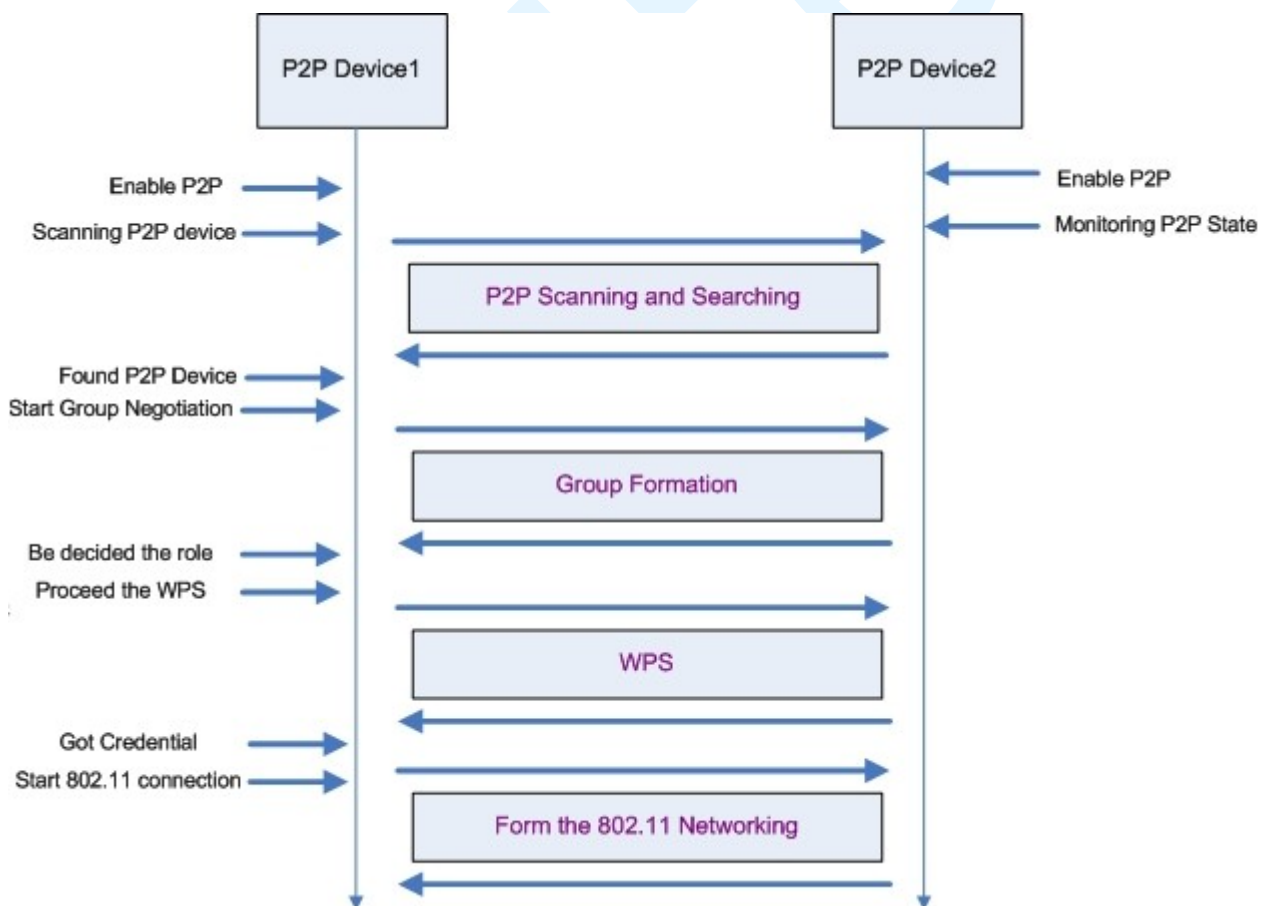

Figure1: Wi-Fi Direct Overview

The figure1 describes the basic Wi-Fi Direct scenario and this document will use this figure to do connection

test.

1. Enable P2P

In this case, there are two Wi-Fi devices which both support the Wi-Fi Direct functionality. We can use start wpa_supplicant to enable the Wi-Fi Direct function of Realtek Wi-Fi driver (enable P2P).

If use driver version 5.8 or later and it supports "rtw_sel_p2p_iface" when insmod, you need to use this parameter to select p2p interface when driver define "CONFIG_CONCURRENT_MODE".

Ex: #> insmod 8821cu.ko rtw_sel_p2p_iface=0

(This example select interface 0 to be p2p interface, otherwise default use interface 1 to be p2p interface)

If use driver version 5.11 or later, it supports modify "rtw_sel_p2p_iface" by Makefile. You need to add the "CONFIG_SEL_P2P_IFACE" to your platform dependent configuration section to select p2p interface when driver define "CONFIG_CONCURRENT_MODE".

Ex:

```
ifeq ($(CONFIG_PLATFORM_RTK119X_AM), y)
EXTRA_CFLAGS += -DCONFIG_PLATFORM_RTK119X_AM
EXTRA_CFLAGS += -DCONFIG_LITTLE_ENDIAN
EXTRA_CFLAGS += -DCONFIG_TRAFFIC_PROTECT
EXTRA_CFLAGS += -DCONFIG_CONCURRENT_MODE -DCONFIG_FULL_CH_IN_P2P_HANDSHAKE
EXTRA_CFLAGS += -DCONFIG_SEL_P2P_IFACE=2
EXTRA_CFLAGS += -DCONFIG_IFACE_NUMBER=3
EXTRA_CFLAGS += -DCONFIG_IOCTL_CFG80211 -DRTW_USE_CFG80211_STA_EVENT

ifeq ($(CONFIG_USB_HCI), y)
EXTRA_CFLAGS += -DCONFIG_USE_USB_BUFFER_ALLOC_TX
endif
```

(This example select interface 2 to be p2p interface, otherwise default use interface 1 to be p2p interface)

Note: The method from the previous paragraph has higher priority than this method.

2. Scanning P2P Device

After enabling the P2P functionality of the Wi-Fi driver, the P2P device1 got to find out how many other P2P devices exist in the environment. We can do the this via wpa_supplicant command.

Ex: #> wpa_cli p2p_find

Ex: #> wpa_cli p2p_listen

3. Start Group Negotiation + Provisioning

In the Wi-Fi Direct scenario, one of the P2P devices will become a group owner (almost the same as the SoftAP) and the other P2P device will become an 802.11 client to connect to that group owner. The stage3 "Start Group Negotiation" is the procedure to determine which P2P device should be the group owner/client.

After confirming the role for both P2P Device1 and P2P Device2, the P2P device the wpa_supplicant in the background and use the PIN CODE or PBC to perform the WPS procedure.

"go_intent" is a value from 0 ~ 15. This value will provide the degree information to want to be the group owner. "intent=15" means this Wi-Fi driver must be the group owner. The default intent value is 1.

"pbc" is wps config method. It can be "pbc" or "pin"

Ex: #> wpa_cli -i wlan0 p2p_connect 00:e0:4c:02:80:6e pbc go_intent=7

4. DHCP

The Wi-Fi Direct Specification required that the P2P device which becomes the group owner should also provide the DHCP server application in their system. The DHCP server should be launched and be ready to provide the IP address to the DHCP client. The specification also required that the P2P device which becomes the P2P client should launch the DHCP client application to acquire the IP address from the P2P group owner after the wpa_supplicant established the 802.11 connection with AP successfully.

(3)   Support version:

- cfg80211 only

(4) How to start wpa_supplicant:

- Please reference another document under document folder. (wpa_cli_with_wpa_supplicant.pdf)

(5) P2P commands

Note: Can reference "README-P2P" file under wpa_supplicant folder

Device Discovery

Command	Description
p2p_find [timeout in seconds]	Enables discovery – start sending probe request frames
p2p_stop_find	Stops discovery, or whatever you are doing (listen mode, connection process etc.)
p2p_listen [timeout in seconds]	Enable listen mode

Device Discovery : Discovered Peers List

Command	Description
p2p_peers [discovered]	Shows list of discovered peers (with 'discovered' – shows only fully discovered peers)
p2p_peer <P2P Device Address>	Show detailed information about discovered peers
p2p_flush	Flush p2p_state, and clears the discovered peer list

Group Formation

Command	Description
p2p_connect <peer device address> <pbc/pin> [GO_intent=<0-15>] [auth/join]	GO_intent – initiate connection to another device (using entered group intent) Auth – WPS authorize incoming connection Join – connect to an existing GO No input – initiate connection using default GO intent
p2p_group_add [freq=<freq in MHz>]	Become an autonomous GO freq=<freq in MHz> can be used to force the GO to be started on a specific

	frequency.
p2p_group_remove	Remove device from group, return to device mode if acting as GO or autonomous GO

Group Formation : GO WPS authorizations

Command	Description
wps_pbc	Start WPS PBC method
wps_pin <any/address> <PIN>	Start WPS PIN method

Others

Command	Description
p2p_prov_disc <peer device address> <display/keypad/pbc> [join/auto]	Send P2P provision discovery request to the specified peer.
p2p_invite	Invite a peer to join a group or to reinvoke a persistent group.

(6)   P2P Use case

Case 1 : Connect in PBC (Push button Control)

Case 1-1 : DUT #1 & DUT #2 use p2p_find to find out each other

Step #	DUT #1	DUT #2	Comments
1	p2p_find 30	p2p_find 30	Find p2p device
2	p2p_peers	p2p_peers	verify p2p candidates MAC ADDRESS
3	p2p_connect <DUT#2_MAC_ADDRESS> pbc go_intent=7		go_intent=7 means that there a same chance for both DUTs to become GO go_intent= 15 means that DUT will become GO go_intent= 0 means that DUT will become Client
4		p2p_connect <DUT#1_MAC_ADDRESS> pbc	

DUT_1_case_1-1	DUT_2_case_1-1
[root]# ./wpa_cli -i wlan0 **status**	**[root]#** ./wpa_cli -i wlan0 status
wpa_state=DISCONNECTED	wpa_state=DISCONNECTED
p2p_device_address=00:e0:4c:02:80:45	p2p_device_address=00:e0:4c:02:80:6e
address=00:e0:4c:02:80:45	address=00:e0:4c:02:80:6e
[root]# ./wpa_cli -i wlan0 **p2p_find** 30	**[root]#** ./wpa_cli -i wlan0 **p2p_find** 30
OK	OK
[root]# ./wpa_cli -i wlan0 **p2p_peers**	**[root]#** ./wpa_cli -i wlan0 **p2p_peers**
00:e0:4c:02:80:6e	00:e0:4c:02:80:45
[root]# ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:6e pbc go_intent=7	
OK	
	[root]# ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:45 pbc
	OK
[root]# ./wpa_cli -i wlan0 **status**	**[root]#** ./wpa_cli **status**
bssid=00:e0:4c:02:80:6e	bssid=00:e0:4c:02:80:6e
ssid=DIRECT-Cc	ssid=DIRECT-Cc
id=2	id=1
mode=station	mode=P2P GO
pairwise_cipher=CCMP	pairwise_cipher=CCMP
group_cipher=CCMP	group_cipher=CCMP
key_mgmt=WPA2-PSK	key_mgmt=WPA2-PSK
wpa_state=COMPLETED	wpa_state=COMPLETED
ip_address=192.168.42.33	ip_address=192.168.42.1
p2p_device_address=00:e0:4c:02:80:45	p2p_device_address=00:e0:4c:02:80:6e
address=00:e0:4c:02:80:45	address=00:e0:4c:02:80:6e

Case 1-2 : DUT #2 listen only DUT #1 use p2p_find to find out DUT #2

Step #	DUT #1	DUT #2	Comments
1	p2p_find 30	p2p_listen 60	Find p2p device
2	p2p_peers		verify p2p candidates MAC ADDRESS
3	p2p_connect <DUT#2_MAC_ADDRESS> pbc		go_intent=7 means that there a same chance for both DUTs to become GO go_intent= 15 means that DUT will become GO go_intent= 0 means that DUT will become Client
4		p2p_connect <DUT#1_MAC_ADDRESS> pbc	

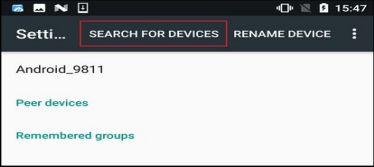| DUT_1_case_1-2	DUT_2_case_1-2
[root]# ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:6e **[root]#** ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:6e pbc OK **[root]#** ./wpa_cli -i wlan0 **status** bssid=00:e0:4c:02:80:6e ssid=DIRECT-5e id=2 mode=station pairwise_cipher=CCMP group_cipher=CCMP key_mgmt=WPA2-PSK wpa_state=COMPLETED ip_address=192.168.42.33 p2p_device_address=00:e0:4c:02:80:45 address=00:e0:4c:02:80:45	**[root]#** ./wpa_cli -i wlan0 **p2p_listen** 60 OK **[root]#** ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:45 pbc OK **[root]#** ./wpa_cli –i wlan0 **status** bssid=00:e0:4c:02:80:6e ssid=DIRECT-5e id=1 mode=P2P GO pairwise_cipher=CCMP group_cipher=CCMP key_mgmt=WPA2-PSK wpa_state=COMPLETED ip_address=192.168.42.1 p2p_device_address=00:e0:4c:02:80:6e address=00:e0:4c:02:80:6e

Case 1-3 : DUT #2 is Android device and use UI to connect P2P

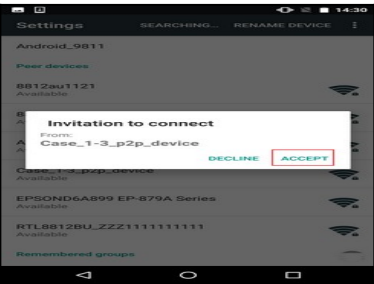| Step #	DUT #1	DUT #2 (Android device)	Comments
		Enter Wi-Fi Direct or P2P page	
1	p2p_find 30	Push "SEARCH FOR DEVICES"	Find p2p device
2	p2p_peers		verify p2p candidates MAC ADDRESS
3	p2p_connect <DUT#2_MAC_ADDRESS> pbc		
4		Push "ACCEPT"	
			Note: When DUT#1 is GO. It maybe need to start a DHCP server to assign a IP to Android device. (Some Android device no IP will trigger disconnect)

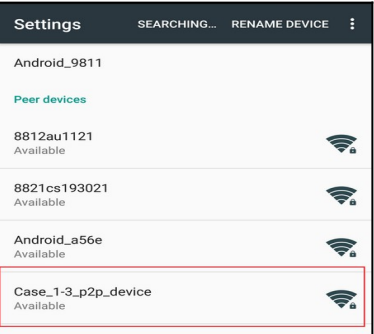| DUT_1 case_1-3	DUT_2_case_1-3
address=00:e0:4c:05:50:c1	1.
[root]# ./wpa_cli -i wlan0 **set device_name** Case_1-3_p2p_device OK	2.
[root]# ./wpa_cli -i wlan0 **p2p_find** 30 OK	
e0:4c:02:80:6e pbc	3.
	4.
mode=P2P GO pairwise_cipher=CCMP group_cipher=CCMP key_mgmt=WPA2-PSK wpa_state=COMPLETED p2p_device_address=00:e0:4c:05:50:c1 address=00:e0:4c:05:50:c1	

Case 2 : Connect in PBC (Push button Control)

where DUT #1 is defined as the Auto Group Owner

Step #	DUT #1	DUT #2	Comments
1	p2p_find 30	p2p_find 30	
2	p2p_peers	p2p_peers	verify p2p candidates MAC ADDRESS
3	p2p_group_add		Define DUT #1 as GO
4	wps_pbc		
5		p2p_connect <DUT#1_MAC_ADDRESS> pbc join	

DUT_1_case_2	DUT_2_case_2
[root]# ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:6e **[root]#** ./wpa_cli -i wlan0 **p2p_group_add** OK **[root]#** ./wpa_cli -i wlan0 **wps_pbc** OK	**[root]#** ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:45 **[root]#** ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:45 pbc join OK

Case 3 : Connect in PIN (PIN Number) where DUT

#1 is defined as the Auto Group Owner

Step #	DUT #1	DUT #2	Comments
1	p2p_find 30	p2p_find 30	
2	p2p_peers	p2p_peers	verify p2p candidates MAC ADDRESS
3	p2p_group_add		Define DUT #1 as GO
4	wps_pin any		verify the <DUT#1_PIN_CODE>, will appear on the terminal
5		p2p_connect <DUT#1_MAC_ADDRESS> <DUT#1_PIN_CODE> join	

DUT_1_case_3	DUT_2_case_3
[root]# ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:6e **[root]#** ./wpa_cli -i wlan0 **p2p_group_add** OK **[root]#** ./wpa_cli -i wlan0 **wps_pin any** 3760533	**[root]#** ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:45 **[root]#** ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:45 3760533 join OK

Case 4 : Connection using PIN code

Step #	DUT #1	DUT #2	Comments
1	p2p_find 30	p2p_find 30	
2	p2p_peers	p2p_peers	verify p2p candidates MAC ADDRESS
3	p2p_connect <DUT#2_MAC_ADDRESS> pin auth		verify the <DUT#1_PIN_CODE>, will appear on the terminal
4		p2p_connect <DUT#1_MAC_ADDRESS> <DUT#1_PIN_CODE>	

DUT_1_case_4	DUT_2_case_4
[root]# ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:6e **[root]#** ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:6e pin auth 53617170	**[root]#** ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:45 **[root]#** ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:45 53617170 OK

Case 5: Connect in PBC (Push button Control) and establish a persistent P2P group

Step #	DUT #1	DUT #2	Comments
1	p2p_find 30	p2p_find 30	
2	p2p_peers	p2p_peers	verify p2p candidates MAC ADDRESS
3	p2p_connect <DUT#2_MAC_ADDRESS> pbc persistent go_intent=7		go_intent=7 means that there a same chance for both DUTs to become GO go_intent= 15 means that DUT will become GO go_intent= 0 means that DUT will become Client
4		p2p_connect <DUT#1_MAC_ADDRESS> pbc persistent	

DUT_1_case_5	DUT_2_case_5
[root]# ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:6e **[root]#** ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:6e pbc persistent go_intent=7 OK	**[root]#** ./wpa_cli -i wlan0 **p2p_find** 30 OK **[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:45 **[root]#** ./wpa_cli -i wlan0 **p2p_connect** 00:e0:4c:02:80:45 pbc persistent OK

Case 6: Re-invoke a persistent P2P group where DUT#1 was the

Group Owner of the persistent P2P group

Step #	DUT #1	DUT #2	Comments
	p2p_find 30	p2p_find	Make **DUT #2** keep in find phase with no time limit
	p2p_peers	p2p_peers	verify p2p candidates MAC ADDRESS
	list_networks	list_networks	verify the persistent P2P group Network ID
	p2p_invite persistent=<Network ID> peer=<DUT#2_MAC_ADDRESS>		
	p2p_find		Make **DUT #1** keep in find phase with no time limit
		p2p_invite persistent=<Network ID> peer=<DUT#1_MAC_ADDRESS>	

DUT_1_case_6	DUT_2_case_6
[root]# ./wpa_cli -i wlan0 **p2p_find** 30 OK	**[root]#** ./wpa_cli -i wlan0 **p2p_find** 30 OK
[root]# ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:6e	**[root]#** ./wpa_cli -i wlan0 **p2p_peers** 00:e0:4c:02:80:45
[root]# ./wpa_cli -i wlan0 **list_networks** network id / ssid / bssid / flags 1 DIRECT-Yj 00:e0:4c:02:80:45 [DISABLED][P2P-PERSISTENT]	**[root]#** ./wpa_cli -i wlan0 **list_networks** network id / ssid / bssid / flags 1 DIRECT-Yj 00:e0:4c:02:80:45 [DISABLED][P2P-PERSISTENT]
[root]# ./wpa_cli -i wlan0 **p2p_invite** persistent=1 peer=00:e0:4c:02:80:6e P2P-INVITATION-RESULT status=1	P2P-INVITATION-RECEIVED sa=00:e0:4c:02:80:45 persistent=1
[root]# ./wpa_cli -i wlan0 **p2p_find** OK	
P2P-INVITATION-ACCEPTED sa=02:e0:4c:08:90:20 persistent=1	**[root]#** ./wpa_cli -i wlan0 **p2p_invite** persistent=1 peer=00:e0:4c:02:80:45 P2P-INVITATION-RESULT status=0 00:e0:4c:02:80:45

(7)   Improve the success rate of P2P nego to be GO

To improve the success rate to be P2P GO, we have to set the go_intent as 14 and set the tie breaker bit. Please follow be low steps.

First, in source code, p2p_connect function. (./src/p2p/p2p.c)
You need patch the red words before the return as below:

```
int p2p_connect(struct p2p_data *p2p, const u8 *peer_addr,
…
{
    …
    dev->tie_breaker = 1;
    return p2p_connect_send(p2p, dev);
}
```

Second, in configure file for P2P, you need add this setting for default value

```
p2p_go_intent=14
```

Finally, in connect flow. Maybe the p2p_connect with the go_intent=14 parameter to overridden the default intent.

```
wpa_cli > p2p_connect <peer device address> pbc go_intent=14
```

If you can not control the go intent by command or configure. You can write the source code, wpa_p2p_connect function (./wpa_supplicant/p2p_supplicant.c)
You need patch the red words before the return as below:

```
int wpas_p2p_connect(struct wpa_supplicant *wpa_s, const u8 *peer_addr,
…
{
    …
    if (go_intent < 0)
        go_intent = wpa_s->conf->p2p_go_intent;
    go_intent = 14;
    …
}
```