Realtek Wi-Fi SDK for Android 14.x
ver. 1.0.0

Contents

Release History

1.0.0	2023/11/30	1. First formal release

SDK packages

- hardware/realtek/*
  Folder to store private code from Realtek.

- supplicant_overlay_configs/*
  Folder to store supplicant overlay config files

Introduction

This document provides a simple guide to help engineers to apply Realtek Wi-Fi solution onto their Android 14.x system. For now, we have supported the following two scenarios:

- **STA/AP** – Switch between STA mode and AP mode
- **(STA+P2P)/AP** – Switch between STA+P2P(Wi-Fi Direct) concurrent mode and AP mode

To port Realtek Wi-Fi driver onto Android 14.x platform, you can go through the following guide with reference codes within our driver package's realtek_wifi_SDK_for_android_14_x_<date>.tgz.

Because Android's SDK may differ from platform to platform, our reference codes may not be applied on every platform without modifications. You should check if our reference code is suitable for you to use.

In this document, $ANDROID_SDK is the path of top folder of the target Android SDK; this term is used in the following paragraphs.

1. **Copy Necessary Files into SDK**

   (1) SDK

   After unzipping realtek_wifi_SDK_for_android_14_x_<date>.tgz, copy the following folder hardware/realtek into $ANDROID_SDK/hardware/folder

   (2) Supplicant overlay config files

   After unzipping supplicant_overlay_config.tar.gz, copy the following files in folder **supplicant_overlay_config** into $ANDROID_SDK/devices/$vendor/$target.

   If supplicant_overlay_config files already exist in $ANDROID_SDK/devices/$vendor/$target, please add contents respectively:

   $ANDROID_SDK/devices/$vendor/$target/wpa_supplicant_overlay.conf

   ```
   p2p_disabled=1
   ```

   $ANDROID_SDK/devices/$vendor/$target/p2p_supplicant_overlay.conf

   ```
   p2p_no_group_iface=1
   ```

   Please refer to Chapter 2.3. Others - Supplicant_overlay config file for further details

2. **Platform Related Files**

   Reference Platform
   Realtek Stark - DHC 1619 platform

2.1 BoardConfig.mk

To apply Realtek Wi-Fi solution onto your Android system, you need to define some compile-time variables in BoardConfig.mk of your platform.

In general, the BoardConfig.mk file is located in:

$ANDROID_SDK /device/<soc_vendor_name>/<board_name>/

If we use Realtek DHC 1619 platform as example, it is located at

$ANDROID_SDK/device/realtek/stark/BoardConfig.mk

```
BOARD_WIFI_VENDOR := realtek
ifeq ($(BOARD_WIFI_VENDOR), realtek)
  WPA_SUPPLICANT_VERSION := VER_0_8_X
  BOARD_WPA_SUPPLICANT_DRIVER := NL80211
  BOARD_WPA_SUPPLICANT_PRIVATE_LIB := lib_driver_cmd_rtl
  BOARD_HOSTAPD_DRIVER := NL80211
  BOARD_HOSTAPD_PRIVATE_LIB := lib_driver_cmd_rtl
  BOARD_WLAN_DEVICE := realtek
endif
WIFI_HIDL_UNIFIED_SUPPLICANT_SERVICE_RC_ENTRY := true
```

- **BOARD_WIFI_VENDOR := realtek**

To distinguish the platform Wi-Fi device from products of other vendors, we define variable BOARD_WIFI_VENDOR as realtek. This is for compile-time choices to be applied for Realtek Wi-Fi solutions.

- **WPA_SUPPLICANT_VERSION := VER_0_8_X**

For Android 14, please set WPA_SUPPLICANT_VERSION as VER_0_8_X   to use wpa_supplicant_8.

- **BOARD_WPA_SUPPLICANT_DRIVER := NL80211**
- **BOARD_WPA_SUPPLICANT_PRIVATE_LIB := lib_driver_cmd_rtl**
- **BOARD_HOSTAPD_DRIVER := NL80211**
- **BOARD_HOSTAPD_PRIVATE_LIB := lib_driver_cmd_rtl**

We use NL80211 as the driver interface for wpa_supplicant and hostapd to communicate with driver and provide lib_driver_cmd_rtl as the private library.

- **BOARD_WLAN_DEVICE**

BOARD_WLAN_DEVICE is used to choose which vendor wifi_hal should be applied. You have to set BOARD_WLAN_DEVICE := realtek to use realtek's wifi_hal.

- **WIFI_HIDL_UNIFIED_SUPPLICANT_SERVICE_RC_ENTRY**

Set WIFI_HIDL_UNIFIED_SUPPLICANT_SERVICE_RC_ENTRY := true to use unified wpa_supplicant run-command script provided by android.

For more detail, please refer to chapter 2.2 – wpa_supplicant

2.2 init.xxx.rc

For Wi-Fi to operate properly, we need some actions and daemons to be defined as service inside init.xxx.rc. In general, the init.xxx.rc file is located in:

$ANDROID_SDK/device/<soc_vendor_name>/
<board_name>/ If we use Realtek DHC 1619 platform as example,
it is located at:

$ANDROID_SDK/device/realtek/stark/common/init.stark.rc

Please add the service definitions below:

● **mkdir for wpa_supplicant and copy config file**

Please make sure the directories used by the Wireless and related wpa_supplicant config filess will be created and copied in init rc.

```
on zygote-start
# Create the directories used by the Wireless subsystem
# Copy wpa_supplicant config files to related dir
   mkdir /data/misc/wifi 0770 wifi wifi
   mkdir /data/misc/wifi/wpa_supplicant 0770 wifi wifi
   mkdir /data/vendor/wifi 0771 wifi wifi
   mkdir /data/vendor/wifi/wpa 0770 wifi wifi
   mkdir /data/vendor/wifi/wpa/sockets 0770 wifi wifi
   mkdir /data/misc/dhcp 0770 dhcp dhcp
   chown dhcp dhcp /data/misc/dhcp
```

● **insmod**

Please select one of action definitions below according to your requirement. Meanwhile, please make sure your wlan.ko has right seclabel and mode as

```
   -rw-r--r-- 1 root root  u:object_r:vendor_file:s0 3429448 1970-01-01 00:19 wlan.ko
```

You can use command "ls -alZ" to check seclabel of wlan.ko.

(For concurrent mode)

```
on boot
    exec u:r:vendor_modprobe:s0 root root -- /vendor/bin/toybox_vendor insmod
   /vendor/modules/wlan.ko ifname=wlan0 if2name=p2p0
```

- **wpa_supplicant**

  Please use android.hardware.wifi.supplicant-service.rc to launch wpa_supplicant service.

  You should use wpa_supplicant/aidl/android.hardware.wifi.supplicant-service.rc as run command

  ```
  service wpa_supplicant /vendor/bin/hw/wpa_supplicant \
       -O/data/vendor/wifi/wpa/sockets -dd \
       -g@android:wpa_wlan0
       #   we will start as root and wpa_supplicant will switch to user wifi
       #   after setting up the capabilities required for WEXT
       #   user wifi
       #   group wifi inet keystore
       interface aidl android.hardware.wifi.supplicant.ISupplicant/default
       class main
       socket wpa_wlan0 dgram 660 wifi wifi
       disabled
       oneshot
  ```

2.3 Others

For topics mentioned here, you can add the following code segments in any .mk file which your platform will use.

- **Add wifi related packages**

These packages are needed for Wifi support, please make sure these packages are added in .mk

If we use Realtek DHC 1619 platform as example, it is located at
$ANDROID_SDK/device/realtekstark/device.mk

```
    PRODUCT_PACKAGES += \
libwpa_client wpa_supplicant hostapd wificond wifilogd wpa_supplicant.conf \
hostapd.conf libwifi-hal \
android.hardware.wifi.supplicant \
android.hardware.wifi.hostapd \
android.hardware.wifi-service \
```

● **Add android.hardware.wifi.xml**

To claim Wi-Fi support for your device, please add the rule in the PRODUCT_COPY_FILES variable to copy the permission definition file of Wi-Fi to the /system/etc/permissions/ folder of your system image.
If we use Realtek DHC 1619 platform as example, it is located at $ANDROID_SDK/device/realtek/stark/common/deviceCommon.mk

```
PRODUCT_COPY_FILES += \

  frameworks/native/data/etc/android.hardware.wifi.xml:system/etc/permissions/android.hardware.wifi.xml
```

● **Add android.hardware.wifi.direct.xml**

To claim Wi-Fi Direct (P2P) support for your device, please add the rule in the PRODUCT_COPY_FILES variable to copy the permission definition file of Wi-Fi Direct to the /system/etc/permissions/ folder of your system image.

```
PRODUCT_COPY_FILES += \

  frameworks/native/data/etc/android.hardware.wifi.direct.xml:system/etc/permissions/android.hardware.wifi.direct.xml
```

Make sure your driver is configured for STA+P2P concurrent mode or you may encounter error when you open the Wi-Fi. Please refer to **"6. Driver Configurations for Android"**

- **Set wifi.interface**

To specify the wifi interface name in Android, a system property named "wifi.interface" is used. For Realtek Wi-Fi driver, Wi-Fi interface name is assigned with "wlan%d". In general, you should set wifi.interface as "wlan0".

```
PRODUCT_PROPERTY_OVERRIDES += \
        wifi.interface=wlan0
```

- **Set wifi.direct.interface**

If you require p2p support, you have to set wifi.direct.interface as "p2p0"

```
PRODUCT_PROPERTY_OVERRIDES += \
        wifi.direct.interface=p2p0
```

- **wifi_hal**

Android 14 using Android.bp to build libwifi-hal
board_wlan_device realtek should be already in
$ANDROID_SDK/frameworks/opt/net/wifi/libwifi_hal/Android.bp
as below :

```
wifi_cc_defaults {
    name: "libwifi_hal_vendor_impl_defaults",
    soong_config_variables: {
       board_wlan_device: {
          bcmdhd: {
             whole_static_libs: ["libwifi-hal-bcm"],
          },
          ....
          realtek: {
             whole_static_libs: ["libwifi-hal-rtk"],
          },
          emulator: {
             whole_static_libs: ["//device/generic/goldfish:libwifi-hal-emu"],
          }
    ....
```

- **Device manifest file**

The Device manifest file is provided by the device. It lives in the Android source tree at device/${VENDOR}/${DEVICE}/manifest.xml and on the device at /vendor/manifest.xml

If we use Realtek DHC 1619 platform as example, it is located at $ANDROID_SDK/device/realtek/stark/common/manifest.xml

RTK wifi needs Iwifi，ISupplicant and IHostapd. Not like Android 10 or earlier versions, in Android 14, these should be declared at
$ANDROID_SDK/hardware/interface/wifi/aidl/default/android.hardware.wifi-service.xml
$ANDROID_SDK/external/wpa_supplicant_8/wpa_supplicant/aidl/android.hardware.wifi.supplicant.xml
$ANDROID_SDK/external/wpa_supplicant_8/hostapd/android.hardware.wifi.hostapd.xml

```
<manifest version="1.0" type="device">
  <hal format="aidl">
     <name>android.hardware.wifi</name>
      <fqname>IWifi/default</fqname>
   </hal>


  <hal format="aidl">
     <name>android.hardware.wifi.supplicant</name>
     <fqname>ISupplicant/default</fqname>
   </hal>


  <hal format="aidl">
     <name>android.hardware.wifi.hostapd</name>
     <fqname>IHostapd/default</fqname>
   </hal>
</manifest>
```

- **Supplicant_overlay config file**

After "1. Copy Necessary Files into SDK." accomplished, you should find out
wpa_supplicant_overlay.conf and **p2p_supplicant_overlay.conf** in folder
$ANDROID_SDK/devices/${VENDOR}/${TARGET}.
Please modify $ANDROID_SDK/devices/${VENDOR}/${TARGET}/BoardConfig.mk as below.

```
PRODUCT_COPY_FILES += device/${VENDOR}/$
{TARGET}/wpa_supplicant_overlay.conf:$(TARGET_COPY_OUT_VENDOR)/etc/
wifi/wpa_supplicant_overlay.conf

PRODUCT_COPY_FILES += device/${VENDOR}/$
{TARGET}/p2p_supplicant_overlay.conf:$(TARGET_COPY_OUT_VENDOR)/etc/
wifi/p2p_supplicant_overlay.conf
```

Please make sure that **wpa_supplicant_overlay.conf** and **p2p_supplicant_overlay.conf**
exist in **/vendor/etc/wifi** on device.
Meanwhile,
"p2p_disabled=1" should be in **wpa_supplicant_overlay.conf**
"p2p_no_group_iface=1" should be in **p2p_supplicant_overlay.conf**

If you need wowlan function, the below line wowlan_triggers=any should be added in
your wpa_supplicant_overlay.conf

3.   System Resource Configurations

You should set the following four resource configurations for your platform to
configure the network function and enable the corresponding UI interface. In general,
you can set the following configurations in your platform dependent config.xml file.
If we use Realtek DHC 1619 platform as example, it is located at
   $ANDROID_SDK/frameworks/base/core/res/res/values/config.xml
Or the global config.xml file:
   $ANDROID_SDK/frameworks/base/core/res/res/values/config.xml

- **networkAttributes**

To define the system's available network interfaces, make sure the wifi interface
items is defined in the networkAttributes resource configuration in the config.xml. For
example:

```
<string-array translatable="false" name="networkAttributes">

    <item>"wifi,1,1,1,-1,true"</item>

    <item>"bluetooth,7,7,0,-1,true"</item>

    <item>"ethernet,9,9,2,-1,true"</item>

</string-array>
```

- **radioAttributes**

To define the system's available network interfaces, we need to define interface items for
wifi in the radioAttributes resource configuration. For example:

```
<string-array translatable="false" name="radioAttributes">

    <item>"1,1"</item>
                                            12
    <item>"7,1"</item>

    <item>"9,1"</item>

</string-array>
```

- **config_tether_wifi_regexs**

The interfaces set here are tetherable Wi-Fi interfaces which will be used as interfaces for Wi-Fi LAN port. We use 'wlan0' by default when our Wi-Fi is set as softap mode. So it needs to set 'wlan0' here. For example:

```
<string-array translatable="false" name="config_tether_wifi_regexs">
    <item>"wlan0"</item>
</string-array>
```

- **config_tether_upstream_types**

The connection types set here are used as the interfaces for WAN port to connect to internet. For example, adding Wi-Fi and Ethernet:

```
<integer-array translatable="false" name="config_tether_upstream_types">
    <item>1</item>
    <item>9</item>
</integer-array>
```

At least one item should be declared here to enable the "Tehtering & portable hotspot" option of WirelessSettings in Settings.apk.

To know the definition and set other upstream connection types, please refer to ANDROID_SDK/frameworks/base/core/java/android/net/ConnectivityManager.java.

- **config_enableWifiDisplay**

To enable Wi-Fi Display(Miracast) function, set config_enableWifiDisplay to true:

```
<bool name="config_enableWifiDisplay">true</bool>
```

4. **wpa_supplicant_8**

   We would not provide wpa_supplicant_8.

   Please use android official release:

   https://android.googlesource.com/platform/external/wpa_supplicant_8/+/refs/
   heads/android14-release

5. **GKI (Generic Kernel Image)**

   Android 11 introduced GKI and KMI(kernel module interface) in kernel 5.4.
   Therefore, kernel API kernel_path(), kernel_write() and kernel_read() were
   prohibited if kernel 5.4 or higher version is used on your android.

   **If kernel module needs to load parameters from the parameter files, the relative
   parameter files must be put under the directory: /vendor/firmware**.

   You cannot specify driver REALTEK_CONFIG_PATH through Makefile

   For example, if you want to load TX Power By Rate and Limit from file system, you
   have to put PHY_REG_PG.txt and TXPWR_LMT.txt under the directory
   /vendor/firmware on DUT

6. Driver Configurations for Android

Android 11.x support two scenarios for Wi-Fi solution:
- **STA/AP – Switch between STA and AP mode**
- **(STA+P2P)/AP – Switch between STA+P2P concurrent and AP mode**

The configuration of driver to fit the requirement of each scenario, see the following table:

MACRO	STA /AP	(STA+P2P)/AP	Kernel ver.
CONFIG_IOCTL_CFG80211	Defined	Defined	ver. >= 2.6.35
RTW_USE_CFG80211_STA_EVENT	Defined	Defined	ver. >= 3.2.0
CONFIG_RADIO_WORK	Defined	Defined	-
CONFIG_CONCURRENT_MODE	Undefined	Defined	-
RTW_ENABLE_WIFI_CONTROL_FUNC	Defined for platform device/driver mechanism		
CONFIG_RTW_WIFI_HAL	Defined if android version >= 8.x		ver. >= 3.18
CONFIG_RTW_ANDROID	Must set this when rtk driver ver >= v5.9, please refer to section 6.1		
CONFIG_SEL_P2P_IFACE	Undefined	Defined with Wi-Fi 5 IC	
CONFIG_CFG80211_REPORT_PROBE_REQ	Define this macro if using Wi-Fi 6 IC. When defining this macro, CONFIG_SEL_P2P_IFACE should be undefined		

- **CONFIG_IOCTL_CFG80211** is used for driver to enable cfg80211 ioctl interface, which is required by Realtek Wi-Fi to operate on Android system.

- **RTW_USE_CFG80211_STA_EVENT** is used for driver to indicate new cfg80211 STA event, which is required by wpa_supplicant_8 on Android. Linux kernel supports this feature after kernel 3.2.

- **CONFIG_RADIO_WORK** is used for driver to fit 'radio work' mechanism of Android's wpa_supplicant_8. If this MACRO doesn't exist in driver's source code, please contact with Realtek technical windows for suitable driver.
  If you use rtk driver ver. >= v5.9, you do not need to set this compile flag.
  Instead, you have to set CONFIG_RTW_ANDROID according to section "6.1 CONFIG_RTW_ANDROID"

- **CONFIG_CONCURRENT_MODE** is used for driver to enable concurrent mode, which is required by STA+P2P concurrent mode.

- **RTW_ENABLE_WIFI_CONTROL_FUNC** is used to register platform driver callbacks. If your platform needs those callbacks, please define this macro to register platform driver callback functions. For example, these functions include:

```
static struct platform_driver wifi_device = {
      .probe = wifi_probe,
      .remove = wifi_remove
}
```

By default, the probe callback is used to set up Wi-Fi power and remove callback is used to close Wi-Fi power.

To compile Realtek Wi-Fi driver with the above setting, please refer to the following document:
document/Quick_Start_Guide_for_Driver_Compilation_and_Installation.pdf Adding platform selection and setting sections for compilation settings of your platform.

For example, if you want to configure Realtek Wi-Fi driver for the (STA+P2P)/AP scenario, make sure the macros: CONFIG_IOCTL_CFG80211, RTW_USE_CFG80211_STA_EVENT, CONFIG_RADIO_WORK and CONFIG_CONCURRENT_MODE are defined into the EXTRA_CFLAGS settings as following:

```
CONFIG_PLATFORM_ANDROID_M60_SAMPLE = y

…

…

…

ifeq ($(CONFIG_PLATFORM_ANDROID_ML0_SAMPLE), y)

EXTRA_CFLAGS += -DCONFIG_LITTLE_ENDIAN

EXTRA_CFLAGS += -DCONFIG_CONCURRENT_MODE

EXTRA_CFLAGS += -DCONFIG_IOCTL_CFG80211 -

DRTW_USE_CFG80211_STA_EVENT EXTRA_CFLAGS += -DCONFIG_RADIO_WORK

ARCH := arm

CROSS_COMPILE := /toolchain/bin/arm-none-linux-gnueabi-

KSRC := / android_sdk/android_l/

kernel endif
```

- **CONFIG_RTW_WIFI_HAL** is defined if android version is >= 8.x :

For supporting Android version >= 8.x, make sure CONFIG_RTW_WIFI_HAL is set to "y" in Makefile as follows,

```
…
CONFIG_RTW_WIFI_HAL = y
…
```

If you use rtk driver ver. >= v5.9, you do not need to set this compile flag.
Instead, you have to set CONFIG_RTW_ANDROID according to section "6.1 CONFIG_RTW_ANDROID"

According to Google's suggestion, you must use kernel specified on Android Common Kernel.
For more detail, you can refer
https://source.android.com/docs/core/architecture/kernel/android-common

● CONFIG_RTW_ANDROID

From Wifi driver version 5.9, a new setting CONFIG_RTW_ANDROID is added in Makefile, We can set CONFIG_RTW_ANDROID with the Android version in Makefile. e.g. CONFIG_RTW_ANDROID = 14

Please note that we must set CONFIG_RTW_ANDROID with correct Android version from wifi driver version 5.9, otherwise there will be problem in wifi driver for Android. And the default value of CONFIG_RTW_ANDROID is 0, which means the driver is for pure linux, not Android.

(CONFIG_RTW_ANDROID=4 means Android 4.4)

Example in Makefile:

```
########################## Android ##########################
# CONFIG_RTW_ANDROID - 0: no Android, 4/5/6/7/8/9/10/11 : Android version
CONFIG_RTW_ANDROID = 14

ifeq ($(shell test $(CONFIG_RTW_ANDROID) -gt 0; echo $$?), 0)
EXTRA_CFLAGS += -DCONFIG_RTW_ANDROID=$(CONFIG_RTW_ANDROID)
endif
```

Then most of the settings mentioned above are set automatically by Android version (CONFIG_RTW_ANDROID) in drv_conf.h, and we don't need to write these setting in Makefile; Except CONFIG_CONCURRENT_MODE and RTW_ENABLE_WIFI_CONTROL_FUNC still need to be set manually, depends on the platforms in Makefile as above as before.
drv_conf.h

```
#define CONFIG_IOCTL_CFG80211

#define RTW_USE_CFG80211_STA_EVENT

#if (CONFIG_RTW_ANDROID > 4)
#ifndef CONFIG_RADIO_WORK
#define CONFIG_RADIO_WORK
#endif
#endif

#if (CONFIG_RTW_ANDROID >= 8)
        #if (LINUX_VERSION_CODE >= KERNEL_VERSION(3,18,0))
        #ifndef CONFIG_RTW_WIFI_HAL
        #define CONFIG_RTW_WIFI_HAL
        #endif
        #else
        #error "Linux kernel version is too old\n"
        #endif
#endif
```

- **CONFIG_SEL_P2P_IFACE** is defined if using Wi-Fi 5 IC.

It is used to fix which interface can use the P2P function.

```
# Fix second interface (p2p0) can use the P2P function
EXTRA_CFLAGS += -DCONFIG_SEL_P2P_IFACE=1
```

- **CONFIG_CFG80211_REPORT_PROBE_REQ** is defined if using Wi-Fi 6 IC.

This macro allows the driver to report probe request frames to wpa_supplicant. With this macro enabled, CONFIG_SEL_P2P_IFACE will no longer be needed. Which interface can use the P2P function is determined by wpa_supplicant, so make sure your supplicant_overlay.conf is used correctly (refer to "Error: Reference source not found").
If you define both CONFIG_SEL_P2P_IFACE and CONFIG_CFG80211_REPORT_PROBE_REQ, the driver will have compilation errors. You should remove the definition of CONFIG_SEL_P2P_IFACE when the following compilation error occurs.

```
#error "CONFIG_SEL_P2P_IFACE has no effect when CONFIG_CFG80211_REPORT_PROBE_REQ is enabled"
```

7. FAQ

7.1. Wi-Fi (STA mode)

7.1.1. Why Wi-Fi can't enable?

The whole Wi-Fi enabling procedure includes the following three main check points. Please check in sequence:

- Is network interface(s) created?
    - insmod driver success
    - Wi-Fi device is recognized
    - wlan0 is created
- Does wpa_supplicant run successfully?
    - wpa_supplicant.conf (and p2p_supplicant.conf) exists and is correct
    - Service definition of wpa_supplicant exists and is correct
    - Binary file wpa_supplicant exists and is executable
- Do connections of communication socket setup?
    - Make sure the communication socket settings is matched below:
        - ctrl_interface in:
          /data/misc/wifi/wpa_supplicant.conf
          (and /data/misc/wifi/p2p_supplicant.conf)
        - Service definition of wpa_supplicant
        - Paths of communication socket in wifi.c

7.2. Portable Wi-Fi hotspot (AP mode)

7.2.1. Why Portable Wi-Fi hotspot can't enable?

The whole Portable Wi-Fi hotspot enabling procedure includes the following three main check points. Please check in sequence:

- Is network interface created?
  - insmod driver success
  - Wi-Fi device is recognized
  - wlan0 is created
- Does netd and hostapd run successfully?
  - /data/misc/wifi/hostapd.conf exists and is correct
  - Binary file netd and hostapd exist and are executable
- Does dnsmasq run successfully?
  - Binary file dnsmasq exist and are executable

7.3. Wi-Fi Direct (P2P mode)

7.3.1. There is no Wi-Fi Direct UI shown?

Please refer to "**Add android.hardware.wifi.direct.xml**" in chapter **2.3. Others** to enable Wi-Fi Direct functionality of Android 14.

7.3.2. Wi-Fi Direct can't scan any peer?

First, make sure you have workable Wi-Fi Direct device nearby. Make them into Wi-Fi Direct scanning state. Push "SEARCH FOR DEVICES" button also in our device and wait for a while.

If there is still no peer shown the problem is usually caused by wrong service definition of wpa_supplicant services. Please refer to "**wpa_supplicant**" in chapter **2.2. init.xxx.rc** to check your service definition of wpa_supplicant.