MTK Wi-Fi STA Software Programming Guide

Version: 1.1
Release date: 2012-10-31

© 2008 - 2012 MediaTek Inc.
This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

Specifications are subject to change without notice.

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

Document Revision History

Revision	Date	Author	Description
1.0	2012/10/29	Pan Liu	Initial Version
1.1	2012/10/31	Pan Liu	Add sample code

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

Table of Contents

MEDIATEK CONFIDENTIAL
FOR 2950901@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

MEDIATEK CONFIDENTIAL
FOR 2250201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

MEDIATEK CONFIDENTIAL
FOR 2990201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

1 Introduction

This document is the Software programming guide for Mediatek Wi-Fi STA driver. The Software Programming guide covers profile setting, command list, and OID examples to demonstrate how to programming the WLAN driver.

MEDIATEK CONFIDENTIAL
FOR 2950201@ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2 Wi-Fi STA driver Profile Default Setting

```
#The word of "Default" must not be removed
Default
CountryRegion=5
CountryRegionABand=7
CountryCode=
ChannelGeography=1
SSID=11n-AP
NetworkType=Infra
WirelessMode=5
Channel=0
BeaconPeriod=100
TxPower=100
BGProtection=0
TxPreamble=0
RTSThreshold=2347
FragThreshold=2346
TxBurst=1
PktAggregate=0
WmmCapable=1
AckPolicy=0;0;0;0
AuthMode=OPEN
EncrypType=NONE
WPAPSK=
DefaultKeyID=1
Key1Type=0
Key1Str=
Key2Type=0
Key2Str=
Key3Type=0
Key3Str=
Key4Type=0
Key4Str=
PSMode=CAM
AutoRoaming=0
RoamThreshold=70
APSDCapable=0
APSDAC=0;0;0;0
HT_RDG=1
HT_EXTCHA=0
HT_OpMode=0
HT_MpduDensity=4
HT_BW=1
HT_AutoBA=1
HT_BADecline=0
HT_AMSDU=0
HT_BAWinSize=64
HT_GI=1
HT_MCS=33
HT_MIMOPSMode=3
HT_DisallowTKIP=1
HT_STBC=0
```

FOR MEDIATEK CONFIDENTIAL
2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

IEEE80211H=0
WirelessEvent=0
CarrierDetect=0
AntDiversity=0
BeaconLostTime=4
PSP_XLINK_MODE=0
WscManufacturer=
WscModelName=
WscDeviceName=
WscModelNumber=
WscSerialNumber=
RadioOn=1
Wsc4digitPinCode=0

2.1 WLAN Profile Description

Syntax is 'Param'='Value' and describes below.

SectionNumber **Param**
Value
. . .
. . .
. . .

The WLAN driver needs to be restart, after WLAN profile has been modified. Otherwise settings will not take any effect.

A interface down/ up could do that.
EX:

ifconfig ra0 down
ifconfig ra0 up

2.1.1 CountryRegion

Description: Country region for WLAN radio 2.4G HZ regulation.
Value:

CountryRegion=5

Region	Channels
0	1-11
1	1-13
2	10-11
3	10-13
4	14
5	1-14
6	3-9
7	5-13
31	1-14
32	1-11 active scan, 12 and 13 passive scan
33	1-14 all active scan, 14 b mode only

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2.1.2 CountryRegionForABand

Description: Country region for WLAN radio 5G HZ regulation.
Value:

CountryRegionABand=7

Region	Channels
0	36, 40, 44, 48, 52, 56, 60, 64, 149, 153, 157, 161, 165
1	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140
2	36, 40, 44, 48, 52, 56, 60, 64
3	52, 56, 60, 64, 149, 153, 157, 161
4	149, 153, 157, 161, 165
5	149, 153, 157, 161
6	36, 40, 44, 48
7	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 149, 153, 157, 161, 165
8	52, 56, 60, 64
9	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 132, 136, 140, 149, 153, 157, 161, 165
10	36, 40, 44, 48, 149, 153, 157, 161, 165
11	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 149, 153, 157, 161

2.1.3 CountryCode

Description: County Code for WLAN radio regulation.
Value: (Default is empty)

CountryCode=

2 characters, like TW for Taiwan.
Please refer to ISO3166 code list for other countries and can be found at
http://www.iso.org/iso/prods-services/iso3166ma/02iso-3166-code-lists/country_names_and_code_elements

Note:

1. This parameter can be set from EEPRM or EFUSE.
2. EEPROM/EFUSE has higher priority than the WLAN Profile.

2.1.4 ChannelGeography

Description: For Channel list builder
Value:

ChannelGeography=1
0: Outdoor
1: Indoor
2: Both

2.1.5 SSID

Description: The target BSSID string name

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.
MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

Value:

SSID=11n-AP

0~z, 1~32 ASCII characters.

2.1.6 NetworkType

Description: Network type

Value:

NetworkType=Infra

Infra: infrastructure mode
Adhoc: adhoc mode

2.1.7 WirelessMode

Description: Wireless Mode

Value:

WirelessMode=5

0: legacy 11b/g mixed
1: legacy 11B only
2: legacy 11A only
3: legacy 11a/b/g mixed
4: legacy 11G only
5: 11ABGN mixed
6: 11N only
7: 11GN mixed
8: 11AN mixed
9: 11BGN mixed
10: 11AGN mixed
11: 11N only in 5G band only
14: 11A/AN/AC mixed 5G band only (Only 11AC chipset support)

2.1.8 Channel

Description: WLAN Radio channel (2.4G Band or 5G band)

Value:

Channel=0

Depends on CountryRegion or CountryRegionForABand.
Default value = 0, the driver scan BSSID's channel automatically.

2.1.9 BeaconPeriod

Description: Beacon period setting (It is SoftAP only)

Value:

BeaconPeriod=100

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2.1.10 TxPower

Description: WLAN Radio Transmit Power setting in percentage
Value:

TxPower=100

0~100

2.1.11 BGProtection

Description: Enable/disable WLAN 11B or 11G protection
Value:

BGProtection=0

0: disable
1: enable

2.1.12 TxPreamble

Description: Enable or disable Tx preamble
Value:

TxPreamble=0

0: disable
1: enable

2.1.13 RTSThreshold

Description: Set RTS Threshold
Value:

RTSThreshold=2347

1~2347

2.1.14 FragThreshold

Description: Set Fragment threshold
Value:

FragThreshold=2346

256~2346

2.1.15 TxBurst

Description: Enable or disable Tx burst
Value:

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

TxBurst=1

0: disable
1: enable

2.1.16 PktAggregate

Description: Enable or disable Tx Aggregate
Value:

PktAggregate=0

0: disable
1: enable

2.1.17 WmmCapable

Description: Enable or disable WMM QOS function
Value:

WmmCapable=1

0: disable
1: enable

2.1.18 AckPolicy

Description: Ack policy supports normal Ack or no Ack (AC_BK, AC_BE, AC_VI, AC_VO)
Value:

AckPolicy=0;0;0;0

0: No ack
1: normal Ack

2.1.19 AuthMode

Description: WLAN security Authentication mode
Value:

AuthMode=OPEN

OPEN	For open system
SHARED	For shared key system
WEPAUTO	Auto switch between OPEN and SHARED
WPAPSK	For WPA pre-shared key (Infra)
WPA2PSK	For WPA2 pre-shared key (Infra)
WPANONE	For WPA pre-shared key (Adhoc)
WPA	For enterprise mode (Need wpa_supplicant)
WPA2	For enterprise mode (Need wpa_supplicant)

MEDIATEK CONFIDENTIAL
FOR 50201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2.1.20 EncrypType

Description: WLAN security Encryption type

Value:

EncrypType=NONE

NONE	For AuthMode=OPEN
WEP	For AuthMode=OPEN or AuthMode=SHARED
TKIP	For AuthMode=WPAPSK or WPA2PSK
AES	For AuthMode=WPAPSK or WPA2PSK

2.1.21 WPAPSK

Description: WLAN Security string for (TKIP/AES)

Value:

WPAPSK=

8~63 ASCII
Or
64 HEX characters

2.1.22 DefaultKeyID

Description: Default key ID

Value:

DefaultKeyID=1

1~4

2.1.23 Key1Type

Description: Key 1 type

Value:

Key1Type=0

0: Hexadecimal type
1: ASCII type

2.1.24 Key1Str

Description: Key 1 string

Value:

Key1Str=

10 or 26 characters (key type=0)
5 or 13 characters (key type=1)

2.1.25 Key2Type

Description: Key 2 type

MEDIATEK CONFIDENTIAL
FOR 89502201@ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

Value:

Key2Type=0

0: Hexadecimal type
1: ASCII type

2.1.26 Key2Str

Description: Key 2 string

Value:

Key2Str=

10 or 26 characters (key type=0)
5 or 13 characters (key type=1)

2.1.27 Key3Type

Description: Key 3 type

Value:

Key3Type=0

0: Hexadecimal type
1: ASCII type

2.1.28 Key3Str

Description: Key 3 string

Value:

Key3Str=

10 or 26 characters (key type=0)
5 or 13 characters (key type=1)

2.1.29 Key4Type

Description: Key 4 type

Value:

Key4Type=0

0: Hexadecimal type
1: ASCII type

2.1.30 Key4Str

Description: Key 4 string

Value:

Key4Str=

10 or 26 characters (key type=0)
5 or 13 characters (key type=1)

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2.1.31 PSMode

Description: WLAN Power Saving mode
Value:

PSMode=CAM

CAM	Constantly Awake Mode
Max_PSP	Max Power Saving
Fast_PSP	Fast Power Saving
Legacy_PSP	Legacy Power Saving

2.1.32 AutoRoaming

Description: Enable or disable WLAN driver auto roaming between APs.
Value:

AutoRoaming=0

0: disable
1: enable

2.1.33 RoamThreshold

Description: WLAN Roaming RSSI threshold
Value:

RoamThreshold=70

2.1.34 APSDCapable

Description: Enable or disable APSD Capable support
Value:

APSDCapable=0

0: disable
1: enable

2.1.35 APSDAC

Description: Enable or disable APSD access control
Value:

APSDAC=0;0;0;0

0: disable
1: enable

2.1.36 FixedTxMode

Description: Fix Transmit mode
Value:

MEDIATEK CONFIDENTIAL
FOR 20300201@gemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

FixedTxMode=ofdm

CCK
OFDM
HT

2.1.37 HT_RDG

Description: Enable or disable HT Reverse Direction Grant
Value:

HT_RDG=1

0: disable
1: enable

2.1.38 HT_EXTCHA

Description: To locate the 40MHz channel in combination with the control
Value:

HT_EXTCHA=0

0: Below
1: Above

2.1.39 HT_OpMode

Description: HT operation mode
Value:

HT_OpMode=0

0: HT mixed mode
1: HT Greenfield mode

2.1.40 HT_MpduDensity

Description: Minimum separation of MPDUs in an A-MPDU
Value:

HT_MpduDensity=4

0~7
0: no restriction
1: 1/4 μs
2: 1/2 μs
3: 1 μs
4: 2 μs
5: 4 μs
6: 8 μs
7: 16 μs

MEDIATEK CONFIDENTIAL
FOR 950201@ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2.1.41 HT_BW

Description: HT channel Bandwidth
Value:

HT_BW=1

0: 20 MHz
1: 40 MHz

2.1.42 HT_BADecline

Description: Enable or disable decline Block Ack to peer
Value:

HT_BADecline=0

0: disable
1: enable

2.1.43 HT_AutoBA

Description: Enable or disable auto build Block Ack section with peer
Value:

HT_AutoBA=1

0: disable
1: enable

2.1.44 HT_AMSDU

Description: Enable or disable AMSDU section
Value:

HT_AMSDU=0

0: disable
1: enable

2.1.45 HT_BAWinSize

Description: Block Ack window size
Value:

HT_BAWinSize=64

1~64

MEDIATEK CONFIDENTIAL
FOR 2950201@ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2.1.46 HT_GI

Description: HT Guard interval support
Value:

HT_GI=1

0: Long guard interval
1: short guard interval

2.1.47 HT_MCS

Description: WLAN Modulation and Coding Scheme (MCS)
Value:

HT_MCS=33

0 ~15, 32: Fix MCS rate for HT rate.
33: Auto Rate Adaption, recommended

2.1.48 HT_MIMOPSMode

Description: 802.11n SM power save mode
Value:

HT_MIMOPSMode=3

0: Static SM Power Save Mode
2: Reserved
1: Dynamic SM Power Save Mode
3: SM enabled
(not fully support yet)

2.1.49 HT_DisallowTKIP

Description: Enable or disable 11N rate with 11N AP when cipher is TKIP or WEP
Value:

HT_DisallowTKIP=1

0: disable
1: enable

2.1.50 HT_STBC

Description: Enable or disable HT STBC support
Value:

HT_STBC=0

0: disable
1: enable

MEDIATEK CONFIDENTIAL
FOR 29502010@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2.1.51 IEEE80211H

Description: Enable or disable IEEE 802.11h function. Spectrum management. This field can only be enabled in A band.

Value:

IEEE80211H=0

0: disable
1: enable

2.1.52 WirelessEvent

Description: Enable or disable driver to send Linux Wireless event support.

Value:

WirelessEvent=0

0: disable
1: enable

2.1.53 CarrierDetect

Description: Enable or disable carrier detection

Value:

CarrierDetect=0

0: disable
1: enable

2.1.54 AntDiversity

Description: Enable or disable antenna diversity (only 1x1 support, please confirm HW first)

Value:

AntDiversity=0

0: disable
1: enable

2.1.55 BeaconLostTime

Description: Becon lost time = 4 seconds then disconnect with AP.

Value:

BeaconLostTime=4

1~60

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

2.1.56 PSP_XLINK_MODE

Description: Enable or disable XLINK mode
Value:

PSP_XLINK_MODE=0

0: disable
1: enable

2.1.57 WscManufacturer

Description: WPS manufacturer string
Value:

WscManufacturer=

Less than 64 characters

2.1.58 WscModelName

Description: WPS Mode name string
Value:

WscModelName=

Less than 32 characters

2.1.59 WscDeviceName

Description: WPS Device name string
Value:

WscDeviceName=

Less than 32 characters

2.1.60 WscModelNumber

Description: WPS Device model number string
Value:

WscModelNumber=

Less than 32 characters

2.1.61 WscSerialNumber

Description: WPS serial number string
Value:

WscSerialNumber=

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

Less than 32 characters

2.1.62 RadioOn

Description: Enable or disable RF
Value:

RadioOn=1

0: disable
1: enable

2.1.63 Wsc4digitPinCode

Description: WPS 4 digit pin code string
Value:

Wsc4digitPinCode=0

4 digit

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

3 Wi-Fi STA driver iwpriv command

Syntax is iwpriv ra0 set [parameters]=[Value]

Note: Execute one iwpriv/set command at a time.

3.1 Debug

Description: config WLAN driver Debug level.
Value:

iwpriv ra0 set Debug=3

0~5
0: Debug Off
1: Debug Error
2: Debug Warning
3: Debug Trace
4: Debug Info
5: Debug Loud

3.2 DriverVersion

Description: Check driver version by iwpriv command. (Need to enable debug mode)
Value:

iwpriv ra0 set DriverVersion=0

Any value

3.3 CountryRegion

Description: Set Country Region
Value:

iwpriv ra0 set CountryRegion=1

Region	Channels
0	1-11
1	1-13
2	10-11
3	10-13
4	14
5	1-14
6	3-9
7	5-13

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

31	1-14
32	1-11 active scan, 12 and 13 passive scan
33	1-14 all active scan, 14 b mode only

3.4 CountryRegionABand

Description: Set Country Region for 5G Hz WLAN regulation
Value:

iwpriv ra0 set CountryRegionABand=7

Region	Channels
0	36, 40, 44, 48, 52, 56, 60, 64, 149, 153, 157, 161, 165
1	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140
2	36, 40, 44, 48, 52, 56, 60, 64
3	52, 56, 60, 64, 149, 153, 157, 161
4	149, 153, 157, 161, 165
5	149, 153, 157, 161
6	36, 40, 44, 48
7	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 149, 153, 157, 161, 165
8	52, 56, 60, 64
9	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 132, 136, 140, 149, 153, 157, 161, 165
10	36, 40, 44, 48, 149, 153, 157, 161, 165
11	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 149, 153, 157, 161

3.5 SSID

Description: Set AP SSID
Value:

iwpriv ra0 set SSID=11n-AP

0~z, 1~32 ASCII characters

3.6 WirelessMode

Description: Set WLAN mode
Value:

iwpriv ra0 set WirelessMode=5

0: legacy 11b/g mixed
1: legacy 11B only
2: legacy 11A only
3: legacy 11a/b/g mixed
4: legacy 11G only
5: 11ABGN mixed
6: 11N only

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

7: 11GN mixed
8: 11AN mixed
9: 11BGN mixed
10: 11AGN mixed
11: 11N only in 5G band only
14: 11A/AN/AC mixed 5G band only (Only 11AC chipset support)

3.7 TxBurst

Description: Enable or disable Tx Burst
Value:

iwpriv ra0 set TxBurst=1

0: disable
1: enable

3.8 TxPreamble

Description: Enable or disable Tx Preamle
Value:

iwpriv ra0 set TxPreamble=0

0: disable
1: enable

3.9 TxPower

Description: Set Transmit Power by percentage
Value:

iwpriv ra0 set TxPower=100

0~100

3.10 Channel

Description: Set WLAN Channel
Value:

iwpriv ra0 set Channel=6

Please follow 2.4G or 5G band regulation on WLAN radio channel.

3.11 BGProtection

Description: Enable or disable 11B, 11G protection
Value:

iwpriv ra0 set BGProtection=0

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

0: disable
1: enable

3.12 RTSThreshold

Description: Set RTS Threshold
Value:

iwpriv ra0 set RTSThreshold=2347

1~2347

3.13 FragThreshold

Description: Set Fragment threshold
Value:

iwpriv ra0 set FragThreshold=2346

256~2346

3.14 HtBw

Description: Set HT WLAN Bandwidth
Value:

iwpriv ra0 set HtBw=1

0: 20 MHz
1: 40MHz

3.15 HtMcs

Description: Set WLAN Modulation and Coding Scheme (MCS)
Value:

iwpriv ra0 set HtMcs=33

0 ~15, 32: Fix MCS rate for HT rate.
33: Auto Rate Adaption, recommended

3.16 HtGi

Description: Set WLAN Guard interval support
Value:

iwpriv ra0 set HtGi=1

0: long guard interval
1: short guard interval

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

3.17 HtOpMode

Description: HT operation Mode
Value:

iwpriv ra0 set HtOpMode=0

0: HT mixed mode
1: HT Greenfield mode

3.18 HtStbc

Description: Enable or disable HT STBC
Value:

iwpriv ra0 set HtStbc=1

0: disable
1: enable

3.19 HtExtcha

Description: To locate the 40MHz channel in combination with the control
Value:

iwpriv ra0 set HtExtcha=0

0: below
1: Above

3.20 HtMpduDensity

Description: Minimum separation of MPDUs in an A-MPDU
Value:

iwpriv ra0 set HtMpduDensity=4

0~7
0: no restriction
1: 1/4 μs
2: 1/2 μs
3: 1 μs
4: 2 μs
5: 4 μs
6: 8 μs
7: 16 μs

3.21 HtBaWinSize

Description: Block Ack window size
Value:

iwpriv ra0 set HtBaWinSize=64

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

1~64

3.22 HtRdg

Description: Enable or disable HT Reverse Direction Grant
Value:

iwpriv ra0 set HtRdg=1

0: disable
1: enable

3.23 HtAmsdu

Description: Enable or disable AMSDU section
Value:

Iwpriv ra0 set HtAmsdu=0

0: disable
1: enable

3.24 HtAutoBa

Description: Enable or disable auto build Block Ack section with peer
Value:

iwpriv ra0 set HtAutoBa=1

0: disable
1: enable

3.25 HtBaDecline

Description: Enable or disable decline Block Ack to peer
Value:

iwpriv ra0 set HtBaDecline=0

0: disable
1: enable

3.26 HtProtect

Description: Enable or disable HT protect
Value:

iwpriv ra0 set HtProtect=0

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

0: disable
1: enable

3.27 HtMimoPs

Description: Enable or disable HT MIMO Power saving mode
Value:

iwpriv ra0 set HtMimoPs=0

0: disable
1: enable

3.28 HtDisallowTKIP

Description: Enable or disable 11N rate with 11N AP when cipher is TKIP or WEP
Value:

iwpriv ra0 set HtDisallowTKIP=0

0: disable
1: enable

3.29 HtBssCoex

Description: Enable or disable HT BSS coexistence
Value:

iwpriv ra0 set HtBssCoex=0

0: disable
1: enable

3.30 PktAggregate

Description: Enable or disable 11B/G packet aggregation
Value:

iwpriv ra0 set PktAggregate=1

0: disable
1: enable

3.31 WmmCapable

Description: Enable or disable WMM support
Value:

iwpriv ra0 set WmmCapable=1

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

0: disable
1: enable

3.32 IEEE80211H

Description: Enable or disable IEEE 802.11h function. Spectrum management. This field can only be enabled in A band.

Value:

iwpriv ra0 set IEEE80211H=0

0: disable
1: enable

3.33 NetworkType

Description: Network type

Value:

iwpriv ra0 set NetworkType=Infra

Infra: infrastructure mode
Adhoc: adhoc mode

3.34 AuthMode

Description: WLAN security Authentication mode

Value:

iwpriv ra0 set AuthMode=OPEN

OPEN	For open system
SHARED	For shared key system
WEPAUTO	Auto switch between OPEN and SHARED
WPAPSK	For WPA pre-shared key (Infra)
WPA2PSK	For WPA2 pre-shared key (Infra)
WPANONE	For WPA pre-shared key (Adhoc)
WPA	For enterprise mode (Need wpa_supplicant)
WPA2	For enterprise mode (Need wpa_supplicant)

3.35 EncrypType

Description: WLAN security Encryption type

Value:

iwpriv ra0 set EncrypType=NONE

NONE	For AuthMode=OPEN
WEP	For AuthMode=OPEN or AuthMode=SHARED
TKIP	For AuthMode=WPAPSK or WPA2PSK
AES	For AuthMode=WPAPSK or WPA2PSK

MEDIATEK CONFIDENTIAL
FOR 2950 @ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

3.36 DefaultKeyID

Description: Default key ID

Value:

iwpriv ra0 set DefaultKeyID=1

1~4

3.37 Key1

Description: Key 1 string

Value:

iwpriv ra0 set Key1=aaaaa

10 or 26 characters
5 or 13 characters

3.38 Key2

Description: Key 2 string

Value:

iwpriv ra0 set Key2=aaaaa

10 or 26 characters
5 or 13 characters

3.39 Key3

Description: Key 3 string

Value:

iwpriv ra0 set Key3=aaaaa

10 or 26 characters
5 or 13 characters

3.40 Key4

Description: Key 4 string

Value:

iwpriv ra0 set Key4=aaaaa

10 or 26 characters
5 or 13 characters

3.41 WPAPSK

Description: WLAN Security string for (TKIP/AES)

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

Value:

iwpriv ra0 set WPAPSK=12345678

8~63 ASCII
Or
64 HEX characters

3.42 ResetCounter

Description: reset WLAN statistic counter
Value:

iwpriv ra0 set ResetCounter=1

1

3.43 PSMode

Description: WLAN Power Saving mode
Value:

iwpriv ra0 set PSMode=CAM

CAM	Constantly Awake Mode
Max_PSP	Max Power Saving
Fast_PSP	Fast Power Saving
Legacy_PSP	Legacy Power Saving

3.44 FixedTXMode

Description: Fixed Transmit mode
Value:

iwpriv ra0 set FixedTxMode=ofdm

CCK/cck
OFDM/ofdm
HT/ht

3.45 BeaconLostTime

Description: Becon lost time = 4 seconds then disconnect with AP.
Value:

iwpriv ra0 set BeaconLostTime=4

1~60

3.46 AutoRoaming

Description: Enable or disable auto roaming

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

Value

iwpriv ra0 set AutoRoaming=0

0: disable
1: enable

3.47 SiteSurvey

Description: Scan with specific SSID after link up
Value:

iwpriv ra0 set SiteSurvey=XXX

0~z, 1~32 ACSII characters

3.48 AutoReconnect

Description: Enable or disable auto reconnect
Value:

iwpriv ra0 set AutoReconnect=0

0: disable
1: enable

3.49 AdhocN

Description: Enable or disable Adhoc to support 11N rate
Value:

iwpriv ra0 set AdhocN=1

0: disable
1: enable

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

4 iwpriv ra0 usage

This section describes parameters set using iwpriv. Please refer to the Readme section for more general data.
A detailed explanation of each parameter for iwpriv is shown subsequently. Refer to the Readme before using this section.

iwpriv ra0 [parameters]

4.1 connStatus

Description: Show WLAN connection status
Value:

iwpriv ra0 connStatus

4.2 driverVer

Description: Show WLAN driver version
Value:

iwpriv ra0 driverVer

4.3 bainfo

Description: Show Block Ack information
Value:

iwpriv ra0 bainfo

4.4 rxbulk

Description: Show STA current rxbluk information
Value:

iwpriv ra0 rxbulk

4.5 txbulk

Description: Show STA current txbluk information
Value:

iwpriv ra0 txbulk

4.6 radio_off

Description: Turn off Radio
Value:

MEDIATEK CONFIDENTIAL
FOR 2950201@ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

iwpriv ra0 radio_off

4.7 radio_on

Description: Turn on Radio
Value:

iwpriv ra0 radio_on

4.8 get_site_survey

Description: get site survey result
Value:

iwpriv ra0 get_site_survey

4.9 stat

Description: Display WLAN static counter
Value:

iwpriv ra0 stat

4.10 bbp (Debug only)

Description: Display/Write bbp content
Value:

//Display
iwpriv ra0 bbp offset

//Write bbp
iwpriv ra0 bbp offset=value

offset = hex address
value= hex value

4.11 mac (Debug only)

Description: Display/Write mac content
Value:

//Display
iwpriv ra0 mac offset

//Write mac
iwpriv ra0 mac offset=value

offset = hex address
value= hex value

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

4.12 rf (Debug only)

Description: Display/Write rf content
Value:

```
//Display
iwpriv ra0 rf offset

//Write
iwpriv ra0 rf offset=value

offset = hex address
value= hex value
```

4.13 e2p (Debug only)

Description: Display/Write EEPROM content
Value:

```
//Display
iwpriv ra0 e2p offset

//Write EEPROM
iwpriv ra0 e2p offset=value

offset = hex address
value= hex value
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

5 iwpriv ra0 show command

This section describes parameters set using iwpriv. Please refer to the Readme section for more general data.
Display parameter which has been currently configured in the WLAN driver.

iwpriv ra0 show [parameters]

[Parameters list]

- SSID
- WirelessMode
- TxBurst
- TxPreamble
- TxPower
- Channel
- BGProtection
- RTSThreshold
- FragThreshold
- HtBw
- HtMcs
- HtGi
- HtOpMode
- HtExtcha
- HtMpduDensity
- HtBaWinSize
- HtRdg
- HtAmsdu
- HtAutoBa
- CountryRegion
- CountryRegionABand
- CountryCode
- PktAggregate
- WmmCapable
- IEEE80211H
- NetworkType
- WpsApBand
- AuthMode
- EncrypType
- DefaultKeyID
- Key1
- Key2
- Key3
- Key4
- WPAPSK

Example: show SSID

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

6 iwpriv examples

6.1 Infrastructure Security Mode

WLAN infrastructure Security mode to connect with AP by iwpriv command.
Command sequence must be exact.

6.1.1 OPEN/NONE

Config STA to link with AP which is OPEN/NONE(Authentication/Encryption)
AP's SSID string= XXXX

```
iwpriv ra0 set NetworkType=Infra
iwpriv ra0 set AuthMode=OPEN
iwpriv ra0 set EncrypType=NONE
iwpriv ra0 set SSID="XXXXX"
```

6.1.2 SHARED/WEP

Config STA to link with AP which is SHARED/WEP(Authentication/Encryption)
AP's SSID string= XXXX, WEP key=12345

```
iwpriv ra0 set NetworkType=Infra
iwpriv ra0 set AuthMode=SHARED
iwpriv ra0 set EncrypType=WEP
iwpriv ra0 set DefaultKeyID=1
iwpriv ra0 set Key1="12345"
iwpriv ra0 set SSID="XXXX"
```

6.1.3 WPAPSK/TKIP

Config STA to link with AP which is WPAPSK/TKIP(Authentication/Encryption)
AP's SSID string= XXXX, WPAPSK=12345678

```
iwpriv ra0 set NetworkType=Infra
iwpriv ra0 set AuthMode=WPAPSK
iwpriv ra0 set EncrypType=TKIP
iwpriv ra0 set SSID="XXXX"
iwpriv ra0 set WPAPSK=12345678
iwpriv ra0 set SSID="XXXX"
```

6.1.4 WPAPSK/AES

Config STA to link with AP which is WPAPSK/AES(Authentication/Encryption)
AP's SSID string= XXXX, WPAPSK=12345678

```
iwpriv ra0 set NetworkType=Infra
iwpriv ra0 set AuthMode=WPAPSK
iwpriv ra0 set EncrypType=AES
iwpriv ra0 set SSID="XXXX"
iwpriv ra0 set WPAPSK=12345678
iwpriv ra0 set SSID="XXXX"
```

MEDIATEK CONFIDENTIAL
FOR 2550XX@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

6.1.5 WPA2PSK/TKIP

Config STA to link with AP which is WPA2PSK/TKIP(Authentication/Encryption)
AP's SSID string= XXXX, WPAPSK=12345678

```
iwpriv ra0 set NetworkType=Infra
iwpriv ra0 set AuthMode=WPA2PSK
iwpriv ra0 set EncrypType=TKIP
iwpriv ra0 set SSID=”XXXX”
iwpriv ra0 set WPAPSK=12345678
iwpriv ra0 set SSID="XXXX”
```

6.2 Ad-hoc mode

WLAN adhoc Security mode to connect with Peer by iwpriv command.
Command sequence must be exact.

6.2.1 OPEN/NONE

Config STA to create/link as adhoc mode, which is OPEN/NONE(Authentication/Encryption)
SSID=XXXX

```
iwpriv ra0 set NetworkType=Adhoc
iwpriv ra0 set AuthMode=OPEN
iwpriv ra0 set EncrypType=NONE
iwpriv ra0 set SSID=XXXX
```

6.2.2 WPANONE/TKIP

Config STA to create/link as adhoc mode, which is WPANONE/TKIP(Authentication/Encryption)
SSID=XXXX, WPAPSK=12345678

```
iwpriv ra0 set NetworkType=Adhoc
iwpriv ra0 set AuthMode=WPANONE
iwpriv ra0 set EncrypType=TKIP
iwpriv ra0 set SSID="XXXX”
iwpriv ra0 set WPAPSK=12345678
iwpriv ra0 set SSID="XXXX"
```

6.3 Get Site Survey result

```
//Issue a scan to the WLAN driver
iwlist ra0 sc

//display site survey result
iwpriv ra0 get_site_survey
```

6.4 Get WLAN Statistics

```
iwpriv ra0 stat
```

MEDIATEK CONFIDENTIAL
FOR 2500201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
//Reset WLAN Statistic counter
iwpriv ra0 set ResetCounter=0
```

6.5 To Link with AP

```
iwpriv ra0 set SSID=”AP’s SSID” //Specific AP

or

iwpriv ra0 set SSID=””  //Any SSID with OPEN/NONE security
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

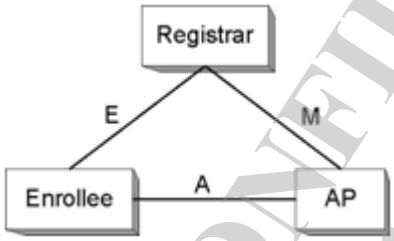
7 WPS Wi-Fi PROTECTED SETUP

Simple Config Architectural Overview
This section presents a high-level description of the Simple Config architecture. Much of the material is taken directly from the Simple Config specification.
Figure 1 depicts the major components and their interfaces as defined by Wi-Fi Simple Config Spec.
There are three logical components involved: the Registrar, the access point (AP), and the Enrollee.

The **Enrollee** is a device seeking to join a WLAN domain. Once an Enrollee obtains a valid credential, it becomes a member.

A **Registrar** is an entity with the authority to issue and revoke domain credentials. A registrar can be integrated into an AP.

The **AP** can be either a WLAN AP or a wireless router.



Registration initiation is ordinarily accomplished by a user action such as powering up the Enrollee and, optionally, running a setup wizard on the Registrar (PC).

Note: The WLAN driver needs to set HAS_WSC=y in order to enable WPS functions.

7.1 WPS iwpriv command

This section describes parameters set using iwpriv. Please refer to the Readme section for more general data.

iwpriv ra0 [commands]=[Value]

Note: Wireless extension private handlers.

7.1.1 wsc_conf_mode

Description: set WPS conf mode
Value:

iwpriv ra0 wsc_conf_mode 1

0: WPS Disabled
1: Enrollee
2: Registrar

MEDIATEK CONFIDENTIAL
FOR 695020120@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

7.1.2 wsc_mode

Description: WPS mode, PBC or PIN
Value:

iwpriv ra0 wsc_mode 1

1: PIN Mode
2: PBC Mode

7.1.3 wsc_pin

Description: set WPS pin code
Value:

iwpriv ra0 wsc_pin xxxxxxxx

xxxxxxx = {00000000 ~ 99999999}

7.1.4 wsc_ssid

Description: set WPS AP SSID
Value:

iwpriv ra0 wsc_ssid xxxxxxx

0~z, 1~32 ASCII characters

7.1.5 wsc_bssid

Description: set BSSID of WSC AP that STA wants to do WPS with
Value:

iwpriv ra0 wsc_bssid xx:xx:xx:xx:xx:xx

xx:xx:xx:xx:xx:xx ←BSSID

7.1.6 wsc_start

Description: Trigger WLAN driver to do WPS process
Value:

iwpriv ra0 wsc_start

7.1.7 wsc_stop

Description: Stop WLAN driver to do WPS process
Value:

iwpriv ra0 wsc_stop

MEDIATEK CONFIDENTIAL
FOR 2950201@gemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

7.1.8 wsc_gen_pincode

Description: Generate new PIN code for WPS usage
Value:

iwpriv ra0 wsc_gen_pincode

7.1.9 wsc_cred_count

Description: Set count of WPS credential, only support one credential for M8 in Registrar mode.
Value:

iwpriv ra0 wsc_cred_count 1

1~8

7.1.10 wsc_cred_ssid

Description: Set SSID into credtentail[idx].
Value:

iwpriv ra0 wsc_cred_ssid “idx ssid_str”

idx: 0 ~ 7
ssid_str: 0~z, 1~32 ascii characters

Example:

iwpriv ra0 wsc_cred_ssid "0 wps_ap1"

7.1.11 wsc_cred_auth

Description: Set AuthMode into credtentail[idx].
Value:

iwpriv ra0 wsc_cred_auth “idx auth_str”
idx: 0 ~ 7
auth_str: OPEN, WPAPSK, WPA2PSK, SHARED, WPA, WPA2

Example:

iwpriv ra0 wsc_cred_auth "0 WPAPSK"

7.1.12 wsc_cred_encr

Description: Set EncrypType into credtentail[idx].
Value:

iwpriv ra0 wsc_cred_encr “idx encr_str”

idx: 0 ~ 7
encr_str: NONE, WEP, TKIP, AES

Example:

iwpriv ra0 wsc_cred_encr "0 TKIP"

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

7.1.13 wsc_cred_keyIdx

Description: Set Key Index into credtentail[idx].

Value:

iwpriv ra0 wsc_cred_keyid “idx key_index”

idx: 0 ~ 7
key_index: 1 ~ 4

Example:

iwpriv ra0 wsc_cred_keyIdx "0 1"

7.1.14 wsc_cred_key

Description: Set Key into credtentail[idx].

Value:

iwpriv ra0 wsc_cred_key “idx key”

idx: 0 ~ 7
key: ASCII string (wep_key_len(=5,13), passphrase_len(=8~63))
OR
Hex string (wep_key_len(=10,26), passphrase_len(=64))

Example:

iwpriv ra0 wsc_cred_key "0 12345678" ;; Passphrase
iwpriv ra0 wsc_cred_key "0 abcd" ;; WEP Key

7.1.15 wsc_cred_mac

Description: Set AP's MAC into credtentail[idx].

Value:

iwpriv ra0 wsc_cred_mac “idx mac_str”

idx: 0 ~ 7
mac_str: xx:xx:xx:xx:xx:xx

Example:

iwpriv ra0 wsc_cred_mac "0 00:11:22:33:44:55"

7.1.16 wsc_conn_by_idx

Description: Connect AP by credential index.

Value:

iwpriv ra0 wsc_conn_by_idx 0

0~7

MEDIATEK CONFIDENTIAL
FOR 2950201@cgeiray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

7.1.17 wsc_auto_conn

Description: If the registration is successful, driver will re-connect to AP or not.
Value:

iwpriv ra0 wsc_auto_conn 1

0: disabled, driver won't re-connect to AP with new configurations.
1: enabled, driver will re-connect to AP with new configurations.

7.1.18 wsc_ap_band

Description: Setting prefer band to do WPS with dual band WPS AP.
Value:

iwpriv ra0 wsc_ap_band 2

0: Prefer 2.4G
1: Prefer 5G
2: Auto (default)

7.1.19 Wsc4digitPinCode

Description: Generate WPS 4-digits PIN
Value:

iwpriv ra0 Wsc4digitPinCode 1

0: disable
1: enable

7.2 WPS STA as an Enrollee or Registrar

7.2.1 Enrollee Mode

7.2.1.1 PIN mode

Running Scenarios (case 'a' and 'b')

a. Adding an Enrollee to AP+Registrar (EAP)
[AP+Registrar]<----EAP--->[Enrollee Client]

b. Adding an Enrollee with external Registrar (UPnP/EAP)
[External Registrar]<----UPnP--->[AP_Proxy]<---EAP--->[Enrollee Client]

Note: 'EAP' indicates to use wireless medium and 'UPnP' indicates to use wired or wireless medium.

(i) [Registrar] or [AP+Registrar]

Enter the Enrollee PinCode on the Registrar and start WPS on the Registrar.

Note: How to get the Enrollee PinCode? Use 'iwpriv ra0 stat' on the Enrollee.

(ii) [Linux WPS STA]

MEDIATEK CONFIDENTIAL
FOR 2950201@egemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
iwpriv ra0 wsc_conf_mode 1 ;; Enrollee
iwpriv ra0 wsc_mode 1 ;; PIN
iwpriv ra0 wsc_ssid "AP's SSID"
iwpriv ra0 wsc_start
```

(iii) If the registration is successful, the Enrollee will be re-configured with the new parameters, and will connect to the AP with these new parameters.

7.2.2 PBC mode

Running Scenarios (case 'a' only)

a. Adding an Enrollee to AP+Registrar (EAP)
[AP+Registrar]<----EAP--->[Client]

(i) [AP+Registrar]

Start PBC on the Registrar.

(ii) [Linux WPS STA]

```
iwpriv ra0 wsc_conf_mode 1 ;; Enrollee
iwpriv ra0 wsc_mode 2 ;; PBC
iwpriv ra0 wsc_start
```

(iii) If the registration is successful, the Enrollee will be re-configured with the new parameters, and will connect to the AP with these new parameters.

7.3 Registrar Mode

7.3.1 PIN Mode

Running Scenarios (case 'a' and 'b')

a. Configure the un-configured AP
[Unconfigured AP]<----EAP--->[Registrar]

b. Configure the configured AP
Configured AP]<----EAP--->[Registrar]

(i) [AP]
Start PIN on the Enrollee WPS AP.

(ii) [Linux WPS STA]

```
iwpriv ra0 wsc_conf_mode 2 ;; Registrar
iwpriv ra0 wsc_mode 1 ;; PIN
iwpriv ra0 wsc_pin xxxxxxxx ;; AP's PIN Code
iwpriv ra0 wsc_ssid "AP's SSID"
iwpriv ra0 wsc_start
```

(iii) If the registration is successful;

In case 'a':
The Registrar will be re-configured with the new parameters, and will connect to the AP with these new parameters;

MEDIATEK CONFIDENTIAL
FOR 2560201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

In case 'b':
The Registrar will be re-configured with AP's configurations, and will connect to the AP with these new parameters.

7.3.2 PBC Mode

Running Scenarios (case 'a' and 'b')
a. Configure the un-configured AP
[Unconfigured AP]<----EAP--->[Registrar]

b. Configure the configured AP
Configured AP]<----EAP--->[Registrar]

(i) [AP]

Start PBC on the Enrollee WPS AP.

(ii) [Linux WPS STA]

```
iwpriv ra0 wsc_conf_mode 2 ;; Registrar
iwpriv ra0 wsc_mode 2 ;; PBC
iwpriv ra0 wsc_start
```

(iii) If the registration is successful;

In case 'a':
The Registrar will be re-configured with the new parameters, and will connect to the AP with these new parameters;

In case 'b':
The Registrar will be re-configured with AP's configurations, and will connect to the AP with these new parameters.

7.4 WPS Command & OID Example

7.4.1 Iwpriv command without argument

iwpriv command:

```
iwpriv ra0 wsc_start
iwpriv ra0 wsc_stop
iwpriv ra0 wsc_gen_pincode
```

OID:

Example:

```
memset(&lwreq, 0, sizeof(lwreq));
sprintf(lwreq.ifr_name, "ra0", 3);
iwreq.u.mode = WSC_STOP;
/* Perform the private ioctl */
if(ioctl(skfd, RTPRIV_IOCTL_SET_WSC_PROFILE_U32_ITEM, &lwreq) < 0)
{
fprintf(stderr, "Interface doesn't accept private ioctl...\n");
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

7.4.2 Iwpriv command with one INT argument

iwpriv command:

```
iwpriv ra0 wsc_cred_count 1
iwpriv ra0 wsc_conn_by_idx 1
iwpriv ra0 wsc_auto_conn 1
iwpriv ra0 wsc_conf_mode 1
iwpriv ra0 wsc_mode 1
iwpriv ra0 wsc_pin 12345678
```

OID:

Example:

```
memset(&lwreq, 0, sizeof(lwreq));
lwreq.u.data.length = 1;
cred_count = 1;
((int *) buffer)[i] = (int) cred_count;
offset = sizeof(int);
sprintf(lwreq.ifr_name, "ra0", 3);
lwreq.u.mode = WSC_CREDENTIAL_COUNT;
memcpy(lwreq.u.name + offset, buffer, IFNAMSIZ - offset);
/* Perform the private ioctl */
if(ioctl(skfd, RTPRIV_IOCTL_SET_WSC_PROFILE_U32_ITEM, &lwreq) < 0)
{
  fprintf(stderr, "Interface doesn't accept private ioctl...\n");
  return -1;
}
```

7.4.3 Iwpriv command with string argument

iwpriv command:

```
iwpriv ra0 wsc_ssid “0 xxxxx”
iwpriv ra0 wsc_cred_ssid “0 xxxxx”
iwpriv ra0 wsc_cred_auth “0 WPAPSK”
iwpriv ra0 wsc_cred_encr “0 TKIP”
iwpriv ra0 wsc_cred_keyIdx “0 1”
iwpriv ra0 wsc_cred_key “0 12345”
iwpriv ra0 wsc_cred_mac “0 00:11:22:33:44:55”
```

OID:

Example:

```
memset(&lwreq, 0, sizeof(lwreq));
memset(buffer, 0, 2048);
sprintf(lwreq.ifr_name, "ra0", 3);
sprintf(buffer, "0 wps_ssid_1");
lwreq.u.data.length = strlen(buffer) + 1;
lwreq.u.data.pointer = (caddr_t) buffer;
lwreq.u.data.flags = WSC_CREDENTIAL_SSID;
/* Perform the private ioctl */
if(ioctl(skfd, RTPRIV_IOCTL_SET_WSC_PROFILE_STRING_ITEM, &lwreq) < 0)
{
```

MEDIATEK CONFIDENTIAL
FOR 2500201@ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
        fprintf(stderr, "Interface doesn't accept private ioctl...\n");
        return -1;
}
```

7.5 WPS OID Sample Program

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <assert.h>
#include <netinet/in.h>    /* for sockaddr_in       */
#include <fcntl.h>
#include <time.h>
#include <sys/times.h>
#include <unistd.h>
#include <sys/socket.h>    /* for connect and socket*/
#include <sys/stat.h>
#include <err.h>
#include <errno.h>
#include <asm/types.h>
#include </usr/include/linux/wireless.h>
#include <sys/ioctl.h>

#define IFNAMSIZ 16

#define RTPRIV_IOCTL_SET_WSC_PROFILE_U32_ITEM      (SIOCIWFIRSTPRIV + 0x14)
#define RTPRIV_IOCTL_SET_WSC_PROFILE_STRING_ITEM    (SIOCIWFIRSTPRIV + 0x16)

enum {
            WSC_CREDENTIAL_COUNT = 1,
            WSC_CREDENTIAL_SSID = 2,
            WSC_CREDENTIAL_AUTH_MODE = 3,
            WSC_CREDENTIAL_ENCR_TYPE = 4,
            WSC_CREDENTIAL_KEY_INDEX = 5,
            WSC_CREDENTIAL_KEY = 6,
            WSC_CREDENTIAL_MAC = 7,
            WSC_SET_DRIVER_CONNECT_BY_CREDENTIAL_IDX = 8,
            WSC_SET_DRIVER_AUTO_CONNECT = 9,
            WSC_SET_CONF_MODE = 10, // Enrollee or Registrar
            WSC_SET_MODE = 11,      // PIN or PBC
            WSC_SET_PIN = 12,
            WSC_SET_SSID = 13,
            WSC_START = 14,
            WSC_STOP = 15,
            WSC_GEN_PIN_CODE = 16,
};

int main()
{
 struct iwreq lwreq;
 char         buffer[2048] = {0};
 int          cred_count;
 int          offset = 0;            /* Space for sub-ioctl index */
 int          skfd, i = 0;                    /* generic raw socket desc. */

 skfd = socket(AF_INET, SOCK_DGRAM, 0);
 if (skfd < 0)
            return -1;

 ////////////// WSC_STOP //////////////
 memset(&lwreq, 0, sizeof(lwreq));
 sprintf(lwreq.ifr_name, "ra0", 3);
 lwreq.u.mode = WSC_STOP;

 /* Perform the private ioctl */
if(ioctl(skfd, RTPRIV_IOCTL_SET_WSC_PROFILE_U32_ITEM, &lwreq) < 0)
 {
            fprintf(stderr, "Interface doesn't accept private ioctl...\n");
            return -1;
 }
  ////////////////////////////////////
```

MEDIATEK CONFIDENTIAL
FOR 2450201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

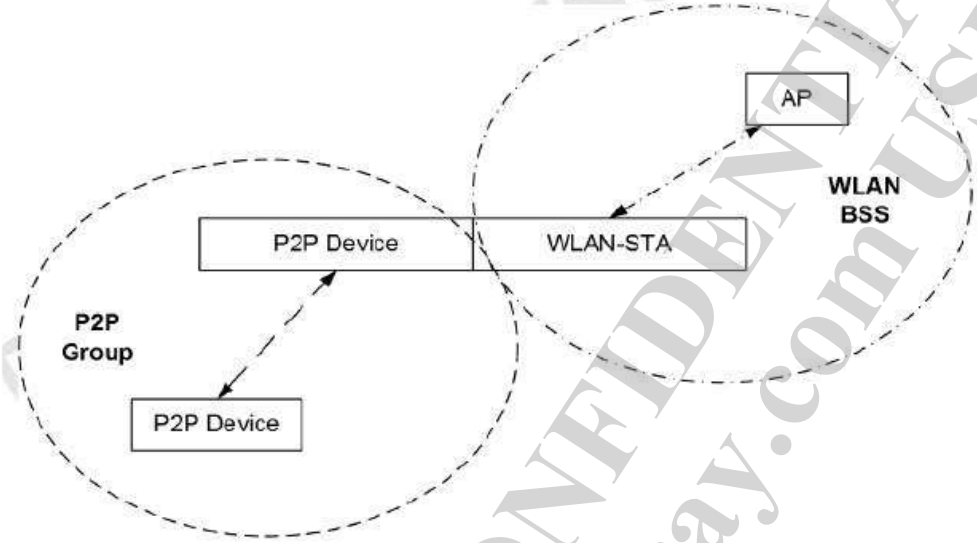
```
        /////// WSC_CREDENTIAL_COUNT ///////
        memset(&lwreq, 0, sizeof(lwreq));
        lwreq.u.data.length = 1;
        cred_count = 1;
        ((int *) buffer)[i] = (int) cred_count;
        offset = sizeof(int);

        sprintf(lwreq.ifr_name, "ra0", 3);
        lwreq.u.mode = WSC_CREDENTIAL_COUNT;
        memcpy(lwreq.u.name + offset, buffer, IFNAMSIZ - offset);

        /* Perform the private ioctl */
        if(ioctl(skfd, RTPRIV_IOCTL_SET_WSC_PROFILE_U32_ITEM, &lwreq) < 0)
        {
                fprintf(stderr, "Interface doesn't accept private ioctl...\n");
                return -1;
        }
         ////////////////////////////////////

         /////// WSC_CREDENTIAL_SSID ///////
        memset(&lwreq, 0, sizeof(lwreq));
        memset(buffer, 0, 2048);
        sprintf(lwreq.ifr_name, "ra0", 3);
        sprintf(buffer, "0 wps_ssid_1");
        lwreq.u.data.length = strlen(buffer) + 1;
        lwreq.u.data.pointer = (caddr_t) buffer;
        lwreq.u.data.flags = WSC_CREDENTIAL_SSID;

        /* Perform the private ioctl */
         if(ioctl(skfd, RTPRIV_IOCTL_SET_WSC_PROFILE_STRING_ITEM, &lwreq) < 0)
        {
                fprintf(stderr, "Interface doesn't accept private ioctl...\n");
                return -1;
        }
         ////////////////////////////////////

        close(skfd);
        return 0;
}
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

8 Wi-Fi DIRECT (P2P)

Wifi direct feature makes direct connections to one another quickly and conveniently to do things like print, sync, and share content even when an access point or router is unavailable.



- A Wi-fi Direct Device may operate concurrently with a WLAN (infrastructure network)
- A P2P Group may operate in the same or different regulatory class and channel as a concurrently operating WLAN BSS

8.1 Wi-Fi DIRECT iwpriv Command

P2P interface name: p2p0
Syntax is iwpriv p2p0 set [parameters]=[Value]

Note: Execute one iwpriv/set command at a time.

8.1.1 P2pOpMode

Description: Set P2P Operation Mode
Value:

iwpriv p2p0 set P2pOpMode=1

0: Device mode
1: Autonomous GO

MEDIATEK CONFIDENTIAL
FOR 2050201@ogemay.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

8.1.2 p2pEnable

Description: Enable or disable P2P
Value:

iwpriv p2p0 set p2pEnable=1

0: disable
1: enable

8.1.3 p2pScan

Description: Start P2P scanning
Value:

iwpriv p2p0 set p2pScan=1

0:disable
1:enable

8.1.4 p2pTable

Description: Display p2p scan table in WLAN driver log.
Value:

iwpriv p2p0 set p2pTable=1

0: disable
1: enable

8.1.5 p2pGoInt

Description: set P2P GO intent value
Value:

iwpriv p2p0 set p2pGoInt=0

0~15

8.1.6 p2pDevName

Description: set p2p device name
Value:

iwpriv p2p0 set p2pDevName=Direct-P2P0

0~Z, less than 32 characters.

8.1.7 p2pWscMode

Description: Set WPS mode for P2P negotiate
Value:

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

iwpriv p2p0 set p2pWscMode=2

1: PIN
2: PBC

8.1.8 p2pWscConf

Description: Set WPS conf method for P2P negotiate
Value:

iwpriv p2p0 set p2pWscConf=1

1: Display
2: KeyPad
3: PBC

8.1.9 p2pLisCh

Description: Set P2P listen Channel
Value:

iwpriv p2p0 set p2pLisCh=1

1,6,11

8.1.10 p2pOpCh

Description: Set P2P Operation Channel
Value:

Depending on country region setting.

iwpriv p2p0 set p2pOpCh=1

8.1.11 p2pLink

Description: Set P2P device ID to do GO negotiation
Value:

iwpriv p2p0 set p2pLink=1

0~29

8.1.12 p2pInv

Description: Select P2P device ID to Invite
Value:

iwpriv p2p0 set p2pInv=1

0~29

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

8.1.13 p2pProv

Description: Select P2P device ID to provision
Value:

iwpriv p2p0 set p2pProv=1

0~29

8.1.14 p2pCfg

Description: Dump out P2P configuration
Value:

iwpriv p2p0 set p2pCfg=1

1

8.1.15 p2pStat

Description: Dump out P2P status
Value:

iwpriv p2p0 set p2pStat=1

1

8.1.16 p2pReset

Description: Reset P2P configuration
Value:

iwpriv p2p0 set p2pReset=1

1

8.1.17 p2pDefConfMthd

Description: Set default WPS Config Method to Provision
Value:

iwpriv p2p0 set p2pDefConfMthd=3

1: Display
2: KeyPad
3: PBC

8.1.18 p2pDevDisc

Description: Select P2P Device ID for discovery

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

Value:

iwpriv p2p0 set p2pDevDisc=1

0~29

8.1.19 p2pLinkDown

Description: Disconnect P2P session and turn back to listen state
Value:

iwpriv p2p0 set p2pLinkDown=1

1

8.2 P2P Example iwpriv command

8.2.1 Configure WLAN driver as Autonomous GO

Enable Autonomous on channel 11 with WPA2PSK/AES key:12345678, SSID=aaapp

```
# interface init
ifconfig p2p0 down
ifconfig ra0 down
ifconfig ra0 up
iwpriv ra0 set Debug=3
ifconfig p2p0 up

# p2p init
iwpriv p2p0 set p2pLinkDown=1
iwpriv p2p0 set p2pReset=1

iwpriv p2p0 set p2pOpCh=11
iwpriv p2p0 set P2pOpMode=1
iwpriv p2p0 set SSID=aaapp
iwpriv p2p0 set AuthMode=WPA2PSK
iwpriv p2p0 set EncrypType=AES
iwpriv p2p0 set WPAPSK=12345678
iwpriv p2p0 set SSID=aaapp

#set p2p ip address
ifconfig p2p0 10.10.10.6
```

8.2.2 Configure WLAN driver as Autonomous GO start WPS PBC

```
# interface init
ifconfig p2p0 down
ifconfig ra0 down
ifconfig ra0 up
```


This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.
MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

```
iwpriv ra0 set Debug=3
ifconfig p2p0 up

# p2p init
iwpriv p2p0 set p2pLinkDown=1
iwpriv p2p0 set p2pReset=1

# p2p PBC Go WPS
iwpriv p2p0 set p2pWscMode=2
iwpriv p2p0 set p2pWscConf=3
iwpriv p2p0 set p2pGoInt=15

# p2p scan and link
iwpriv p2p0 set p2pScan=1
sleep 25
iwpriv p2p0 set p2pLink=0

# set p2p ip
ifconfig p2p0 10.10.10.6
```

8.2.3 Configure WLAN driver as autonomous GO start WPS PIN-Display

```
# interface init
ifconfig p2p0 down
ifconfig ra0 down
ifconfig ra0 up
iwpriv ra0 set Debug=3
ifconfig p2p0 up

# p2p init
iwpriv p2p0 set p2pLinkDown=1
iwpriv p2p0 set p2pReset=1

# p2p display WPS
iwpriv p2p0 set p2pWscConf=1
iwpriv p2p0 set p2pGoInt=0
iwpriv p2p0 set p2pScan=1

# set p2p ip
ifconfig p2p0 10.10.10.222
```

8.2.4 Configure WLAN driver as autonomous GO start WPS PIN-KeyPad

```
# interface init
ifconfig p2p0 down
ifconfig ra0 down
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
ifconfig ra0 up
iwpriv ra0 set Debug=3
ifconfig p2p0 up

# p2p init
iwpriv p2p0 set p2pLinkDown=1
iwpriv p2p0 set p2pReset=1

# p2p keypad WPS
iwpriv p2p0 set p2pWscConf=2
iwpriv p2p0 set p2pGoInt=15
iwpriv p2p0 set p2pEnterPIN=32240979  #Read enrollee PIN code

# p2p scan and link
iwpriv p2p0 set p2pScan=1
sleep 25
iwpriv p2p0 set p2pLink=0

# set p2p ip
ifconfig p2p0 10.10.10.6
```

8.2.5 Configure WLAN driver as P2P device start WPS PIN-Display

```
# interface init
ifconfig p2p0 down
ifconfig ra0 down
ifconfig ra0 up
iwpriv ra0 set Debug=3
ifconfig p2p0 up

# p2p init
iwpriv p2p0 set p2pLinkDown=1
iwpriv p2p0 set p2pReset=1

# p2p keypad WPS
iwpriv p2p0 set p2pWscConf=2
iwpriv p2p0 set p2pGoInt=15
iwpriv p2p0 set p2pScan=1

# waiting....Win7 (GO)
# Check Win7 PIN code and Set PIN
iwpriv p2p0 set p2pEnterPIN=32240979
```

8.2.6 Configure WLAN driver as P2P device start WPS PBC

```
# interface init
```

MEDIATEK CONFIDENTIAL
FOR 2560201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
ifconfig p2p0 down
ifconfig ra0 down
ifconfig ra0 up
iwpriv ra0 set Debug=3
ifconfig p2p0 up

# p2p init
iwpriv p2p0 set p2pLinkDown=1
iwpriv p2p0 set p2pReset=1

# p2p PBC dev WPS
iwpriv p2p0 set p2pWscMode=2
iwpriv p2p0 set p2pWscConf=3
iwpriv p2p0 set p2pGoInt=0
iwpriv p2p0 set p2pScan=1

# set p2p ip
ifconfig p2p0 10.10.10.22
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

9 OID programming

Please refer to WLAN driver source code include/oid.h for OID data structure.

9.1 OID Set Data

Command and IOCTL Function		
Set Data		
Function Type	**Command**	**IOCTL**

RTPRIV_IOCTL_SET	Iwpriv ra0 **set SSID=RT3572AP**	sprintf(name, "ra0");
		strcpy(data, "**SSID=RT3572AP**");
		strcpy(wrq.ifr_name, name);
		wrq.u.data.length = strlen(data);
		wrq.u.data.pointer = data;
		wrq.u.data.flags = 0;
		ioctl(socket_id, RTPRIV_IOCTL_SET, &wrq);

9.2 OID Get Data

Command and IOCTL Function		
Get Data		
Function Type	**Command**	**IOCTL**

RTPRIV_IOCTL_STATISTICS	Iwpriv ra0 **stat**	sprintf(name, "ra0");
		strcpy(data, "**stat**");
		strcpy(wrq.ifr_name, name);
		wrq.u.data.length = strlen(data);
		wrq.u.data.pointer = data;
		wrq.u.data.flags = 0;
		ioctl(socket_id, RTPRIV_IOCTL_STATISTICS, &wrq);

RTPRIV_IOCTL_GSITESURVEY	Iwpriv ra0 **get_site_survey**	sprintf(name, "ra0");
		strcpy(data, "**get_site_survey**");
		strcpy(wrq.ifr_name, name);
		wrq.u.data.length = strlen(data);
		wrq.u.data.pointer = data;
		wrq.u.data.flags = 0;
		ioctl(socket_id, RTPRIV_IOCTL_GSITESURVEY, &wrq);

9.3 OID set Raw Data with flags

IOCTL Function	
Set Raw Data by I/O Control Interface with Flags	
Function Type	**IOCTL**

RT_OID_802_11_COUNTRY_REGION	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(UCHAR));
	wrq.u.data.length = sizeof(UCHAR);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_802_11_COUNTRY_REGION;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

OID_802_11_BSSID_LIST_SCAN	sprintf(name, "ra0");

MEDIATEK CONFIDENTIAL
FOR 95020120@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = 0;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_BSSID_LIST_SCAN**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_SSID	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_SSID));
	wrq.u.data.length = sizeof(NDIS_802_11_SSID);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_SSID**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_BSSID	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_MAC_ADDRESS));
	wrq.u.data.length = sizeof(NDIS_802_11_MAC_ADDRESS);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_BSSID**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_RADIO	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(BOOLEAN));
	wrq.u.data.length = sizeof(BOOLEAN);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_RADIO**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_PHY_MODE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RT_802_11_PHY_MODE));
	wrq.u.data.length = sizeof(RT_802_11_PHY_MODE);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_PHY_MODE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_STA_CONFIG	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RT_802_11_STA_CONFIG));
	wrq.u.data.length = sizeof(RT_802_11_STA_CONFIG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_STA_CONFIG**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_DESIRED_RATES	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_RATES));
	wrq.u.data.length = sizeof(NDIS_802_11_RATES);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_DESIRED_RATES**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_PREAMBLE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RT_802_11_PREAMBLE));
	wrq.u.data.length = sizeof(RT_802_11_PREAMBLE);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_PREAMBLE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_WEP_STATUS	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_WEP_STATUS));
	wrq.u.data.length = sizeof(NDIS_802_11_WEP_STATUS);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_WEP_STATUS**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_AUTHENTICATION_MODE	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0,
	sizeof(NDIS_802_11_AUTHENTICATION_MODE));
	wrq.u.data.length =
	sizeof(NDIS_802_11_AUTHENTICATION_MODE);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_AUTHENTICATION_MODE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

MEDIATEK CONFIDENTIAL
FOR 29500201@qq.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

OID_802_11_INFRASTRUCTURE_MODE	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_NETWORK_INFRASTRUCTURE));
	wrq.u.data.length = sizeof(NDIS_802_11_NETWORK_INFRASTRUCTURE);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_INFRASTRUCTURE_MODE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_REMOVE_WEP	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_KEY_INDEX));
	wrq.u.data.length = sizeof(NDIS_802_11_KEY_INDEX);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_REMOVE_WEP**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_RESET_COUNTERS	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = 0;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_RESET_COUNTERS**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_RTS_THRESHOLD	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_RTS_THRESHOLD));
	wrq.u.data.length = sizeof(NDIS_802_11_RTS_THRESHOLD);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_RTS_THRESHOLD**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_FRAGMENTATION_THRESHOLD	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_FRAGMENTATION_THRESHOLD));
	wrq.u.data.length = sizeof(NDIS_802_11_FRAGMENTATION_THRESHOLD);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_FRAGMENTATION_THRESHOLD**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_POWER_MODE	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_POWER_MODE));
	wrq.u.data.length = sizeof(NDIS_802_11_POWER_MODE);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_POWER_MODE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_TX_POWER_LEVEL	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_TX_POWER_LEVEL));
	wrq.u.data.length = sizeof(NDIS_802_11_TX_POWER_LEVEL);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_TX_POWER_LEVEL**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_TX_POWER_LEVEL_1	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_TX_POWER_LEVEL_1**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_NETWORK_TYPE_IN_USE	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_NETWORK_TYPE));
	wrq.u.data.length = / sizeof(NDIS_802_11_NETWORK_TYPE);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_NETWORK_TYPE_IN_USE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_RX_ANTENNA_SELECTED	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_ANTENNA));

MEDIATEK CONFIDENTIAL
FOR 2950201@qq.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

	wrq.u.data.length = sizeof(NDIS_802_11_ANTENNA);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_RX_ANTENNA_SELECTED**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_TX_ANTENNA_SELECTED	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_ANTENNA));
	wrq.u.data.length = sizeof(NDIS_802_11_ANTENNA);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_TX_ANTENNA_SELECTED**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_ADD_WPA	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, 32);
	wrq.u.data.length = 32;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_ADD_WPA**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_REMOVE_KEY	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_REMOVE_KEY));
	wrq.u.data.length = sizeof(NDIS_802_11_REMOVE_KEY);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_REMOVE_KEY**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_ADD_KEY	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, keylength); //5,10,13,26
	wrq.u.data.length = keylength L;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_ADD_KEY**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_SET_IEEE8021X	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(BOOLEAN));
	wrq.u.data.length = sizeof(BOOLEAN);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_SET_IEEE8021X**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_SET_IEEE8021X_REQUIRE_KEY	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(BOOLEAN));
	wrq.u.data.length = sizeof(BOOLEAN);
	wrq.u.data.pointer = data;
	wrq.u.data.flags =
	OID_802_11_SET_IEEE8021X_REQUIRE_KEY;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_ADD_WEP	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, keylength); //5,10,13,26
	wrq.u.data.length = keylength;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_RADIO**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_CONFIGURATION	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_CONFIGURATION));
	wrq.u.data.length = sizeof(NDIS_802_11_CONFIGURATION);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_CONFIGURATION**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_SET_COUNTERMEASURES	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = 0;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_SET_COUNTERMEASURES**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_DISASSOCIATE	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = 0;

MEDIATEK CONFIDENTIAL
FOR 2950201@qq.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_DISASSOCIATE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_PMKID	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = keylength; //follow your setting
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_PMKID**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_WPA_SUPPLICANT_SUPPORT	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(BOOLEAN));
	wrq.u.data.length = sizeof(BOOLEAN);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_WPA_SUPPLICANT_SUPPORT**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_WPA_SUPPLICANT_SUPPORT	printf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_WPA_SUPPLICANT_SUPPORT**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_SET_DEL_MAC_ENTRY	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0xdd, 6);
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = 6;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_SET_DEL_MAC_ENTRY**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_SET_HT_PHYMODE | OID_GET_SET_TOGGLE	typedef struct {
	RT_802_11_PHY_MODE PhyMode;
	UCHAR TransmitNo;
	UCHAR HtMode; //HTMODE_GF or HTMODE_MM
	UCHAR ExtOffset;//extension channel above or below
	UCHAR MCS;
	UCHAR BW;
	UCHAR STBC;
	UCHAR SHORTGI;
	UCHAR rsv;
	} OID_SET_HT_PHYMODE ;
	
	RT_802_11_PHY_MODE tmp_ht_mode;
	sprintf(wrq.ifr_name, "ra0");
	wrq.u.data.pointer = (caddr_t) & tmp_ht_mode;
	wrq.u.data.length = sizeof(RT_802_11_PHY_MODE);
	wrq.u.data.flags = **RT_OID_802_11_SET_HT_PHYMODE | OID_GET_SET_TOGGLE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

IOCTL Function

Set Raw Data by I/O Control Interface with Flags

Function Type	IOCTL

RT_OID_802_11_SET_HT_PHYMODE | OID_GET_SET_TOGGLE	typedef struct {
	RT_802_11_PHY_MODE PhyMode;
	UCHAR TransmitNo;
	UCHAR HtMode; //HTMODE_GF or HTMODE_MM
	UCHAR ExtOffset;//extension channel above or below
	UCHAR MCS;
	UCHAR BW;
	UCHAR STBC;
	UCHAR SHORTGI;
	UCHAR rsv;

MEDIATEK CONFIDENTIAL FOR 2950201@oecirai.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

	} OID_SET_HT_PHYMODE ;
	
	RT_802_11_PHY_MODE tmp_ht_mode;
	sprintf(wrq.ifr_name, "ra0");
	wrq.u.data.pointer = (caddr_t) & tmp_ht_mode;
	wrq.u.data.length = sizeof(RT_802_11_PHY_MODE);
	wrq.u.data.flags = **RT_OID_802_11_SET_HT_PHYMODE | OID_GET_SET_TOGGLE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

9.4 OID Get Raw Data with Flags

IOCTL Function	
Get Raw Data by I/O Control Interface with Flags	
Function Type	**IOCTL**

RT_OID_DEVICE_NAME	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, 255);
	wrq.u.data.length = 255;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_DEVICE_NAME**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_VERSION_INFO	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RT_VERSION_INFO));
	wrq.u.data.length = sizeof(RT_VERSION_INFO);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_VERSION_INFO**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_BSSID_LIST	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, BssLen);
	wrq.u.data.length = BssLen;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_BSSID_LIST**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_3_CURRENT_ADDRESS	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(CurrentAddress));
	wrq.u.data.length = sizeof(CurrentAddress);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_3_CURRENT_ADDRESS**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_GEN_MEDIA_CONNECT_STATUS	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_MEDIA_STATE));
	wrq.u.data.length = sizeof(NDIS_MEDIA_STATE);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_GEN_MEDIA_CONNECT_STATUS**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_BSSID	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_MAC_ADDRESS));
	wrq.u.data.length = sizeof(NDIS_802_11_MAC_ADDRESS);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_BSSID**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_SSID	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_SSID));
	wrq.u.data.length = sizeof(NDIS_802_11_SSID);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_SSID**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_QUERY_LINK_STATUS	sprintf(name, "ra0");

MEDIATEK CONFIDENTIAL
FOR 2950201@qq.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RT_802_11_LINK_STATUS));
	wrq.u.data.length = sizeof(RT_802_11_LINK_STATUS);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_QUERY_LINK_STATUS**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_CONFIGURATION	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_CONFIGURATION));
	wrq.u.data.length = sizeof(NDIS_802_11_CONFIGURATION);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_CONFIGURATION**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_RSSI_TRIGGER	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ulInfo));
	wrq.u.data.length = sizeof(ulInfo);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_RSSI_TRIGGER**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_RSSI	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ulInfo));
	wrq.u.data.length = sizeof(ulInfo);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_RSSI**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_RSSI_1	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ulInfo));
	wrq.u.data.length = sizeof(ulInfo);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_RSSI_1**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_RSSI_2	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ulInfo));
	wrq.u.data.length = sizeof(ulInfo);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_RSSI_2**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_STATISTICS	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_STATISTICS));
	wrq.u.data.length = sizeof(NDIS_802_11_STATISTICS);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_STATISTICS**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_GEN_RCV_OK	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ulInfo));
	wrq.u.data.length = sizeof(ulInfo);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_GEN_RCV_OK**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_GEN_RCV_NO_BUFFER	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ulInfo));
	wrq.u.data.length = sizeof(ulInfo);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_GEN_RCV_NO_BUFFER**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_PHY_MODE	typedef enum _RT_802_11_PHY_MODE { PHY_11BG_MIXED = 0, PHY_11B, PHY_11A, PHY_11ABG_MIXED, PHY_11G, PHY_11ABGN_MIXED, // both band 5 PHY_11N, // 6

MEDIATEK CONFIDENTIAL
FOR 2950201@qq.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

	PHY_11GN_MIXED, // 2.4G band 7
	PHY_11AN_MIXED, // 5G band 8
	PHY_11BGN_MIXED, // if check 802.11b. 9
	PHY_11AGN_MIXED, // if check 802.11b. 10
	} RT_802_11_PHY_MODE
	
	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ulInfo));
	wrq.u.data.length = sizeof(ulInfo);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_PHY_MODE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_STA_CONFIG	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RT_802_11_STA_CONFIG));
	wrq.u.data.length = sizeof(RT_802_11_STA_CONFIG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_STA_CONFIG**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_RTS_THRESHOLD	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RtsThresh));
	wrq.u.data.length = sizeof(RtsThresh);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_RTS_THRESHOLD**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_FRAGMENTATION_THRESHOLD	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(FragThresh));
	wrq.u.data.length = sizeof(FragThresh);
	wrq.u.data.pointer = data;
	wrq.u.data.flags =
	OID_802_11_FRAGMENTATION_THRESHOLD;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_POWER_MODE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(PowerMode));
	wrq.u.data.length = sizeof(PowerMode);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_POWER_MODE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_RADIO	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(RadioState));
	wrq.u.data.length = sizeof(RadioState);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_RADIO**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_INFRASTRUCTURE_MODE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(BssType));
	wrq.u.data.length = sizeof(BssType);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_INFRASTRUCTURE_MODE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_PREAMBLE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(PreamType));
	wrq.u.data.length = sizeof(PreamType);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_PREAMBLE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_AUTHENTICATION_MODE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(AuthMode));
	wrq.u.data.length = sizeof(AuthMode);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_AUTHENTICATION_MODE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

MEDIATEK CONFIDENTIAL
FOR 29502010@qq.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

OID_802_11_WEP_STATUS	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(WepStatus));
	wrq.u.data.length = sizeof(WepStatus);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_WEP_STATUS**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_TX_POWER_LEVEL	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_TX_POWER_LEVEL**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_TX_POWER_LEVEL_1	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_TX_POWER_LEVEL_1**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_NETWORK_TYPES_SUPPORTED	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, 16);
	wrq.u.data.length = 16;
	wrq.u.data.pointer = data;
	wrq.u.data.flags =
	OID_802_11_NETWORK_TYPES_SUPPORTED;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

OID_802_11_NETWORK_TYPE_IN_USE	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **OID_802_11_NETWORK_TYPE_IN_USE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_QUERY_EEPROM_VERSION	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags =
	RT_OID_802_11_QUERY_EEPROM_VERSION;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_QUERY_FIRMWARE_VERSION	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags =
	RT_OID_802_11_QUERY_FIRMWARE_VERSION;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_QUERY_NOISE_LEVEL	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(UCHAR));
	wrq.u.data.length = sizeof(UCHAR);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_QUERY_NOISE_LEVEL**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_EXTRA_INFO	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_EXTRA_INFO**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_QUERY_PIDVID	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(ULONG));
	wrq.u.data.length = sizeof(ULONG);

MEDIATEK CONFIDENTIAL
FOR 2950201@qq.com ONLY USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_802_11_QUERY_PIDVID**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_WE_VERSION_COMPILED	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(UINT));
	wrq.u.data.length = sizeof(UINT);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = **RT_OID_WE_VERSION_COMPILED**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_QUERY_LAST_TX_RATE	HTTRANSMIT_SETTING tmpHT;
	sprintf(wrq.ifr_name, "ra0");
	wrq.u.data.pointer = (caddr_t) & tmpHT;
	wrq.u.data.flags = **RT_OID_802_11_QUERY_LAST_TX_RATE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

RT_OID_802_11_QUERY_LAST_RX_RATE	HTTRANSMIT_SETTING tmpHT;
	sprintf(wrq.ifr_name, "ra0");
	wrq.u.data.pointer = (caddr_t) & tmpHT;
	wrq.u.data.flags = **RT_OID_802_11_QUERY_LAST_RX_RATE**;
	ioctl(socket_id, **RT_PRIV_IOCTL**, &wrq);

SHOW_CONN_STATUS	u_char buffer[IW_PRIV_SIZE_MASK];
	sprintf(wrq.ifr_name, "ra0");
	wrq.u.data.pointer = (caddr_t) buffer;
	wrq.u.data.flags = **SHOW_CONN_STATUS**;
	ioctl(socket_id, **RTPRIV_IOCTL_SHOW**, &wrq);

MEDIATEK CONFIDENTIAL
FOR 2950201@ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

10 IOCTL Sample Code

10.1 Check connection status

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <unistd.h>                                 /* for close */
#include <linux/wireless.h>
//============================================================================
#if WIRELESS_EXT <= 11
#ifndef SIOCDEVPRIVATE
#define SIOCDEVPRIVATE                              0x8BE0
#endif
#define SIOCIWFIRSTPRIV                             SIOCDEVPRIVATE
#endif
//SET/GET CONVENTION :
// * ------------------
// * Simplistic summary :
// *          o even numbered ioctls are SET, restricted to root, and should not
// *     return arguments (get_args = 0).
// *          o odd numbered ioctls are GET, authorised to anybody, and should
// *     not expect any arguments (set_args = 0).
//
#define RT_PRIV_IOCTL                               (SIOCIWFIRSTPRIV + 0x0E)
//-----------------------------------------------------------
// IEEE 802.11 OIDs
#define    OID_GEN_MEDIA_CONNECT_STATUS                                     0x060B
//-----------------------------------------------------------
#ifndef    TRUE
#define    TRUE                                     1
#endif

#ifndef    FALSE
#define    FALSE                                    0
#endif
#define PACKED  __attribute__ ((packed))
typedef unsigned int                        NDIS_MEDIA_STATE;
#define NdisMediaStateConnected                             1
#define NdisMediaStateDisconnected                  0
//============================================================================
int main( int argc, char ** argv )
{
#define DATA_BUFFER_SIZE 8192
        char        name[25];
        int     socket_id;
        struct  iwreq wrq;
        int     ret = 1, cmd = 0;
        char        *base;//, *base1;
        char        *data;
        // open socket based on address family: AF_NET ----------------------------
        socket_id = socket(AF_INET, SOCK_DGRAM, 0);
        if(socket_id < 0)
        {
                printf("\nrtuser::error::Open socket error!\n\n");
```

MEDIA TEK CONFIDENTIAL
FOR 20130201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
		return -1;
	}
	data = malloc(DATA_BUFFER_SIZE);
	if (data == NULL)
	{
		printf("unable to alloc data buffer, size (%d)\n", DATA_BUFFER_SIZE);
		return -1;
	}
	// set interface name as "ra0" ---------------------------------------------
	sprintf(name, "ra0");
	memset(data, 0, DATA_BUFFER_SIZE);
	//
	//example of ioctl function ========================================
	//
	base = argv[1];
	//base1 = argv[2];
	if(argc != 2)
		goto rtuser_exit;
	if(strstr(base, "OID_GEN_MEDIA_CONNECT_STATUS"))
		cmd = OID_GEN_MEDIA_CONNECT_STATUS;
	switch(cmd)
	{

	case OID_GEN_MEDIA_CONNECT_STATUS:
		{
			// Get media connect status
			printf("\nrtuser::set OID_GEN_MEDIA_CONNECT_STATUS \n\n");
			strcpy(wrq.ifr_name, name);
			wrq.u.data.length = sizeof(NDIS_MEDIA_STATE);;
			wrq.u.data.pointer = data;
			wrq.u.data.flags = OID_GEN_MEDIA_CONNECT_STATUS;
			ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
			if(ret != 0)
			{
				printf("\nrtuser::error::media connect status\n\n");
				goto rtuser_exit;
			}
			printf("Media connect status=%u\n\n", (NDIS_MEDIA_STATE)(*data));
			if ( (NDIS_MEDIA_STATE)(*data) == NdisMediaStateConnected )
				printf("NdisMediaStateConnected\n");
			else
				printf("NdisMediaStateDisconnected");
		}
		break;
	default:
		printf("\nUnsupported OID type\n");
		break;
	}
	rtuser_exit:
	if(socket_id >= 0)
		close(socket_id);
	if (data != NULL)
		free(data);
	if(ret)
		return ret;
	else
		return 0;
}// endof main()
```

MEDIATEK CONFIDENTIAL
FOR 2950020(@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

10.2 Polling WPS status

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <netinet/in.h> /* for sockaddr_in */
#include <fcntl.h>
#include <time.h>
#include <sys/times.h>
#include <unistd.h>
#include <sys/socket.h>
#include <sys/stat.h>
#include <err.h>
#include <errno.h>
#include <asm/types.h>

#define IFNAMSIZ 16
#include </usr/include/linux/wireless.h>
#include <sys/ioctl.h>

#define RT_PRIV_IOCTL (SIOCIWFIRSTPRIV + 0x01)          // copy from raconfig.h
#define RT_OID_WSC_QUERY_STATUS 0x0751  // copy from webui src
typedef int WSC_STATE;
#define DBGPRINT printf
WSC_STATE rt_oid_get_wsc_status(char *ifname, int skfd){
  struct iwreq wrq;
  int data;
  memset(&wrq, 0, sizeof(wrq));
  strcpy(wrq.ifr_ifrn.ifrn_name, ifname);
  wrq.u.data.length = sizeof(int);
  wrq.u.data.pointer = &data;
  wrq.u.data.flags = RT_OID_WSC_QUERY_STATUS ;
  ioctl(skfd, RT_PRIV_IOCTL ,&wrq);
  DBGPRINT("wsc query status = %d \n", data);
  return data;
}

int main(int argc, char *argv[])
{
  unsigned num_sec = 0;
  int skfd;

  skfd = socket(AF_INET, SOCK_DGRAM, 0);
  if (skfd < 0)
    return -1;
  if (argc < 2)
  {
    printf("Usage: %s <# secs to poll wps status> \n", argv[0]);
    return 1;
  }
  num_sec = atoi(argv[1]);
  while (num_sec-- > 0)
  {
    rt_oid_get_wsc_status("ra0", skfd);
    sleep(1);
  }
  return 0;
}
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

10.3 Site survey

```
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <sys/socket.h>
#include <sys/ioctl.h>
#include <unistd.h> /* for close */
#include <linux/wireless.h>

#define RT_PRIV_IOCTL       (SIOCIWFIRSTPRIV + 0x0E)
#define    OID_GET_SET_TOGGLE                   0x8000
#define    OID_802_11_BSSID_LIST_SCAN           (0x0508 | OID_GET_SET_TOGGLE)
#define    OID_802_11_BSSID_LIST                0x0609
#define PACKED  __attribute__ ((packed))

//security
#define WPS_OUI_TYPE                0x04F25000
#define WPS_PDID                    0x1210
#define WPA_OUI_TYPE                0x01F25000
#define WPA_OUI                     0x00F25000
#define WPA2_OUI                    0x00AC0F00
#define CISCO_OUI                   0x00964000

//===============
#define NDIS_802_11_LENGTH_SSID         32
#define NDIS_802_11_LENGTH_RATES_EX     16
#define NDIS_802_11_LENGTH_RATES        8

typedef long    NDIS_802_11_RSSI;           // in dBm
typedef unsigned char   NDIS_802_11_MAC_ADDRESS[6];
typedef unsigned char   NDIS_802_11_RATES_EX[NDIS_802_11_LENGTH_RATES_EX];  // Set of 16 data rates
typedef unsigned char   NDIS_802_11_RATES[NDIS_802_11_LENGTH_RATES];        // Set of 8 data rates

typedef struct PACKED _NDIS_802_11_FIXED_IEs
{
    unsigned char Timestamp[8];
    unsigned short BeaconInterval;
    unsigned short Capabilities;
} NDIS_802_11_FIXED_IEs, *PNDIS_802_11_FIXED_IEs;

typedef struct _NDIS_802_11_VARIABLE_IEs
{
    unsigned char ElementID;
    unsigned char Length;    // Number of bytes in data field
    unsigned char data[1];
} NDIS_802_11_VARIABLE_IEs, *PNDIS_802_11_VARIABLE_IEs;

typedef enum _NDIS_802_11_NETWORK_INFRASTRUCTURE
{
   Ndis802_11IBSS,
   Ndis802_11Infrastructure,
   Ndis802_11AutoUnknown,
   Ndis802_11Monitor,
   Ndis802_11InfrastructureMax     // Not a real value, defined as upper bound
} NDIS_802_11_NETWORK_INFRASTRUCTURE, *PNDIS_802_11_NETWORK_INFRASTRUCTURE;

typedef struct _NDIS_802_11_CONFIGURATION_FH
{
   unsigned long         Length;            // Length of structure
```

MEDIATEK CONFIDENTIAL
FOR 2500201@ ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
  unsigned long         HopPattern;       // As defined by 802.11, MSB set
  unsigned long         HopSet;           // to one if non-802.11
  unsigned long         DwellTime;        // units are Kusec
} NDIS_802_11_CONFIGURATION_FH, *PNDIS_802_11_CONFIGURATION_FH;

typedef struct _NDIS_802_11_CONFIGURATION
{
  unsigned long                 Length;           // Length of structure
  unsigned long                 BeaconPeriod;     // units are Kusec
  unsigned long                 ATIMWindow;       // units are Kusec
  unsigned long                 DSConfig;         // Frequency, units are kHz
  NDIS_802_11_CONFIGURATION_FH  FHConfig;
} NDIS_802_11_CONFIGURATION, *PNDIS_802_11_CONFIGURATION;


typedef enum _NDIS_802_11_NETWORK_TYPE
{
  Ndis802_11FH,
  Ndis802_11DS,
   Ndis802_11OFDM5,
   Ndis802_11OFDM5_N,
   Ndis802_11OFDM24,
   Ndis802_11OFDM24_N,
  Ndis802_11Automode,
   Ndis802_11NetworkTypeMax    // not a real type, defined as an upper bound
} NDIS_802_11_NETWORK_TYPE, *PNDIS_802_11_NETWORK_TYPE;

typedef struct PACKED
{
   unsigned int   SsidLength;       // length of SSID field below, in bytes;
                  // this can be zero.
   unsigned char   Ssid[NDIS_802_11_LENGTH_SSID];         // SSID information field
} NDIS_802_11_SSID, *PNDIS_802_11_SSID;


typedef struct PACKED
{
   unsigned long                Length;          // Length of this structure
   NDIS_802_11_MAC_ADDRESS      MacAddress;      // BSSID
   unsigned char                Reserved[2];
   NDIS_802_11_SSID             Ssid;            // SSID
   unsigned int                 Privacy;         // WEP encryption requirement
   NDIS_802_11_RSSI             Rssi;            // receive signal
   NDIS_802_11_NETWORK_TYPE     NetworkTypeInUse;
   NDIS_802_11_CONFIGURATION    Configuration;
   NDIS_802_11_NETWORK_INFRASTRUCTURE  InfrastructureMode;
   NDIS_802_11_RATES_EX         SupportedRates;
   unsigned long                IELength;
   unsigned char                IEs[1];
} NDIS_WLAN_BSSID_EX, *PNDIS_WLAN_BSSID_EX;

typedef struct PACKED _NDIS_WLAN_BSSID
{
  unsigned long                 Length;     // Length of this structure
  NDIS_802_11_MAC_ADDRESS       MacAddress; // BSSID
  unsigned char                 Reserved[2];
  NDIS_802_11_SSID              Ssid;       // SSID
  unsigned long                 Privacy;    // WEP encryption requirement
  NDIS_802_11_RSSI              Rssi;       // receive signal strength in dBm
  NDIS_802_11_NETWORK_TYPE      NetworkTypeInUse;
```

FOR 2950201@ogemray.com USE ONLY
MEDIATEK CONFIDENTIAL

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
    NDIS_802_11_CONFIGURATION          Configuration;
    NDIS_802_11_NETWORK_INFRASTRUCTURE  InfrastructureMode;
    NDIS_802_11_RATES                  SupportedRates;
} NDIS_WLAN_BSSID, *PNDIS_WLAN_BSSID;

typedef struct PACKED
{
    unsigned int                NumberOfItems;      // in list below, at least 1
    NDIS_WLAN_BSSID_EX      Bssid[1];
} NDIS_802_11_BSSID_LIST_EX, *PNDIS_802_11_BSSID_LIST_EX;
//===============



void checkSecurity(PNDIS_WLAN_BSSID_EX pBssid){

                                    // work with NDIS_WLAN_BSSID_EX
                                    int bTKIP = 0;
                                    int bAESWRAP = 0;
                                    int bAESCCMP = 0;
                                    int bWPA = 0;
                                    int bWPAPSK = 0;
                                    int bWPANONE = 0;
                                    int bWPA2 = 0;
                                    int bWPA2PSK = 0;
                                    int bWPA2NONE = 0;
                                    int bCCKM = 0; // CCKM for Cisco

                                    char strAuth[32] = "";
                                    char strEncry[32] = "";

    if ((pBssid->Length > sizeof(NDIS_WLAN_BSSID)) && (pBssid->IELength > sizeof(NDIS_802_11_FIXED_IEs))){
                                        unsigned int lIELoc = 0;
                                        PNDIS_802_11_FIXED_IEs pFixIE = (PNDIS_802_11_FIXED_IEs)pBssid->IEs;
                                        PNDIS_802_11_VARIABLE_IEs                   pVarIE                  =
(PNDIS_802_11_VARIABLE_IEs)((char*)pFixIE + sizeof(NDIS_802_11_FIXED_IEs));
                                        lIELoc += sizeof(NDIS_802_11_FIXED_IEs);

                                        while (pBssid->IELength > (lIELoc + sizeof(NDIS_802_11_VARIABLE_IEs)))
                                        {
                                            if ((pVarIE->ElementID == 221) && (pVarIE->Length >= 16))
                                            {
                                                //unsigned int* pOUI = (unsigned int*)((char*)pVarIE + 2);
                                                unsigned int* pOUI = (unsigned int*)((char*)pVarIE->data);
                                                if(*pOUI != WPA_OUI_TYPE)
                                                {
                                                    lIELoc += pVarIE->Length;
                                                    lIELoc += 2;
                                                    pVarIE                                                  =
(PNDIS_802_11_VARIABLE_IEs)((char*)pVarIE + pVarIE->Length + 2);

                                                    if(pVarIE->Length <= 0)
                                                        break;
                                                    continue;
                                                }
                                                unsigned int* plGroupKey;
                                                unsigned short* pdPairKeyCount;
                                                unsigned int* plPairwiseKey=NULL;
                                                unsigned int* plAuthenKey=NULL;
                                                unsigned short* pdAuthenKeyCount;
```

MEDIATEK CONFIDENTIAL
FOR 2950201@cmray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
                                        plGroupKey = (unsigned int*)((char*)pVarIE + 8);

                                        unsigned int lGroupKey = *plGroupKey & 0x00ffffff;
                                        if (lGroupKey == WPA_OUI)
                                        {
                                                lGroupKey = (*plGroupKey & 0xff000000) >> 0x18;
                                                if (lGroupKey == 2)
                                                        bTKIP = 1;
                                                else if (lGroupKey == 3)
                                                        bAESWRAP = 1;
                                                else if (lGroupKey == 4)
                                                        bAESCCMP = 1;
                                        }
                                        else
                                        {
                                                lIELoc += pVarIE->Length;
                                                lIELoc += 2;
                                                pVarIE                                          =
(PNDIS_802_11_VARIABLE_IEs)((char*)pVarIE + pVarIE->Length + 2);

                                                if(pVarIE->Length <= 0)
                                                        break;

                                                continue;
                                        }

                                        pdPairKeyCount = (unsigned short*)((char*)plGroupKey + 4);
                                        plPairwiseKey = (unsigned int*) ((char*)pdPairKeyCount + 2);
                                        unsigned short k = 0;
                                        for( k = 0; k < *pdPairKeyCount; k++)
                                        {
                                                unsigned int lPairKey = *plPairwiseKey & 0x00ffffff;
                                                if(lPairKey == WPA_OUI )//|| (lPairKey & 0xffffff00) ==
WPA_OUI_1)

                                                {
                                                        lPairKey = (*plPairwiseKey & 0xff000000) >>
0x18;

                                                        if(lPairKey == 2){
                                                                bTKIP = 1;
                                                        }
                                                        else if(lPairKey == 3){
                                                                bAESWRAP = 1;
                                                        }
                                                        else if(lPairKey == 4){
                                                                bAESCCMP = 1;
                                                        }
                                                }
                                                else
                                                        break;

                                                ++plPairwiseKey;
                                        }

                                        pdAuthenKeyCount = (unsigned short*)((char*)pdPairKeyCount +
2 + 4 * (*pdPairKeyCount));

                                        plAuthenKey = (unsigned int*)((char*)pdAuthenKeyCount + 2);
                                        for(k = 0; k < *pdAuthenKeyCount; k++)
                                        {
                                                unsigned int lAuthenKey = *plAuthenKey & 0x00ffffff;
                                                if(lAuthenKey == CISCO_OUI)
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
                                        {
                                                bCCKM = 1; // CCKM for Cisco
                                        }
                                        else if (lAuthenKey == WPA_OUI)
                                        {
                                                lAuthenKey  =  (*plAuthenKey  &  0xff000000)
>> 0x18;

                                                if(lAuthenKey == 1){
                                                        bWPA = 1;
                                                }
                                                else if(lAuthenKey == 0 || lAuthenKey == 2)
                                                {
                                                        if(pBssid->InfrastructureMode){
                                                                bWPAPSK = 1;
                                                        }
                                                        else
                                                                bWPANONE = 1;
                                                }
                                        }

                                        ++plAuthenKey;
                                }
                        }
                        else if(pVarIE->ElementID == 48 && pVarIE->Length >= 12)
                        {
                                unsigned int* plGroupKey;
                                unsigned int* plPairwiseKey;
                                unsigned short* pdPairKeyCount;
                                unsigned int* plAuthenKey;
                                unsigned short* pdAuthenKeyCount;
                                plGroupKey = (unsigned int*)((char*)pVarIE + 4);

                                unsigned int lGroupKey = *plGroupKey & 0x00ffffff;
                                if(lGroupKey == WPA2_OUI)
                                {
                                        lGroupKey = (*plGroupKey & 0xff000000) >> 0x18;
                                        if(lGroupKey == 2){
                                                bTKIP = 1;
                                        }
                                        else if(lGroupKey == 3){
                                                bAESWRAP = 1;
                                        }
                                        else if(lGroupKey == 4){
                                                bAESCCMP = 1;
                                        }
                                }
                                else
                                {
                                        lIELoc += pVarIE->Length;
                                        lIELoc += 2;
                                        pVarIE                                                  =
(PNDIS_802_11_VARIABLE_IEs)((char*)pVarIE + pVarIE->Length + 2);
                                        if(pVarIE->Length <= 0)
                                                break;

                                        continue;
                                }

                                pdPairKeyCount = (unsigned short*)((char*)plGroupKey + 4);
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogeiray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
                                        plPairwiseKey = (unsigned int*)((char*)pdPairKeyCount + 2);
                                        unsigned short k = 0;
                                        for( k = 0; k < *pdPairKeyCount; k++)
                                        {
                                                unsigned int lPairKey = *plPairwiseKey & 0x00ffffff;
                                                if(lPairKey == WPA2_OUI)
                                                {
                                                        lPairKey = (*plPairwiseKey & 0xff000000) >>
0x18;

                                                        if(lPairKey == 2){
                                                                bTKIP = 1;
                                                        }
                                                        else if(lPairKey == 3){
                                                                bAESWRAP = 1;
                                                        }
                                                        else if(lPairKey == 4){
                                                                bAESCCMP = 1;
                                                        }
                                                }
                                                else
                                                        break;

                                                ++plPairwiseKey;
                                        }

                                        pdAuthenKeyCount = (unsigned short*)((char*)pdPairKeyCount +
2 + 4 * *pdPairKeyCount);
                                        plAuthenKey = (unsigned int*)((char*)pdAuthenKeyCount + 2);

                                        for(k = 0; k < *pdAuthenKeyCount; k++)
                                        {
                                                unsigned int lAuthenKey = *plAuthenKey & 0x00ffffff;
                                                if(lAuthenKey == CISCO_OUI)
                                                {
                                                        bCCKM = 1; // CCKM for Cisco
                                                }
                        else if (lAuthenKey == WPA2_OUI)
                                                {
                                                        lAuthenKey = (*plAuthenKey & 0xff000000)
>> 0x18;

                                                        if(lAuthenKey == 1)
                                                                bWPA2 = 1;
                                                        else if(lAuthenKey == 0 || lAuthenKey == 2)
                                                        {
                                                                if(pBssid->InfrastructureMode)
                                                                        bWPA2PSK = 1;
                                                                else
                                                                        bWPA2NONE = 1;
                                                        }
                                                }

                                                ++plAuthenKey;
                                        }
                                }

                                lIELoc += pVarIE->Length;
                                lIELoc += 2;
                                pVarIE = (PNDIS_802_11_VARIABLE_IEs)((char*)pVarIE + pVarIE->Length
+ 2);

                                if(pVarIE->Length <= 0)
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
                                        break;
                        }
                }
                //print security
                        //if(bCCKM)
                                //printf("CCKM; ");
                        if (bWPA)
                                strcat(strAuth, "WPA");
                        if (bWPAPSK)
                                strcat(strAuth, "WPAPSK");
                        if (bWPANONE)
                                strcat(strAuth, "WPA-NONE");
                        if (bWPA2)
                                strcat(strAuth, "WPA2");
                        if (bWPA2PSK)
                                strcat(strAuth, "WPA2PSK");
                        if (bWPA2NONE)
                                strcat(strAuth, "WPA2-NONE");

                        //if (strlen(strAuth) > 0)
                        //{
                                //strncpy((char *)tmpAuth, strAuth, strlen(strAuth) - 2);
                                //strcpy(strAuth, (char *)tmpAuth);
                                //printf("Unknown 01");
                        //}
                        //else
                        //{
                                //printf("Unknown 02");
                        //}

                        if(bTKIP)
                                strcat(strEncry, "TKIP");
                        if(bAESWRAP || bAESCCMP)
                                strcat(strEncry, "AES");


                        //if(strlen(strEncry) > 0)
                        //{
                                //strncpy((char *)tmpEncry, strEncry, strlen(strEncry) - 2);
                                //strcpy(strEncry, (char *)tmpEncry);
                                //printf("Unknown 03");
                        //}
                        //else
                        //{
                                if (pBssid->Privacy)
                                {
                                        if(strlen(strEncry) < 2)
                                        {
                                           printf("WEP");
                                        }
                                        else
                                        {
                                                printf("%s/%s", strAuth,strEncry);
                                        }
                                }
                                else
                                {
                                        printf("NONE");
                                }
                        //}
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogmray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
}


int main(int argc, char** argv)
{
        int i, socket_id;
        struct iwreq wrq, wrq1;
        int ret;
        unsigned int bufLen = 1024*100;
        PNDIS_802_11_BSSID_LIST_EX pBssidList;
        PNDIS_WLAN_BSSID_EX pBssid;
        PNDIS_802_11_FIXED_IEs pFixIE;
   PNDIS_802_11_VARIABLE_IEs pVarIE;

        socket_id = socket(AF_INET,SOCK_DGRAM,0);
        if(socket_id<0)
        {
                printf("\n Open socket error!\n");
                goto error_exit;
        }

        //Scanning
        strcpy(wrq1.ifr_name, "ra0");
        wrq1.u.data.flags = OID_802_11_BSSID_LIST_SCAN;
        ret = ioctl(socket_id,RT_PRIV_IOCTL,&wrq1);
        if(ret != 0)
        {
                printf("\n Scan fail %d\n", ret);
                goto error_exit;
        }

        //Get List
        pBssidList = (PNDIS_802_11_BSSID_LIST_EX)malloc(bufLen); //64k
        memset(pBssidList, 0x00, bufLen);

        strcpy(wrq.ifr_name, "ra0");
        wrq.u.data.length = bufLen;
        wrq.u.data.pointer = pBssidList;
        wrq.u.data.flags = OID_802_11_BSSID_LIST;
        ret = ioctl(socket_id,RT_PRIV_IOCTL,&wrq);
        do{
                ret = ioctl(socket_id,RT_PRIV_IOCTL,&wrq);
                printf("Scanning...\n");
                if(ret == EAGAIN)
                        printf("EAGAIN...\n");
                else if(ret == E2BIG)
                        printf("E2BIG...\n");
                else
                        sleep(1);
        }while(ret == -1);

        printf("\n=======GET BSSID LIST========\n");
        pBssid = (PNDIS_WLAN_BSSID_EX)(pBssidList->Bssid);
        for (i = 0; i < pBssidList->NumberOfItems; i++){
                if(pBssid->Ssid.SsidLength == 0)
                        printf("%d: %-33s\t",i+1, " ");
                else{
                        printf("%d: %-33s\t", i+1, pBssid->Ssid.Ssid);
                }
```

MEDIATEK CONFIDENTIAL
FOR 95060161@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
                printf("MAC:%02x:%02x:%02x:%02x:%02x:%02x\t",
                                                pBssid->MacAddress[0],
                                                pBssid->MacAddress[1],
                                                pBssid->MacAddress[2],
                                                pBssid->MacAddress[3],
                                                pBssid->MacAddress[4],
                                                pBssid->MacAddress[5]);
                printf("Rssi:%d\t",pBssid->Rssi);
                //Networktype
                if(pBssid->InfrastructureMode == 0)
                        printf("Ad-hoc\t");
                else
                        printf("Infra\t",pBssid->Rssi);

                //wirelessMode
                if(pBssid->NetworkTypeInUse == 1)
                        printf("11b\t");
                else if(pBssid->NetworkTypeInUse == 3)
                        printf("11bg\t", pBssid->NetworkTypeInUse);
                else if(pBssid->NetworkTypeInUse == 6)
                        printf("11bgn\t", pBssid->NetworkTypeInUse);
                else
                        printf("unknown\t", pBssid->NetworkTypeInUse);
        //Security
                pFixIE = (PNDIS_802_11_FIXED_IEs)pBssid->IEs;
                pVarIE = (PNDIS_802_11_VARIABLE_IEs)((char*)pFixIE + sizeof(NDIS_802_11_FIXED_IEs));
                checkSecurity(pBssid);

                printf("\n");
                //move forward
                pBssid = (PNDIS_WLAN_BSSID_EX)((char *)pBssid + pBssid->Length);
        }

        error_exit:
                if(socket_id>=0)
                        close(socket_id);
                return 1;
}
```

10.4 Display Rate and BW

```
        HTTRANSMIT_SETTING HTSetting;
        Double Rate;
        double b_mode[] ={1, 2, 5.5, 11};
        float g_Rate[] = { 6,9,12,18,24,36,48,54};
        switch(HTSetting.field.MODE)
        {
                case 0:
                        if (HTSetting.field.MCS >=0 && HTSetting.field.MCS<=3)
                                Rate = b_mode[HTSetting.field.MCS];
                        else if (HTSetting.field.MCS >=8 && HTSetting.field.MCS<=11)
                                Rate = b_mode[HTSetting.field.MCS-8];
                        else
                                Rate = 0;
                        break;
                case 1:
                        if ((HTSetting.field.MCS >= 0) && (HTSetting.field.MCS < 8))
                                Rate = g_Rate[HTSetting.field.MCS];
                        else
                                Rate = 0;
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.

```
                break;
        case 2:
        case 3:
                if (0 == bGetHTTxRateByBW_GI_MCS(HTSetting.field.BW, HTSetting.field.ShortGI,
                                HTSetting.field.MCS,
                                &Rate))
                        Rate = 0;
                break;
        default:
                Rate = 0;
                break;
}

char bGetHTTxRateByBW_GI_MCS(int nBW, int nGI, int nMCS, double* dRate)
{
        double HTTxRate20_800[16]={6.5, 13.0, 19.5, 26.0, 39.0, 52.0, 58.5, 65.0, 13.0, 26.0, 39.0, 52.0, 78.0, 104.0,
        117.0, 130.0};
        double HTTxRate20_400[16]={7.2, 14.4, 21.7, 28.9, 43.3, 57.8, 65.0, 72.2, 14.444, 28.889, 43.333, 57.778,
        86.667, 115.556, 130.000, 144.444};
        double HTTxRate40_800[18]={13.5, 27.0, 40.5, 54.0, 81.0, 108.0, 121.5, 135.0, 27.0, 54.0, 81.0, 108.0,
        162.0, 216.0, 243.0, 270.0, 6.0, 39.0};
        double HTTxRate40_400[18]={15.0, 30.0, 45.0, 60.0, 90.0, 120.0, 135.0, 150.0, 30.0, 60.0, 90.0, 120.0,
        180.0, 240.0, 270.0, 300.0, 6.7, 43.3};

        // no TxRate for (BW = 20, GI = 400, MCS = 32) & (BW = 20, GI = 400, MCS = 32)
        if (((nBW == BW_20) && (nGI == GI_400) && (nMCS == 32)) ||
                        ((nBW == BW_20) && (nGI == GI_800) && (nMCS == 32)))
                return 0; //false

        if( nBW == BW_20 && nGI == GI_800)
                *dRate = HTTxRate20_800[nMCS];
        else if( nBW == BW_20 && nGI == GI_400)
                *dRate = HTTxRate20_400[nMCS];
        else if( nBW == BW_40 && nGI == GI_800)
                *dRate = HTTxRate40_800[nMCS];
        else if( nBW == BW_40 && nGI == GI_400)
                *dRate = HTTxRate40_400[nMCS];
        else
                return 0; //false

        return 1; //true
}
```

MEDIATEK CONFIDENTIAL
FOR 2950201@ogemray.com USE ONLY

This document contains information that is proprietary to MediaTek Inc.
Unauthorized reproduction or disclosure of this information in whole or in part is strictly prohibited.