Mediatek Wi-Fi AP Software Programming Guide

Version: 4.4
Release date: 2014-10-27

Document Revision History

Revision	Date	Author	Description
1.0	2012/11/08	Pan Liu	Initial Version
1.1	2012/11/13	Pan Liu	Update iwpriv command
1.2	2012/12/11	Pan Liu	Add NoForwardingMBCast
1.3	2013/01/04	Pan Liu	Add VHT_BW and VhtBW
1.4	2013/1/14	Pan Liu	Update Apclient WPS command sample
1.5	2013/1/22	Pan Liu	Add FAQ- FixTxMode iwpriv command sample
1.6	2013/1/23	Pan Liu	Add new DAT item VHT_DisallowNonVHT and SingleSKU.dat sample.
1.7	2013/3/6	Pan Liu	Add MAC Repeater section
1.8	2013/3/8	Pan Liu	Add command and profile, DFS debug example
1.9	2013/3/13	Pan Liu	Add Singlesku.dat 5G and 2.4G sample profile and DFS example update
2.0	2013/3/15	Pan Liu	Add IgmpAdd1, WPS command and NEW BSSID Mode MAC address limitation. Update BGProtection
2.1	2013/3/27	Pan Liu	Add EfuseUploadToHost
2.2	2013/3/28	Pan Liu	Add FAQ for TX/RX unbalance issue.
2.3	2013/4/23	Pan Liu	Add iNIC system address configuration for WLAN profile support
2.4	2013/4/23	Pan Liu	Add iwpriv command AP2040Rescan, WLAN profile updates
2.5	2013/5/27	Pan Liu	Add WLAN profile and iwpriv parameters for VHT support.
2.6	2013/6/20	Pan Liu	Update WirelessMode=15, correct NoForwardingMBCast, Add AutoChannelSkipList
2.7	2013/7/4	Pan Liu	Add WLAN profile “EtherTrafficBand”
2.8	2013/7/26	Pan Liu	Add iNIC only profile and iwpriv command
2.9	2013/8/23	Pan Liu	Add iNIC only profile IsolateCard, EnhanceMultiClient, and BGMultiClient.
3.0	2013/8/29	Pan Liu	Add iwpriv command fpga_on, dataphy, databw, databasize, datagi, dataldpc for vht mode data rate setting.
3.1	2013/9/03	Pan Liu	Correct TYPO on DisConnectAllSta
3.2	2013/10/03	Pan Liu	Add VHT MCS table in Q&A
3.3	2013/11/20	Pan Liu	Update Multiple Radiu server usage
3.4	2014/01/08	Pan Liu	Add iNIC only new profile parameters
3.5	2014/01/20	Pan Liu	Update iwpriv commands and APClient command example
3.6	2014/02/11	Pan Liu	Add note for WpaMixPairCipher
3.7	2014/02/27	Pan Liu	Add iwpriv command ApCliAutoConnect and update SiteSurvey
3.8	2014/03/07	Pan Liu	Remove RadioOn from profile SoftAP is not support this option
3.9	2014/03/07	Pan Liu	Add iNIC profile TX&RTS retry counter and EDCCA profile
4.0	2014/04/01	Pan Liu	Update BADeline, datamcs and FixTxMode iwpriv command samples
4.1	2014/05/29	Hughes Kang	Add EDCCA testing
4.2	2014/07/01	Hughes Kang	Add HT_PROTECT, BASetup, BAOriTearDown, BARecTearDown, HT_TxStream, HT_RxStream, HtTxStream, HtRxStream, EntryLifeCheck, WAPI related parameters,

			WscStop
4.3	2014/09/16	Hughes Kang	Add PMF
4.4	2014/10/24	Money Wang	Update ● WDS ● WMM ● PMF ● Security ● AP-Client ● MAC Repeater ● IGMP Snooping ● MBSSID ● How to Fix Data Rate ● FAQ

Table of Contents

1 Introduction

This document is a software programming guide for Mediatek Wi-Fi SoftAP driver and it teaches you how to configure your own settings. We do provide two kinds of configuration method, profile and iwpriv. Later we show you the profile parameter list, the iwpriv command list, and some OID examples to demonstrate how to fully utilize the WLAN driver.

2 WLAN SoftAP Driver Profile

2.1 Sample Profile

```
#The word of "Default" must not be removed
Default
CountryRegion=5
CountryRegionABand=7
CountryCode=TW
BssidNum=1
SSID=RT2860AP
WirelessMode=9
TxRate=0
Channel=11
BasicRate=15
BeaconPeriod=100
DtimPeriod=1
TxPower=100
DisableOLBC=0
BGProtection=0
TxAntenna=
RxAntenna=
TxPreamble=0
RTSThreshold=2347
FragThreshold=2346
TxBurst=1
PktAggregate=0
TurboRate=0
WmmCapable=0
APSDCapable=0
DLSCapable=0
APAifsn=3;7;1;1
APCwmin=4;4;3;2
APCwmax=6;10;4;3
APTxop=0;0;94;47
APACM=0;0;0;0
BSSAifsn=3;7;2;2
BSSCwmin=4;4;3;2
BSSCwmax=10;10;4;3
BSSTxop=0;0;94;47
BSSACM=0;0;0;0
AckPolicy=0;0;0;0
NoForwarding=0
NoForwardingBTNBSSID=0
HideSSID=0
StationKeepAlive=0
ShortSlot=1
AutoChannelSelect=0
IEEE8021X=0
IEEE80211H=0
CSPeriod=10
WirelessEvent=0
IdsEnable=0
AuthFloodThreshold=32
```

```
AssocReqFloodThreshold=32
ReassocReqFloodThreshold=32
ProbeReqFloodThreshold=32
DisassocFloodThreshold=32
DeauthFloodThreshold=32
EapReqFooldThreshold=32
PreAuth=0
AuthMode=OPEN
EncrypType=NONE
RekeyInterval=0
RekeyMethod=DISABLE
PMKCachePeriod=10
WPAPSK=
DefaultKeyID=1
Key1Type=0
Key1Str=
Key2Type=0
Key2Str=
Key3Type=0
Key3Str=
Key4Type=0
Key4Str=
AccessPolicy0=0
AccessControlList0=
AccessPolicy1=0
AccessControlList1=
AccessPolicy2=0
AccessControlList2=
AccessPolicy3=0
AccessControlList3=
WdsEnable=0
WdsEncrypType=NONE
WdsList=
WdsKey=
RADIUS_Server=192.168.2.3
RADIUS_Port=1812
RADIUS_Key=ralink
own_ip_addr=192.168.5.234
EAPifname=br0
PreAuthifname=br0
HT_HTC=0
HT_RDG=0
HT_EXTCHA=0
HT_LinkAdapt=0
HT_OpMode=0
HT_MpduDensity=5
HT_BW=1
VHT_BW=1
VHT_SGI=1
VHT_STBC=0
VHT_BW_SIGNAL=0
VHT_DisallowNonVHT=0
VHT_LDPC=
HT_AutoBA=1
HT_AMSDU=0
HT_BAWinSize=64
HT_GI=1
```

HT_MCS=33
WscManufacturer=
WscModelName=
WscDeviceName=
WscModelNumber=
WscSerialNumber=

2.2 Common WLAN Profile Parameters

As you could see in *Section 2.1 Sample Profile*, all the settings obey the following syntax.

[Syntax]

Parameter=Value

The WLAN driver needs to be restarted after changing the profile. Otherwise, settings would not take effect and an interface down/up cycle could help.

ifconfig ra0 down
ifconfig ra0 up

2.2.1 CountryRegion

Description: Country region for WLAN radio 2.4 GHz regulation (G band)
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

2.2.2 CountryRegionABand

Description: Country region for WLAN radio 5 GHz regulation (A band)
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

2.2.3 CountryCode

Description: County code for WLAN radio regulation
Value:

CountryCode=

Note:
Default is empy.
2 characters, like TW for Taiwan.
Please refer to the following link for ISO3166 code list for other countries.
http://www.iso.org/iso/prods-services/iso3166ma/02iso-3166-code-lists/country_names_and_code_elements
This parameter can also be configured in EEPROM or eFuse.
Configuration in EEPROM or eFuse has higher priority than that in WLAN Profile.

2.2.4 ChannelGeography

Description: For Channel list builder
Value:

ChannelGeography=1

0: Outdoor
1: Indoor
2: Both

2.2.5 SSID

Description: The target BSSID string name configuration
Value:

SSID=11n-AP

0~z, 1~32 ASCII characters

2.2.6 WirelessMode

Description: Wireless mode configuration
Value:

WirelessMode=9

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 in 2.4G
7: 11GN mixed
8: 11AN mixed
9: 11BGN mixed
10: 11AGN mixed
11: 11N only in 5G
14: 11A/AN/AC mixed 5G band only (Only 11AC chipset support)
15: 11 AN/AC mixed 5G band only (Only 11AC chipset support)

2.2.7 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.

2.2.8 BasicRate

Description: Basic rate support
Value:

BasicRate=15

0~4095

Note:

A bitmap represent basic support rate (A mode not support)
1: Basic rate-1Mbps
2: Basic rate-2Mbps
3: Basic rate-1Mbps, 2Mbps
4: Basic rate-5.5Mbps
15: Basic rate-1Mbps, 2Mbps, 5.5Mbps, 11Mbps
Examples:

Basic Rate Bit Map (max. 12-bit, represent max. 12 basic rates)												
Bit	11	10	9	8	7	6	5	4	3	2	1	0
Rate	54	48	36	24	18	12	9	6	11	5.5	2	1
Set	0	1	0	1	0	1	0	1	1	1	1	1
Hex	5				5				F			
Decimal	1375											

Note:

Set correct basic rates set before changing wireless mode.

11B/G Mixed, 11B/G/N Mixed, and 11N Only:
iwpriv ra0 set BasicRate=15 ➔ (0x0F: 1, 2, 5.5, 11 Mbps)

11B:
iwpriv ra0 set BasicRate=3 ➔ (0x03: 1, 2 Mbps)
11G-Only and 11G/N Mixed:
iwpriv ra0 set BasicRate=351 ➔ (0x15F: 1, 2, 5.5, 11, 6, 12, 24 Mbps)

2.2.9 BeaconPeriod

Description: Beacon period setting (It is SoftAP only)
Value:

BeaconPeriod=100

2.2.10 DtimPeriod

Description: DTIM period
Value:

DtimPeriod=1

1~255

2.2.11 TxPower

Description: WLAN Radio Transmit Power setting in percentage
Value:

TxPower=100

0~100

2.2.12 DisableOLBC

Description: Enable or disable OLBC (Overlapping Legacy BSS Condition)
Value:

DisableOLBC=0

0: disable
1: enable

2.2.13 BGProtection

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

BGProtection=0

0: AUTO
1: On
2: Off

2.2.14 MaxStaNum

Description: Configure Maximun numbder of station that could connect with this SoftAP
Value:

MaxStaNum=0

0: disable
1~32

2.2.15 TxAntenna

Description: Configure Tx antenna number
Value:

TxAntenna=1

1: 1Tx1R
2: 2Tx2R
3: 3Tx3R

2.2.16 RxAntenna

Description: Configure Rx antenna number
Value:

RxAntenna=1

1: 1Tx1R
2: 2Tx2R
3: 3Tx3R

2.2.17 TxPreamble

Description: Enable or disable Tx preamble
Value:

TxPreamble=0

0: disable
1: enable

2.2.18 RTSThreshold

Description: Set RTS Threshold
Value:

RTSThreshold=2347

1~2347

2.2.19 FragThreshold

Description: Set Fragment threshold
Value:

FragThreshold=2346

256~2346

2.2.20 TxBurst

Description: Enable or disable Tx burst
Value:

TxBurst=1

0: disable
1: enable

2.2.21 PktAggregate

Description: Enable or disable Tx Aggregate
Value:

PktAggregate=0

0: disable
1: enable

2.2.22 NoForwarding

Description: enable or disable No forwarding STA packet within the same BSSID
Value:

NoForwarding=0

0: disable
1: enable

2.2.23 NoForwardingBTNBSSID

Description: enable or disable No Forwarding between each BSSID interface.
Value:

NoForwardingBTNBSSID=0

0: disable
1: enable

2.2.24 NoForwardingMBCast

Description: enable or disable No Forwarding multicast/broadcast packets between the same BSSID interface.
Value:

NoForwardingMBCast=0

0: disable
1: enable

2.2.25 HideSSID

Description: enable or disable Hidden SSID support
Value:

HideSSID=0

0: disable
1: enable

2.2.26 StationKeepAlive

Description: enable or disable Auto-detect the alive status of the station periodically
Value:

StationKeepAlive=0

0: disable
1~65535 seconds

2.2.27 ShortSlot

Description: enable or disable short slot time
Value:

ShortSlot=1

0: disable
1: enable

2.2.28 AutoChannelSelect

Description: Enable or disable Auto Channel Selection support
Value:

AutoChannelSelect=0

0: disable
1: Old Channel Selection Algorithm
2: New Channel Selection Algorithm

2.2.29 AutoChannelSkipList

Description: Configure channels you want to skip when Auto Channel Selection function is enabled
Value:

AutoChannelSkipList=<channel_list>

Example:

<channel_list>=2;3;4;5;7;8;10;

2.2.30 IEEE80211H

Description: enable or disable IEEE 802.11H support (DFS)
Value:

IEEE80211H=0

0: disable
1: enable

2.2.31 CSPeriod

Description: Set how many beacons with Channel Switch Announcement Element will be sent before changing a new channel.
Value:

CSPeriod=10

0 ~ 255. The default is 10.

Note: Channel switch period (Beacon count), unit is based on Beacon interval.

2.2.32 WirelessEvent

Description: enable or disable sending wireless event to the system log (Linux only)
Value:

WirelessEvent=0

0: disable
1: enable

2.2.33 IdsEnable

Description: enable or disable intrusion detection system
Value:

IdsEnable=0

0: disable
1: enable

2.2.34 AuthFloodThreshold

Description: enable or disable Authentication frame flood threshold
Value:

AuthFloodThreshold=32

0: disable
1~65535. (default=32)

2.2.35 ReassocReqFloodThreshold

Description: enable or disable Reassocation request frame flood threshold
Value:

ReassocReqFloodThreshold=32

0: disable
1~65535. (default=32)

2.2.36 ProbeReqFloodThreshold=32

Description: enable or disable Probe request frame flood threshold
Value:

ProbeReqFloodThreshold=32

0: disable
1~65535. (default=32)

2.2.37 DisassocFloodThreshold

Description: enable or disable disassocation frame flood threshold
Value:

DisassocFloodThreshold=32

0: disable
1~65535. (default=32)

2.2.38 DeauthFloodThreshold

Description: enable or disable deauthentication frame flood threshold
Value:

DeauthFloodThreshold=32

0: disable
1~65535. (default=32)

2.2.39 EapReqFooldThreshold

Description: enable or disable EAP request frame flood threshold
Value:

EapReqFooldThreshold=32

0: disable
1~65535. (default=32)

2.2.40 AccessPolicy0

Description: Set the access policy of ACL table 0.
Value:

AccessPolicy0=0

0: Disable this function
1: Allow all entries of ACL table to associate AP
2: Reject all entries of ACL table to associate AP

2.2.41 AccessControlList0

Description: Set the entry's MAC address into ACL table 0.
Value:

AccessControlList0=

[Mac Address];[Mac Address];...

Example:
00:10:20:30:40:50;0A:0b:0c:0D:0e:0f;1a:2b:3c:4d:5e:6f

Note: ACL for Bssid0, max=64

2.2.42 AccessPolicy1

Description: Set the access policy of ACL table 1.
Value:

AccessPolicy1=0

0: Disable this function
1: Allow all entries of ACL table to associate AP
2: Reject all entries of ACL table to associate AP

2.2.43 AccessControlList1

Description: Set the entry's MAC address into ACL table 1.
Value:

AccessControlList1=

[Mac Address];[Mac Address];...

Example:
00:10:20:30:40:50;0A:0b:0c:0D:0e:0f;1a:2b:3c:4d:5e:6f

Note: ACL for Bssid0, max=64

2.2.44 AccessPolicy2

Description: Set the access policy of ACL table 2.
Value:

AccessPolicy2=0

0: Disable this function
1: Allow all entries of ACL table to associate AP
2: Reject all entries of ACL table to associate AP

2.2.45 AccessControlList2

Description: Set the entry's MAC address into ACL table2.
Value:

AccessControlList2=

[Mac Address];[Mac Address];...

Example:
00:10:20:30:40:50;0A:0b:0c:0D:0e:0f;1a:2b:3c:4d:5e:6f

Note: ACL for Bssid0, max=64

2.2.46 AccessPolicy3

Description: Set the access policy of ACL table 3.
Value:

AccessPolicy3=0

0: Disable this function
1: Allow all entries of ACL table to associate AP
2: Reject all entries of ACL table to associate AP

2.2.47 AccessControlList3

Description: Set the entry's MAC address into ACL table 3.
Value:

AccessControlList3=

[Mac Address];[Mac Address];...

Example:
00:10:20:30:40:50;0a:0b:0c:0d:0e:0f;1a:2b:3c:4d:5e:6f

Note: ACL for Bssid0, max=64

2.2.48 RADIUS_Server

Description: Configure radius server IP address
Value:

RADIUS_Server=

IP address.
Example: RADIUS_Server=192.168.2.3

2.2.49 RADIUS_Port

Description: Configure radius server port number
Value:

RADIUS_Port=1812
Deafult: 1812

2.2.50 RADIUS_Key

Description: Configure radius key string
Value:

RADIUS_Key=

Example:

RADIUS_Key=ralink

2.2.51 own_ip_addr

Descripion: Configure SoftAP itself IP Address
Value:

own_ip_addr=

Example:

own_ip_addr=192.168.1.1

2.2.52 EAPifname

Description: EAPifname is assigned as the binding interface for EAP negotiation
Value:

EAPifname=

Example:

EAPifname=br0

2.2.53 PreAuthifname

Description: PreAuthifname is assigned as the binding interface for WPA2 Pre-authentication
Value:

PreAuthifname=

Example:

PreAuthifname=br0

2.2.54 HT_HTC

Description: enable or disable Support the HT control field
Value:

HT_HTC=0

0: disable
1: enable

Note: HTC Control field (4-octet) is following QoS field. An MPDU that contains the HT control field is referred to as a +HTC frame.

2.2.55 HT_RDG

Description: Enable or disable HT Reverse Direction Grant
Value:

HT_RDG=1

0: disable
1: enable

2.2.56 HT_EXTCHA

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

HT_EXTCHA=0

0: Below
1: Above

2.2.57 HT_LinkAdapt

Decription: enable ir disable HT Link Adaptation Control
Value:

HT_LinkAdapt=0

0: disable
1: enable

2.2.58 HT_OpMode

Description: HT operation mode
Value:

HT_OpMode=0

0: HT mixed mode
1: HT Greenfield mode

2.2.59 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

2.2.60 HT_BW

Description: HT channel bandwidth configuration
Value:

HT_BW=1

0: 20 MHz
1: 20/40 MHz

2.2.61 HT_PROTECT

Description: Enable or disable 802.11n protection mechanism
Value:

HT_PROTECT=1

0: Disable
1: Enable

2.2.62 HT_BSSCoexistence

Description: Enable or disable HT BSS coexistence support
Value:

HT_BSSCoexistence=1

0: Disable
1: Enable

2.2.63 HT_TxStream

Description: Set the number of spatial streams for transmission.
Value:

HT_TxStream=1/2/3

1~3: valid spatial streams

2.2.64 HT_RxStream

Description: Set the number of spatial streams for reception.
Value:

HT_RxStream=1/2/3

1~3: valid spatial streams

2.2.65 HT_BADecline

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

HT_BADecline=0

0: disable
1: enable

2.2.66 HT_AutoBA

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

HT_AutoBA=1

0: disable
1: enable

2.2.67 HT_AMSDU

Description: Enable or disable AMSDU section
Value:

HT_AMSDU=0

0: disable
1: enable

2.2.68 HT_BAWinSize

Description: Block Ack window size
Value:

HT_BAWinSize=64

1~64

2.2.69 HT_GI

Description: HT Guard interval support
Value:

HT_GI=1

0: Long guard interval
1: short guard interval

2.2.70 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

2.2.71 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)

2.2.72 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

2.2.73 HT_STBC

Description: Enable or disable HT STBC support
Value:

HT_STBC=0

0: disable
1: enable

2.2.74 VHT_BW

Description: Enable or disable 11ac 80MHz bandwidth
Value:

VHT_BW=1

0: disable
1: enable

Note: 11AC chipset only

2.2.75 VHT_STBC

Description: Enable or disable 11ac STBC
Value:

VHT_STBC=1

0: disable
1: enable

Note: 11AC chipset only

2.2.76 VHT_BW_SIGNAL

Description: Enable or disable 11ac bandwidth signaling
Value:

VHT_BW_SIGNAL=1

0: disable
1: enable

Note: 11AC chipset only

2.2.77 VHT_LDPC

Description: Enable or disable LDPC on received packets with 11ac MCS
Value:

VHT_LDPC=1

0: disable
1: enable

Note: 11AC chipset only

2.2.78 VHT_DisallowNonVHT

Description: Enable or disable the function of rejecting connection attempt from non-VHT STA
Value:

VHT_DisallowNonVHT=1

0: disable
1: enable

Note: 11AC chipset only

2.2.79 WscManufacturer

Description: WPS manufacturer string
Value:

WscManufacturer=

Less than 64 characters

2.2.80 WscModelName

Description: WPS Mode name string
Value:

WscModelName=

Less than 32 characters

2.2.81 WscDeviceName

Description: WPS Device name string
Value:

WscDeviceName=

Less than 32 characters

2.2.82 WscModelNumber

Description: WPS Device model number string
Value:

WscModelNumber=

Less than 32 characters

2.2.83 WscSerialNumber

Description: WPS serial number string
Value:

WscSerialNumber=

Less than 32 characters

2.2.84 Wsc4digitPinCode

Description: WPS 4 digit pin code string
Value:

Wsc4digitPinCode=0

4 digit

2.2.85 VLANID

Description: set VLAN ID
Value:

VLANID=0

0: Disable

2.2.86 VLANPriority

Description: set VLAN Priority
Value:

VLANPriority=0

0: Disable

2.2.87 E2pAccessMode

Description: Select the EEPROM access mode from interface start-up
Value:

E2pAccessMode=2

0: NONE
1: EFUSE mode
2: FLASH mode
3: EEPROM mode
4: BIN FILE mode

2.2.88 EntryLifeCheck

Description: Set how many continued TX failure packets per STA can be ignored. Over the value, AP will tear down this STA, because it shall be gone.
Value:

EntryLifeCheck=20

Example:
EntryLifeCheck=1 ~ 65535. Default is 20.

2.2.89 EtherTrafficBand

Description: To bind enthernet packets with specific RF band
Value:

EtherTrafficBand=2G

2G: Bind enthernet packets with 2.4GHz RF Band
5G: Bind enthernet packets with 5GHz RF Band

Note: only available after SoftAP driver v3.0.1.2. or after version

2.3 WAPI Specific

2.3.1 Wapiifname

Description: Assign an interface name to process the WAI frame. The WAPID daemon shall be bound on this interface. If it doesn't specify, the default interface is "br0".
Value:

br0: default binding interface

2.3.2 WapiAsCertPath

Description: Assign the path of the AS certificate for the WAPI certificate authentication.
Value:

WapiAsCertPath=/etc/as.cer

2.3.3 WapiAsIpAddr

Description: Assign the IP address of the AS for the WAPI certificate authentication.
Value:

WapiAsIpAddr=192.168.222.174

2.3.4 WapiAsPort

Description: Assign the port number of the AS for the WAPI certificate authentication.
Value:

WapiAsPort=3810

2.3.5 WapiMskRekeyMethod

Description: Set the method for WAPI group key renew mechanism
Value:

DISABLE : Disable the rekey mechanism
TIME : time-based
PKT : packet-based

2.3.6 WapiMskRekeyThreshold

Description: Set the period of WAPI group key updating
Value:

0 : Disable this mechanism
10 ~ 0x3ffffff, Default is 3600.

2.3.7 WapiPsk1

Description: Set the WAPI pre-shared key
Value:

8~64 characters

2.3.8 WapiPskType

Description: Set the WAPI key type
Value:

0: HEX mode
1: ASCII mode

2.3.9 WapiUserCertPath

Description: Assign the path of the user certificate for the WAPI certificate authentication
Value:

WapiUserCertPath=/etc/user.cer

2.3.10 WapiUskRekeyMethod

Description: Set the method for WAPI unicast key renew mechanism
Value:

DISABLE : Disable the rekey mechanism
TIME : time-based
PKT : packet-based

2.3.11 WapiUskRekeyThreshold

Description: Set the period of WAPI unicast key updating
Value:

0 : Disable this mechanism
10 ~ 0x3ffffff, Default is 3600

2.4 iNIC Specific

2.4.1 Ext_LNA

Description: support External or internal LNA
Value:

Ext_LNA

0: Internal LNA
1: External LNA

Note: MT7620 iNIC driver only profile

2.4.2 Ext_PA

Description: support External or internal PA
Value:

Ext_PA

0: Internal PA
1: External PA

Note: MT7620 iNIC driver only profile

2.4.3 ExtEEPROM

Description: Support driver to read EEPROM from an external file
Value:

ExtEEPROM=1

0: read EERPM data from EEPROM chip
1: read EEPROM data from an external file

Note: The external EEPROM file must be exactly the same format as EEPROM format.
iNIC driver only profile.

2.4.4 Mem

Description: Support WLAN profile can configure iNIC system address value
Value:

Mem=addr1,value1;addr2,value2;

Example:
Mem=b0110014,ff7f5555;b011008c,2404040;
iNIC firmware will Set

1. memory address (0xb0110014) value (0xff7f5555);
2. memory address (0xb011008c) value (0x2404040);

Note: This parameter is only for iNIC driver.

2.4.5 DetectPhy

Description: Disable/Enable iNIC Phy link detection. if Phy link down will reset iNIC to load firmware.
Value:

DetectPhy=0

0: disable
1: enable

Note: only available on iNIC MT76XX FW v2.7.0.8 and after.

2.4.6 Thermal

Description: Disable/Enable iNIC thermal function
Value:

Thermal=0

0: disable
1: enable

Note:
Thermal function will be according to criteria with current temperature to configure Ant.
Criteria Value: 1~1000
default:80

Example:
iwpriv ra0 set tpc =80
Only available on iNIC MT76XX FW v2.7.0.8 and after.

2.4.7 %s_DfsSwAddCheck%d

Description: WLAN profile parameter to check DFS false alarm.
The first string is RDRegion. RDRegion string can be "CE", "FCC", "JAP", "JAP_W53", "JAP_W56".
The second integer is channel index. Channel index can be from 0 to 4.

Value:
There are four parameter (Period low, Period High, Width low, Width high) in one rule. Multiple rules can be used. At least one rule must be used. Each parameter is separated by semicolon. T_Low;T_High;W_Low;W_High

For example:
CE_DfsSwAddCheck0=100;200;50;500
FCC_DfsSwAddCheck0=100;200;50;500;70;700;30;300

Note: only available on iNIC MT76XX FW v2.7.0.8 and after.

2.4.8 IsolateCard

Description: Disable/Enable for iNIC isolate concurrent card traffic.
Value:
IsolateCard=0

0: disable
1: enable (iNIC concurrent card traffic can't forward to each other)

Note: only available on iNIC MT76XX FW v2.7.0.9 and after.

2.4.9 EnhanceMultiClient

Description: Disable/Enable multiple N client related configuration.
Value:
EnhanceMultiClient=0

0: disable
1: enable

Note: only available on iNIC MT76XX FW v2.7.0.9 and after.

2.4.10 BGMultiClient

Description: Disable/Enable multiple legacy client related configuration.
Value:

BGMultiClient=0

0: disable
1: enable

Note: only available on iNIC MT76XX FW v2.7.0.9 and after.

2.4.11 RssiDisauth

Description: Disable or Enable RSSI disassociate feature..
Value:

RssiDisauth=0

0: disable
1: enable

Default : 0 (disable);

If Enable RSSI disassociate feature.
Two scenarios for this feature:
(1.) STA was exceeded the RssiThreshold value. AP will disassociate STA.
(2.) Periodically Checking:
After client was associated. AP will check RSSI periodically base on PollingRssiInterval. If STA was exceeding the RssiThreshold. A counter will be increase. STA will be disassociated when STA's own counter was exceeded TimeExceedRssiThreshold. The counter will be reset if AP found STA didn't exceed the RSSI threshold.

Note: only available on iNIC MT76XX FW v2.7.1.0 and after.

2.4.12 RssiThreshold

Description: Minimum RSSI disassociate threshold.
Value:

RssiThreshold=0

Default : 0 (disable);
value : -100 ~ -1

Note: only available on iNIC MT76XX FW v2.7.1.0 and after.

2.4.13 PollingRssiInterval

Description: Polling time interval for check STA RSSI(in second).
Value:

PollingRssiInterval=0

Default : 0 (disable);
value : 1 ~ 3600

Note: only available on iNIC MT76XX FW v2.7.1.0 and after.

2.4.14 TimeExceedRssiThreshold

Description: Time of user exceed the RSSI threshold before disassociate
Value:

TimeExceedRssiThreshold=0

Default : 0 (disable);
value : 1 ~ 10000

Note: only available on iNIC MT76XX FW v2.7.1.0 and after.

2.4.15 SiteSurveyRssi

Description: Disable or Enable get RSSI for each Site Survey APT
Value:

SiteSurveyRssi=0

Default : 0
value : 0/1

Note: only available on iNIC MT76XX FW v2.7.1.0 and after.

2.4.16 AssociationInfoEvent

Description: Disable or Enable association send event include wireless mode/PHY rate/RSSI
Value:

AssociationInfoEvent=0

Default : 0
value : 0/1

Note: only available on iNIC MT76XX FW v2.7.1.0 and after.

2.4.17 EDCCA

Description: Disable or Enable EDCCA function
Value:

EDCCA=0

Default : 0
value : 0/1

Note: only available on iNIC MT76XX FW v2.7.1.0 and after.

2.4.18 TX_RETRY_NUM

Description: Tx retry number
Value:

TX_RETRY_NUM=3

Default:0

Note: only available on iNIC MT76XX FW v3.0.0.2 and after.

2.4.19 RTS_RETRY_NUM

Description: RTC retry number
Value:

RTS_RETRY_NUM=3

Default=0

Note: only available on iNIC MT76XX FW v3.0.0.2 and after.

2.4.20 EDCCA_AP_STA_TH

Description: STA count on SoftAP
Value:

EDCCA_AP_STA_TH=1

Default:1

Note: only available on iNIC MT76XX FW v3.0.0.2 and after.

2.4.21 EDCCA_AP_AP_TH

Description: SoftAP count on the same working channel
Value:

EDCCA_AP_AP_TH=1

Default:1

Note: only available on iNIC MT76XX FW v3.0.0.2 and after.

2.4.22 EDCCA_AP_RSSI_TH

Description: SoftAP count threshold on the same working channel, only when SofAP RSSI is greater than the configured level.
Value:

```
EDCCA_AP_RSSI_TH=-80
```

Note: only available on iNIC MT76XX FW v3.0.0.2 and after.

3 WLAN SoftAP Driver iwpriv set command

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

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

3.1.1 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

3.1.2 DriverVersion

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

iwpriv ra0 set DriverVersion=0

Any value

3.1.3 CountryRegion

Description: Country region for WLAN radio 2.4 GHz regulation (G band)
Value:

iwpriv ra0 set 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

3.1.4 CountryRegionABand

Description: Country region for WLAN radio 5 GHz regulation (A band)
Value:

iwpriv rai0 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

3.1.5 CountryCode

Description: County code for WLAN radio regulation
Value:

iwpriv ra0 set CountryCode=TW

Note:
2 characters, like TW for Taiwan.
Please refer to the following link for ISO3166 code list for other countries.
http://www.iso.org/iso/prods-services/iso3166ma/02iso-3166-code-lists/country_names_and_code_elements

3.1.6 AccessPolicy

Description: Configure access policy of ACL table
Value:

iwpriv ra0 set AccessPolicy=0

0: Disable this function
1: Allow all entries of ACL table to associate AP
2: Reject all entries of ACL table to associate AP

3.1.7 ResetCounter

Description:Reset all statistic counter
Value:

iwpriv ra0 set ResetCounter=1

3.1.8 SiteSurvey

Description: Make a site survey request to the driver
Value:

iwpriv ra0 set SiteSurvey=

Note:
Passive scan: Use empty string as argument, like “iwpriv ra0 set SiteSurvey=”
Active scan: Use legal SSID as argument, like “iwpriv ra0 set SiteSurvey=Target_SSID”

3.1.9 CountryString

Description: configure country string
Value:

iwpriv ra0 set CountryString=TAIWAN

32 characters, ex:Taiwan, case insensitive

Note: Please refer to ISO3166 code list for other countries and can be found at
http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html#sz

Item	Country Number	ISO Name	Country Name (CountryString)	Support 802.11A	802.11A Country Region	Support 802.11G	802.11G Country Region
	0	DB	Debug	Yes	A_BAND_REGION_7	Yes	G_BAND_REGION_5
	8	AL	ALBANIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	12	DZ	ALGERIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	32	AR	ARGENTINA	Yes	A_BAND_REGION_3	Yes	G_BAND_REGION_1
	51	AM	ARMENIA	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	36	AU	AUSTRALIA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	40	AT	AUSTRIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	31	AZ	AZERBAIJAN	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	48	BH	BAHRAIN	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	112	BY	BELARUS	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	56	BE	BELGIUM	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	84	BZ	BELIZE	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	68	BO	BOLIVIA	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	76	BR	BRAZIL	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	96	BN	BRUNEI DARUSSALAM	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	100	BG	BULGARIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	124	CA	CANADA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	152	CL	CHILE	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	156	CN	CHINA	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	170	CO	COLOMBIA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	188	CR	COSTA RICA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	191	HR	CROATIA	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	196	CY	CYPRUS	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	203	CZ	CZECH REPUBLIC	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	208	DK	DENMARK	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	214	DO	DOMINICAN REPUBLIC	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	218	EC	ECUADOR	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	818	EG	EGYPT	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	222	SV	EL SALVADOR	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	233	EE	ESTONIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	246	FI	FINLAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	250	FR	FRANCE	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	268	GE	GEORGIA	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	276	DE	GERMANY	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1

	300	GR	GREECE	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	320	GT	GUATEMALA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	340	HN	HONDURAS	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	344	HK	HONG KONG	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	348	HU	HUNGARY	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	352	IS	ICELAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	356	IN	INDIA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	360	ID	INDONESIA	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	364	IR	IRAN	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	372	IE	IRELAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	376	IL	ISRAEL	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	380	IT	ITALY	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	392	JP	JAPAN	Yes	A_BAND_REGION_9	Yes	G_BAND_REGION_1
	400	JO	JORDAN	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	398	KZ	KAZAKHSTAN	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	408	KP	KOREA DEMOCRATIC	Yes	A_BAND_REGION_5	Yes	G_BAND_REGION_1
	410	KR	KOREA REPUBLIC OF	Yes	A_BAND_REGION_5	Yes	G_BAND_REGION_1
	414	KW	KUWAIT	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	428	LV	LATVIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	422	LB	LEBANON	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	438	LI	LIECHTENSTEIN	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	440	LT	LITHUANIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	442	LU	LUXEMBOURG	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	446	MO	MACAU	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	807	MK	MACEDONIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	458	MY	MALAYSIA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	484	MX	MEXICO	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	492	MC	MONACO	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	504	MA	MOROCCO	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	528	NL	NETHERLANDS	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	554	NZ	NEW ZEALAND	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	578	NO	NORWAY	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	512	OM	OMAN	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	586	PK	PAKISTAN	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	591	PA	PANAMA	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	604	PE	PERU	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	608	PH	PHILIPPINES	Yes	A_BAND_REGION_4	Yes	G_BAND_REGION_1
	616	PL	POLAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	620	PT	PORTUGAL	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	630	PR	PUERTO RICO	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	634	QA	QATAR	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	642	RO	ROMANIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	643	RU	RUSSIA FEDERATION	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	682	SA	SAUDI ARABIA	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	702	SG	SINGAPORE	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	703	SK	SLOVAKIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	705	SI	SLOVENIA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	710	ZA	SOUTH AFRICA	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	724	ES	SPAIN	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	752	SE	SWEDEN	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	756	CH	SWITZERLAND	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	760	SY	SYRIAN ARAB REPUBLIC	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	158	TW	TAIWAN	Yes	A_BAND_REGION_3	Yes	G_BAND_REGION_0
	764	TH	THAILAND	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	780	TT	TRINIDAD AND TOBAGO	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	788	TN	TUNISIA	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	792	TR	TURKEY	Yes	A_BAND_REGION_2	Yes	G_BAND_REGION_1
	804	UA	UKRAINE	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	784	AE	UNITED ARAB EMIRATES	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1

	826	GB	UNITED KINGDOM	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_1
	840	US	UNITED STATES	Yes	A_BAND_REGION_0	Yes	G_BAND_REGION_0
	858	UY	URUGUAY	Yes	A_BAND_REGION_5	Yes	G_BAND_REGION_1
	860	UZ	UZBEKISTAN	Yes	A_BAND_REGION_1	Yes	G_BAND_REGION_0
	862	VE	VENEZUELA	Yes	A_BAND_REGION_5	Yes	G_BAND_REGION_1
	704	VN	VIET NAM	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	887	YE	YEMEN	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1
	716	ZW	ZIMBABWE	No	A_BAND_REGION_0	Yes	G_BAND_REGION_1

3.1.10 SSID

Description: Set AP SSID

Value:

iwpriv ra0 set SSID=11n-AP

0~z, 1~32 ASCII characters

3.1.11 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
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)
15: 11 AN/AC mixed 5G band only (Only 11AC chipset support)

3.1.12 FixedTxMode

Description: Fix Tx mode to CCK or OFDM for MCS rate selection

Value:

iwpriv ra0 set FixedTxMode=CCK

CCK
OFDM

3.1.13 BasicRate

Description: configure basic rate

Value:

iwpriv ra0 set BasicRate=

0~4095

Basic Rate Bit Map (max. 12-bit, represent max. 12 basic rates)												
Bit	11	10	9	8	7	6	5	4	3	2	1	0
Rate	54	48	36	24	18	12	9	6	11	5.5	2	1
Set	0	1	0	1	0	1	0	1	1	1	1	1
Hex	5				5				F			
Decimal	1375											

Note: Be careful to set this value, if you don't know what this is, please don't set this field.

3.1.14 Channel

Description: Configure wireless channel

Value:

iwpriv ra0 set Channel=6

802.11b/g: 1 ~ 14 (it must agree with the CountryRegion setting)

802.11a: 36~165 (it must agree with the CountryRegionABand setting)

3.1.15 BeaconPeriod

Description: configure Beacon period

Value:

iwpriv ra0 set BeaconPeriod=100

20 ~ 1024 (unit is in milli-seconds)

3.1.16 DtimPeriod

Description: Configure DTIM period

Value:

iwpriv ra0 set DtimPeriod=1

1~5

3.1.17 TxPower

Description: Set Transmit Power by percentage

Value:

iwpriv ra0 set TxPower=100

0~100

Note:

91 ~ 100% & AUTO, treat as 100% in terms of mW

61 ~ 90%, treat as 75% in terms of mW	-1dBm
31 ~ 60%, treat as 50% in terms of mW	-3dBm
16 ~ 30%, treat as 25% in terms of mW	-6dBm
10 ~ 15%, treat as 12.5% in terms of mW	-9dBm
0 ~ 9 %, treat as MIN(~3%) in terms of mW	-12dBm

3.1.18 BGProtection

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

iwpriv ra0 set BGProtection=0

0: disable
1: Always on
2:Always off

3.1.19 DisableOLBC

Description: enable or disable OLBC
Value:

iwpriv ra0 set DisableOLBC=0

0: disable
1: enable

3.1.20 TxPreamble

Description: enable or disable Tx preamble
Value:

iwpriv ra0 set TxPreamble=1

0: disable
1: enable

3.1.21 RTSThreshold

Description: Set RTS Threshold
Value:

iwpriv ra0 set RTSThreshold=2347

1~2347

3.1.22 FragThreshold

Description: Set Fragment threshold
Value:

iwpriv ra0 set FragThreshold=2346

256~2346

3.1.23 TxBurst

Description: enable or disable Tx burst mode
Value:

iwpriv ra0 set TxBurst=0

0: disable
1: enable

3.1.24 PktAggregate

Description: enable or disable packet aggregation (Ralink to Ralink only)
Value:

iwpriv ra0 set PktAggregate=1

0: disable
1: enable

3.1.25 NoForwarding

Description: enable or disable no forwarding packet between STAs in the same BSSID
Value:

iwpriv ra0 set NoForwarding=0

0: disable
1: enable

3.1.26 NoForwardingBTNBSSID

Description: enable or disable No Forwarding between each BSSID interface.
Value:

iwpriv ra0 set NoForwardingBTNBSSID=1

0: disable
1: enable

3.1.27 NoForwardingMBCast

Description: enable or disable No Forwarding multicast/broadcast packets between each BSSID interface.
Value:

iwpriv ra0 set NoForwardingMBCast=1

0: disable

1: enable

3.1.28 HideSSID

Description: enable or disable hidden SSID
Value:

iwpriv ra0 set HideSSID=1

0: disable
1: enable

3.1.29 ShortSlot

Description: enable or disabllle short slot time
Value:

iwpriv ra0 set ShortSlot=0

0: disable
1: enable

3.1.30 DisConnectSta

Description: Disconnect one specific STA which connected with this SoftAP manually
Value:

iwpriv ra0 set DisConnectSta=00:11:22:33:44:55

[MAC address]

3.1.31 DisConnectAllSta

Description: Disconnect all STAs which connected with this SoftAP manually.
Value:

iwpriv ra0 set DisConnectAllSta=1

1: disconnect all STAs

3.1.32 McastPhyMode

Description: Configure multicast physical mode
Value:

iwpriv ra0 set McastPhyMode=0

0: Disable
1: CCK
2: OFDM
3 HTMIX

3.1.33 McastMcs

Description: Specify the MCS of multicast packets.
Value:

iwpriv ra0 set McastMcs=0

0~15

3.1.34 WscVendorPinCode

Description: Set vendor pin code as pin code of WPS AP's enrollee
Value:

iwpriv ra0 WscVendorPinCode=xxxxxxx

xxxxxxxx //Valid PIN code

3.1.35 ACLAddEntry

Description: To insert one or several MAC addresses into Access control MAC table list, up to 64 MAC address at one time.
Value:

iwpriv ra0 set ACLAddEntry="xx:xx:xx:xx:xx:xx"

[MAC address];[MAC address];...;[MAC address]"
Example:

iwpriv ra0 set ACLAddEntry="00:0c:43:28:aa:12;00:0c:43:28:aa:11;00:0c:43:28:aa:10"

3.1.36 ACLClearAll

Description: To clear all the MAC address entries in an Access control MAC table list.
Value:

iwpriv ra0 set ACLClearAll=1

1: indicate to clear the table
Other value is invalid.

3.1.37 MaxStaNum

Description: To limit the maximum number of associated clients per BSS.
Value:

iwpriv ra0 set MaxStaNum=0

0: disable this function
1~32 (default:32)

3.1.38 AutoFallBack

Description: enable or disable auto fall back rate control function
Value:

iwpriv ra0 set AutoFallBack=1

0: disable
1: enable

3.1.39 GreenAP

Description: enable or disable Green AP fucntion
Value:

iwpriv ra0 set GreenAP=0

0: disable
1: enable

3.1.40 AutoChannelSel

Description: auto channel select when driver is loaded
Value:

iwpriv ra0 set AutoChannelSel=2

0: Disable
1: Old Channel Selection Algorithm
2: New Channel Selection Algorithm

3.1.41 ACSCheckTime

Description: Set a periodic check time for auto channel selection (unit: hour)
Value:

iwpriv ra0 set ACSCheckTime=3

0: Disable

3.1.42 MBSSWirelessMode

Description: Set MBSS Wireless phy Mode. Only support in v2.5.0.0 and after version.
Value:

0: 802.11 B/G mixed
1: 802.11 B only
2: 802.11 A only
4: 802.11 G only
6: 802.11 N only
7: 802.11 G/N mixed
8: 802.11 A/N mixed
9: 802.11 B/G/N mixed
10: 802.11 A/G/N mixed
11: 802.11 N in 5G band only

Example:

ra0: B/G/N fixed
ra1: B only
ra2: B/G mixed
ra3: G only
Must set main BSS (ra0) first then set other MBSS WirelessMode. Can't have A & B mode fixed in MBSS.

iwpriv ra0 set WirelessMode=9
iwpriv ra1 set MBSSWirelessMode=1
iwpriv ra2 set MBSSWirelessMode=0
iwpriv ra3 set MBSSWirelessMode=4

3.1.43 HwAntDiv

Description: enable or disable Hardware antenna diversity
Value:

iwpriv ra0 set HwAntDiv=0

0: disable
1: enable

Note: RT5350 only

3.1.44 HtBw

Description: HT channel bandwidth configuration
Value:

iwpriv ra0 set HtBw=1

0: 20 MHz
1: 20/40 MHz

3.1.45 VhtBw

Description: Enable or disable 11AC 80MHz Bandwidth support
Value:

iwpriv ra0 set VhtBw=1

0: disable
1: enable

Note: 11AC chipset only

3.1.46 VhtStbc

Description: Enable/disable 11AC STBC Support
Value:

iwpriv ra0 set VhtStbc=1

0: disable
1: enable

Note: 11AC chipset only

3.1.47 VhtBwSignal

Description: Enable/disable 11 AC BandWidth signaling
Value:

iwpriv ra0 set VhtBwSignal=1

0: disable
1: enable

Note: 11AC chipset only.

3.1.48 VhtDisallowNonVHT

Description: Enable/disable to reject non-VHT STA to connect
Value:

iwpriv ra0 set VhtDisallowNonVHT=1

0: disable
1: enable to reject non-VHT STA

Note: 11AC chipset only.

3.1.49 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

HT Mixed Mode, Refer to IEEE P802.11n Figure n67 HT Greenfield, Refer to IEEE P802.11n Figure n68	
MCS = 0 (1S)	(BW=0, SGI=0) 6.5Mbps
MCS = 1	(BW=0, SGI=0) 13Mbps
MCS = 2	(BW=0, SGI=0) 19.5Mbps
MCS = 3	(BW=0, SGI=0) 26Mbps
MCS = 4	(BW=0, SGI=0) 39Mbps
MCS = 5	(BW=0, SGI=0) 52Mbps
MCS = 6	(BW=0, SGI=0) 58.5Mbps
MCS = 7	(BW=0, SGI=0) 65Mbps
MCS = 8 (2S)	(BW=0, SGI=0) 13Mbps
MCS = 9	(BW=0, SGI=0) 26Mbps
MCS = 10	(BW=0, SGI=0) 39Mbps
MCS = 11	(BW=0, SGI=0) 52Mbps

MCS = 12	(BW=0, SGI=0) 78Mbps
MCS = 13	(BW=0, SGI=0) 104Mbps
MCS = 14	(BW=0, SGI=0) 117Mbps
MCS = 15	(BW=0, SGI=0) 130Mbps
MCS = 32	(BW=1, SGI=0) HT duplicate 6Mbps
Notes: When BW=1, PHY_RATE = PHY_RATE * 2 When SGI=1, PHY_RATE = PHY_RATE * 10/9 The effects of BW and SGI are accumulative. When MCS=0~7(1S, One Tx Stream), SGI option is supported. BW option is supported. When MCS=8~15(2S, Two Tx Stream), SGI option is supported. BW option is supported. When MCS=32, only SGI option is supported. BW option is not supported. (BW =1) Other MCS code in HT mode are reserved.	

3.1.50 HtGi

Description: Set WLAN Guard interval support
Value:

iwpriv ra0 set HtGi=1

0: long guard interval
1: short guard interval

3.1.51 HtOpMode

Description: HT operation Mode
Value:

iwpriv ra0 set HtOpMode=0

0: HT mixed mode
1: HT Greenfield mode

3.1.52 HtStbc

Description: Enable or disable HT STBC
Value:

iwpriv ra0 set HtStbc=1

0: disable
1: enable

3.1.53 HtExtcha

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

iwpriv ra0 set HtExtcha=0

0: below
1: Above

3.1.54 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

3.1.55 HtBaWinSize

Description: Block Ack window size
Value:

iwpriv ra0 set HtBaWinSize=64

1~64

3.1.56 HtTxBASize

Description: Set the number of AMPDU aggregation size of one transmission burst.
Value:

iwpriv ra0 set HtTxBASize=64

1~64: valid value

3.1.57 HtRdg

Description: Enable or disable HT Reverse Direction Grant
Value:

iwpriv ra0 set HtRdg=1

0: disable
1: enable

3.1.58 HtAmsdu

Description: Enable or disable AMSDU section
Value:

Iwpriv ra0 set HtAmsdu=0

0: disable
1: enable

3.1.59 HtAutoBa

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

iwpriv ra0 set HtAutoBa=1

0: disable
1: enable

3.1.60 BADecline

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

iwpriv ra0 set BADecline=0

0: disable
1: enable

3.1.61 HtProtect

Description: Enable or disable HT protect
Value:

iwpriv ra0 set HtProtect=0

0: disable
1: enable

3.1.62 HtMimoPs

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

iwpriv ra0 set HtMimoPs=0

0: disable
1: enable

3.1.63 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

3.1.64 AP2040Rescan

Description: Trigger HT20/40 coexistence to rescan
Value:

iwpriv ra0 set AP2040Rescan=1

1: trigger to rescan

3.1.65 HtBssCoex

Description: Enable or disable HT BSS coexistence
Value:

iwpriv ra0 set HtBssCoex=0

0: disable
1: enable

3.1.66 HtTxStream

Description: Set the number of spatial streams for transmission
Value:

iwpriv ra0 set HtTxStream=1 or 2 or 3

1~3: valid spatial streams

3.1.67 HtRxStream

Description: Set the number of spatial streams for reception
Value:

iwpriv ra0 set HtRxStream=1 or 2 or 3

1~3: valid spatial streams

3.1.68 BASetup

Description: Add an Originator BA entry into the BA table manually.
Value:

iwpriv ra0 set BASetup=00:0c:43:01:02:03-1

→The six 2 digit hex-decimal number(xx) previous are the Mac address,
→The seventh decimal number(d) is the tid value.

3.1.69 BAOriTearDown

Description: Remove an Originator BA entry from the BA table manually.
Value:

iwpriv ra0 set BAOriTearDown=00:0c:43:01:02:03-1

→The six 2 digit hex-decimal number(xx) previous are the Mac address,
→The seventh decimal number(d) is the tid value.

3.1.70 BARecTearDown

Description: Remove an Recipient BA entry from the BA table manually.
Value:

iwpriv ra0 set BARecTearDown=00:0c:43:01:02:03-1

→The six 2 digit hex-decimal number(xx) previous are the Mac address,
→The seventh decimal number(d) is the tid value.

3.1.71 PktAggregate

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

iwpriv ra0 set PktAggregate=1

0: disable
1: enable

3.1.72 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

3.1.73 KickStaRssiLow

Description: Set the lowest limitation for AP kicking out STA.
Value:

iwpriv ra0 set KickStaRssiLow=0

0: Disable
0 ~ -100

3.1.74 AssocReqRssiThres

Description: Set AssocReq RSSI Threshold to reject STA with weak signal
Value:

Iwpriv ra0 set AssocReqRssiThres=0

0: Disable
0~ -100

3.2 iNIC specific

3.2.1 QAEnable

Description: enable or disable QA test tool function.

iwpriv ra0 set QAEnable=1

0: disable
1: enable

3.2.2 Console

Description: redirect console information to host.

iwpriv ra0 set Console=1

0: disable
1: enable

3.2.3 EfuseUploadToHost

Description: This command is specific to iNIC solution.
The content of efuse will be uploaded to the iNIC host in iNIC_e2p.bin or iNIC_e2p1.bin .

iwpriv ra0 set EfuseUploadToHost=1

0: disable
1: enable

3.2.4 tpc

Description: Thermal function will be according to criteria with current temperature to configure Ant.
Criteria Value:1~1000
default: 80

Iwpriv ra0 set tpc=80

3.2.5 DfsSwAddCheck

Description: This command is used to add an entry to prevent false detection in specific range.
“ch” is the bbp dfs detection engine ID
“T_Low” is the Radar Period low boundary to filter out.
“T_High” is the Radar Period high boundary to filter out.
“W_Low” is the Radar Width low boundary to filter out.
“W_High” is the Radar Width high boundary to filter out.

iwpriv ra0 set DfsSwAddCheck=ch:T_Low:T_High:W_Low:W_high

Example:
iwpriv ra0 set DfsSwAddCheck=0:100:200:50:500

3.2.6 DfsSwDelCheck

Description: This command is used to delete an entry which was added to filter out radar in specific range.

iwpriv ra0 set DfsSwDelCheck=ch:T_Low:T_High:W_Low:W_high

Example:
iwpriv ra0 set DfsSwDelCheck=0:100:200:50:500

4 Other iwpriv Command

4.1 stat

Description: Show WLAN statistics
Value:

iwpriv ra0 stat

Note:
You can use “iwpriv ra0 set ResetCounter=1” to reset statistics
Also, you can use the following command line shell script to get per-second statistics.
while [1]; do iwpriv ra0 set ResetCounter=1; sleep 1; iwpriv ra0 stat; done;

4.2 get_site_survey

Description: Show site survey result
Value:

iwpriv ra0 get_site_survey

Note: You need to use “iwpriv ra0 set SiteSurvey=” to collect information first

4.3 get_mac_table

Description: Show MAC addresses of connected stations
Value:

iwpriv ra0 get_mac_table

4.4 get_ba_table

Description: Show raw data of the BlockAck table
Value:

iwpriv ra0 get_ba_table

4.5 get_wsc_profile

Description: Show WPS profile information
Value:

iwpriv ra0 get_wsc_profile

4.6 e2p

Description: Read/Write EEPROM content
Value:

```
// Read
iwpriv ra0 e2p offset
// Write
iwpriv ra0 e2p offset=value
```

Note:

offset = hexidecimal address
value = hexidecimal value

4.7 show

You could use iwpriv ra0 show command to display general or specific information. As to specific information, you have to turn on the corresponding function in driver config.

[Format]
iwpriv ra0 show [parameter]

[Parameter list]

1. driverinfo - show driver version
2. stat - show statistics counter
3. stainfo - show MAC address of associated STAs
4. stacountinfo - show TRx byte count of associated STAs
5. stasecinfo - show security information of associated STAs
6. bainfo - show BlockAck information
7. connStatus - show AP-Client connection status
8. reptinfo - show MAC Repeater information
9. wdsinfo - show WDS link list
10. igmpinfo - show all entries in the IGMP Snooping Table
11. mbss - show MBSS PHY mode information
12. blockch - show DFS blocked channel list

[Example]

```
# iwpriv ra0 show driverinfo
Driver version: 2.7.1.6
```

5 TBD

To Be Defined.

6 WPS

Wi-Fi Protected Setup (WPS) also known as Wi-Fi Simple Config (WSC)

Simple Config Architectural Overview

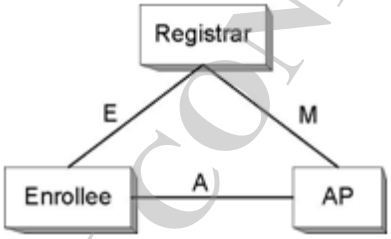
This section presents a high-level description of the Simple Config architecture. Much of the material is taken directly from the Simple Config specification.

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.

6.1 WPS Profile settings

6.1.1 WscConfMode

Description: Configure WPS role (bitwise OR)
Value:

WscConfMode=7

b’000: 0 Disable
b’001: 1 Enrollee
b’010: 2 Proxy
b’100: 4 Registrar

6.1.2 WscConfStatus

Description: Configure WPS state

Value:

WscConfStatus=1

1: AP is unconfigured
2: AP is configured

6.1.3 WscConfMethods

Description: Setup the configuration methods which Enrollee or Registrar supports
Value:

WscConfMethods=238c

Note:

Hexadecimal value only.
// Bitwise OR all values which DUT supports
0x238c = 0x2008 + 0x0280 + 0x0100 + 0x0004
Virtual Display PIN + Virtual Push Button + Keypad + Label PIN

Config Method	Value
Label PIN	0x0004
External NFC Token	0x0010
Integrated NFC Token	0x0020
NFC Interface	0x0040
Keypad	0x0100
Virtual Push Button	0x0280
Physical Push Button	0x0480
Virtual Display PIN	0x2008
Physical Display PIN	0x4008

6.1.4 WscKeyASCII

Description: Define WPS WPAPSK format and key length for un-configured internal WPS Registrar AP
Value:

WscKeyASCII=0

0: Hex (64-bytes)
1: ASCII (Random length)
8 ~ 63: ASCII length

6.1.5 WscSecurityMode

Description: Define WPS Regitrar’s unconfiguraed -> configuraed security mode.
Value:

WscSecurityMode=0

0: WPA2PSK AES
1: WPA2PSK TKIP
2: WPAPSK AES
3: WPAPSK TKIP

6.1.6 WscDefaultSSID0

Description: Default WPS SSID for AP. After WPS process completes with Enrollee when AP acts as un-configured Registrar, AP will use this SSID as new SSID.

Value:

WscDefaultSSID0=SSID

1~32 characters

6.1.7 WscV2Support

Description: Enable or disable WPS v2.0 support
Value:

WscV2Support=1

0: disable
1: enable

6.2 WPS iwpriv command

6.2.1 WscConfMode

Description: Configure WPS role (bitwise OR)
Value:

iwpriv ra0 set WscConfMode=7

b'000: 0 Disable
b'001: 1 Enrollee
b'010: 2 Proxy
b'100: 4 Registrar

6.2.2 WscConfStatus

Description: Configure WPS state
Value:

iwpriv ra0 set WscConfStatus=1

1: AP is unconfigured
2: AP is configured

6.2.3 WscMode

Description: Configure WPS mode
Value:

iwpriv ra0 set WscMode=1

1: PIN Mode
2: PBC Mode

6.2.4 WscStatus

Description: Get WPS Configured Methods.
Value:

iwpriv ra0 set WscStatus=0

0: Not Used
1: Idle
2: WSC Process Fail
3: Start WSC Process
4: Received EAPOL-Start
5: Sending EAP-Req(ID)
6: Receive EAP-Rsp(ID)
7: Receive EAP-Req with wrong WSC SMI Vendor Id
8: Receive EAPReq with wrong WSC Vendor Type
9: Sending EAP-Req(WSC_START)
10: Send M1
11: Received M1
12: Send M2
13: Received M2
14: Received M2D
15: Send M3
16: Received M3
17: Send M4
18: Received M4
19: Send M5
20: Received M5
21: Send M6
22: Received M6
23: Send M7
24: Received M7
25: Send M8
26: Received M8
27: Processing EAP Response (ACK)
28: Processing EAP Request (Done)
29: Processing EAP Response (Done)
30: Sending EAP-Fail
31: WSC_ERROR_HASH_FAIL
32: WSC_ERROR_HMAC_FAIL

33: WSC_ERROR_DEV_PWD_AUTH_FAIL
34: Configured

6.2.5 WscPinCode

Description: Input Enrollee's Pin Code to AP-Registrar.
Value:

iwpriv ra0 WscPinCode xxxxxxxx

xxxxxxx = {00000000 ~ 99999999}

6.2.6 WscOOB

Description: Reset WPS AP to the OOB (out-of-box) configuration.
Value:

iwpriv ra0 set WscOOB=1

0: disable
1: enable

6.2.7 WscGetConf

Description: Trigger WPS AP to do simple config with WPS Client.
Value:

iwpriv ra0 set WscGetConf=1

0: disable
1: enable

6.2.8 WscGenPinCode

Description: Randomly generate enrollee PIN code
Value:

iwpriv ra0 set WscGenPinCode=1

1

6.2.9 WscVendorPinCode

Description: Input vendor's Pin Code to AP-Registrar.
Value:

iwpriv ra0 set WscVendorPinCode=xxxxxxxx

xxxxxxxx: 8 digit pin code

6.2.10 WscSecurityMode

Description: Set WPS regitrar's unconfiguraed -> configuraed security mode.

Value:

iwpriv ra0 set WscSecurityMode=0

0 : WPA2PSK AES
1 : WPA2PSK TKIP
2 : WPAPSK AES
3 : WPAPSK TKIP

6.2.11 WscMultiByteCheck

Description: Set multi byte check is enabled or disabled.
Value:

iwpriv ra0 set WscMultiByteCheck=1

0: disable
1: enable

6.2.12 WscVersion

Description: Set WPS support version
Value:

iwpriv ra0 set WscVersion=10

xx: Hex value

6.2.13 WscVersion2

Description: Set WPS version of V2 support
Value:

iwpriv ra0 set WscVersion2=10

xx: Hex Value

6.2.14 WscV2Support

Description: enable or disable WPS V2.0 support
Value:

iwpriv ra0 WscV2Support=1

0: disable
1: enable

6.2.15 WscFragment

Description: enable or disable WPS fragment
Value:

iwpriv ra0 WscFragment=0

0: disable
1: enable

6.2.16 WscFragmentSize

Description: Set the size of WPS fragmentation.
Value:

iwpriv ra0 set WscFragmentSize=128

128~300

6.2.17 WscSetupLock

Description: enable or disable WPS setup lock
Value:

iwpriv ra0 set WscSetupLock=1

0: disable
1: enable

6.2.18 WscSetupLockTime

Description: Configure WPS setup lock time
Value:

iwpriv ra0 set WscSetupLockTime=0

0: lock forever
Unit: minute

6.2.19 WscMaxPinAttack

Description: Configure WPS pin attack Max time.
Value:

iwpirv ra0 set WscMaxPinAttack

0:Disable
1-10

6.2.20 WscExtraTlvTag

Description: Add extra TLV tag to Beacon, probe response and WSC EAP messages
Value:

iwpriv ra0 set WscExtraTlvTag=1088

Hex value: 0000 ~ FFFF
Example: 1088

6.2.21 WscExtraTlvType

Description: Define data format of extra TLV value
Value:

iwpriv ra0 set WscExtraTlvType=1

0: ASCII string
1: Hex string

6.2.22 WscExtraTlvData

Description: Add extra TLV data to Beacon, probe response and WSC EAP messages
Value:

iwpriv ra0 set WscExtraTlvData=

ASCII string or Hex string

6.2.23 WscStop

Description: Stop WPS process.
Value:

iwpriv ra0 set WscStop

6.2.24 WPS iwpriv command example

6.2.24.1 Disable WPS support

iwpriv ra0 set WscConfMode=0

6.2.24.2 Enable WPS Function

iwpriv ra0 set WscConfMode =7 (Binary: 111)
(AP could be Registrar(0x4), Proxy(0x2) or Enrollee(0x1))

6.2.24.3 WPS AP SC (Simple Config) State

iwpriv ra0 set WscConfStatus=1 (AP is un-configured)
iwpriv ra0 set WscConfStatus=2 (AP is configured)

6.2.24.4 WPS Configured Methods

iwpriv ra0 set WscMode =1 (use PIN code)
iwpriv ra0 set WscMode =2 (use PBC)

6.2.24.5 Input Enrollee's Pin Code to AP-Registrar

iwpriv ra0 set WscPinCode=xxxxxxxx

6.2.24.6 Reset WPS AP to the OOB configuration

iwpriv ra0 set WscOOB=1
(Security: WPAPSK/TKIP, psk: "RalinkInitialAPxx1234" ; SC state: 0x1)

(SSID: RalinkInitialAPxxxxxx, last three characters of AP MAC address)

6.2.24.7 Trigger WPS AP to do simple config with WPS Client

iwpriv ra0 set WscGetConf=1

6.2.24.8 AP services as Enrollee by using PIN code

iwpriv ra0 set WscMode=1
iwpriv ra0 set WscGetConf=1

6.2.24.9 AP services as Enrollee by using PBC

iwpriv ra0 set WscMode=2
iwpriv ra0 set WscGetConf=1

6.2.24.10 AP services as Internal Registrar using PIN code

iwpriv ra0 set WscMode=1
iwpriv ra0 set WscPinCode=xxxxxxxx (PIN code from Enrollee, len=8)
iwpriv ra0 set WscGetConf=1

6.2.24.11 AP services as Internal Registrar using PBC

iwpriv ra0 set WscMode=2
iwpriv ra0 set WscGetConf=1

6.2.24.12 Get WPS Profile from external registrar

iwpriv ra0 get_wsc_profile

6.3 WPS AP Setup Procedure

To run the Access Point (as Enrollee or with Registrar capabilities).
The following scenarios are currently supported:

1. Initial Access Point (AP) setup, with the Registrar configuring the Access Point
   1.1. One WiFi-enabled laptop is setup as the AP acting as an Enrollee
   1.2. Another WiFi-enabled laptop is setup as a station acting as the Registrar
   1.3. Two sub cases are 1a) using EAP transport and 1b) using UPnP transport

2. Configuration of a WiFi client, using an AP with a built-in registrar
   2.1. One WiFi-enabled laptop is setup as the AP with registrar functionality Another WiFi-enabled laptop is setup as a station acting as an Enrollee

3. Configuration of a WiFi client using an external registrar. AP acts as a proxy and communicates with the client over EAP and with the Registrar over UPnP.
   3.1. One WiFi-enabled laptop is setup as a station acting as an Enrollee
   3.2. Second WiFi-enabled laptop is setup as the AP with proxy functionality
   3.3. Third laptop is setup as the registrar. The registrar and the AP are connected over Ethernet.

6.3.1 Running the WPS command-line application

Run the protocol from the console.
First, run UPNP deamon like below:
wscd -w /etc/xml -m 1 -d 3 & (if your xml file in /etc/xml)

use iwpriv command trigger wps, like below:

iwpriv ra0 set WscConfMode=7
iwpriv ra0 set WscConfStatus=1
iwpriv ra0 set WscMode=1
iwpriv ra0 set WscPinCode=31668576
iwpriv ra0 set WscGetConf=1
iwpriv ra0 set WscStatus=0

1. AP services as Enrollee:
   1.1. If AP-Enrollee SC state is 0x1, AP will restart with new configurations.
   1.2. If AP-Enrollee SC state is 0x2, AP sends own configurations to external-registrar and ignores configurations from external-registrar.

2. AP services as Registrar:
   2.1. If AP-Registrar SC state is 0x1, the security mode will be WPAPSK/TKIP and generate random 64bytes psk; after process, AP will restart with new security.

3. WPS AP only services one WPS client at a time.
   3.1. WPS AP only can work in ra0.
   3.2. After WPS configuration finishes, Ralink AP driver writes new configuration to Cfg structure and DAT file.

4. Write items to MBSSID Cfg structure are as below:
   4.1. Ssid
   4.2. AuthMode
   4.3. WepStatus
   4.4. PMK
   4.5. DefaultKeyId.

5. Write items to SharedKey table are as below:
   5.1. Key
   5.2. CipherAlg

6. Write items to DAT file are as belw:
   6.1. SSID
   6.2. AuthMode
   6.3. EncrypType
   6.4. WPAPSK
   6.5. WscConfStatus
   6.6. DefaultKeyID

Note: wscd daemon must be ported to the target platform first.

6.3.2 Initial AP setup with Registrar Configuring AP (EAP/UPnP)

To run command-line console in this mode do:

[Unconfigured AP] ← EAP/UPnP → [Registrar]

Note:

Please make sure upnp deamon is running. After the success of WPS registration, Configured AP will act as a proxy forward EAP and Upnp.)

1. **PIN**
   - **(1) on AP side**
     - iwpriv ra0 set WscConfMode=7
     - iwpriv ra0 set WscConfStatus=1
     - iwpriv ra0 set WscMode=1
     - iwpriv ra0 set WscGetConf=1
   - **(2) on Registrar side**
     - When prompted for the enrollee's PIN, Enter the AP's PIN. Enter the new SSID and new Security for the AP when prompted.
     - The registration process will start, and the application will display the result of the process on completion.
2. **PBC**
   - **(1) on AP side**
     - iwpriv ra0 set WscConfMode=7
     - iwpriv ra0 set WscConfStatus=1
     - iwpriv ra0 set WscMode=2
     - iwpriv ra0 set WscGetConf=1
   - **(2) on Registrar side**
     - Select push-button".
     - The registration process will start, and the application will display the result of the process on completion.

The security config will be written out to the AP and registrar config files.

6.3.3 Adding an Enrollee to AP+Registrar (EAP)

To run command-line console in this mode do:

[AP+Registrar] ← EAP → [Client]

Note:

Please make sure WPS AP configure status is configured, if AP is un-configure, when WPS AP configure client, it will change configure status to configured and auth mode are WPA-PSK)

1. **PIN**
   - **(1) on AP side**
     - iwpriv ra0 set WscConfMode=7
     - iwpriv ra0 set PinCode=31668576 (enter the enrollee's PIN, the PIN from WPS client)
     - iwpriv ra0 set WscMode=1
     - iwpriv ra0 set WscGetConf=1.
     - The registration process will begin, and the console will display the result of the process on completion.
   - **(2) on Client (Enrollee) side**
     - Select PIN process.
     - The process will start, and the application will display the result of the process on completion
2. **PBC**
   - **(1) on AP side**
     - iwpriv ra0 set WscConfMode=7
     - iwpriv ra0 set WscMode=2
     - iwpriv ra0 set WscGetConf=1.
     - The registration process will start, and the application will display the result of the process on completion.
   - **(2) on Client (Enrollee) side**
     - Select PBC process.

- The process will start, and the application will display the result of the process on completion

If the registration is successful, on the client will be re-configured with the new parameters, and will connect to the AP with these new parameters.

6.3.4 Adding an Enrollee with Eternal Registrar (UPnP/EAP)

To run command-line console in this mode do:

[Registrar] ← PnP → [AP] ← EAP → [Client]

1. **PIN**
   - **(1) on Registrar side**
     - When prompted for the enrollee's PIN, Enter the enrollee's PIN.
     - AP Nothing to be selected..
     - The registration process will begin, and the application will display the result of the process on completion.
   - **(2) on Client (Enrollee) side**
     - Select PIN process
     - The process will start, and the application will display the result of the process on completion
2. **PBC**
   - **(1) on Registrar side**
     - Select "push-button".
     - AP Nothing to be selected.
     - The registration process will begin, and the application will display the result of the process on completion.
   - **(2) on Client (Enrollee) side**
     - Select PBC process
     - The registration process will start, and the application will display the result of the process on completion.

6.3.5 WPS Config status

6.3.5.1 Over View

The 'Simple Config State' of WPS attribute in WPS IEs contained in beacon and probe response indicates if a device is configured.If an AP is shipped from the factory in the Not-Configured state (Simple Config State set to 0x01), then the AP must transition to the Configured state (Simple Config State set to 0x02) if any of the following occur:

1. Configuration by an external registrar.

The AP sends the WSC_Done message in the External Registrar configuration process.

2. Automatic configuration by internal registrar.

The AP receives the WSC_Done response in the Enrollee Registration Process from the first Enrollee.
Note:
The internal registrar waits until successful completion of the protocol before applying the automatically generated credentials to avoid an accidental transition from unconfigured to configured in the case that a neighbouring device tries to run WSC before the real enrollee, but fails. A failed attempt does not change the configuration of the AP, nor the Simple Config State.

3. Manual configuration by user.

A user manually configures the AP using whatever interface(s) it provides to modify any one of the following:

- the SSID
- the encryption algorithm
- the authentication algorithm
- any key or pass phrase

If the AP is shipped from the factory in the Not Configured state (Simple Config State set to 0x01), then a factory reset must revert the Simple Config State to Not Configured.

If the AP is shipped from the factory pre-configured with WPA2-Personal mixed mode and a randomly generated key, the Simple Config State may be set to 'Configured' (0x2) to prevent an external registrar from overwriting the factory settings. A factory reset must restore the unit to the same configuration as when it was shipped.

6.4 Basic operation of Ralink WPS AP

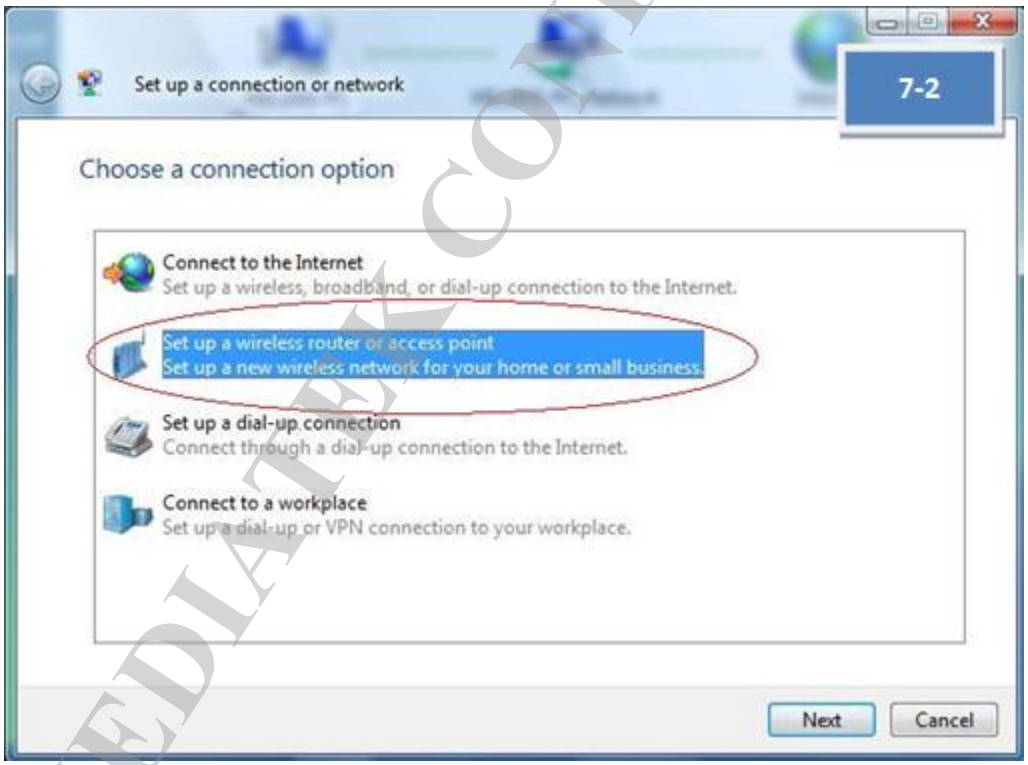
6.4.1 Configure APUT using PIN method through a WLAN external Registrar

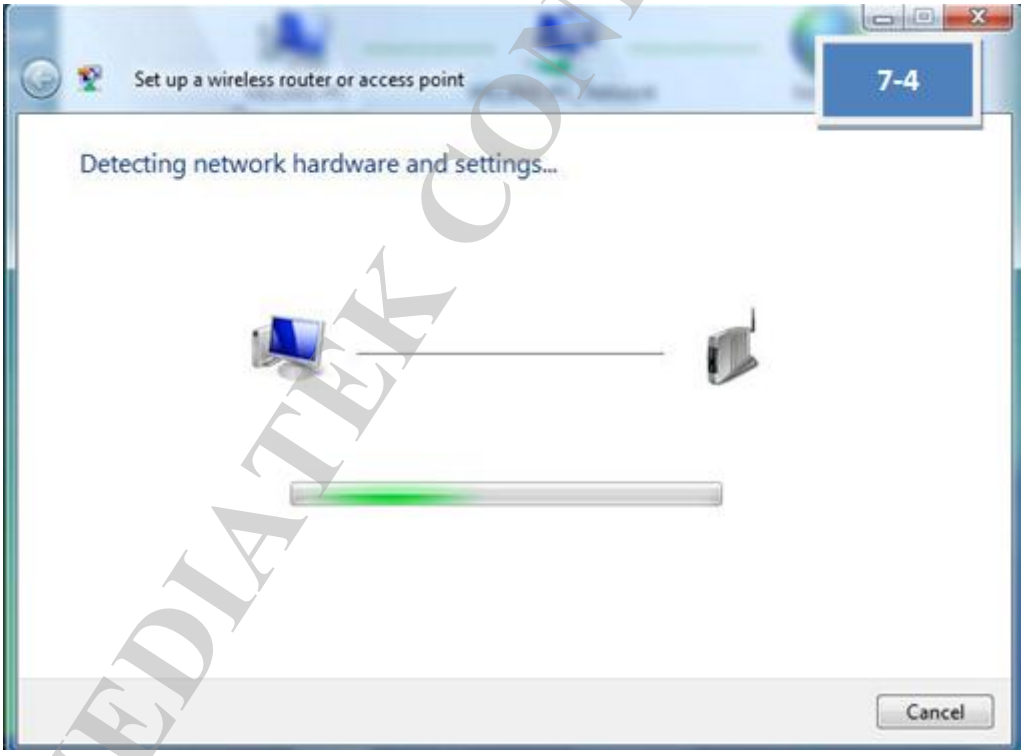
1. [Ralink AP] - Turn on the Ralink AP
2. [Ralink AP] - To change AP ability “iwpriv ra0 set WscConfMode=7”
3. [Ralink AP] - To change from configured to un-configured state: “iwpriv ra0 set WscConfStatus=1 “
4. [Ralink AP] - To change config method to PIN “iwpriv ra0 set WscMode=1”
5. [Ralink AP] - Trigger Ralink AP start process WPS protocol “iwpriv ra0 set WscGetConf=1”
6. [Intel WPS STA] - The Registrar on Intel STA will be configured with the new parameters (SSID = “scaptest4.1.2ssid” and WPA(2)-PSK=”scaptest4.1.2psk”) which should be entered when prompted
7. [Intel WPS STA] - Read AP's PIN from console and enter the PIN at Intel STA.
8. [Intel WPS STA] - Verify that Intel STA successes to ping to Ralink AP
9. [Ralink STA] - Manually configure Ralink STA with the new parameters (SSID = “scaptest4.1.2ssid” and WPA (2)-PSK = “scaptest4.1.2psk”).
10. [Intel WPS STA] - Verify that Intel STA successes to ping to Ralink STA

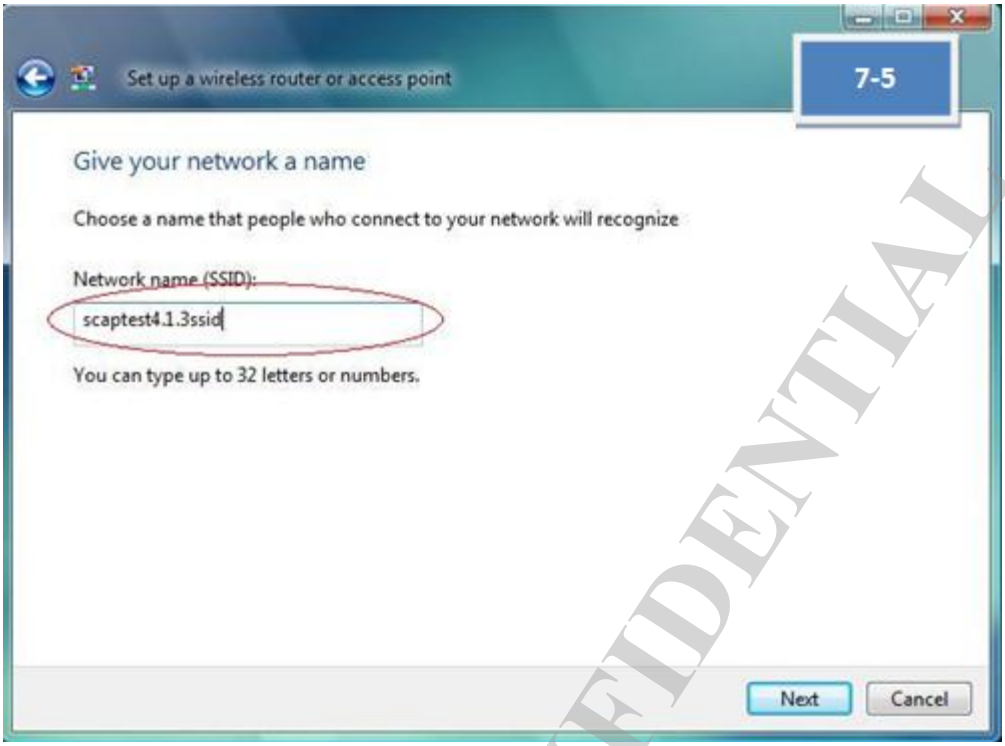
6.4.2 Configure APUT using PIN method through a wired external registrar

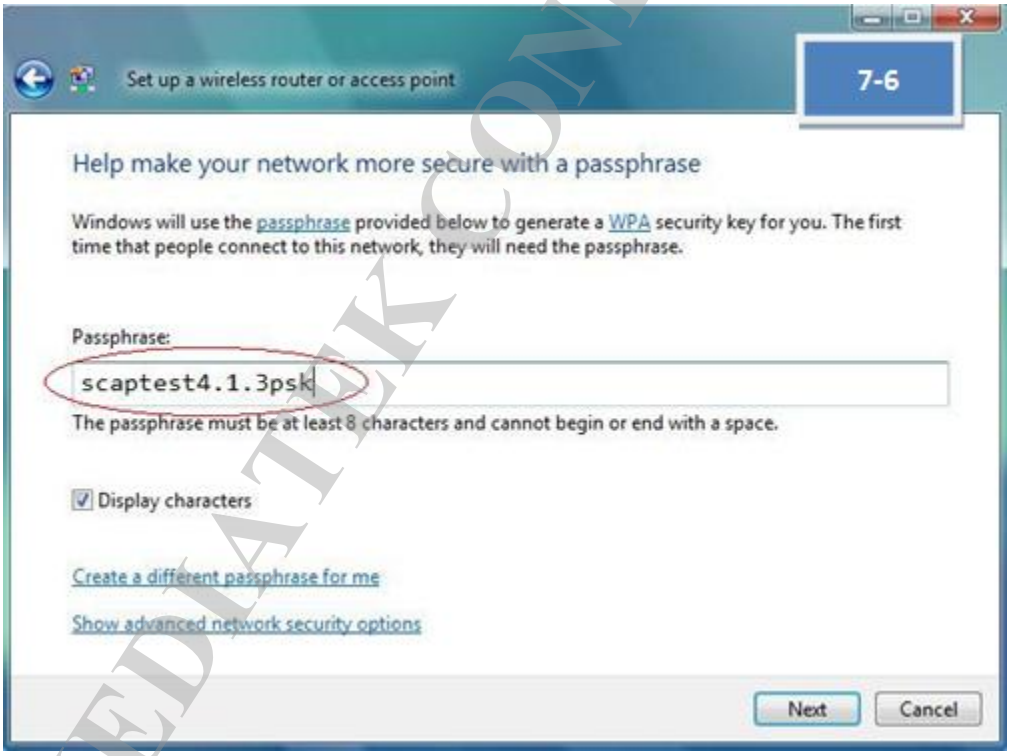
1. [Ralink AP] - Turn on the Ralink AP
2. [Ralink AP] - Connect the Ethernet cable between AP and extern registrar(Windows Vista) and make sure you can pin our device from extern registrar first!
3. [Ralink AP] - To change AP ability “iwpriv ra0 set WscConfMode=7”
4. [Ralink AP] - To change from configured to un-configured state: “iwpriv ra0 set WscConfStatus=1 “
5. [Ralink AP] - To change config method to PIN “iwpriv ra0 set WscMode=1”
6. [Ralink AP] - Trigger Ralink AP start process WPS protocol “iwpriv ra0 set WscGetConf=1”
7. [Microsoft STA] - The Registrar on Microsoft STA will be configured with the new wireless configuration settings (SSID = “scaptest4.1.3ssid” and WPA (2)-PSK=”scaptest4.1.3psk”), which should be entered when prompted.

Please refer to below figures [7-1] to [7-6].



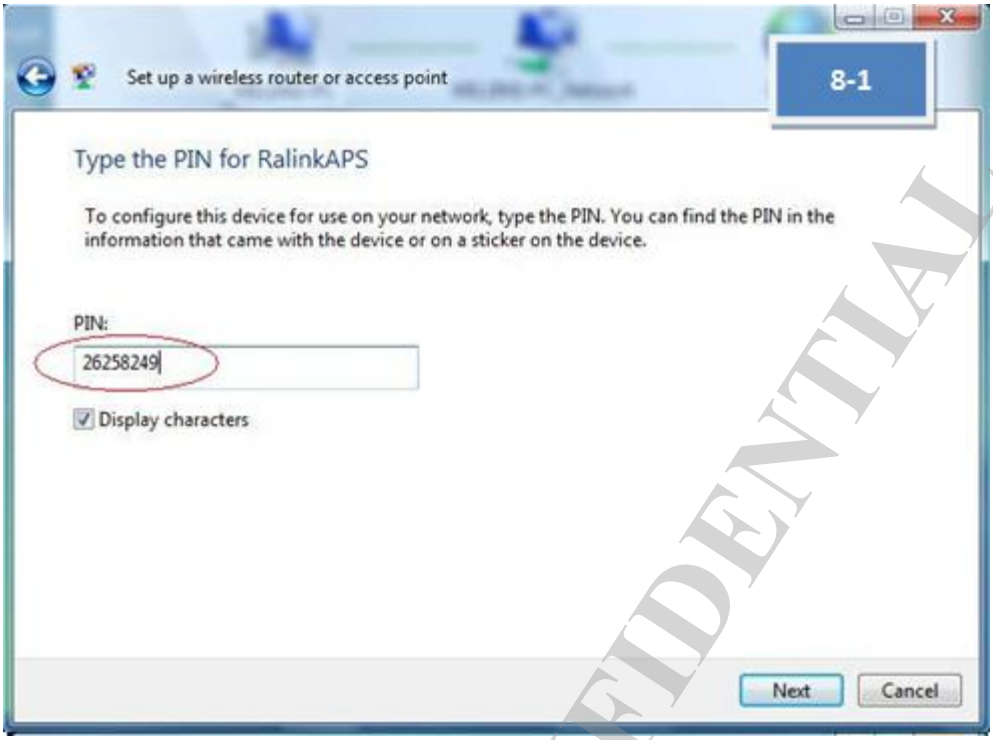


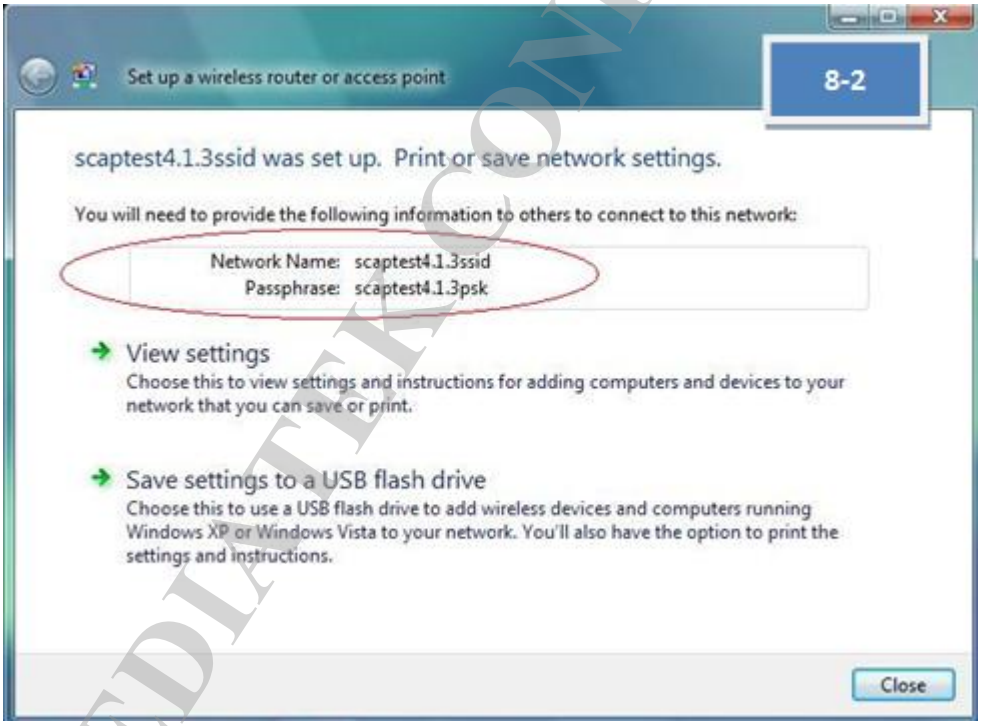




1. [Microsoft STA] - Read AP's PIN from console and enter the PIN at Microsoft STA.

Please refer to below figures [8-1] to [8-2].





1. [Ralink STA] - Manually configure Ralink STA with the new parameters (SSID = “scaptest4.1.3ssid” and WPA (2)-PSK passphrase= “scaptest4.1.3psk”).
2. [Ralink STA] - Verify that Ralink STA successes to ping to Microsoft STA.

6.4.3 Add devices using external Registrars

1. [Ralink AP] - Turn on the APUT.
2. [Ralink STA] - Turn on the Ralink STA.

3. [Ralink STA] - Push PIN button.
4. [Microsoft STA] - Search will be configure enrollee (you can in control->network and internet->network and sharing center->add a device to the network). Enter the enrollee's PIN(Ralink STA) at Microsoft STA when prompted.
5. [Ralink AP] - Do not thing.
6. [Ralink STA] - Verify that Ralink STA successes to ping Ralink A.

6.4.4 How to know WPS AP services as Internal Registrar, Enrollee or Proxy

It depends on the content of EAP-Response/Identity from WPS Client.

- ⇨ When identity is "WFA-SimpleConfig-Registrar-1-0":
  WPS AP would service as Enrollee. (After set trigger command)
- ⇨ When identity is "WFA-SimpleConfig-Enrollee-1-0":
  WPS AP would service as Internal Registrar and Proxy.
  Without trigger command, WPS AP services as proxy only.

6.4.5 How to know WPS AP PinCode

Use ioctl query **RT_OID_WSC_PIN_CODE** OID to get AP PinCode.

6.4.6 Notes for WPS

1. AP services as Enrollee:
   1.1. If AP-Enrollee SC state is 0x1, AP's configuration is changeable and will restart with new configurations.
   1.2. If AP-Enrollee SC state is 0x2, AP's configuration is un-changeable. AP sends own configurations to external-registrar and ignores configurations from external-registrar.
2. AP services as Registrar:
   2.1. If AP-Registrar SC state is 0x1, the security mode will be WPAPSK/TKIP and generate random 64bytes psk; after process, AP will restart with new security.
3. AP services as Proxy:
   3.1. The value of SC state has no effect in proxy mode.
   3.2. WPS AP only services one WPS client at a time.
   3.3. WPS AP only can work in ra0.

6.4.7 Compile flag for WPS AP

```
WFLAGS += -DWSC_SUPPORT
```

6.4.8 WPS related Document

1. Wi-Fi Protected Setup Specification v1.0 (member only)
2. Wi-Fi Protected Setup White Paper
3. Introducing Wi-Fi Protected Setup
4. WSC Linux* Reference Implementation
5. How to Use Windows Connect Now Configuration to Enable Simple Setup for Consumer Wi-Fi Networks [WinHEC 2006; 5.83 MB]
6. Network Infrastructure Device Implementer's Guide

6.5 UPNP Daemon HOWTO

6.5.1 Build WPS UPnP Daemon

Requirements:

1. Linux platform
2. Ralink wireless driver version which support WPS
3. Libupnp
   - ⇨ You can download the libupnp source code from the following URL: http://upnp.sourceforge.net/
   - ⇨ libupnp-1.3.1 is preferred version. For other versions, you may need to patch our modification to the library yourself.
4. POSIX thread library
   - ⇨ Both libupnp and our WPS UPnP daemon need the POSIX thread library, following are recommended pthread library version.
     - For uCLibc, need the version >= 0.9.27
     - For GLIBC, need the version >= 2.3.2
   - ⇨ If your pthread library is older than upper list, you may need to upgrade it.

Build and Run:

1. Modify the "$(work_directory)/wsc_upnp/Makefile" and change the compile flags depends on your target platform.
   - ⇨ Ex. For arm-Linux target platform, you may need to set the following fags:
     - CROSS_COMPILE = arm-Linux-
     - TARGET_HOST = arm-Linux
     - **WIRELESS_H_INCLUDE_PATH = /usr/src/kernels/2.6.11-1.1369_FC4-smp-i686/include/**
2. Modify the "$(work_directory)/wsc_upnp/libupnp-1.3.1/Makefile.src" and change the configure parameters.
   - ⇨ Ex. For big-endian system, you may need to add CFAGS as following:
     - ./configure --host=$(TARGET_HOST) CFLAGS="-mbig-endian"
3. Compile it
   - ⇨ Run "make" in "$(work_directory)/wsc_upnp", after successful compilation, you will get an execution file named "wscd".
4. Install
   - ⇨ Create a sub-directory named "xml" in the "/etc" of your target platform
   - ⇨ Copy all files inside in "$(work_directory)/wsc_upnp/xml" to "/etc/xml"
     - Copy the "wscd" to the target platform.
5. Run it
   - ⇨ Before run it, be sure the target platform already **has set the default route or has a route entry for subnet 239.0.0.0 (For UuPnP Multicast)**. Or the WPS daemon will failed when do initialization.
   - ⇨ Now you can run it by following command:
     - /bin/wscd –m 1 –d 3

Related Document:

1. WPS Specification (Simple_Config_v1.0g.pdf)
2. UPnP Device Architecture 1.0
3. Windows Connect Now-NET Version 1.0
4. WFAWLANConfig:1 Service Template Version 1.01
5. WFA Device:1 Device Template Version 1.01

6.6 WPS Command & OID Example

6.6.1 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");
```

7 WMM

7.1 Introduction

IEEE 802.11e amendment is to provide basic QoS features to 802.11 network and Wi-Fi Multimedia (WMM) is a WFA interoperability certification based on the IEEE 802.11e standard. WMM prioritizes wireless traffic according to four Access Categories, including Voice (VO), Video (VI), Best Effort (BE) and Background (BK).

7.2 WMM iwpriv command

7.2.1 WmmCapable

Description: Enable or disable WMM QoS function
Value:

iwpriv ra0 set WmmCapable=1

0: disable
1: enable

7.3 Parameters in RT2860AP.dat

7.3.1 WmmCapable

Description: Enable or disable WMM QoS function
Value:

WmmCapable=1

0: disable
1: enable

Note: Only WmmCapable has iwpriv command support

7.3.2 APSDCapable

Description: WMM Automatic Power Save Delivery (APSD) function configuration
Value:

APSDCapable=0

0: disable
1: enable

7.3.3 APAifsn

Description: AP arbitration interframe space number configuration
Value:

APAifsn=3;7;1;1

AC_BE;AC_BK;AC_VI;AC_VO

7.3.4 APCwmin

Description: AP contention window minimum (exponent) configuration
Value:

APCwmin=4;4;3;2

AC_BE;AC_BK;AC_VI;AC_VO

7.3.5 APCwmax

Description: AP contention window maximum (exponent) configuration
Value:

APCwmax=6;10;4;3

AC_BE;AC_BK;AC_VI;AC_VO

7.3.6 APTxop

Description: AP Transmit Opportunity configuration (unit: 32μs)
Value:

APTxop=0;0;94;47

AC_BE;AC_BK;AC_VI;AC_VO

7.3.7 APACM

Description: AP Admission Control Mandatory configuration
Value:

APACM=0;0;0;0

AC_BE;AC_BK;AC_VI;AC_VO

7.3.8 BSSAifsn

Description: STA arbitration interframe space number configuration
Value:

BSSAifsn=3;7;2;2

AC_BE;AC_BK;AC_VI;AC_VO

7.3.9 BSSCwmin

Description: STA contention window minimum (exponent) configuration
Value:

BSSCwmin=4;4;3;2

AC_BE;AC_BK;AC_VI;AC_VO

7.3.10 BSSCwmax

Description: STA contention window maximum (exponent) configuration
Value:

BSSCwmax=10;10;4;3

AC_BE;AC_BK;AC_VI;AC_VO

7.3.11 BSSTxop

Description: STA Transmit Opportunity configuration (unit: 32μs)
Value:

BSSTxop=0;0;94;47

AC_BE;AC_BK;AC_VI;AC_VO

7.3.12 BSSACM

Description: STA Admission Control Mandatory configuration
Value:

BSSACM=0;0;0;0

AC_BE;AC_BK;AC_VI;AC_VO

7.3.13 AckPolicy

Description: Acknowledgement policy configuration
Value:

AckPolicy=0;0;0;0

0: Normal Ack or Implicit Block Ack Request
1: No Ack
2: No explicit acknowledgement
3: Block Ack

AC_BE;AC_BK;AC_VI;AC_VO

7.4 How to Run WMM test

1. WmmCapable=1
2. TxBurst=**0**
3. Parameters for AP
   APAifsn=3;7;1;1 // AC_BE;AC_BK;AC_VI;AC_VO
   APCwmin=4;4;3;2 // AC_BE;AC_BK;AC_VI;AC_VO
   APCwmax=6;10;4;3 // AC_BE;AC_BK;AC_VI;AC_VO
   APTxop=0;0;94;47 // AC_BE;AC_BK;AC_VI;AC_VO
   APACM=0;0;0;0 // AC_BE;AC_BK;AC_VI;AC_VO
4. Parameters for all STAs
   BSSAifsn=3;7;2;2 // AC_BE;AC_BK;AC_VI;AC_VO
   BSSCwmin=4;4;3;2 // AC_BE;AC_BK;AC_VI;AC_VO
   BSSCwmax=10;10;4;3 // AC_BE;AC_BK;AC_VI;AC_VO
   BSSTxop=0;0;94;47 // AC_BE;AC_BK;AC_VI;AC_VO
   BSSACM=0;0;0;0 // AC_BE;AC_BK;AC_VI;AC_VO
5. Ack policy
   AckPolicy=0;0;0;0 // AC_BE;AC_BK;AC_VI;AC_VO;

All default values comply with the Wi-Fi spec.

8 IEEE802.11d & IEEE802.11h

8.1 IEEE802.11d

Regulatory Domains

To turn on IEEE802.11d, just fill up the parameter of 'CountryCode', according to ISO3166 code list. This parameter can work in A/B/G band.

The parameter of "CountryCode' needs to match with 'CountryRegion' or 'CountryRegionABand' depends on A or B/G band

Wi-Fi test requirement for IEEE802.11d

Country code IE (0x07) includes in beacon frame and probe response

Power constraint IE (32) includes in beacon frame and probe response

8.2 IEEE802.11h

Spectrum and Transmit Power Management

1. To turn on IEEE802.11h, just fill up the parameters of 'IEEE80211H', 'AutoChannelSelect' as 1, WirelessMode set as 3 to support A band. This parameter can work in only A band.
2. Use 'CSPeriod' to determine how many beacons before channel switch
3. Driver will turn off BBP tuning temporarily in radar detection mode
4. If turn on IEEE802.11h, AP will have 60sec to do channel available check, and will not send beacon and can not be connect.
5. Wi-Fi test requirement for IEEE802.11h
   - Force AP switch channel, AP will stop beacon transmit between 15 sec
   - At least five beacon includes channel switch announcement IE (37)in beacon frame
6. ETSI test requirement, please refer to ETSI EN 301 893 for V1.2.3 detail

Table D.1: DFS requirement values

Parameter	Value
Channel Availability Check Time	60 s
Channel Move Time	10 s
Channel Closing Transmission Time	260 ms

Table D.2: Interference Threshold values, Master

Maximum Transmit Power	Value (see note)
≥ 200 mW	-64 dBm
< 200 mW	-62 dBm
NOTE: This is the level at the input of the receiver assuming a 0 dBi receive antenna.	

Table D.3: Interference Threshold values, Slave

Maximum Transmit Power	Value (see note)
≥ 200 mW	-64 dBm
< 200 mW	N/A
NOTE: This is the level at the input of the receiver assuming a 0 dBi receive antenna.	

9 SECURITY

9.1 All possible combinations of security policy

Type I. Without Radius (IEEE8021X has to be **False**)

	OPEN	SHARED	WEPAUTO
NONE	V	X	X
WEP	V	V	V
802.1x daemon	Off	Off	Off

Type II. With Radius (Non-WiFi standard) (IEEE8021X has to be **True**)

	OPEN
NONE	V
WEP	V
802.1x daemon	On

Type III. With WFA WPA/WPA2 (IEEE8021X has to be **False**)

	WPAPSK	WPA2PSK	WPAPSK WPA2PSK	WPA	WPA2	WPA WPA2
TKIP	V	V	V	V	V	V
AES	V	V	V	V	V	V
TKIPAES	V	V	V	V	V	V
802.1x daemon	Off	Off	Off	**On**	**On**	**On**

9.2 Security iwpriv command

9.2.1 AuthMode

Description: WLAN security authentication mode
Value:

iwpriv ra0 set AuthMode=OPEN

OPEN	Open system
SHARED	Shared key system
WEPAUTO	Auto switch between OPEN and SHARED
WPAPSK	WPA Pre-Shared Key (Infra)
WPA2PSK	WPA2 Pre-Shared Key (Infra)
WPAPSKWPA2PSK	WPAPSK/WPA2PSK mixed mode (Infra)
WPA	WPA Enterprise mode (Need wpa_supplicant)
WPA2	WPA2 Enterprise mode (Need wpa_supplicant)
WPA1WPA2	WPA/WPA2 mixed mode (Need wpa_supplicant)

9.2.2 EncrypType

Description: WLAN security encryption type
Value:

iwpriv ra0 set EncrypType=NONE

NONE	No encryption
WEP	Wired Equivalent Privacy
TKIP	Temporal Key Integrity Protocol
AES	Advanced Encryption Standard
TKIPAES	Mixed cipher

9.2.3 DefaultKeyID

Description: Default key ID (WEP only)
Value:

iwpriv ra0 set DefaultKeyID=1

The ID range is 1~4

9.2.4 Key1

Description: Key 1 string (WEP only)
Value:

iwpriv ra0 set Key1=aaaaa

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.2.5 Key2

Description: Key 2 string (WEP only)
Value:

iwpriv ra0 set Key2=aaaaa

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.2.6 Key3

Description: Key 3 string (WEP only)
Value:

iwpriv ra0 set Key3=aaaaa

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.2.7 Key4

Description: Key 4 string (WEP only)
Value:

iwpriv ra0 set Key4=aaaaa

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.2.8 WPAPSK

Description: WLAN security password for TKIP/AES
Value:

iwpriv ra0 set WPAPSK=12345678

8~63 ASCII characters
64 hexadecimal characters

9.2.9 WpaMixPairCipher

Description: Providing more flexible combination of cipher suite
Value:

iwpriv ra0 set WpaMixPairCipher=WPA_TKIP_WPA2_AES

WPA_AES_WPA2_TKIPAES
WPA_AES_WPA2_TKIP
WPA_TKIP_WPA2_AES
WPA_TKIP_WPA2_TKIPAES
WPA_TKIPAES_WPA2_AES
WPA_TKIPAES_WPA2_TKIPAES
WPA_TKIPAES_WPA2_TKIP

9.3 Parameters in RT2860AP.dat

9.3.1 AuthMode

Description: WLAN security authentication mode
Value:

AuthMode=OPEN

OPEN	Open system
SHARED	Shared key system
WEPAUTO	Auto switch between OPEN and SHARED
WPAPSK	WPA Pre-Shared Key (Infra)
WPA2PSK	WPA2 Pre-Shared Key (Infra)
WPAPSKWPA2PSK	WPAPSK/WPA2PSK mixed mode (Infra)
WPA	WPA Enterprise mode (Need wpa_supplicant)
WPA2	WPA2 Enterprise mode (Need wpa_supplicant)

WPA1WPA2	WPA/WPA2 mixed mode (Need wpa_supplicant)

9.3.2 EncrypType

Description: WLAN security encryption type
Value:

EncrypType=NONE

NONE	No encryption
WEP	Wired Equivalent Privacy
TKIP	Temporal Key Integrity Protocol
AES	Advanced Encryption Standard
TKIPAES	Mixed cipher

9.3.3 IEEE8021X

Description: Enable or disable 8021X-WEP mode

Value:

IEEE8021X=0

0: disable
1: enable

this field is enabled only when
-WEP or Radius-NONE mode on, otherwise must disable.

9.3.4 RekeyMethod

Description: Configuration of rekey method for WPA/WPA2
Value:

RekeyMethod=DISABLE

TIME:	Time rekey
PKT:	Packet rekey
DISABLE:	Disable rekey

9.3.5 RekeyInterval

Description: Rekey interval configuration for WPA/WPA2
Value:

RekeyInterval=0

The value range is 0 ~ 0x3FFFFF. (Unit: 1 second or 1000 packets)
Use 0 to disable rekey.

9.3.6 PMKCachePeriod

Description: PMK cache life time configuration for WPA/WPA2
Value:

PMKCachePeriod=10

The value range is 0 ~ 65535. (Unit: minute)

9.3.7 WPAPSK

Description: WLAN security password for TKIP/AES
Value:

WPAPSK=01234567

8~63 ASCII characters
64 hexadecimal characters

9.3.8 DefaultKeyID

Description: Default key ID (WEP only)
Value:

DefaultKeyID=1

The ID range is 1~4

9.3.9 Key1Type

Description: Key 1 type
Value:

Key1Type=0

0: Hexadecimal
1: ASCII

9.3.10 Key1Str

Description: Key 1 string
Value:

Key1Str=

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.3.11 Key2Type

Description: Key 2 type
Value:

Key2Type=0

0: Hexadecimal
1: ASCII

9.3.12 Key2Str

Description: Key 2 string
Value:

Key2Str=

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.3.13 Key3Type

Description: Key 3 type
Value:

Key3Type=0

0: Hexadecimal
1: ASCII

9.3.14 Key3Str

Description: Key 3 string
Value:

Key3Str=

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.3.15 Key4Type

Description: Key 4 type
Value:

Key4Type=0

0: Hexadecimal
1: ASCII

9.3.16 Key4Str

Description: Key 4 string
Value:

Key4Str=

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.3.17 WpaMixPairCipher

Description: Providing more flexible combination of cipher suite
Value:

WpaMixPairCipher=WPA_TKIP_WPA2_AES

WPA_AES_WPA2_TKIPAES
WPA_AES_WPA2_TKIP
WPA_TKIP_WPA2_AES
WPA_TKIP_WPA2_TKIPAES
WPA_TKIPAES_WPA2_AES
WPA_TKIPAES_WPA2_TKIPAES
WPA_TKIPAES_WPA2_TKIP

9.3.18 PreAuth

Description: Enable or disable WPA2 pre-authentication mode
Value:

PreAuth=0

0: disable
1: enable

9.4 New WFA Security Rules

		2013/12/31	2014/1/1
Personal			
WPA-PSK Only	TKIP	V	X
	AES	△	X
WPA2-PSK Only	TKIP	△	X
	AES	V	V
WPA-PSK/WPA2-PSK Mixed			
WPA-PSK	TKIP	V	V
	AES	△	X
WPA2-PSK	TKIP	△	X
	AES	V	V
Enterprise			
WPA Only	TKIP	V	X
	AES	△	X
WPA2 Only	TKIP	△	X
	AES	V	V
WPA/WPA2 Mixed			
WPA	TKIP	V	V
	AES	△	X
WPA2	TKIP	△	X
	AES	V	V

V = Allowed by WFA
X = Prohibited by WFA
△ = It was not prohibited by WFA, but no test case use it.

Note: Please check 9.5.5 for the correct settings of mixed mode.

9.5 iwpriv command examples

Please specify SSID at last step to trigger the AP restart procedure which would reload new security settings.

9.5.1 OPEN/NONE

1. iwpriv ra0 set AuthMode=OPEN
2. iwpriv ra0 set EncrypType=NONE
3. iwpriv ra0 set IEEE8021X=0
4. iwpriv ra0 set SSID=myownssid

9.5.2 SHARED/WEP

1. iwpriv ra0 set AuthMode=SHARED
2. iwpriv ra0 set EncrypType=WEP
3. iwpriv ra0 set Key1=0123456789
4. iwpriv ra0 set DefaultKeyID=1
5. iwpriv ra0 set IEEE8021X=0
6. iwpriv ra0 set SSID=myownssid

9.5.3 WPAPSK/TKIP

1. iwpriv ra0 set AuthMode=WPA2PSK
2. iwpriv ra0 set EncrypType=TKIP
3. iwpriv ra0 set IEEE8021X=0
4. iwpriv ra0 set SSID=myownssid
5. iwpriv ra0 set WPAPSK=myownpresharedkey
6. iwpriv ra0 set DefaultKeyID=2
7. iwpriv ra0 set SSID=myownssid

Note: Deprecated by WFA since 2014.01.01

9.5.4 WPA2PSK/AES

1. iwpriv ra0 set AuthMode=WPA2PSK
2. iwpriv ra0 set EncrypType=AES
3. iwpriv ra0 set IEEE8021X=0
4. iwpriv ra0 set SSID=MySsid
5. iwpriv ra0 set WPAPSK=MyPassword
6. iwpriv ra0 set DefaultKeyID=2
7. iwpriv ra0 set SSID=MySsid

9.5.5 WPAPSKWPA2PSK/TKIPAES

1. iwpriv ra0 set AuthMode=WPAPSKWPA2PSK
2. iwpriv ra0 set EncrypType=TKIPAES
3. iwpriv ra0 set IEEE8021X=0
4. iwpriv ra0 set WpaMixPairCipher=**WPA_TKIP_WPA2_AES**
5. iwpriv ra0 set SSID=MySsid
6. iwpriv ra0 set WPAPSK=MyPassword
7. iwpriv ra0 set DefaultKeyID=2
8. iwpriv ra0 set SSID=MySsid

10 Authenticator

rt2860apd - IEEE 802.1X Authenticator (user space utility)
rt2860apd is an optional user space component for the SoftAP driver. It provides IEEE 802.1X Authenticator feature when you choose to use external RADIUS Authentication Server (AS).

10.1 IEEE 802.1X features in rt2860apd

IEEE Std. 802.1X-2001 is a standard for port-based network access control. It introduces an extensible mechanism for authenticating and authorizing users. rt2860apd implements part of IEEE 802.1X features which help AS authorizing Supplicant and also prove itself a valid Authenticator to AS. Please be noted that rt2860apd does not include the state machine for key management. The key management function is included in the SoftAP driver. rt2860apd relays the frames between the Supplicant and the AS. Not until either one timeout or Success or Fail frame indicated does rt2860apd finish the authentication process. The port control entity is implemented in the SoftAP driver.

10.2 How to start rt2860apd

Please run “rt2860apd” in your system script.

10.3 rt2860apd configuration for IEEE 802.1X

When rt2860apd starts, it reads the configuraion file to derive parameters. For any changes to make, one need to first edit the configuration file, then restart rt2860apd.
Please add 4 required parameters in the configuration file for WLAN SoftAP driver (RT2860AP.dat/RT2870AP.dat).
RADIUS_Server='192.168.2.3'
RADIUS_Port='1812'
RADIUS_Key='password'
own_ip_addr='your_ip_addr'
The word in ' ' must be replaced with your own correct setting. Please make sure 'your_ip_addr' and RADIUS_Server is connected and RADIUS_Server's IAS (or related) services are started.

The optional variables as below,

- session_timeout_interval is for 802.1x reauthentication setting.
  - set to zero to disable 802.1x reauthentication service for each session.
  - session_timeout_interval unit is second and must be larger than 60.
  - For example,
    - session_timeout_interval = 120
      reauthenticate each session every 2 minutes.
    - session_timeout_interval = 0
      disable reauthenticate service.
- EAPifname is assigned as the binding interface for EAP negotiation.

- Its default value is "br0". But if the wireless interface doesn't attach to bridge interface or the bridge interface name isn't "br0", please modify it.
- For example,
  - EAPifname=br0

- PreAuthifname is assigned as the binding interface for WPA2 Pre-authentication.
  - Its default value is "br0". But if the ethernet interface doesn't attach to bridge interface or the bridge interface name isn't "br0", please modify it.
  - For example,
    - PreAuthifname=br0

10.4 Support Multiple RADIUS Servers

We use complier option to turn on/off the multiple RADIUS servers for 802.1x.
If you want to enable the feature, make sure that "MULTIPLE_RADIUS" is defined in Makefile. Default is disabled. Besides, you must modify the file "RT2860AP.dat" to co-operate with 802.1x. We extend some variables to support individual RADIUS server IP address, port and secret key for MBSS.
E.g.
RADIUS_Server=192.168.2.1;192.168.2.2;192.168.2.3;192.168.2.4
RADIUS_Port=1811;1812;1813;1814
RADIUS_Key=ralink_1;ralink_2;ralink_3;ralink_4
RADIUS_Server=10.10.10.1; 10.10.10.2; 10.10.10.3; 10.10.10.4
RADIUS_Port=1812;1812;1812;1812
RADIUS_Key=ralink_5;ralink_6;ralink_7;ralink_8

Or
RADIUS_Key1=ralink_1;
RADIUS_Key1=ralink_5;
RADIUS_Key2=ralink_2;
RADIUS_Key2=ralink_6;
RADIUS_Key3=ralink_3;
RADIUS_Key3=ralink_7;
RADIUS_Key4=ralink_4;
RADIUS_Key4=ralink_8;

For backward compatibility, the driver parses "RADIUS_Key" or RADIUS_KeyX"(X=1~4) for radius key usage. But the paramter "RADIUS_Key" has the first priority.
This implies,
The RADIUS server IP of ra0 is 192.168.2.1, its port is 1811 and its secret key is ralink_1.
The RADIUS server IP of ra1 is 192.168.2.2, its port is 1812 and its secret key is ralink_2.
The RADIUS server IP of ra2 is 192.168.2.3, its port is 1813 and its secret key is ralink_3.
The RADIUS server IP of ra3 is 192.168.2.4, its port is 1814 and its secret key is ralink_4.
If your wireless interface prefix is not "ra", please modify these variables.

Setup Multiple RADIUS Server failover by iwpriv:

iwpriv ra0 set RADIUS_Server=192.168.1.1;192.168.1.2
iwpriv ra0 set RADIUS_Port=1812;1813
iwpriv ra0 set RADIUS_Key=mediatek123;mediatek456

10.5 Enhance dynamic wep keying

In OPEN-WEP with 802.1x mode, the authentication process generates broadcast and unicast key. The unicast key is unique for every individual client so it is always generated randomly by 802.1x daemon. But the broadcast key is shared for all associated clients; it can be pre-set manually by users or generated randomly by 802.1x daemon.

Through the parameter "DefaultKeyID" and its corresponding parameter "KeyXStr"(i.e. X = the value of DefaultKeyID) in RT2860Ap.dat, the 802.1x daemon would use it as the broadcast key material. But if the corresponding parameter "KeyXStr" is empty or unsuitable, the broadcast key would be generated randomly by the 802.1x daemon.

The 802.1x daemon need to read RT2860AP.dat to decide whether the broadcast key is generated randomly or not, so please update the RT2860AP.dat and restart rt2860apd if those correlative parameters are changed.

10.6 Examples for Radius server configuration

10.6.1 Example I

This is a step-by-step guide to set SoftAP using WPA security mechanism. Assume RT2800 SoftAP has ip address 192.168.1.138, AS (Authentication Server) has IP address 192.168.1.1, Radius Secret is myownkey.

1. load WLAN SoftAP driver
   - $insmod rt2860ap.o
2. First edit configuration file with correct value, esp. the following parameters that relate to the authentication features of RT2800AP.dat
   RADIUS_Server=192.168.1.1
   RADIUS_Port=1812
   RADIUS_Key=myownkey
   own_ip_addr=192.168.1.138
3. start RT2800apd daemon by typing.
   - $rt2860apd
4. iwpriv ra0 set AuthMode=WPA
5. iwpriv ra0 set EncrypType=TKIP
6. iwpriv ra0 set DefaultKeyID=2
7. iwpriv ra0 set IEEE8021X=0
8. iwpriv ra0 set SSID=myownssid

10.6.2 Example II

Change 802.1x settings to WPA with TKIP, using 802.1x authentication.

1. Modify 4 parameters
   RADIUS_Server=192.168.2.3
   RADIUS_Port=1812
   RADIUS_Key=password
   own_ip_addr=192.168.1.123

in the RT2860AP.dat and save.

2. iwpriv ra0 set AuthMode=WPA
3. iwpriv ra0 set EncrypType=TKIP
4. iwpriv ra0 set IEEE8021X=0
5. iwpriv ra0 set SSID=myownssid

Note:

Step 4 restarts the rt2860apd, and is essential.

10.6.3 Example III

Change setting to OPEN/WEP with 802.1x.

1. iwpriv ra0 set AuthMode= OPEN
2. iwpriv ra0 set EncrypType= WEP
3. iwpriv ra0 set IEEE8021X=1

Note:

"IEEE8021X=1" only when Radius-WEP or Radius-NONE mode on, otherwise must "IEEE8021X=0".

10.6.4 Example V

Change setting to OPEN/NONE with 802.1x.

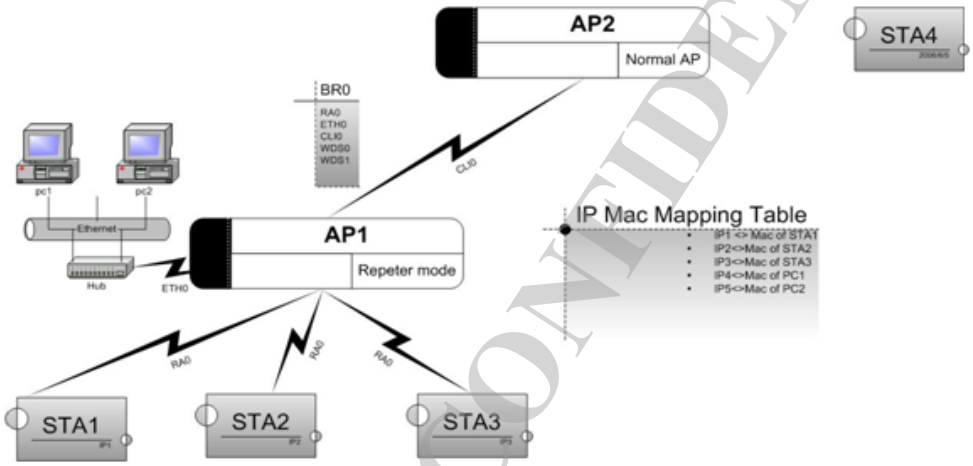
1. iwpriv ra0 set AuthMode= OPEN
2. iwpriv ra0 set EncrypType= NONE
3. iwpriv ra0 set IEEE8021X=1

Note:

"IEEE8021X=1" only when Radius-WEP or Radius-NONE mode on , otherwise must "IEEE8021X=0".

11 AP-CLIENT

The AP-Client function provides a simulated and virtual STA interface while the original AP interface is working simultaneously. Its application is usually a wireless repeater or a wireless extender. AP-Client mainly provides a 1-to-N MAC address mapping mechanism such that multiple stations connected to the AP can transparently communicate with another AP, which we usually call RootAP. When AP-Client function is enabled, besides the original AP interface named ra0, a virtual interface named apcli0 will be created. In a repeater application, the software bridge, like br0, is used to relay packets between these two interfaces.The following figure shows the common network topology and operation module of our AP-Client function.



AP1 is an Access Point which enabled AP-Client and therefore has two wireless interfaces, ra0 and apcli0, providing the AP and station function respectively. AP2 is just a tranditional Access Point that provides normal AP function. In the figure, you can see that STA1 associated to AP1 and STA4 associated to AP2. In the old days, if STA1 wants to communicate with STA4, AP1 and AP2 must have some kind of connection between them to relay traffic, like Ethernet LAN (wired) or WDS (wireless). Now with the new AP-Client feature, AP1 can use the simulated STA interface apcli0 to connect to AP2, thus creating the link, and then STA1 can communicate with STA4 transparently and wired stations connected to AP1 through Ethernet could also communicate with STA4.

Here are some reminders for you before using AP-Client.

- AP-Client only supports the following protocols due to the limitation of 1-to-N MAC address mapping mechanism
  - All IP-based network applications
  - ARP
  - DHCP
  - PPPoE
- The last hexadecimal number of the MAC address must be a multiple of 2

11.1 How to Setup AP-Client

- Turn on **APCLI_SUPPORT** in driver config
- Use "**ifconfig apcli0 up**" to bring up your AP-Client interface
- In a repeater application, you may use the following commands to bridge ra0 and apcli0

  - **brctl addif br0 ra0**
  - **brctl addif br0 apcli0**
- The security policy support for AP-Client include
  - OPEN
  - SHARED (WEP)
  - WPAPSK (TKIP, AES)
  - WPA2PSK (TKIP, AES)
- Please be noted that AP-Client is also a virtual interface. When you use AP-Client with MBSSID simultaneously, AP-Client will consume one position and the parameter “BssidNum” should be larger than 1 and less than 7 (1 < BssidNum < 7)
- Use “**iwpriv apcli0 show connStatus**” to display connection status with RootAP

11.2 Parameters in RT2860AP.dat

11.2.1 ApCliEnable

Description: Enable or disable AP-Client function
Value:

ApCliEnable=1

0: disable
1: enable

11.2.2 ApCliSsid

Description: Configure the target/RootAP SSID which AP-Client wants to connect with
Value:

ApCliSsid=target_ssid

target_ssid: 1~32 characters

11.2.3 ApCliBssid

Description: Configure the target BSSID which AP-Client wants to join
Value:

ApCliBssid=00:11:22:33:44:55

Note: It is an optional command. Users can use this command to indicate the desired BSSID. Otherwise, AP-Client would get correct BSSID according to configured SSID automatically.

11.2.4 ApCliAuthMode

Description: AP-Client authentication mode configuration
Value:

ApCliAuthMode=OPEN

OPEN
SHARED
WPAPSK
WPA2PSK

11.2.5 ApCliEncrypType

Description: AP-Client encryption type configuration
Value:

ApCliEncrypType=NONE

NONE
WEP
TKIP
AES

11.2.6 ApCliWPAPSK

Description: WPA/WPA2 Pre-Shared Key configuration
Value:

ApCliWPAPSK=12345678

8~63 ASCII characters
64 hexadecimal characters

11.2.7 ApCliDefaultKeyID

Description: Default key index configuration
Value:

ApCliDefaultKeyID=1

The ID range is 1~4

11.2.8 ApCliKey1Type

Descripion: Set the WEP key type of AP-Client for key index 1
Value:

ApCliKey1Type=0

0: Hexadecimal
1: ASCII

11.2.9 ApCliKey1Str

Description: Set the WEP key string of AP-Client for key 1
Value:

ApcliKey1Str=012345678

10 or 26 hexadecimal characters
5 or 13 ASCII characters

11.2.10 ApCliKey2Type

Descripion: Set the WEP key type of AP-Client for key index 2
Value:

ApCliKey2Type=0

0: Hexadecimal
1: ASCII

11.2.11 ApCliKey2Str

Description: Set the WEP key string of AP-Client for key 2
Value:

ApcliKey2Str=012345678

10 or 26 hexadecimal characters
5 or 13 ASCII characters

11.2.12 ApCliKey3Type

Descripion: Set the WEP key type of AP-Client for key index 3
Value:

ApCliKey3Type=0

0: Hexadecimal
1: ASCII

11.2.13 ApCliKey3Str

Description: Set the WEP key string of AP-Client for key 3
Value:

ApcliKey3Str=012345678

10 or 26 hexadecimal characters
5 or 13 ASCII characters

11.2.14 ApCliKey4Type

Descripion: Set the WEP key type of AP-Client for key index 4
Value:

ApCliKey4Type=0

0: Hexadecimal
1: ASCII

11.2.15 ApCliKey4Str

Description: Set the WEP key string of AP-Client for key 4
Value:

ApcliKey4Str=012345678

10 or 26 hexadecimal characters
5 or 13 ASCII characters

11.2.16 ApCliTxMode

Description: Fixed transmission mode configuration
Value:

ApCliTxMode=HT

cck|CCK,
ofdm|OFDM,
ht|HT

11.2.17 ApCliTxMcs

Description: AP-Client Tx MCS configuration
Value:

ApCliTxMcs=33

0~15, 32: Fixed MCS
33: Auto MCS

11.2.18 ApCliWscSsid

Description: Configure the SSID which AP-Client wants to do WPS negotiation
Value:

ApCliWscSsid=target_ssid

targer_ssid: 1~32 characters

11.3 AP-Client iwpriv command

11.3.1 ApCliEnable

Description: Enable or disable AP-Client function
Value:

iwpriv apcli0 set ApCliEnable=0

0: disable
1: enable

11.3.2 ApCliSsid

Description: Configure the target/RootAP SSID which AP-Client wants to connect with
Value:

iwpriv apcli0 set ApCliSsid=target_ssid

target_ssid: 1~32 characters

11.3.3 ApCliBssid

Description: Configure the target BSSID which AP-Client wants to join
Value:

iwpriv apcli0 set ApCliBssid=00:11:22:33:44:55

Note: It is an optional command. Users can use this command to indicate the desired BSSID. Otherwise, AP-Client would get correct BSSID according to configured SSID automatically.

11.3.4 ApCliAuthMode

Description: AP-Client authentication mode configuration
Value:

iwpriv apcli0 set ApCliAuthMode=OPEN

OPEN
SHARED
WPAPSK
WPA2PSK

11.3.5 ApCliEncrypType

Description: AP-Client encryption type configuration
Value:

iwpriv apcli0 set ApCliEncrypType=NONE

NONE
WEP
TKIP
AES

11.3.6 ApCliWPAPSK

Description: WPA/WPA2 Pre-Shared Key configuration
Value:

iwpriv apcli0 set ApCliWPAPSK=12345678

8~63 ASCII characters
64 hexadecimal characters

11.3.7 ApCliDefaultKeyID

Description: Default key index configuration
Value:

iwpriv apcli0 set ApCliDefaultKeyID=1

The ID range is 1~4

11.3.8 ApCliKey1

Description: Set the WEP key string of AP-Client for key 1
Value:

iwpriv apcli0 set ApcliKey1=012345678

10 or 26 hexadecimal characters
5 or 13 ASCII characters

11.3.9 ApCliKey2

Description: Set the WEP key string of AP-Client for key 2
Value:

iwpriv apcli0 set ApcliKey2=012345678

10 or 26 hexadecimal characters
5 or 13 ASCII characters

11.3.10 ApCliKey3

Description: Set the WEP key string of AP-Client for key 3
Value:

iwpriv apcli0 set ApcliKey3=012345678

10 or 26 hexadecimal characters
5 or 13 ASCII characters

11.3.11 ApCliKey4

Description: Set the WEP key string of AP-Client for key 4
Value:

iwpriv apcli0 set ApcliKey4=012345678

10 or 26 hexadecimal characters
5 or 13 ASCII characters

11.3.12 ApCliTxMode

Description: Fixed transmission mode configuration
Value:

iwpriv apcli0 set ApCliTxMode=HT

CCK
OFDM
HT

11.3.13 ApCliTxMcs

Description: AP-Client Tx MCS configuration
Value:

iwpriv apcli0 set ApCliTxMcs=33

0~15, 32: Fixed MCS
33: Auto MCS

11.3.14 ApCliWscSsid

Description: Configure the SSID which AP-Client wants to do WPS negotiation
Value:

iwpriv apcli0 set ApCliWscSsid=target_ssid

targer_ssid: 1~32 characters

11.3.15 ApCliAutoConnect

Description: Enable or disable the auto-connection function to find the configured SSID
Value:

iwpriv ra0 set ApCliAutoConnect=1

0: disable
1: enable

Note: APCLI_AUTO_CONNECT_SUPPORT must be turned on

11.4 AP-Client normal connection examples

11.4.1 OPEN/NONE

```
iwpriv apcli0 set ApCliEnable=0
iwpriv apcli0 set ApCliAuthMode=OPEN
iwpriv apcli0 set ApCliEncrypType=NONE
iwpriv apcli0 set ApCliSsid=ROOTAP_SSID
iwpriv apcli0 set ApCliEnable=1
```

11.4.2 OPEN/WEP

```
iwpriv apcli0 set ApCliEnable=0
iwpriv apcli0 set ApCliAuthMode=OPEN
iwpriv apcli0 set ApCliEncrypType=WEP
iwpriv apcli0 set ApCliDefaultKeyID=1
iwpriv apcli0 set ApCliKey1=1234567890
iwpriv apcli0 set ApCliSsid=ROOTAP_SSID
iwpriv apcli0 set ApCliEnable=1
```

11.4.3 WPAPSK/TKIP

```
iwpriv apcli0 set ApCliEnable=0
iwpriv apcli0 set ApCliAuthMode=WPAPSK
iwpriv apcli0 set ApCliEncrypType=TKIP
iwpriv apcli0 set ApCliSsid=ROOTAP_SSID
iwpriv apcli0 set ApCliWPAPSK=12345678
iwpriv apcli0 set ApCliSsid=ROOTAP_SSID
iwpriv apcli0 set ApCliEnable=1
```

11.4.4 WPA2PSK/AES

```
iwpriv apcli0 set ApCliEnable=0
iwpriv apcli0 set ApCliAuthMode=WPA2PSK
iwpriv apcli0 set ApCliEncrypType=AES
iwpriv apcli0 set ApCliSsid=ROOTAP_SSID
iwpriv apcli0 set ApCliWPAPSK=12345678
iwpriv apcli0 set ApCliSsid=ROOTAP_SSID
iwpriv apcli0 set ApCliEnable=1
```

11.5 AP-Client WPS connection examples

11.5.1 PIN mode

```
iwpriv apcli0 set ApCliEnable=1
iwpriv apcli0 set WscConfMode=1          // Enrollee
```

```
iwpriv apcli0 set WscMode=1          // PIN mode
iwpriv apcli0 set WscGetConf=1       // Trigger
iwpriv apcli0 set ApCliEnable=1
```

11.5.2 PBC Mode

```
iwpriv apcli0 set ApCliEnable=1
iwpriv apcli0 set WscConfMode=1      // Enrollee
iwpriv apcli0 set WscMode=2          // PBC mode
iwpriv apcli0 set WscGetConf=1       // Trigger
iwpriv apcli0 set ApCliEnable=1
```

12 WDS

A Wireless Distribution System is a system enabling the wireless interconnection of acess points. Each WDS AP needs to be in the same channel, using the same encryption type.

Actually, there is no test plan to ensure the inter-operability of all WDS products from different Vendors. Mediatek's implementation provides two modes of AP-to-AP connectivity. One is **Bridge mode**, in which WDS APs communicate only with each other and do not allow wireless stations to access them. The other is **Repeater mode**, in which WDS APs communicate with each other and with wireless stations.

In case you want to have an auto-learning WDS peer, we also provide the **Lazy mode** in which you do not need to thoroughly configure the WDS settings. However, be noted that you cannot configure all APs to be in Lazy mode, otherwise no 4-address frame will be transmitted at all and auto-learning would be impossible. This means that there should be at least one AP being configured to Bridge mode or Repeater mode.

12.1 How to Steup WDS

1. Edit the driver profile in each WDS peer
   WDS Peer-A with the MAC address 00:0C:43:aa:bb:cc
   - WdsEnable=3
   - WdsPhyMode=HTMIX;
   - WdsList=00:0C:43:11:22:33;
   - WdsEncrypType=NONE;

   WDS Peer-B with the MAC address 00:0C:43:11:22:33
   - WdsEnable=3
   - WdsPhyMode=HTMIX;
   - WdsList=00:0C:43:aa:bb:cc;
   - WdsEncrypType=NONE;

2. Edit your networking script file, like bridge_setup.sh, according to the number of WDS link. Add "**brctl addif br0 wds0**" and "**ifconfig wds0 0.0.0.0**" to relative places

3. Use "**iwpriv ra0 show wdsinfo**" to display WDS link information

12.2 WDS Security

WDS security is **PSK-only**, and it does not support mixed mode, like WPAPSKWPA2PSK.

When WDS is in Lazy mode, all WDS links (wds0 ~ wds3) shall share the same encryption type and key material (referring to wds0 settings). Otherwise, each WDS link has its own security settings. No matter what WDS mode you use, it has nothing to do with the encryption of the main BSSID (ra0).

WdsKey:
It is used for all WDS interfaces and supports only AES and TKIP configuration. If you want to use WEP, key settings will be retrieved from the main BSSID.

Wds0Key/Wds1Key/Wds2Key/Wds3Key:
They are used to configure key settings for each WDS interface.

The following example is to create one WDS link (wds0) with AES encryption.
WdsEnable=3
WdsPhyMode=HTMIX;HTMIX;HTMIX;HTMIX
WdsList=00:0c:43:12:34:56;
WdsEncrypType=AES;NONE;NONE;NONE
Wds0Key=12345678
Wds1Key=
Wds2Key=
Wds3Key=

12.3 Parameters in RT2860AP.dat

12.3.1 WdsEnable

Description: WDS function configuration
Value:

WdsEnable=0

0: **Disable** - Disable WDS function.
1: Restrict mode - Same as Repeater mode.
2: **Bridge mode** - Enable WDS and work like a bridge.
The MAC address of peer WDS APs should be configured in the "WdsList" field.
In this mode, AP is just a bridge and will not send any beacon and will not respond to any probe request packet. Therefore STA will not be able to connect with it.
3: **Repeater mode** - Enable WDS and work like a repeater.
The MAC address of peer WDS APs should be configured in the "WdsList" field.
4: **Lazy mode** - Enable WDS function.
It automatically learns from 4-address format frames sent by the WDS peer and you do not have to configure WdsList manually.

12.3.2 WdsList

Description: WDS peer MAC address configuration
Value:

WdsList=00:10:20:30:40:50;

The maximum WDS link number is 4.
wds0;wds1;wds2;wds3

12.3.3 WdsEncrypType

Description: WDS encryption configuration
Value:

WdsEncrypType=NONE;

The option includes NONE, WEP, TKIP and AES.

Example:

WdsEncrypType=OPEN;WEP;TKIP;AES

The encrptytion of wds0 is OPEN
The encrptytion of wds1 is WEP
The encrptytion of wds2 is TKIP
The encrptytion of wds3 is AES

12.3.4 WdsKey

Description: WDS key configuration

Value:

WdsKey=12345678

8 ~ 63 ASCII characters (eg: 12345678) for TKIP or AES
64 hexadecimal characters for TKIP or AES

WdsKey is kept for backward-compatibility and it only supports TKIP and AES.
You can use either WdsKey or Wds[0-4]Key but not both.

Note: Combinations of WDS security mode

EncrypType	WdsEncrypType	WdsEncrypType of the WDS peer	Note
NONE	NONE	NONE	
WEP	WEP	WEP	Using legacy key setting method
TKIP	TKIP	TKIP	WDS's key is from WdsKey
TKIP	AES	AES	WDS's key is from WdsKey
AES	TKIP	TKIP	WDS's key is from WdsKey
AES	AES	AES	WDS's key is from WdsKey
TKIPAES	TKIP	TKIP	WDS's key is from WdsKey
TKIPAES	AES	AES	WDS's key is from WdsKey

12.3.5 Wds0Key

Description: WDS key for Link-0

Value:

Wds0Key=12345678

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters (eg: 12345678) for TKIP or AES
64 hexadecimal characters for TKIP or AES

12.3.6 Wds1Key

Description: WDS key for Link-1

Value:

Wds1Key=12345678

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters (eg: 12345678) for TKIP or AES
64 hexadecimal characters for TKIP or AES

12.3.7 Wds2Key

Description: WDS key for Link-2
Value:

Wds2Key=12345678

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters (eg: 12345678) for TKIP or AES
64 hexadecimal characters for TKIP or AES

12.3.8 Wds3Key

Description: WDS key for Link-3
Value:

Wds3Key=12345678

10 or 26 hexadecimal characters (eg: 1234567890) for WEP
5 or 13 ASCII characters (eg: 12345) for WEP
8 ~ 63 ASCII characters (eg: 12345678) for TKIP or AES
64 hexadecimal characters for TKIP or AES

12.3.9 WdsPhyMode

Description: WDS link physical mode configuration
Value:

WdsPhyMode=HTMIX;

The option includes CCK, OFDM, HTMIX and GREENFIELD.

Example:

WdsPhyMode=CCK;OFDM;HTMIX;GREENFIELD

The PHY mode of wds0 is CCK
The PHY mode of wds1 is OFDM
The PHY mode of wds2 is HTMIX
The PHY mode of wds3 is GREENFIELD

13 IGMP SNOOPING

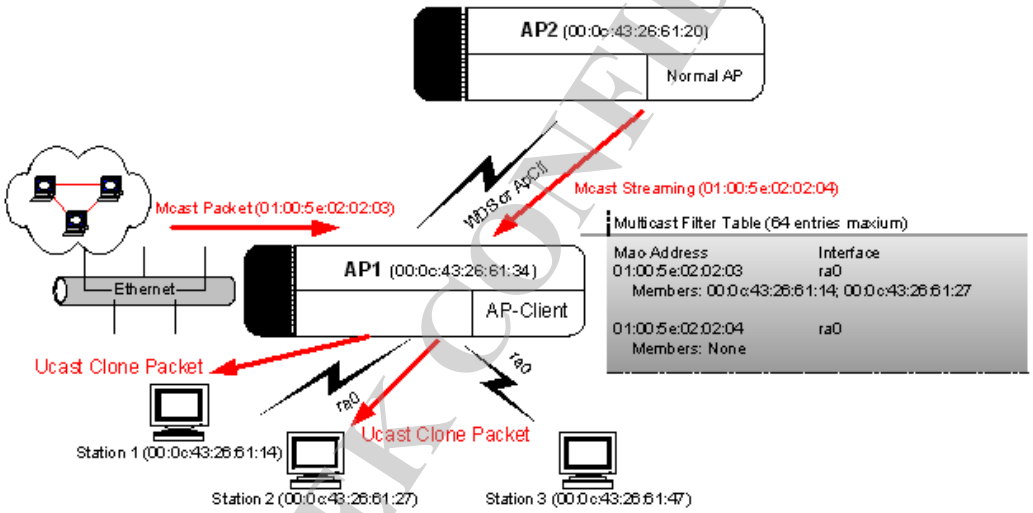
13.1 Basic

Please check the following two Wiki links.

- http://en.wikipedia.org/wiki/Multicast_address
- http://en.wikipedia.org/wiki/Internet_Group_Management_Protocol

IGMP Snooping provides a mechanism converting multicast traffic into unicast traffic. When AP receives incoming multicast traffic, the conversion would be done based on an IGMP Snooping table (Multicast Filter Table).

13.2 Introduction to IGMP Snooping Table



An IGMP Snooping table (a.k.a. Multicast Filter Table) entry consists of 3 components, Group-ID (Multicast MAC Address), Network-Interface and Member-List. Taking above figure for example, you can see that Multicast Filter Table of AP1 has two table entries. One is “01:00:5e:02:02:03” with two members on interface ra0 and the other is “01:00:5e:02:02:04” without any member on interface ra0.

In our implementation, AP will automatically maintain the Multicast Filter Table through packet snooping. The IGMP-Membership-Report packets sent from connected stations would be checked and parsed. You can also manually add and delete an entry through iwpriv command.

13.3 Multicast Packet Parsing Process

When AP receives multicast packets, it will check whether the multicast destination address matches any Group-ID in the Multicast Filter Table. AP will drop the packet if no match found. Otherwise, there are two cases how AP handles a multicast packet. The first one is that Member-List of the matching entry is empty and then AP just forwards multicast packets to all stations connected to the Network-

Interface. In the second case, there are members in the Member-List and AP will do the MC-to-UC conversion based on the membership.

Taking the previous figure for example, AP1 received an Ethernet multicast packet with Group-ID being 01:00:5e:02:02:03. Firstly AP1 checked the Multicast Filter Table and found the first entry matched. Therefore, AP1 cloned every multicast packet into two unicast packets destined to Station 1 and Station 2 respectively.

In the same figure, a multicast streaming sent from AP2 to AP1 with Group-ID 01:00:5e:02:02:04 was forwarded to all stations connected to AP1 (ra0) since the matching entry had no member at all.

<Multicast Filter Table Example>

Group-ID	Network-Interface	Member-List
01:00:5e:02:02:03	ra0	00:0c:43:26:61:14 (Station 1) 00:0c:43:26:61:27 (Station 2)
01:00:5e:02:02:04	ra0	

13.4 Parameters in RT2860AP.dat

13.4.1 IgmpSnEnable

Description: Enable or disable IGMP Snooping function
Value:

IgmpSnEnable=1

0: disable
1: enable

Note: Please make sure that IGMP_SNOOP_SUPPORT is turned on in driver config

13.5 IGMP Snooping iwpriv command

13.5.1 IgmpSnEnable

Description: Enable or disable IGMP Snooping function
Value:

iwpriv ra0 set IgmpSnEnable=1

0: disable
1: enable

13.5.2 IgmpAdd

Description: Create a new group or add a new member to the existing group
Format:

// Create a new group <Group-ID> which can be a MAC address or an IP address

iwpriv ra0 set IgmpAdd=<Group-ID>
// Add a new member to the existing group. [Member] can only be a MAC address
iwpriv ra0 set IgmpAdd=<Group-ID-[Member]-…>

Value:

// Create a new group via either IP or MAC address
iwpriv ra0 set IgmpAdd=**226.2.2.3**
iwpriv ra0 set IgmpAdd=**01:00:5e:02:02:03**

// Add a new member to the existing group
iwpriv ra0 set IgmpAdd=**226.2.2.3-00:0c:43:26:61:11**
// Add 2 new members to the existing group
iwpriv ra0 set IgmpAdd=**01:00:5e:02:02:03-00:0c:43:26:61:27-00:0c:43:26:61:28**

13.5.3 IgmpDel

Description: Delete a group or remove a member from the existing group
Format:

// Delete a group <Group-ID> which can be a MAC address or an IP address
iwpriv ra0 set IgmpDel=<Group-ID>
// Remove a member from the existing group. [Member] can only be a MAC address
iwpriv ra0 set IgmpDel=<Group-ID-[Member]-…>

Value:

// Delete a new group via either IP or MAC address
iwpriv ra0 set IgmpDel=**226.2.2.3**
iwpriv ra0 set IgmpDel=**01:00:5e:02:02:03**

// Remove a member from the existing group
iwpriv ra0 set IgmpDel=**226.2.2.3-00:0c:43:26:61:11**

// Remove members from the existing group
iwpriv ra0 set IgmpDel=**01:00:5e:02:02:03-00:0c:43:26:61:27-00:0c:43:26:61:28**

14 MAC Repeater

The MAC Repeater is a variation of the original AP-Client funcation and it acts as a wireless proxy for its clients. The repeater will create a corresponding upstream connection to the RootAP for each downstream client connected to it. An upstream connection is created according to its own wireless capability and security mode. When a client disconnects from the repeater, the repeater must also disconnect its corresponding upstream connection with the RootAP. All communication between downstream clients and upstream RootAP utilizes one “AP-Client” interface on the repeater.

For example, if there are 3 clients connected to the repeater, 3 upstream connections will be created accordingly. Besides these “proxy connection”, the repeater itself would also create a connection with RootAP. Therefore, in this case there would be totally 3 downstream and 4 upstream connections.

Please be noted that MAC Repeater has the following limitation.

- Roaming of STAs between different BSSs is not supported
- WPA2-Enterprise Security is not supported
- Supported protocols: IPv4 / ARP / DHCP
- The MAC Repeater supports up to 16 clients
- Impact CPU utilization due to parsing all received packets from the STA and all multicast and broadcast packets

14.1 MAC Repeater iwpriv command

14.1.1 MACRepeaterEn

Description: Enable or disable MAC Repeater function
Value:

iwpriv ra0 set MACRepeaterEn=1

0: disable
1: enable

14.1.2 Example

- **iwpriv ra0 set MACRepeaterEn=1**
- ifconfig apcli0 up
- brctl addif br0 apcli0
- iwpriv apcli0 set ApCliEnable=0
- iwpriv apcli0 set ApCliAuthMode=OPEN
- iwpriv apcli0 set ApCliEncrypType=NONE
- iwpriv apcli0 set ApCliSsid=RootAP_SSID
- iwpriv apcli0 set ApCliEnable=1

14.2 Parameter in RT2860AP.dat

14.2.1 MACRepeaterEn

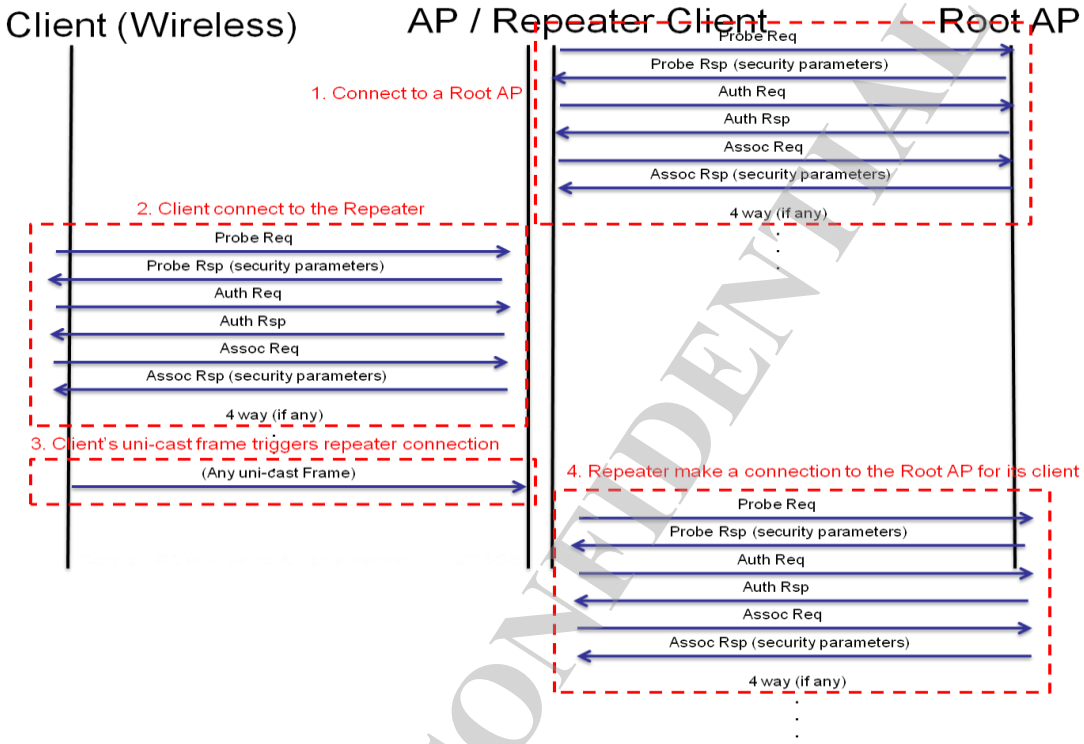
Description: Enable or disable the MAC Repeater function.
Value:

MACRepeaterEn=0

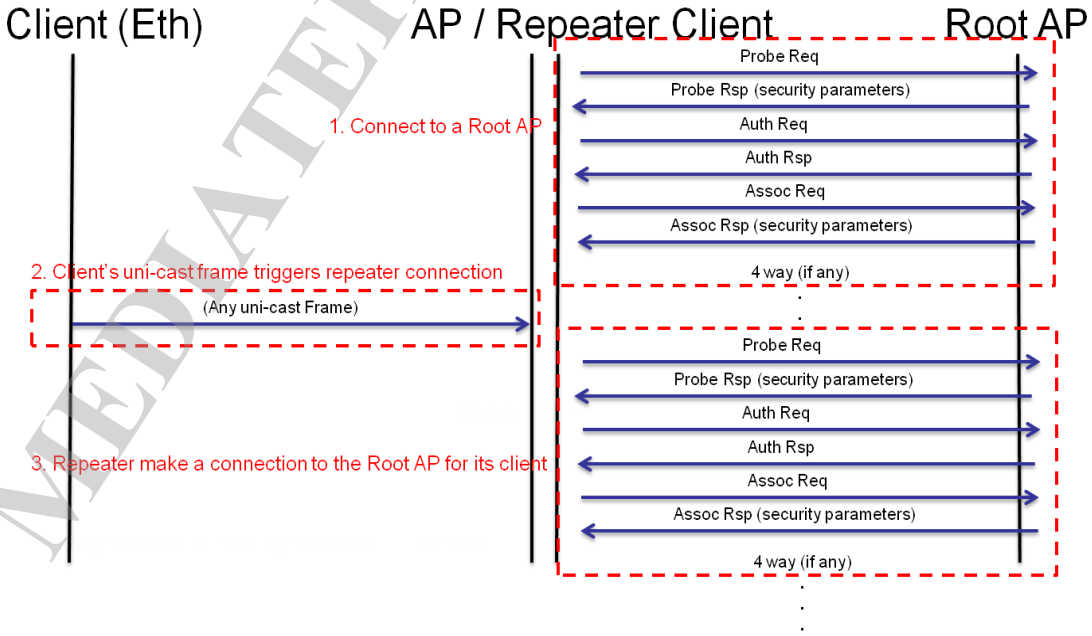
0: disable
1: enable

14.3 Management Frame Flow

14.3.1 Wireless client

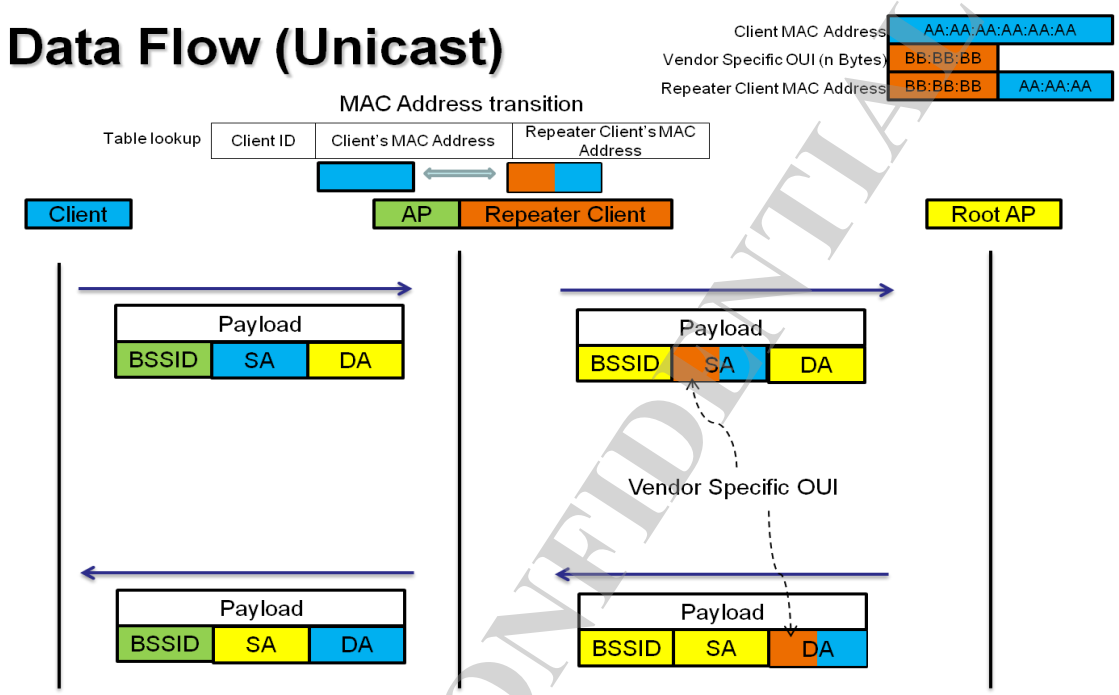


14.3.2 Ethernet client

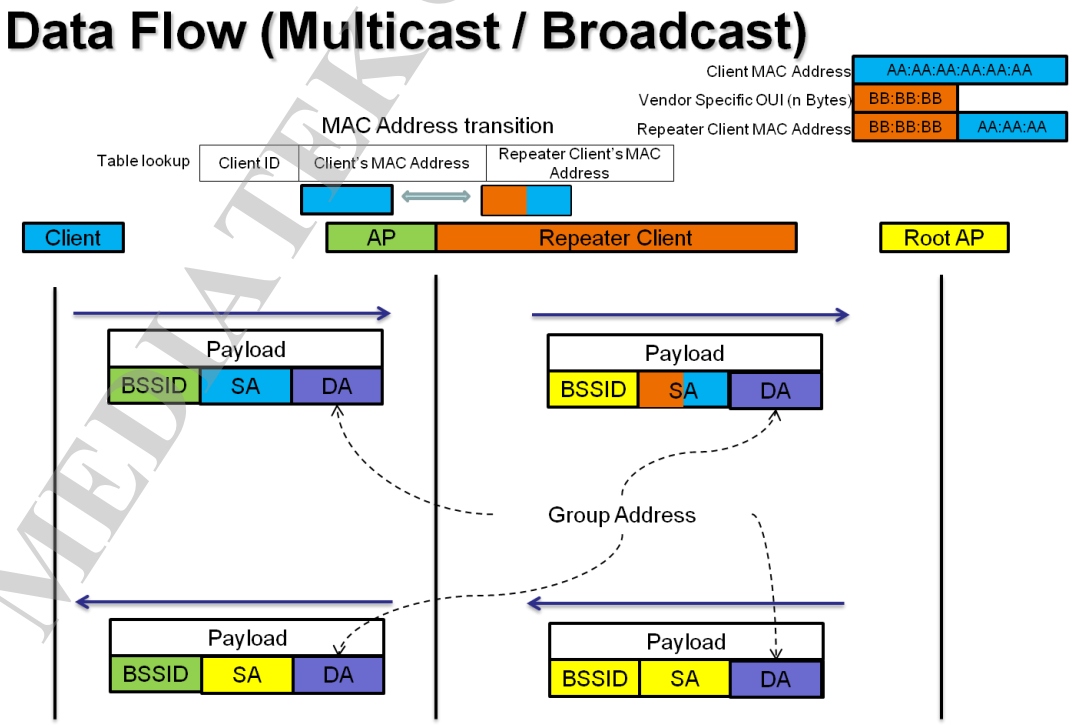


14.4 Data Frame Flow

14.4.1 Unicast



14.4.2 Multicast / Broadcast



15 PMF

PMF stands for Protected Management Frame and IEEE 802.11w is the PMF standard. Its objective is to increase the security of 802.11 management frames.

15.1 PMF iwpriv command

15.1.1 PMFMFPC

Description: Enable or disable Protection Management Frame Capable
Value:

iwpriv ra0 set PMFMFPC=1

0: disable
1: enable

15.1.2 PMFMFPR

Description: Enable or disable Protection Management Frame Required
Value:

iwpriv ra0 set PMFMFPR=1

0: disable
1: enable

15.1.3 PMFSHA256

Description: Enable or disable use SHA256 for Encryption
Value:

iwpriv ra0 set PMFSHA256=1

0: disable
1: enable

Note: SHA stands for Secure Hash Algorithm

15.2 Parameters in RT2860AP.dat

15.2.1 PMFMFPC

Description: Disable or enable Protection Management Frame Capable
Value:

PMFMFPC=0

0: Disable
1: Enable

15.2.2 PMFMFPR

Description: Disable or enable Protection Management Frame Required
Value:

PMFMFPR=0

0: Disable
1: Enable

15.2.3 PMFSHA256

Description: Disable or enable use SHA256 for Encryption
Value:

PMFSHA256=0

0: Disable
1: Enable

15.3 Wi-Fi PMF Testing Note

15.3.1 DUT Requirement

PMF is a mandatory testing item to TGac but an optional one to TGn. Actually you can refer to the following table for the correct combination in a dual band router.

Combination	11ac 5GHz	11n 5GHz	11n 2.4GHz
Correct	PMF supported	PMF supported	PMF supported
Not acceptable	PMF supported	PMF supported	PMF Not Available
Correct	PMF supported	PMF Not Available	PMF Not Available
Not acceptable	PMF supported	PMF Not Available	PMF supported

15.3.2 PMF Test Section 4.3.3.3

Verification of CCMP to protect transmitted unicast deauthentication/disassociation frames

- iwpriv ra0 set PMFMFPC=1
- iwpriv ra0 set PMFMFPR=0
- iwpriv ra0 set PMFSHA256=0
- iwpriv ra0 set SSID=PMF-4.3.3.3
- iwpriv ra0 set **DisConnectSta=00:0C:43:35:93:00**

15.3.3 PMF Test Section 4.4

Verify use of BIP (Broadcast Integrity Protocol) to protect broadcast management frames

- iwpriv ra0 set PMFMFPC=1
- iwpriv ra0 set PMFMFPR=0
- iwpriv ra0 set PMFSHA256=0
- iwpriv ra0 set SSID=PMF-4.4
- iwpriv ra0 set **DisConnectAllSta=2**

16 MBSSID

The Multiple BSSID (MBSSID) function is a feature providing additional virtual WLANs which look like real WLANs to users. Its common application is to create one Main and several Guest Networks simultaneously. You may setup each BSSID with different configuration.

16.1 How to Setup

Please turn on MBSS_SUPPORT in driver config.

We also suggest turn on NEW_MBSSID_MODE which changes how the driver creates extended MAC addresses for these virtual BSSID.

16.2 Parameter in RT2860AP.dat

16.2.1 BssidNum

Description: Multiple BSSID number configuration
Value:

BssidNum=1

1/2/4/8/16

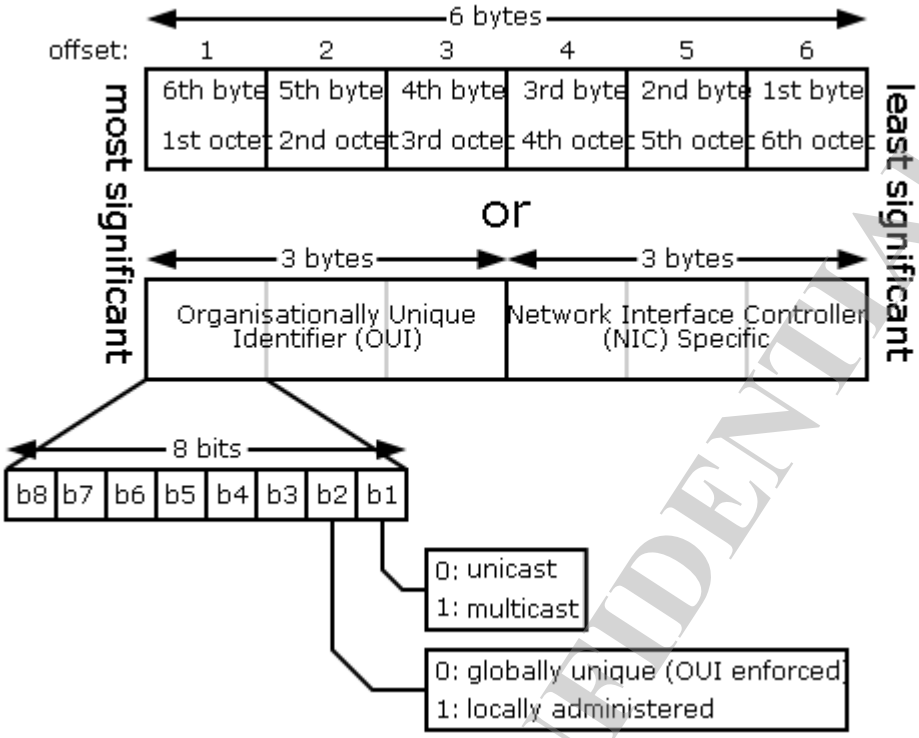
Note:

1. It depends on MBSS_SUPPORT
2. It should be placed before other configuration in the profile
3. 16-BSSID is supported only in new products

16.3 Important Note

16.3.1 MAC Address Format

The following MAC address format figure is from http://en.wikipedia.org/wiki/MAC_address and all subsequent discussion is based on this format.



16.3.2 Old MBSSID Mode

As to main BSSID, **the 1st byte** of its MAC address should be:

- Multiple of 2 for 2-BSSID
- Multiple of 4 for 4-BSSID
- Multiple of 8 for 8-BSSID

Taking BssidNum=4 for example, address extension would be done on 1st byte.

- ra0: 00:0c:43:00:00:00 — 00 is multiple of 4
- ra1: 00:0c:43:00:00:01 — 01 comes from (1st byte 0x00) + 1
- ra2: 00:0c:43:00:00:02 — 02 comes from (1st byte 0x00) + 2
- ra3: 00:0c:43:00:00:03 — 03 comes from (1st byte 0x00) + 3

Other possible address extension:

Multiple of 4	1st BSSID	2nd BSSID	3rd BSSID	4th BSSID
0x00	AA-BB-CC-DD-EE-F0	AA-BB-CC-DD-EE-F1	AA-BB-CC-DD-EE-F2	AA-BB-CC-DD-EE-F3
0x04	AA-BB-CC-DD-EE-F4	AA-BB-CC-DD-EE-F5	AA-BB-CC-DD-EE-F6	AA-BB-CC-DD-EE-F7
0x08	AA-BB-CC-DD-EE-F8	AA-BB-CC-DD-EE-F9	AA-BB-CC-DD-EE-FA	AA-BB-CC-DD-EE-FB
0x0C	AA-BB-CC-DD-EE-FC	AA-BB-CC-DD-EE-FD	AA-BB-CC-DD-EE-FE	AA-BB-CC-DD-EE-FF

Please be noted that all these MAC addresses should be reserved because they are global MAC addresses.

16.3.3 New MBSSID Mode

Since there is MAC address reservation problem in the old MBSSID mode, we provide the new MBSSID mode which will utilize b2 of 6th byte of a virtual MAC address to claim it as locally administered. Address extension would be done on 6th byte.

As to main BSSID, **the b[4:2] of 6th byte** of its MAC address should be:

- Multiple of 2 for 2-BSSID
- Multiple of 4 for 4-BSSID
- Multiple of 8 for 8-BSSID

Taking BssidNum=4 for example:

- ra0: 0**0**:0c:43:00:00:00
- ra1: 0**2**:0c:43:00:00:00 — 0**2** comes from (6th byte 0x00 | b'000000**1**0)
- ra2: 0**6**:0c:43:00:00:00 — 0**6** comes from (6th byte 0x00 | b'000001**1**0)
- ra3: 0**a**:0c:43:00:00:00 — 0**a** comes from (6th byte 0x00 | b'000010**1**0)

16.3.4 Enhanced New MBSSID Mode

The enhanced new MBSSID mode removes the restriction of using the 6th byte since OUI (Consists of 6th, 5th, 4th bytes) is not controllable. Local Administration bit would be turned on and address extension would be done on 3rd byte. This is supported only in new products.

Taking BssidNum=4 for example:

- ra0: 00:0c:43:0**0**:00:00 — 00 is multiple of 4
- ra1: 0**2**:0c:43:0**0**:00:00 — 00 comes from (3rd byte 0x00) + 0
- ra2: 0**2**:0c:43:0**1**:00:00 — 01 comes from (3rd byte 0x00) + 1
- ra3: 0**2**:0c:43:0**2**:00:00 — 02 comes from (3rd byte 0x00) + 2

16.4 Configuration

BssidNum can be configured only through profile and you must restart the interface to make it to work. Other parameters can be configured dynamically through iwpriv command per interface. MBSSID-supported parameters are SSID, AuthMode, EncrypType, WPAPSK, etc.

16.4.1 Example

BssidNum=4
SSID=SSID_A;SSID_B;SSID_C;SSID_D
AuthMode=OPEN;SHARED;WPAPSK;WPA2PSK
EncrypType=NONE;WEP;TKIP;AES

17 Concurrent A+G Settings

Below table is brief example for two interfaces.
For example, Linux HotPlug system found new device would create one driver instance (create new space for driver image) for new device to hold private information (memory consumed).

Interface Bring Up Sequence						
NIC#	Sequence	Normal	WDS			
			1	2	3	4
Two	ifconfig ra0 up	ra0	wds0	wds1	wds2	wds3
	ifconfig ra1 up	ra1	wds4	wds5	wds6	wds7

NIC#	Sequence	Normal	MBSSID			WDS			
						1	2	3	4
Two	ifconfig ra0 up	ra0	ra2	ra3	ra4	wds0	wds1	wds2	wds3
	ifconfig ra1 up	ra1	ra5	ra6	ra7	wds4	wds5	wds6	wds7

WDS IS A VIRTUAL INTERFACE WITHOUT IOCTL FUNCTIONALITY.

18 SNMP MIBs Support List

18.1 RT2860AP Supported v.s. IEEE802dot11-MIB

IEEE802dot11-MIB	Access	Support	OID	RT2860AP.dat
ieee802dot11				
dot11smt		-		
dot11StationConfigTable	not-accessible	-		
dot11StationConfigEntry	not-accessible	-		
dot11StationID	read-write	Y	OID_802_3_CURRENT_ADDRESS	N
dot11MediumOccupancyLimit	read-write	N		N
dot11CFPollable	read-only	N		N
dot11CFPPeriod	read-write	N		N
dot11CFPMaxDuration	read-write	N		N
dot11AuthenticationResponseTimeOut	read-write	N		N
dot11PrivacyOptionImplemented	read-only	Y	RT_OID_802_11_PRIVACYOPTIONIMPLEMENTED	N
dot11PowerManagementMode	read-write	Y	RT_OID_802_11_POWERMANAGEMENTMODE	N
dot11DesiredSSID	read-write	N		N
dot11DesiredBSSType	read-write	N		N
dot11OperationalRateSet	read-write	N		N
dot11BeaconPeriod	read-write	N		N
dot11DTIMPeriod	read-write	N		N
dot11AssociationResponseTimeOut	read-write	N		N
dot11DisassociateReason	read-only	N		N
dot11DisassociateStation	read-only	N		N
dot11DeauthenticateReason	read-only	N		N
dot11DeauthenticateStation	read-only	N		N
dot11AuthenticateFailStatus	read-only	N		N
dot11AuthenticateFailStation	read-only	N		N
dot11AuthenticationAlgorithmsTable	not-accessible	-		-
dot11AuthenticationAlgorithmsEntry	not-	-		-

	accessible			
dot11AuthenticationAlgorithmsIndex	not-accessible	Y		N
dot11AuthenticationAlgorithm	read-only	Y		N
dot11AuthenticationAlgorithmsEnable	read-write	Y		N
dot11WEPDefaultKeysTable	not-accessible	-		-
dot11WEPDefaultKeysEntry	not-accessible	-		-
dot11WEPDefaultKeyIndex	not-accessible	Y		N
dot11WEPDefaultKeyValue	read-write	Y	OID_802_11_WEPDEFAULTKEYVALUE	Y
dot11WEPKeyMappingsTable	not-accessible	-		-
dot11WEPKeyMappingsEntry	not-accessible	-		-
dot11WEPKeyMappingIndex	not-accessible	N		N
dot11WEPKeyMappingAddress	read-create	N		N
dot11WEPKeyMappingWEPOn	read-create	N		N
dot11WEPKeyMappingValue	read-create	N		N
dot11WEPKeyMappingStatus	read-create	N		N
dot11PrivacyTable	not-accessible	-		
dot11PrivacyEntry	not-accessible	-		
dot11PrivacyInvoked	read-write	Y		N
dot11WEPDefaultKeyID	read-write	Y	OID_802_11_WEPDEFAULTKEYID	Y
dot11WEPKeyMappingLength	read-write	Y	RT_OID_802_11_WEPKEYMAPPINGLENGTH	N
dot11ExcludeUnencrypted	read-write	N		N
dot11WEPICVErrorCount	read-only	N		N
dot11WEPExcludedCount	read-only	N		N
dot11SMTnotification	-	-		
dot11Disassociate	-	N		N
dot11Deauthenticate	-	N		N
dot11AuthenticateFail	-	N		N
dot11mac				
dot11OperationTable	not-access	-		

	ible			
dot11OperationEntry	not-accessible	-		
dot11MACAddress	read-only	Y	RT_OID_802_11_MAC_ADDRESS	N
dot11RTSThreshold	read-write	Y	OID_802_11_RTS_THRESHOLD	Y
dot11ShortRetryLimit	read-write	Y	OID_802_11_SHORTRETRYLIMIT	N
dot11LongRetryLimit	read-write	Y	OID_802_11_LONGRETRYLIMIT	N
dot11FragmentationThreshold	read-write	Y	OID_802_11_FRAGMENTATION_THRESHOLD	Y
dot11MaxTransmitMSDULifetime	read-write	N		N
dot11MaxReceiveLifetime	read-write	N		N
dot11ManufacturerID	read-only	Y	RT_OID_802_11_MANUFACTUREID	N
dot11ProductID	read-only	Y	RT_OID_802_11_PRODUCTID	N
dot11CountersTable	not-accessible	-		
dot11CountersEntry	not-accessible	-		
dot11TransmittedFragmentCount	read-only	Y	OID_802_11_STATISTICS	N
dot11MulticastTransmittedFrameCount	read-only	Y	OID_802_11_STATISTICS	N
dot11FailedCount	read-only	Y	OID_802_11_STATISTICS	N
dot11RetryCount	read-only	Y	OID_802_11_STATISTICS	N
dot11MultipleRetryCount	read-only	Y	OID_802_11_STATISTICS	N
dot11FrameDuplicateCount	read-only	Y	OID_802_11_STATISTICS	N
dot11RTSSuccessCount	read-only	Y	OID_802_11_STATISTICS	N
dot11RTSFailureCount	read-only	Y	OID_802_11_STATISTICS	N
dot11ACKFailureCount	read-only	Y	OID_802_11_STATISTICS	N
dot11ReceivedFragmentCount	read-only	Y	OID_802_11_STATISTICS	N
dot11MulticastReceivedFrameCount	read-only	Y	OID_802_11_STATISTICS	N
dot11FCSErrorCount	read-only	Y	OID_802_11_STATISTICS	N
dot11TransmittedFrameCount	read-only	N		N
dot11WEPUndecryptableCount	read-only	N		N
dot11GroupAddressesTable	not-accessible	-		-
dot11GroupAddressesEntry	not-accessible	-		-

dot11GroupAddressesIndex	not-accessible	N		N
dot11Address	read-create	N		N
dot11GroupAddressesStatus	read-create	N		N
dot11res				
dot11resAttribute				
dot11ResourceTypeIDName	read-only	-		
dot11ResourceInfoTable	not-accessible	-		
dot11ResourceInfoEntry	not-accessible	-		
dot11manufacturerOUI	read-only	Y	RT_OID_802_11_MANUFACTUREROUI	N
dot11manufacturerName	read-only	Y	RT_OID_802_11_MANUFACTURERNAME	N
dot11manufacturerProductName	read-only	Y	RT_OID_DEVICE_NAME	N
dot11manufacturerProductVersion	read-only	Y	RT_OID_VERSION_INFO	N
dot11phy				
dot11PhyOperationTable	not-accessible	-		
dot11PhyOperationEntry	not-accessible	-		
dot11PHYType	read-only	Y	RT_OID_802_11_PHY_MODE	N
dot11CurrentRegDomain	read-write	Y		Y
dot11TempType	read-only	N		N
dot11PhyAntennaTable	not-accessible	-		
dot11PhyAntennaEntry	not-accessible	-		
dot11CurrentTxAntenna	read-write	Y	OID_802_11_TX_ANTENNA_SELECTED	N
dot11DiversitySupport	read-only	Y	OID_802_11_RX_ANTENNA_SELECTED	N
dot11CurrentRxAntenna	read-write	Y	OID_802_11_RX_ANTENNA_SELECTED	N
dot11PhyTxPowerTable	not-accessible	-		
dot11PhyTxPowerEntry	not-accessible	-		
dot11NumberSupportedPowerLevels	read-only	N		N
dot11TxPowerLevel1	read-only	N		N
dot11TxPowerLevel2	read-only	N		N

dot11TxPowerLevel3	read-only	N		N
dot11TxPowerLevel4	read-only	N		N
dot11TxPowerLevel5	read-only	N		N
dot11TxPowerLevel6	read-only	N		N
dot11TxPowerLevel7	read-only	N		N
dot11TxPowerLevel8	read-only	N		N
dot11CurrentTxPowerLevel	read-write	N		N
dot11PhyFHSSTable	not-accessible	-		
dot11PhyFHSSEntry	not-accessible	-		
dot11HopTime	read-only	N		N
dot11CurrentChannelNumber	read-write	N		N
dot11MaxDwellTime	read-only	N		N
dot11CurrentDwellTime	read-write	N		N
dot11CurrentSet	read-write	N		N
dot11CurrentPattern	read-write	N		N
dot11CurrentIndex	read-write	N		N
dot11PhyDSSSTable	not-accessible	-		
dot11PhyDSSSEntry	not-accessible	-		
dot11CurrentChannel	read-write	Y	OID_802_11_CURRENTCHANNEL	Y
dot11CCAModeSupported	read-only	N		N
dot11CurrentCCAMode	read-write	N		N
dot11EDThreshold	read-write	N		N
dot11PhyIRTable	not-accessible	-		
dot11PhyIREntry	not-accessible	-		
dot11CCAWatchdogTimerMax	read-write	N		N
dot11CCAWatchdogCountMax	read-write	N		N
dot11CCAWatchdogTimerMin	read-write	N		N
dot11CCAWatchdogCountMin	read-write	N		N

dot11RegDomainsSupportedTable	not-accessible	-		
dot11RegDomainsSupportEntry	not-accessible	-		
dot11RegDomainsSupportIndex	not-accessible	Y		N
dot11RegDomainsSupportValue	read-only	Y		N
dot11AntennasListTable	not-accessible	-		
dot11AntennasListEntry	not-accessible	-		
dot11AntennaListIndex	not-accessible	Y		N
dot11SupportedTxAntenna	read-write	Y	OID_802_11_TX_ANTENNA_SELECTED	N
dot11SupportedRxAntenna	read-write	Y	OID_802_11_RX_ANTENNA_SELECTED	N
dot11DiversitySelectionRx	read-write	Y	OID_802_11_RX_ANTENNA_SELECTED	N
dot11SupportedDataRatesTxTable	not-accessible	-		
dot11SupportedDataRatesTxEntry	not-accessible	-		
dot11SupportedDataRatesTxIndex	not-accessible	Y		N
dot11SupportedDataRatesTxValue	read-only	Y	OID_802_11_DESIRED_RATES	N
dot11SupportedDataRatesRxTable	not-accessible	-		
dot11SupportedDataRatesRxEntry	not-accessible	-		
dot11SupportedDataRatesRxIndex	not-accessible	Y	OID_802_11_DESIRED_RATES	
dot11SupportedDataRatesRxValue	read-only	Y		
dot11PhyOFDMTable	not-accessible	-		
dot11PhyOFDMEntry	not-accessible	-		
dot11CurrentFrequency	read-write	N	OID_802_11_CURRENTCHANNEL	Y
dot11TIThreshold	read-write	N		N
dot11FrequencyBandsSupported	read-only	N		N

18.2 RALINK OID for SNMP MIB

RALINK OID for SNMP		
Value	Name	Structure
0x010B	OID_802_11_NUMBER_OF_ANTENNAS	USHORT numant;
0x010C	OID_802_11_RX_ANTENNA_SELECTED	USHORT whichant;
0x010D	OID_802_11_TX_ANTENNA_SELECTED	USHORT whichant;
0x050C	RT_OID_802_11_PHY_MODE	ULONG linfo;
0x050E	OID_802_11_DESIRED_RATES	typedef UCHAR NDIS_802_11_RATES[NDIS_802_11_LENGTH_RATES]; #define NDIS_802_11_LENGTH_RATES 8
0x0514	OID_802_11_RTS_THRESHOLD	ULONGlinfo;
0x0515	OID_802_11_FRAGMENTATION_THRESHOLD	ULONGlinfo;
0x0607	RT_OID_DEVICE_NAME	char name[128];
0x0608	RT_OID_VERSION_INFO	typedef struct PACKED _RT_VERSION_INFO{ UCHAR DriverVersionW; UCHAR DriverVersionX; UCHAR DriverVersionY; UCHAR DriverVersionZ; UINT DriverBuildYear; UINT DriverBuildMonth; UINT DriverBuildDay; } RT_VERSION_INFO, *PRT_VERSION_INFO;
0x060A	OID_802_3_CURRENT_ADDRESS	char addr[128];

0x060E	OID_802_11_STATISTICS	typedef struct _NDIS_802_11_STATISTICS
		{
		ULONG Length; // Length of structure
		ULONG TransmittedFragmentCount;
		ULONG MulticastTransmittedFrameCount;
		ULONG FailedCount;
		ULONG RetryCount;
		ULONG MultipleRetryCount;
		ULONG RTSSuccessCount;
		ULONG RTSFailureCount;
		ULONG ACKFailureCount;
		ULONG FrameDuplicateCount;
		ULONG ReceivedFragmentCount;

		ULONG MulticastReceivedFrameCount; ULONG FCSErrorCount; } NDIS_802_11_STATISTICS, PNDIS_802_11_STATISTICS;
0x0700	RT_OID_802_11_MANUFACTUREROUI	char oui[128];
0x0701	RT_OID_802_11_MANUFACTURERNAME	char name[128];
0x0702	RT_OID_802_11_RESOURCETYPEIDNAME	char name[128];
0x0703	RT_OID_802_11_PRIVACYOPTIONIMPLEMENTED	ULONG linfo;
0x0704	RT_OID_802_11_POWERMANAGEMENTMODE	ULONG linfo;
0x0705	OID_802_11_WEPDEFAULTKEYVALUE	typedef struct _DefaultKeyIdxValue { UCHARKeyIdx; UCHARValue[16]; }DefaultKeyIdxValue;
0x0706	OID_802_11_WEPDEFAULTKEYID	UCHARkeyid;
0x0707	RT_OID_802_11_WEPKEYMAPPINGLENGTH	UCHAR len;
0x0708	OID_802_11_SHORTRETRYLIMIT	ULONGlinfo;
0x0709	OID_802_11_LONGRETRYLIMIT	ULONGlinfo;
0x0710	RT_OID_802_11_PRODUCTID	char id[128];
0x0711	RT_OID_802_11_MANUFACTUREID	char id[128];
0x0712	OID_802_11_CURRENTCHANNEL	UCHAR channel
0x0713	RT_OID_802_11_MAC_ADDRESS	char macaddress[128]

19 IOCTL I/O Control Interface

19.1 Parameters for iwconfig's IOCTL

Access	Description	ID	Parameters
Get	BSSID, MAC Address	SIOCGIFHWADDR	wrq->u.name, (length = 6)
	WLAN Name	SIOCGIWNAME	wrq->u.name = "RT2800 SoftAP", length = strlen(wrq->u.name)

	SSID	SIOCGIWESSID	struct iw_point *erq = &wrq->u.essid;
			erq->flags=1;
			erq->length = pAd->PortCfg.MBSSID[pAd->IoctlIF].SsidLen;
			if(erq->pointer)
			{
			if(copy_to_user(erq->pointer,
			pAd->PortCfg.MBSSID[pAd->IoctlIF].Ssid,
			erq->length))
			{
			Status = -EFAULT;
			break;
			}
			}

	Channel / Frequency (Hz)	SIOCGIWFREQ	wrq->u.freq.m = pAd->PortCfg.Channel;
			wrq->u.freq.e = 0;
			wrq->u.freq.i = 0;

	Bit Rate (bps)	SIOCGIWRATE	wrq->u.bitrate.value = RateIdTo500Kbps[pAd->PortCfg.MBSSID[pAd->IoctlIF].TxRate] * 500000; wrq->u.bitrate.disabled = 0;
	AP's MAC address	SIOCGIWAP	wrq->u.ap_addr.sa_family = ARPHRD_ETHER; memcpy(wrq->u.ap_addr. sa_data, &pAd->PortCfg.MBSSID[pAd->IoctlIF].Bssid, ETH_ALEN);
	Operation Mode	SIOCGIWMODE	wrq->u.mode = IW_MODE_INFRA;
	Range of Parameters	SIOCGIWRANGE	range.we_version_compiled = WIRELESS_EXT; range.we_version_source = 14;

	Scanning Results	SIOCGIWSCAN	typedef struct _NDIS_802_11_SITE_SURVEY_TABLE
			{
			LONG Channel;
			LONG Rssi;
			UCHAR Ssid[33];
			UCHAR Bssid[18];
			UCHAR EncrypT[8];
			} NDIS_802_11_SITE_SURVEY_TABLE, *PNDIS_802_11_SITE_SURVEY_TABLE;
			
			wrq->u.data.length = N* sizeof(NDIS_802_11_SITE_SURVEY_TABLE);
			copy_to_user(wrq->u.data.pointer, site_survey_table, wrq->u.data.length);

	Client	SIOCGIWAPLIST	typedef struct _NDIS_802_11_STATION_TABLE

	Association List		{
			UCHAR MacAddr[18];
			ULONG Aid;
			ULONG PsMode;
			ULONG LastDataPacketTime;
			ULONG RxByteCount;
			ULONG TxByteCount;
			ULONG CurrTxRate;
			ULONG LastTxRate;
			} NDIS_802_11_STATION_TABLE, *PNDIS_802_11_STATION_TABLE;
			
			wrq->u.data.length = i * sizeof(NDIS_802_11_STATION_TABLE);
			copy_to_user(wrq->u.data.pointer, sta_list_table, wrq->u.data.length);

Set	Trigger Scanning	SIOCSIWSCAN	ApSiteSurvey(pAd);

19.2 Parameters for iwpriv's IOCTL

Please refer section 4 and 5 to have iwpriv parameters and values.

Parameters:

```
int     socket_id;
char    name[25];                   // interface name
char    data[255];                  // command string
struct  iwreq wrq;
```

Default setting:

```
wrq.ifr_name = name = "ra0";    // interface name
wrq.u.data.pointer = data;      // data buffer of command string
wrq.u.data.length = strlen(data);   // length of command string
wrq.u.data.flags = 0;
```

19.2.1 Iwpriv Set DATA

THESE PARAMETERS ARE THE SAME AS IWPRIV

Command and IOCTL Function		
Set Data		
Function Type	Command	IOCTL

RTPRIV_IOCTL_SET	iwpriv ra0 set SSID=RT2800AP	sprintf(name, "ra0");
		strcpy(data, "SSID=RT2800AP");
		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);

19.2.2 Iwpriv Get DATA

THESE PARAMETERS ARE THE SAME AS IWPRIV

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);

RTPRIV_IOCTL_GET_MAC_TABLE	Iwpriv ra0 get_mac_table	sprintf(name, "ra0");
		strcpy(data, "get_mac_table");
		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_GET_MAC_TABLE, &wrq);

RTPRIV_IOCTL_SHOW	Iwpriv ra0 show	sprintf(name, "ra0");
		strcpy(data, "get_mac_table");
		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_SHOW, &wrq);

RTPRIV_IOCTL_WSC_PROFILE	Iwpriv ra0 get_wsc_profile	sprintf(name, "ra0");
		strcpy(data, "get_mac_table");
		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_WSC_PROFILE, &wrq);

RTPRIV_IOCTL_QUERY_BATABLE	Iwpriv ra0 get_ba_table	sprintf(name, "ra0");
		strcpy(data, "get_mac_table");
		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_QUERY_BATABLE, &wrq);

19.2.3 Iwpriv Set Data: BBP, MAC and EEPROM

Command and IOCTL Function		
Set Data: BBP, MAC and EEPROM, Parameters is Same as iwpriv		
Type	Command	IOCTL

RTPRIV_IOCTL_BBP
(Set BBP Register Value)
	Iwpriv ra0 bbp 17=32	sprintf(name, "ra0");
		strcpy(data, " bbp 17=32”);
		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_BBP, &wrq);

RTPRIV_IOCTL_MAC
(Set MAC Register Value)
	Iwpriv ra0 mac 3000=12345678	sprintf(name, "ra0");
		strcpy(data, " mac 3000=12345678”);
		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_MAC, &wrq);

RTPRIV_IOCTL_E2P
(Set EEPROM Value)
	Iwpriv ra0 e2p 40=1234	sprintf(name, "ra0");
		strcpy(data, " e2p 40=1234”);
		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_E2P, &wrq);

19.2.4 Iwpriv Get Data: BBP, MAC and EEPROM

Command and IOCTL Function		
Get Data: BBP, MAC and EEPROM , Parameters is Same as iwpriv		
Type	Command	IOCTL

RTPRIV_IOCTL_BBP
(Get BBP Register Value)
	Iwpriv ra0 bbp 17	sprintf(name, "ra0");
		strcpy(data, " bbp 17”);
		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_BBP, &wrq);

RTPRIV_IOCTL_MAC
(Get MAC Register Value)
	Iwpriv ra0 mac 3000	sprintf(name, "ra0");
		strcpy(data, " mac 3000”);
		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_MAC, &wrq);

RTPRIV_IOCTL_E2P	Iwpriv ra0 e2p 40	sprintf(name, "ra0");

(Get EEPROM Value)		strcpy(data, " e2p 40”);
		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_E2P, &wrq);

19.2.5 Iwpriv Set Raw Data

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

RTPRIV_IOCTL_RADIUS_DATA	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0x55, 100);
	wrq.u.data.length = 100;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ioctl(socket_id, RTPRIV_IOCTL_RADIUS_DATA, &wrq);

RTPRIV_IOCTL_ADD_WPA_KEY	NDIS_802_11_KEY *vp;
	
	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_KEY));
	vp = (NDIS_802_11_KEY *)&data;
	vp->Length = sizeof(NDIS_802_11_KEY);
	memset(vp->addr, 0x11, 6);
	vp->KeyIndex = 2;
	vp->KeyLength = 32;
	memset(vp->KeyMaterial, 0xAA, 32);
	wrq.u.data.length = sizeof(NDIS_802_11_KEY);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ioctl(socket_id, RTPRIV_IOCTL_ADD_WPA_KEY, &wrq);

RTPRIV_IOCTL_ADD_PMKID_CACHE	NDIS_802_11_KEY *vp;
	
	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, sizeof(NDIS_802_11_KEY));
	vp = (NDIS_802_11_KEY *)&data;
	vp->Length = sizeof(NDIS_802_11_KEY);
	memset(vp->addr, 0x11, 6);
	vp->KeyIndex = 2;
	vp->KeyLength = 32;
	memset(vp->KeyMaterial, 0xBB, 32);
	wrq.u.data.length = sizeof(NDIS_802_11_KEY);
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ioctl(socket_id, RTPRIV_IOCTL_ADD_PMKID_CACHE, &wrq);

19.2.6 Set Raw Data with Flags

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

RT_SET_APD_PID	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, 0, 4);
	data[0] = 12;
	wrq.u.data.length = 4;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_SET_APD_PID;
	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_WSC_SET_SELECTED_REGISTRAR	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, decodeStr, decodeLen);
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = decodeLen;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_WSC_SET_SELECTED_REGISTRAR;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

RT_OID_WSC_EAPMSG	sprintf(name, "ra0");
	strcpy(wrq.ifr_name, name);
	memset(data, wscU2KMsg, wscU2KMsgLen);
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = wscU2KMsgLen;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = RT_OID_WSC_EAPMSG;
	ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

19.2.7 Get Raw Data with Flags

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

RT_QUERY_ATE_TXDONE_COUNT	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_QUERY_ATE_TXDONE_COUNT; ioctl(socket_id, RT_PRIV_IOCTL, &wrq);

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

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

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

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

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

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

RT_OID_GET_LLTD_ASSO_TANLE	sprintf(name, "ra0");

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

19.3 Sample user space Applications

```
//==========================================================================
//
// rtuser:
//          1. User space application to demo how to use IOCTL function.
//          2. Most of the IOCTL function is defined as "CHAR" type and return with string message.
//          3. Use sscanf to get the raw data back from string message.
//          4. The command format "parameter=value" is same as iwpriv command format.
//          5. Remember to insert driver module and bring interface up prior execute rtuser.
//                    change folder path to driver "Module"
//                    dos2unix *                    ; in case the files are modified from other OS environment
//                    chmod 644 *
//                    chmod 755 Configure
//                    make config
//                    make
//                    insmod RT2800ap.o
//                    ifconfig ra0 up
//
// Refer Linux/if.h to have
//                    #define ifr_name    ifr_ifrn.ifrn_name                    /* interface name    */
//
// Make:
//                    cc -Wall -ortuser rtuser.c
//
// Run:
//                    ./rtuser
//
//==========================================================================

#include <stdio.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 + 0x01)
#define RTPRIV_IOCTL_SET                                (SIOCIWFIRSTPRIV + 0x02)
#define RTPRIV_IOCTL_BBP                (SIOCIWFIRSTPRIV + 0x03)
#define RTPRIV_IOCTL_MAC                (SIOCIWFIRSTPRIV + 0x05)
#define RTPRIV_IOCTL_E2P                (SIOCIWFIRSTPRIV + 0x07)
#define RTPRIV_IOCTL_STATISTICS                 (SIOCIWFIRSTPRIV + 0x09)
#define RTPRIV_IOCTL_ADD_PMKID_CACHE    (SIOCIWFIRSTPRIV + 0x0A)
#define RTPRIV_IOCTL_RADIUS_DATA                (SIOCIWFIRSTPRIV + 0x0C)
#define RTPRIV_IOCTL_GSITESURVEY                (SIOCIWFIRSTPRIV + 0x0D)
#define RTPRIV_IOCTL_ADD_WPA_KEY                (SIOCIWFIRSTPRIV + 0x0E)
#define RTPRIV_IOCTL_GET_MAC_TABLE              (SIOCIWFIRSTPRIV + 0x0F)


#define OID_GET_SET_TOGGLE                              0x8000

#define RT_QUERY_ATE_TXDONE_COUNT               0x0401
#define RT_QUERY_SIGNAL_CONTEXT                 0x0402
#define RT_SET_APD_PID                                  (OID_GET_SET_TOGGLE + 0x0405)
#define RT_SET_DEL_MAC_ENTRY                    (OID_GET_SET_TOGGLE + 0x0406)

//----------------------------------------------------------

#ifndef     TRUE
#define     TRUE                1
#endif

#ifndef     FALSE
#define     FALSE     0
#endif

#define MAC_ADDR_LEN                            6
#define ETH_LENGTH_OF_ADDRESS           6
#define MAX_LEN_OF_MAC_TABLE                    64

//----------------------------------------------------------

typedef struct _COUNTERS
{
        unsigned long       TxSuccessTotal;;
        unsigned long       TxSuccessWithRetry;
        unsigned long       TxFailWithRetry;
        unsigned long       RtsSuccess;
        unsigned long       RtsFail;
        unsigned long       RxSuccess;
        unsigned long       RxWithCRC;
        unsigned long       RxDropNoBuffer;
        unsigned long       RxDuplicateFrame;
        unsigned long       FalseCCA;
        unsigned long       RssiA;
        unsigned long       RssiB;
}       COUNTERS;
```

PS. User can check with “iwpriv ra0 stat” to make sure the TXRX status is correct when porting the ATE related test program.

//--

```
typedef    struct _SITE_SURVEY
{
        unsigned char                   channel;
        unsigned short                  rssi;
        unsigned char                   ssid[33];
        unsigned char                   bssid[6];
        unsigned char                   security[9];
}       SITE_SURVEY;

//------------------------------------------------------------

typedef union  _MACHTTRANSMIT_SETTING {
        struct      {
        unsigned short                  MCS:7;          // MCS
        unsigned short                  BW:1;                   //channel bandwidth 20MHz or 40 MHz
        unsigned short                  ShortGI:1;
        unsigned short                  STBC:2;         //SPACE
        unsigned short                  rsv:3;
        unsigned short                  MODE:2;         // Use definition MODE_xxx.
        }           field;
        unsigned short                  word;
 } MACHTTRANSMIT_SETTING, *PMACHTTRANSMIT_SETTING;

typedef struct _RT_802_11_MAC_ENTRY {
   unsigned char                        Addr[6];
   unsigned char                        Aid;
   unsigned char                        Psm;                    // 0:PWR_ACTIVE, 1:PWR_SAVE
   unsigned char                        MimoPs;         // 0:MMPS_STATIC, 1:MMPS_DYNAMIC, 3:MMPS_Enabled
   MACHTTRANSMIT_SETTING        TxRate;
} RT_802_11_MAC_ENTRY, *PRT_802_11_MAC_ENTRY;

typedef struct _RT_802_11_MAC_TABLE {
   unsigned long                        Num;
   RT_802_11_MAC_ENTRY Entry[MAX_LEN_OF_MAC_TABLE];
} RT_802_11_MAC_TABLE, *PRT_802_11_MAC_TABLE;

// Key mapping keys require a BSSID
typedef struct _NDIS_802_11_KEY
{
        unsigned long                   Length;         // Length of this structure
        unsigned char                   addr[6];
        unsigned long                   KeyIndex;
        unsigned long                   KeyLength;      // length of key in bytes
        unsigned char                   KeyMaterial[32];  // variable length depending on above field
} NDIS_802_11_KEY, *PNDIS_802_11_KEY;

typedef struct _RT_SIGNAL_STRUC {
        unsigned short                  Sequence;
        unsigned char                   MacAddr[MAC_ADDR_LEN];
        unsigned char                   CurrAPAddr[MAC_ADDR_LEN];
        unsigned char                   Sig;
} RT_SIGNAL_STRUC, *PRT_SIGNAL_STRUC;

//------------------------------------------------------------

COUNTERS                counter;
SITE_SURVEY     SiteSurvey[100];
char                    data[4096];

//============================================================================
```

```
int main( int argc, char ** argv )
{
	char		name[25];
	int		socket_id;
	struct	iwreq wrq;
	int		ret;

	// 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");
		return -1;
	}

	// set interface name as "ra0" ----------
	sprintf(name, "ra0");
	memset(data, 0x00, 255);
//
//example of iwconfig ioctl function ==========
//
	// get wireless name ----------
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = 255;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ret = ioctl(socket_id, SIOCGIWNAME, &wrq);
	if(ret != 0)
	{
		printf("\nrtuser::error::get wireless name\n\n");
		goto rtuser_exit;
	}

	printf("\nrtuser[%s]:%s\n", name, wrq.u.name);
//
//example of iwpriv ioctl function ==========
//
	//WPAPSK, remove "set" string ----------
	memset(data, 0x00, 255);
	strcpy(data, "WPAPSK=11223344");
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = strlen(data)+1;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ret = ioctl(socket_id, RTPRIV_IOCTL_SET, &wrq);
	if(ret != 0)
	{
		printf("\nrtuser::error::set wpapsk\n\n");
		goto rtuser_exit;
	}
	//set e2p, remove "e2p" string ----------
	memset(data, 0x00, 255);
	strcpy(data, "80=1234");
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = strlen(data)+1;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ret = ioctl(socket_id, RTPRIV_IOCTL_E2P, &wrq);
	if(ret != 0)
```

```
{
        printf("\nrtuser::error::set eeprom\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
        int addr, value, p1;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\n[%dx%02X]:%04X  ", &p1, &addr, &value);
        printf("\nSet EEP[0x%02X]:0x%04X\n", addr, value);
}

//get e2p, remove "e2p" string ----------------------------------------
memset(data, 0x00, 255);
strcpy(data, "80");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_E2P, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get eeprom\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
        int addr, value, p1, p2;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\n[%dx%04X]:%dx%X  ", &p1, &addr, &p2, &value);
        printf("\nGet EEP[0x%02X]:0x%04X\n", addr, value);
}

//set mac, remove "mac" string ----------------------------------------
memset(data, 0x00, 255);
strcpy(data, "2b4f=1");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_MAC, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::set mac register\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
        int addr, value, p1;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\n[%dx%08X]:%08X  ", &p1, &addr, &value);
        printf("\nSet MAC[0x%08X]:0x%08X\n", addr, value);
}
```

```
//get mac, remove "mac" string ------------------------------------------
memset(data, 0x00, 255);
strcpy(data, "2b4f");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_MAC, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get mac register\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
        int addr, value, p1;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\n[%dx%08X]:%08X  ", &p1, &addr, &value);
        printf("\nGet MAC[0x%08X]:0x%08X\n", addr, value);
}

//set bbp, remove "bbp" string ------------------------------------------
memset(data, 0x00, 255);
strcpy(data, "17=32");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_BBP, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::set bbp register\n\n");
        goto rtuser_exit;
}

//printf("\n%s\n", wrq.u.data.pointer);
{
        int id, addr, value, p1;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\nR%02d[%dx%02X]:%02X\n", &id, &p1, &addr, &value);
        printf("\nSet BBP R%02d[0x%02X]:0x%02X\n", id, addr, value);
}

//get bbp, remove "bbp" string ------------------------------------------
memset(data, 0x00, 255);
strcpy(data, "17");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = strlen(data)+1;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_BBP, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get bbp register\n\n");
        goto rtuser_exit;
}
```

```
//printf("\n%s\n", wrq.u.data.pointer);
{
        int id, addr, value, p1;

        // string format: "\n[0x%02X]:0x%04X  " ==> "[0x20]:0x0C02"
        sscanf(wrq.u.data.pointer, "\nR%02d[%dx%02X]:%02X  ", &id, &p1, &addr, &value);
        printf("\nGet BBP R%02d[0x%02X]:0x%02X\n", id, addr, value);
}

//get statistics, remove "stat" string ----------------------------------
memset(data, 0x00, 2048);
strcpy(data, "");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = 0;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_STATISTICS, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get statistics\n\n");
        goto rtuser_exit;
}

printf("\n========= Get AP Statistics ==========\n");
{
        int i;
        char *sp = wrq.u.data.pointer;
        unsigned long *cp = (unsigned long *)&counter;

        for (i = 0 ; i < 13 ; i++)
        {
                sp = strstr(sp, "= ");
                sp = sp+2;
                sscanf(sp, "%ul", (unsigned int *)&cp[i]);
        }
  printf("Tx success                                = %u\n", (unsigned int)counter.TxSuccessTotal);
  printf("Tx success without retry                  = %u\n", (unsigned int)

counter.TxSuccessWithoutRetry);
  printf("Tx success after retry          = %u\n", (unsigned int)counter.TxSuccessWithRetry);
  printf("Tx fail to Rcv ACK after retry   = %u\n", (unsigned int)counter.TxFailWithRetry);
  printf("RTS Success Rcv CTS                       = %u\n", (unsigned int)counter.RtsSuccess);
  printf("RTS Fail Rcv CTS                          = %u\n", (unsigned int)counter.RtsFail);
  printf("Rx success                                = %u\n", (unsigned int)counter.RxSuccess);
  printf("Rx with CRC                               = %u\n", (unsigned int)counter.RxWithCRC);
  printf("Rx drop due to out of resource= %u\n", (unsigned int)counter.RxDropNoBuffer);
  printf("Rx duplicate frame                        = %u\n", (unsigned int)counter.RxDuplicateFrame);
  printf("False CCA (one second)                    = %u\n", (unsigned int)counter.FalseCCA);
  printf("RSSI-A                                    = %d\n", (  signed int)counter.RssiA);
  printf("RSSI-B (if available)           = %d\n", (  signed int)counter.RssiB);
}

#if 0
        //set AP to do site survey, remove "set" string -------------------------
        memset(data, 0x00, 255);
        strcpy(data, "SiteSurvey=1");
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = strlen(data)+1;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = 0;
```

```
        ret = ioctl(socket_id, RTPRIV_IOCTL_SET, &wrq);
#endif

        //get AP's site survey, remove "get_site_survey" string ----------
        memset(data, 0x00, 2048);
        strcpy(data, "");
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = 4096;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = 0;
        ret = ioctl(socket_id, RTPRIV_IOCTL_GSITESURVEY, &wrq);
        if(ret != 0)
        {
                printf("\nrtuser::error::get site survey\n\n");
                goto rtuser_exit;
        }

        //printf("\n%s\n", wrq.u.data.pointer);
        printf("\n========== Get Site Survey AP List ==========");
        if(wrq.u.data.length > 0)
        {
                int     i, apCount;
                char *sp, *op;
                int     len = wrq.u.data.length;

                op = sp = wrq.u.data.pointer;
                sp = sp+1+8+8+35+19+8+1;
                i = 0;
                // santy check
                //      1. valid char data
                //      2. rest length is larger than per line length ==> (1+8+8+35+19+8+1)
                while(*sp && ((len - (sp-op)) > (1+8+8+35+19+8)))
                {
                        //if(*sp++ == '\n')
                        //      continue;
                        //printf("\n\nAP Count: %d\n", i);

                        sscanf(sp, "%d", (int *)&SiteSurvey[i].channel);
                        //printf("channel: %d\n", SiteSurvey[i].channel);

                        sp = strstr(sp, "-");
                        sscanf(sp, "-%d", (int *)&SiteSurvey[i].rssi);
                        //printf("rssi: -%d\n", SiteSurvey[i].rssi);

                        sp = sp+8;
                        strncpy((char *)&SiteSurvey[i].ssid, sp, 32);
                        SiteSurvey[i].ssid[32] = '\0';
                        //printf("ssid: %s\n", SiteSurvey[i].ssid);

                        sp = sp+35;
                        sscanf(sp, "%02x:%02x:%02x:%02x:%02x:%02x",
                                        (int *)&SiteSurvey[i].bssid[0], (int *)&SiteSurvey[i].bssid[1],
                                        (int *)&SiteSurvey[i].bssid[2], (int *)&SiteSurvey[i].bssid[3],
                                        (int *)&SiteSurvey[i].bssid[4], (int *)&SiteSurvey[i].bssid[5]);
                        //printf("bssid: %02x:%02x:%02x:%02x:%02x:%02x\n",
                        //              SiteSurvey[i].bssid[0], SiteSurvey[i].bssid[1],
                        //              SiteSurvey[i].bssid[2], SiteSurvey[i].bssid[3],
                        //              SiteSurvey[i].bssid[4], SiteSurvey[i].bssid[5]);

                        sp = sp+19;
```

```
                strncpy((char *)&SiteSurvey[i].security, sp, 8);
                SiteSurvey[i].security[8] = '\0';
                //printf("security: %s\n", SiteSurvey[i].security);

                sp = sp+8+1;
                i = i+1;
        }

        apCount = i;
        printf("\n%-4s%-8s%-8s%-35s%-20s%-8s\n",
                "AP", "Channel", "RSSI", "SSID", "BSSID", "Security");
        for(i = 0 ; i < apCount ; i++)
        {//4+8+8+35+20+8
                printf("%-4d", i+1);
                printf("%-8d", SiteSurvey[i].channel);
                printf("-%-7d", SiteSurvey[i].rssi);
                printf("%-35s", SiteSurvey[i].ssid);
                printf("%02X:%02X:%02X:%02X:%02X:%02X   ",
                                SiteSurvey[i].bssid[0], SiteSurvey[i].bssid[1],
                                SiteSurvey[i].bssid[2], SiteSurvey[i].bssid[3],
                                SiteSurvey[i].bssid[4], SiteSurvey[i].bssid[5]);
                printf("%-8s\n", SiteSurvey[i].security);
        }
}

//get AP's mac table, remove "get_mac_table" string ----------------------
memset(data, 0x00, 2048);
strcpy(data, "");
strcpy(wrq.ifr_name, name);
wrq.u.data.length = 2048;
wrq.u.data.pointer = data;
wrq.u.data.flags = 0;
ret = ioctl(socket_id, RTPRIV_IOCTL_GET_MAC_TABLE, &wrq);
if(ret != 0)
{
        printf("\nrtuser::error::get mac table\n\n");
        goto rtuser_exit;
}

printf("\n========== Get Associated MAC Table ==========");
{
        RT_802_11_MAC_TABLE                     *mp;
        int                             i;

        mp = (RT_802_11_MAC_TABLE *)wrq.u.data.pointer;
        printf("\n%-4s%-20s%-4s%-10s%-10s%-10s\n",
                "AID", "MAC_Address", "PSM", "LastTime", "RxByte", "TxByte");

        for(i = 0 ; i < mp->Num ; i++)
        {
                printf("%-4d", mp->Entry[i].Aid);
                printf("%02X:%02X:%02X:%02X:%02X:%02X   ",
                                mp->Entry[i].Addr[0], mp->Entry[i].Addr[1],
                                mp->Entry[i].Addr[2], mp->Entry[i].Addr[3],
                                mp->Entry[i].Addr[4], mp->Entry[i].Addr[5]);
                printf("%-4d", mp->Entry[i].Psm);
                printf("%-10u", (unsigned int)mp->Entry[i].HSCounter.LastDataPacketTime);
                printf("%-10u", (unsigned int)mp->Entry[i].HSCounter.TotalRxByteCount);
                printf("%-10u", (unsigned int)mp->Entry[i].HSCounter.TotalTxByteCount);
                printf("\n");
```

```
                }
                printf("\n");
        }

//set: raw data
//      RTPRIV_IOCTL_RADIUS_DATA
//      RTPRIV_IOCTL_ADD_WPA_KEY
//      RTPRIV_IOCTL_ADD_PMKID_CACHE

        //set RADIUS Data ------------------------------------------------------
        printf("\nrtuser::set radius data\n\n");
        memset(data, 0x55, 100);
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = 100;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = 0;
        ret = ioctl(socket_id, RTPRIV_IOCTL_RADIUS_DATA, &wrq);
        if(ret != 0)
        {
                printf("\nrtuser::error::set radius data\n\n");
                goto rtuser_exit;
        }

        //add WPA Key ----------------------------------------------------------
        printf("\nrtuser::add wpa key\n\n");
        {
                NDIS_802_11_KEY         *vp;

                memset(data, 0, sizeof(NDIS_802_11_KEY));
                vp = (NDIS_802_11_KEY *)&data;

                vp->Length = sizeof(NDIS_802_11_KEY);
                memset(vp->addr, 0x11, 6);
                vp->KeyIndex = 2;
                vp->KeyLength = 32;
                memset(vp->KeyMaterial, 0xAA, 32);

                strcpy(wrq.ifr_name, name);
                wrq.u.data.length = sizeof(NDIS_802_11_KEY);
                wrq.u.data.pointer = data;
                wrq.u.data.flags = 0;
                ret = ioctl(socket_id, RTPRIV_IOCTL_ADD_WPA_KEY, &wrq);
                if(ret != 0)
                {
                        printf("\nrtuser::error::add wpa key\n\n");
                        goto rtuser_exit;
                }
        }

        //add PMKID_CACHE ------------------------------------------------------
        printf("\nrtuser::add PMKID_CACHE\n\n");
        {
                NDIS_802_11_KEY         *vp;

                memset(data, 0, sizeof(NDIS_802_11_KEY));
                vp = (NDIS_802_11_KEY *)&data;

                vp->Length = sizeof(NDIS_802_11_KEY);
                memset(vp->addr, 0x11, 6);
                vp->KeyIndex = 2;
```

```
                vp->KeyLength = 32;
                memset(vp->KeyMaterial, 0xBB, 32);

                strcpy(wrq.ifr_name, name);
                wrq.u.data.length = sizeof(NDIS_802_11_KEY);
                wrq.u.data.pointer = data;
                wrq.u.data.flags = 0;
                ret = ioctl(socket_id, RTPRIV_IOCTL_ADD_PMKID_CACHE, &wrq);
                if(ret != 0)
                {
                        printf("\nrtuser::error::add PMKID_CACHE\n\n");
                        goto rtuser_exit;
                }
        }

//set: raw data
//      RT_SET_APD_PID
//      RT_SET_DEL_MAC_ENTRY

        //set APD_PID ----------------------------------------------------------
        printf("\nrtuser::set APD_PID\n\n");
        memset(data, 0, 4);
        data[0] = 12;
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = 4;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = RT_SET_APD_PID;
        ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
        if(ret != 0)
        {
                printf("\nrtuser::error::set APD_PID\n\n");
                goto rtuser_exit;
        }

        //set DEL_MAC_ENTRY ----------------------------------------------------
        printf("\nrtuser::set DEL_MAC_ENTRY\n\n");
        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;
        ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
        if(ret != 0)
        {
                printf("\nrtuser::error::set DEL_MAC_ENTRY\n\n");
                goto rtuser_exit;
        }

//get: raw data
//      RT_QUERY_ATE_TXDONE_COUNT
//      RT_QUERY_SIGNAL_CONTEXT

        //get ATE_TXDONE_COUNT -------------------------------------------------
        printf("\nrtuser::get ATE_TXDONE_COUNT\n\n");
        memset(data, 0, 4);
        strcpy(wrq.ifr_name, name);
        wrq.u.data.length = 4;
        wrq.u.data.pointer = data;
        wrq.u.data.flags = RT_QUERY_ATE_TXDONE_COUNT;
        ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
```

```
	if(ret != 0)
	{
		printf("\nrtuser::error::get ATE_TXDONE_COUNT\n\n");
		goto rtuser_exit;
	}
	printf("\nATE_TXDONE_COUNT:: %08lx\n\n", (unsigned long)*wrq.u.data.pointer);

	//get SIGNAL_CONTEXT ------------------------------------------------------
	printf("\nrtuser::get SIGNAL_CONTEXT\n\n");
	{
		RT_SIGNAL_STRUC			*sp;

		memset(data, 0, sizeof(RT_SIGNAL_STRUC));
		strcpy(wrq.ifr_name, name);
		wrq.u.data.length = sizeof(RT_SIGNAL_STRUC);
		wrq.u.data.pointer = data;
		wrq.u.data.flags = RT_QUERY_SIGNAL_CONTEXT;
		ret = ioctl(socket_id, RT_PRIV_IOCTL, &wrq);
		if(ret != 0)
		{
			printf("\nrtuser::error::get SIGNAL_CONTEXT\n\n");
			goto rtuser_exit;
		}
		sp = (RT_SIGNAL_STRUC *)wrq.u.data.pointer;
		printf("\n===== SIGNAL_CONTEXT =====\n\n");
		printf("Sequence    = 0x%04x\n", sp->Sequence);
		printf("Mac.Addr    = %02x:%02x:%02x:%02x:%02x:%02x\n",
					sp->MacAddr[0], sp->MacAddr[1],
					sp->MacAddr[2], sp->MacAddr[3],
					sp->MacAddr[4], sp->MacAddr[5]);
		printf("CurrAP.Addr = %02x:%02x:%02x:%02x:%02x:%02x\n",
					sp->CurrAPAddr[0], sp->CurrAPAddr[1],
					sp->CurrAPAddr[2], sp->CurrAPAddr[3],
					sp->CurrAPAddr[4], sp->CurrAPAddr[5]);
		printf("Sig         = %d\n\n", sp->Sig);
	}

	//SSID, remove "set" string -----------------------------------------------
	memset(data, 0x00, 255);
	strcpy(data, "SSID=rtuser");
	strcpy(wrq.ifr_name, name);
	wrq.u.data.length = strlen(data)+1;
	wrq.u.data.pointer = data;
	wrq.u.data.flags = 0;
	ret = ioctl(socket_id, RTPRIV_IOCTL_SET, &wrq);
	if(ret != 0)
	{
		printf("\nrtuser::error::set SSID\n\n");
		goto rtuser_exit;
	}

rtuser_exit:
	if (socket_id >= 0)
		close(socket_id);

	if(ret)
		return ret;
	else
		return 0;
}
```

20 SingleSKU Example file (New feature for MT76XX)

20.1 2.4GHz example SingleSKU.dat

```
# Single SKU Max Power Table
# |CCK 1~11 | |  OFDM 6 ~ 54  | |  HT20 MCS 0 ~ 15  | |  HT40 MCS 0 ~ 15  |
ch1 23 23 23 23 21 21 21 21 21 20 20 16 16 16 16 16 16 16 16 16 16 16 16 16 16
ch2 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch3 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch4 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch5 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch6 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch7 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch8 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch9 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch10 23 23 23 23 22 22 22 22 22 22 20 20 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch11 23 23 23 23 19 19 19 19 19 19 19 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch12 23 23 23 23 19 19 19 19 19 19 19 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch13 23 23 23 23 19 19 19 19 19 19 19 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
ch14 23 23 23 23
```

Note: default SingleSKU profile path in driver is defined “/etc_ro/Wireless/RT2860AP/SingleSKU.dat”

For the detailed usage of SingleSKU in profile support, please refer to the MTK_SingleSKU_InProfile_User_manual.pdf and contact with MTK support windows.

20.2 5GHz example SingleSKU.dat

```
# Single SKU Max Power Table
#  OFDM 6 ~ 54  | |  HT20 MCS 0 ~ 15  | |  HT40 MCS 0 ~ 15  | |  VHT80 MCS 0 ~ 9  |
ch36 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
ch38 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
ch40 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
ch42 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13 13 13
ch44 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13
ch46 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13
ch48 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13
ch52 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13
ch54 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13
ch56 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13
ch58 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13
ch60 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
ch62 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
ch64 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
ch100 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
```

ch102 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
13 13 13 13 13 13
ch104 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
13 13 13 13 13 13
ch106 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13
13 13 13 13 13 13
ch108 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13
13 13 13 13 13 13
ch110 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13
13 13 13 13 13 13
ch112 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13
13 13 13 13 13 13
ch116 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch118 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch120 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch122 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch124 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch126 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch128 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch132 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 18 18 18 18
18 18 18 18 18 18
ch134 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 18 18 18 18
18 18 18 18 18 18
ch136 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 18 18 18 18
18 18 18 18 18 18
ch140 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch149 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch151 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch153 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch155 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch157 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch159 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch161 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch165 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch169 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
ch173 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18
End of Single SKU Table

Note: default SingleSKU profile path in driver is defined "/etc_ro/Wireless/RT2860AP/SingleSKU.dat"

For the detailed usage of SingleSKU in profile support, please refer to the MTK_SingleSKU_InProfile_User_manual.pdf and contact with MTK support windows.

21 How to Fix Data Rate

21.1 802.11n Data Rate Table

MCS index	Spatial streams	Modulation type	Coding rate	Data rate (Mbit/s)			
				20 MHz channel		40 MHz channel	
				800 ns GI	400 ns GI	800 ns GI	400 ns GI
0	1	BPSK	1/2	6.50	7.20	13.50	15.00
1	1	QPSK	1/2	13.00	14.40	27.00	30.00
2	1	QPSK	3/4	19.50	21.70	40.50	45.00
3	1	16-QAM	1/2	26.00	28.90	54.00	60.00
4	1	16-QAM	3/4	39.00	43.30	81.00	90.00
5	1	64-QAM	2/3	52.00	57.80	108.00	120.00
6	1	64-QAM	3/4	58.50	65.00	121.50	135.00
7	1	64-QAM	5/6	65.00	72.20	135.00	150.00
8	2	BPSK	1/2	13.00	14.40	27.00	30.00
9	2	QPSK	1/2	26.00	28.90	54.00	60.00
10	2	QPSK	3/4	39.00	43.30	81.00	90.00
11	2	16-QAM	1/2	52.00	57.80	108.00	120.00
12	2	16-QAM	3/4	78.00	86.70	162.00	180.00
13	2	64-QAM	2/3	104.00	115.60	216.00	240.00
14	2	64-QAM	3/4	117.00	130.00	243.00	270.00
15	2	64-QAM	5/6	130.00	144.40	270.00	300.00
16	3	BPSK	1/2	19.50	21.70	40.50	45.00
17	3	QPSK	1/2	39.00	43.30	81.00	90.00
18	3	QPSK	3/4	58.50	65.00	121.50	135.00
19	3	16-QAM	1/2	78.00	86.70	162.00	180.00
20	3	16-QAM	3/4	117.00	130.00	243.00	270.00
21	3	64-QAM	2/3	156.00	173.30	324.00	360.00
22	3	64-QAM	3/4	175.50	195.00	364.50	405.00
23	3	64-QAM	5/6	195.00	216.70	405.00	450.00
24	4	BPSK	1/2	26.00	28.80	54.00	60.00
25	4	QPSK	1/2	52.00	57.60	108.00	120.00
26	4	QPSK	3/4	78.00	86.80	162.00	180.00
27	4	16-QAM	1/2	104.00	115.60	216.00	240.00
28	4	16-QAM	3/4	156.00	173.20	324.00	360.00
29	4	64-QAM	2/3	208.00	231.20	432.00	480.00
30	4	64-QAM	3/4	234.00	260.00	486.00	540.00
31	4	64-QAM	5/6	260.00	288.80	540.00	600.00

21.2 2.4g

21.2.1 B only

```
iwpriv ra0 set FixedTxMode=CCK
iwpriv ra0 set WirelessMode=1          // 11b only
iwpriv ra0 set BasicRate=3             // 1, 2 Mbps
iwpriv ra0 set HtMcs=0                 // Please check Note-11b
iwpriv ra0 set SSID=11B_only_AP        // Restart AP
```

Note-11b:

HtMcs	**0**	**1**	**2**	**3**
Rate	1 Mbps	2 Mbps	5.5 Mbps	11 Mbps

21.2.2 G only

```
iwpriv ra0 set FixedTxMode=OFDM
iwpriv ra0 set WirelessMode=4          // 11g only
iwpriv ra0 set BasicRate=351           // 1, 2, 5.5, 11, 6, 12, 24 Mbps
iwpriv ra0 set HtMcs=0                 // Please check Note-11g
iwpriv ra0 set SSID=11G_only_AP        // Restart AP
```

Note-11g:

HtMcs	0	1	2	3	4	5	6	7
Rate	6 Mbps	9 Mbps	12 Mbps	18 Mbps	24 Mbps	36 Mbps	48 Mbps	54 Mbps

21.2.3 N only

```
iwpriv ra0 set FixedTxMode=HT
iwpriv ra0 set WirelessMode=6          // 2.4g 11n only
iwpriv ra0 set BasicRate=15            // 1, 2, 5.5, 11 Mbps
iwpriv ra0 set HtMcs=0                 // Please check Note-11n
iwpriv ra0 set HtGi=0
iwpriv ra0 set HtBw=0
iwpriv ra0 set SSID=11GN_only_AP       // Restart AP
```

Note-11n:
HtMcs=<0-15> + HtGi=<0-1> + HtBw=<0-1>
Please check all possible combination of above set in Section 21.1.

21.2.4 B/G/N mixed

```
iwpriv ra0 set FixedTxMode=HT
iwpriv ra0 set WirelessMode=9          // 11bgn mixed
iwpriv ra0 set BasicRate=15            // 1, 2, 5.5, 11 Mbps
iwpriv ra0 set HtMcs=0                 // Please check Note-11n
iwpriv ra0 set HtGi=0
iwpriv ra0 set HtBw=0
iwpriv ra0 set SSID=11BGN_mixed_AP     // Restart AP
```

Note-11n:
HtMcs=<0-15> + HtGi=<0-1> + HtBw=<0-1>
Please check all possible combination of above set in Section 21.1.

21.3 5g

21.3.1 A only

```
iwpriv ra0 set FixedTxMode=OFDM
iwpriv ra0 set WirelessMode=2          // 11a only
iwpriv ra0 set BasicRate=336           // 6, 12, 24 Mbps
```

iwpriv ra0 set HtMcs=0 // Please check Note-11a
iwpriv ra0 set SSID=11A_only_AP // Restart AP

Note-11a:

HtMcs	0	1	2	3	4	5	6	7
Rate	6 Mbps	9 Mbps	12 Mbps	18 Mbps	24 Mbps	36 Mbps	48 Mbps	54 Mbps

21.3.2 N only

iwpriv ra0 set FixedTxMode=HT
iwpriv ra0 set WirelessMode=11 // 5g 11n only
iwpriv ra0 set BasicRate=336 // 6, 12, 24 Mbps
iwpriv ra0 set HtMcs=0
iwpriv ra0 set HtGi=0
iwpriv ra0 set HtBw=0
iwpriv ra0 set SSID=11AN_only_AP // Restart AP

Note-11n:
HtMcs=<0-15> + HtGi=<0-1> + HtBw=<0-1>
Please check all possible combination of above set in Section 21.1.

21.4 11ac

21.4.1 VHT Fixed Rate iwpriv command

21.4.1.1 fpga_on

Description: Turn on or off VHT fixed rate
Value:

iwpriv rai0 set fpga_on=6

0: Disable
6: Enable

21.4.1.2 dataphy

Description: PHY mode configuration
Value:

iwpriv rai0 set dataphy=4

0 = CCK
1 = OFDM
2 = HT-MM
3 = HT-GF
4 = VHT

21.4.1.3 databw

Description: Bandwidth configuration
Value:

iwpriv rai0 set databw=2

0 = 20M
1 = 40M
2 = 80M

21.4.1.4 datamcs

Description: MCS configuration
Value:

iwpriv rai0 set datamcs=24

Note
bit[3:0] stands for Modulation Coding Scheme (MCS)
Range: 0 - 9
bit[6:4] stands for Number of Spatial Stream (NSS)
0: 1SS
1: 2SS

Example:
datamcs=24 → 2SS MCS8
24 (dec) = 0x18 = b'0001,1000
bit[6:4] = b'001 = 1 (dec) → 2SS
bit[3:0] = b'1000 = 8 (dec) → MCS8

1SS & 2SS MCS Rate mapping table:

1SS

MCS Index	Modulation	Value (Dec)
0	BPSK	0
1	QPSK	1
2	QPSK	2
3	16-QAM	3
4	16-QAM	4
5	64-QAM	5
6	64-QAM	6
7	64-QAM	7
8	256-QAM	8
9	256-QAM	9

2SS

MCS Index	Modulation	Value (Dec)
0	BPSK	16
1	QPSK	17
2	QPSK	18
3	16-QAM	19
4	16-QAM	20
5	64-QAM	21
6	64-QAM	22
7	64-QAM	23
8	256-QAM	24
9	256-QAM	25

21.4.1.5 datagi

Description: Guard Interval configuration

Value:

iwpriv rai0 set datagi=0

0 = Short GI,
1 = Long GI

21.4.2 VHT Fixed Rate example

```
iwpriv rai0 set WirelessMode=14
iwpriv rai0 set fpga_on=6        // Enable VHT fixed rate
iwpriv rai0 set dataphy=4        // VHT
iwpriv rai0 set databw=2         // 80MHz
iwpriv rai0 set datagi=0         // SGI
iwpriv rai0 set datamcs=25       // 2SS MCS9
```

The following 802.11ac rate table is from http://www.revolutionwifi.net/.

802.11ac OFDM Data Rates

MCS	Modulation	Bits per Symbol	Coding Ratio	20-MHz		40-MHz		80-MHz	
				800ns	400ns	800ns	400ns	800ns	400ns
1 Spatial Stream				Data Rate (Mbps)					
MCS 0	BPSK	1	1/2	6.5	7.2	13.5	15.0	29.3	32.5
MCS 1	QPSK	2	1/2	13.0	14.4	27.0	30.0	58.5	65.0
MCS 2	QPSK	2	3/4	19.5	21.7	40.5	45.0	87.8	97.5
MCS 3	16-QAM	4	1/2	26.0	28.9	54.0	60.0	117.0	130.0
MCS 4	16-QAM	4	3/4	39.0	43.3	81.0	90.0	175.5	195.0
MCS 5	64-QAM	6	2/3	52.0	57.8	108.0	120.0	234.0	260.0
MCS 6	64-QAM	6	3/4	58.5	65.0	121.5	135.0	263.3	292.5
MCS 7	64-QAM	6	5/6	65.0	72.2	135.0	150.0	292.5	325.0
MCS 8	256-QAM	8	3/4	78.0	86.7	162.0	180.0	351.0	390.0
MCS 9	256-QAM	8	5/6	N/A	N/A	180.0	200.0	390.0	433.3
2 Spatial Streams				Data Rate (Mbps)					
MCS 0	BPSK	1	1/2	13.0	14.4	27.0	30.0	58.5	65.0
MCS 1	QPSK	2	1/2	26.0	28.9	54.0	60.0	117.0	130.0
MCS 2	QPSK	2	3/4	39.0	43.3	81.0	90.0	175.5	195.0
MCS 3	16-QAM	4	1/2	52.0	57.8	108.0	120.0	234.0	260.0
MCS 4	16-QAM	4	3/4	78.0	86.7	162.0	180.0	351.0	390.0
MCS 5	64-QAM	6	2/3	104.0	115.6	216.0	240.0	468.0	520.0
MCS 6	64-QAM	6	3/4	117.0	130.0	243.0	270.0	526.5	585.0
MCS 7	64-QAM	6	5/6	130.0	144.4	270.0	300.0	585.0	650.0
MCS 8	256-QAM	8	3/4	156.0	173.3	324.0	360.0	702.0	780.0
MCS 9	256-QAM	8	5/6	N/A	N/A	360.0	400.0	780.0	866.7

22 Q&A

22.1 Why does WPAPSK not work?

Please make sure the parameter **"DefaultKeyID" is set to 2** in the configuration file.

22.2 How to switch driver to operate in 5G band?

Please make sure the IC supports 5G band.
Also, please configure the WirelessMode and Channel correctly.

22.3 How do I check my channel list?

Please check CountryRegion or CountryRegionABand.

22.4 How can I know the version of current WLAN Driver?

Please use the following command.
iwpriv ra0 show driverinfo

22.5 Can SoftAP support Antenna diversity?

No, SoftAP do not support antenna diversity even EEPROM has set antenna enabled.

22.6 DFS Test example

Case 1: Band 2 & 3 select one channel for test
Test Condition:
Run 30% throughput between STA and AP.

DFS Debug command:
iwpriv ra0 set RadarDebug=0x10

DFS CE certification setting in the profile:
IEEE80211H=1
DfsOutdoor=0
RDRegion=CE
CountryCode=GB

Result:
All major test items are all passed.

Case 2: Band 2 & 3 select one channel for test.

Test condition:
Run video stream throughput between STA and AP. (Set AP Fix Tx Rate to MCS0)
Bandwidth setting 20MHz and 20/40MHz Auto.

DFS Debug command:
iwpriv ra0 set RadarDebug=0x10

DFS FCC certification setting in the profile:
IEEE80211H=1
DfsOutdoor=0
RDRegion=FCC
CountryCode=US

Result:
When Radar signal run in 5498~5502MHz, Radar type 3 & 4 fail in BW 40MHz test.
Radar type 1 fail in BW 20MHz test, Recommend to make the Radar signal run in 5495~5525MHz with BW 40MHz test. In 5494~5506MHz in BW 20MHz test. All major test items are all passed.

Case 3: Detect DFS signal without move channel. (For Lab testing)
Command Example:

iwpriv ra0 set Debug=3
iwpriv ra0 set Channel=100
iwpriv ra0 set RadarDebug=0x10
iwpriv ra0 set ChMovTime=2
iwpriv ra0 set DfsSwDisable=0

Result:
When Radar signals run in channl 100, the AP will display DFS detected information on the console.

DFS deteched consloe log may look like below:

DFS HW check channel = 0x4
T= XXXXX W= XXX detected by ch 2

22.7 TX & RX performance is always unbalance

When encounter TX & RX performance unbalance issue during Wi-Fi performance test, please check the TxBurst option is off or on. When TxBurst is on, the TX packets will have higher priority than RX packets. In the result, the WLAN TX performance will be higher than RX. This problem usual appears in Fast Ethernet + WLAN solution. GiGaBit Ethernet + WLAN solution doesn't have such problem.

How to turn off TxBurst?
By profile:
TxBurst=0
By iwpriv command:
iwpriv ra0 set TxBurst=0

22.8 Why can't I configure a SSID containing comma ","?

Please modify your code as follows.

```
==========
INT RTMPAPPrivIoctlSet(
    IN RTMP_ADAPTER *pAd,
    IN RTMP_IOCTL_INPUT_STRUCT *ploctlCmdStr)
{
    PSTRING this_char;
    PSTRING value;
    INT Status = NDIS_STATUS_SUCCESS;

    while ((this_char = strsep((char **)&ploctlCmdStr->u.data.pointer, "\0")) != NULL)
    {
        if (!*this_char)
            continue;

        if ((value = strchr(this_char, '=')) != NULL)
            *value++ = 0;
```

22.9 Why throughput is low when using 1SS to send traffic with legacy rate or MCS0-7?

Using 2SS to send traffic with legacy rate and MCS0-7 is our design by default. If you intend to change from 2SS to 1SS, please use TC instead of TSSI.

22.10 TGn 4.2.10 failed. Why does DUT not send MC traffic?

4.2.10 Group traffic with WPA2-PSK Only Mode and WPA/WPA2-PSK Mixed Mode
If this item fails, please turn off IGMP Snooping first.

22.11 TGn 4.2.29 failed. Why the performance cannot reach the criteria?

Please make sure that the following items are correctly configured.
<Profile>
TxPreamble=1
PktAggregate=0

<Driver Config>
-CONFIG_RA_NETWORK_WORKQUEUE_BH=y
+CONFIG_RA_NETWORK_TASKLET_BH=y

<Kernel Config>
Please check items in Networking Option & Core Netfilter in your kernel config. Remove those you do not use or know.