Mediatek Wi-Fi AP Software Programming Guide

Version:	4.6
Release date:	2015-08-25

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
4.5	2015/03/25	Money Wang	Update ● WPS ● PMF ● IEEE802.11h ● Authenticator Add ● ACL
4.6	2015/08/25	Money Wang	Update ACS-related parameters ACS stands for Automatic Channel Selection Add maximum support rate parameters Update CountryRegionABand to support Ch144 Update MBSSID chapter Remove iNIC

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 all active scan
6	3-9
7	5-13
31	1-11 active scan, 12-14 passive scan
32	1-11 active scan, 12-14 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
12	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 144
13	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 144, 149, 153, 157, 161, 165
14	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 132, 136, 140, 144, 149, 153, 157, 161, 165

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
```

Note:
The range of configurable values depends on CountryRegion or CountryRegionForABand
Its default value is zero and the driver configures a working channel automatically

2.2.8 AutoChannelSelect

Description: Algorithm configuration of automatic channel selection
Value:

```
AutoChannelSelect=1
```

0: Disable
1: Old Channel Selection Algorithm (AP count)
2: New Channel Selection Algorithm (CCA)

2.2.9 AutoChannelSkipList

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

```
AutoChannelSkipList=<channel_list>
```

Example:

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

2.2.10 ACSCheckTime

Description: Configuration of periodic check time for automatic channel selection

Value:

iwpriv ra0 set ACSCheckTime=1

0: Disable

Note: Unit is hour

2.2.11 BasicRate

Description: Basic rate support

Value:

BasicRate=15

0~4095

Note:

A bitmap represent basic support rate

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 rate set before changing wireless mode.

2.2.12 SupportRate

Description: Maximum support rate configuration for 11bg

Value:

SupportRate=0xFFF

Legacy 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	1	1	1	1	1	1	1	1	1	1	1	1
Hex	F				F				F			

Note:

Unlike BasicRate, the SupportRate bitmap must be composed of consecutive 1s.

For example, if SupportRate=0x7F, it means the maximum support rate is OFDM 12M.
Also, this settings will be applied globally which means no per-SSID configuration is allowed.
Only RT5x92 supports this. Its macro is DYNAMIC_RX_RATE_ADJ.

2.2.13 SupportHTRate

Description: Maximum support rate configuration for 11n
Value:

SupportHTRate=0xFFFF

HTRate Bit Map (max. 16-bit, represent max. 16 rates)																
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCS	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Set	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Hex	F				F				F				F			

Note:
Unlike BasicRate, the SupportHTRate bitmap must be composed of consecutive 1s.
For example, if SupportHTRate=0x7F, it means the maximum support rate is MCS 6.
Also, this settings will be applied globally which means no per-SSID configuration is allowed.
Only RT5x92 supports this. Its macro is DYNAMIC_RX_RATE_ADJ.

2.2.14 BeaconPeriod

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

BeaconPeriod=100

2.2.15 DtimPeriod

Description: DTIM period
Value:

DtimPeriod=1

1~255

2.2.16 TxPower

Description: WLAN Radio Transmit Power setting in percentage
Value:

TxPower=100

0~100

2.2.17 DisableOLBC

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

```
DisableOLBC=0

0: disable
1: enable
```

2.2.18 BGProtection

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

```
BGProtection=0

0: AUTO
1: On
2: Off
```

2.2.19 MaxStaNum

Description: Configure maximun number of station that could connect with this SoftAP
Value:

```
MaxStaNum=0

0: disable
1~32
```

2.2.20 TxAntenna

Description: Configure Tx antenna number
Value:

```
TxAntenna=1

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

2.2.21 RxAntenna

Description: Configure Rx antenna number
Value:

```
RxAntenna=1

1: 1Tx1R
2: 2Tx2R
```

3: 3Tx3R

2.2.22 TxPreamble

Description: Enable or disable Tx preamble
Value:

```
TxPreamble=0

0: disable
1: enable
```

2.2.23 RTSThreshold

Description: RTS threshold configuration
Value:

```
RTSThreshold=2347

1~2347
```

2.2.24 FragThreshold

Description: Fragment threshold configuration
Value:

```
FragThreshold=2346

256~2346
```

2.2.25 TxBurst

Description: Enable or disable Tx Burst (Mediatek-proprietary acceleration method)
Value:

```
TxBurst=1

0: disable
1: enable
```

2.2.26 PktAggregate

Description: Enable or disable Tx Aggregate
Value:

```
PktAggregate=0

0: disable
1: enable
```

2.2.27 NoForwarding

Description: Enable or disable No-Packet-Forwarding within a BSSID
Value:

NoForwarding=0

0: disable
1: enable

2.2.28 NoForwardingBTNBSSID

Description: Enable or disable No-Packet-Forwarding between each BSSID
Value:

NoForwardingBTNBSSID=0

0: disable
1: enable

2.2.29 NoForwardingMBCast

Description: Enable or disable No-MC-BC-Packet-Forwarding within a BSSID
Value:

NoForwardingMBCast=0

0: disable
1: enable

2.2.30 HideSSID

Description: Enable or disable stopping sending Beacons to hide SSID
Value:

HideSSID=0

0: disable
1: enable

2.2.31 StationKeepAlive

Description: Enable or disable auto detection of aliveness of connected stations periodically
Value:

StationKeepAlive=0

0: disable
1~65535 seconds

2.2.32 ShortSlot

Description: Enable or disable short slot time
Value:

    ShortSlot=1

    0: disable
    1: enable

2.2.33 WirelessEvent

Description: Enable or disable sending wireless event to the system log
Value:

    WirelessEvent=0

    0: disable
    1: enable

2.2.34 IdsEnable

Description: Enable or disable Intrusion Detection System
Value:

    IdsEnable=0

    0: disable
    1: enable

2.2.35 AuthFloodThreshold

Description: Authentication frame flooding threshold configuration
Value:

    AuthFloodThreshold=32

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

2.2.36 ReassocReqFloodThreshold

Description: Reassocation request frame flooding threshold configuration
Value:

    ReassocReqFloodThreshold=32

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

2.2.37 ProbeReqFloodThreshold

Description: Probe request frame flooding threshold configuration
Value:

ProbeReqFloodThreshold=32

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

2.2.38 DisassocFloodThreshold

Description: Disassocation frame flooding threshold configuration
Value:

DisassocFloodThreshold=32

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

2.2.39 DeauthFloodThreshold

Description: Deauthentication frame flooding threshold configuration
Value:

DeauthFloodThreshold=32

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

2.2.40 EapReqFooldThreshold

Description: EAP request frame flooding threshold configuration
Value:

EapReqFooldThreshold=32

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

2.2.41 HT_HTC

Description: Enable or disable HT control field
Value:

HT_HTC=0

0: disable
1: enable

Note: HT control field (4-octet) is following QoS field

2.2.42 HT_RDG

Description: Enable or disable Reverse Direction Grant
Value:

HT_RDG=1

0: disable
1: enable

2.2.43 HT_EXTCHA

Description: Locate the 40MHz extension channel in combination with the main channel
Value:

HT_EXTCHA=0

0: Below
1: Above

2.2.44 HT_LinkAdapt

Decription: Enable or disable HT Link Adaptation Control
Value:

HT_LinkAdapt=0

0: disable
1: enable

2.2.45 HT_OpMode

Description: HT operation mode configuration
Value:

HT_OpMode=0

0: HT mixed mode (MM)
1: HT Greenfield mode (GF)

2.2.46 HT_MpduDensity

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

HT_MpduDensity=4

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

2.2.47 HT_BW

Description: HT channel bandwidth configuration
Value:

```
HT_BW=1
```

0: 20 MHz
1: 20/40 MHz

2.2.48 HT_PROTECT

Description: Enable or disable 802.11n protection mechanism
Value:

```
HT_PROTECT=1
```

0: disable
1: enable

2.2.49 HT_BSSCoexistence

Description: Enable or disable HT BSS coexistence support
Value:

```
HT_BSSCoexistence=1
```

0: disable
1: enable

2.2.50 HT_TxStream

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

```
HT_TxStream=2
```

1~3: valid spatial streams

2.2.51 HT_RxStream

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

```
HT_RxStream=2
```

1~3: valid spatial streams

2.2.52 HT_BADecline

Description: Configure whether always declining Block Ack Request sent from the peer
Value:

```
HT_BADecline=0

0: disable
1: enable
```

2.2.53 HT_AutoBA

Description: Enable or disable automatically building Block Ack session with the peer
Value:

```
HT_AutoBA=1

0: disable
1: enable
```

2.2.54 HT_AMSDU

Description: Enable or disable AMSDU transmission
Value:

```
HT_AMSDU=0

0: disable
1: enable
```

2.2.55 HT_BAWinSize

Description: Block Ack window size configuration
Value:

```
HT_BAWinSize=64

1~64
```

2.2.56 HT_GI

Description: HT guard interval configuration
Value:

```
HT_GI=1

0: LGI (long guard interval)
1: SGI (short guard interval)
```

2.2.57 HT_MCS

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

```
HT_MCS=33

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

2.2.58 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)
```

2.2.59 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
```

2.2.60 HT_STBC

Description: Enable or disable HT STBC support
Value:

```
HT_STBC=0

0: disable
1: enable
```

2.2.61 HT_LDPC

Description: Enable or disable HT LDPC support
Value:

```
HT_LDPC=0

0: disable
1: enable
```

Note: MT76x2E chipset only

2.2.62 VHT_BW

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

VHT_BW=1

0: disable
1: enable

Note: 11AC chipset only

2.2.63 VHT_SGI

Description: VHT Guard Interval support configuration
Value:

VHT_SGI=1

0: Long guard interval
1: Short guard interval

Note: 11AC chipset only

2.2.64 VHT_STBC

Description: Enable or disable 11ac STBC
Value:

VHT_STBC=1

0: disable
1: enable

Note: 11AC chipset only

2.2.65 VHT_BW_SIGNAL

Description: Enable or disable 11ac bandwidth signaling
Value:

VHT_BW_SIGNAL=1

0: disable
1: enable

Note: 11AC chipset only

2.2.66 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

2.2.67 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

2.2.68 VLANID

Description: set VLAN ID
Value:

    VLANID=0

    0: Disable

2.2.69 VLANPriority

Description: set VLAN Priority
Value:

    VLANPriority=0

    0: Disable

2.2.70 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

2.2.71 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.

2.2.72 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 later 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
```

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 all active scan
6	3-9
7	5-13
31	1-11 active scan, 12-14 passive scan
32	1-11 active scan, 12-13 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
12	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 144
13	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 120, 124, 128, 132, 136, 140, 144, 149, 153, 157, 161, 165
14	36, 40, 44, 48, 52, 56, 60, 64, 100, 104, 108, 112, 116, 132, 136, 140, 144, 149, 153, 157, 161, 165

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: WLAN mode configuration
Value:

```
iwpriv ra0 set 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 band
7: 11gn mixed
8: 11an mixed
9: 11bgn mixed
10: 11AGN mixed
11: 11n only in 5g band
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
HT

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

```
iwpriv ra0 set Channel=6
```

Note:
The range of configurable values depends on CountryRegion or CountryRegionForABand

3.1.15 AutoChannelSel

Description: Algorithm configuration of automatic channel selection
Value:

```
iwpriv ra0 set AutoChannelSel=2
```

0:	Disable
1:	Old Channel Selection Algorithm (AP count)
2:	New Channel Selection Algorithm (CCA)

3.1.16 ACSCheckTime

Description: Configuration of periodic check time for automatic channel selection
Value:

```
iwpriv ra0 set ACSCheckTime=1
```

0:	Disable

Note: Unit is hour

3.1.17 BeaconPeriod

Description: configure Beacon period

Value:

```
iwpriv ra0 set BeaconPeriod=100
```

20 ~ 1024 (unit is in milli-seconds)

3.1.18 DtimPeriod

Description: Configure DTIM period

Value:

```
iwpriv ra0 set DtimPeriod=1
```

1~5

3.1.19 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

3.1.20 BGProtection

Description: Enable or disable 11bg protection

Value:

```
iwpriv ra0 set BGProtection=0
```

0: disable

1: Always on

2: Always off

3.1.21 DisableOLBC

Description: Enable or disable OLBC

Value:

```
iwpriv ra0 set DisableOLBC=0

0: disable
1: enable
```

3.1.22 TxPreamble

Description: enable or disable Tx preamble

Value:

```
iwpriv ra0 set TxPreamble=1

0: disable
1: enable
```

3.1.23 RTSThreshold

Description: Set RTS Threshold

Value:

```
iwpriv ra0 set RTSThreshold=2347

1~2347
```

3.1.24 FragThreshold

Description: Set Fragment threshold

Value:

```
iwpriv ra0 set FragThreshold=2346

256~2346
```

3.1.25 TxBurst

Description: enable or disable Tx burst mode

Value:

```
iwpriv ra0 set TxBurst=0

0: disable
1: enable
```

3.1.26 PktAggregate

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

Value:

```
iwpriv ra0 set PktAggregate=1
```

0: disable
1: enable

3.1.27 NoForwarding

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

```
iwpriv ra0 set NoForwarding=0
```

0: disable
1: enable

3.1.28 NoForwardingBTNBSSID

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

```
iwpriv ra0 set NoForwardingBTNBSSID=1
```

0: disable
1: enable

3.1.29 NoForwardingMBCast

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

```
iwpriv ra0 set NoForwardingMBCast=1
```

0: disable
1: enable

3.1.30 HideSSID

Description: enable or disable hidden SSID
Value:

```
iwpriv ra0 set HideSSID=1
```

0: disable
1: enable

3.1.31 ShortSlot

Description: enable or disabllle short slot time
Value:

```
iwpriv ra0 set ShortSlot=0
```

0: disable
1: enable

3.1.32 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]

3.1.33 DisConnectAllSta

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

```
iwpriv ra0 set DisConnectAllSta=1
```

1: disconnect all STAs

3.1.34 McastPhyMode

Description: Configure multicast physical mode
Value:

```
iwpriv ra0 set McastPhyMode=0
```

0:	Disable
1:	CCK
2:	OFDM
3:	HTMIX

3.1.35 McastMcs

Description: Specify the MCS of multicast packets.
Value:

```
iwpriv ra0 set McastMcs=0
```

0~15

3.1.36 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)

3.1.37 AutoFallBack

Description: enable or disable auto fall back rate control function

Value:

```
iwpriv ra0 set AutoFallBack=1
```

0: disable
1: enable

3.1.38 GreenAP

Description: enable or disable Green AP fucntion

Value:

```
iwpriv ra0 set GreenAP=0
```

0: disable
1: enable

3.1.39 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
```

3.1.40 HwAntDiv

Description: Enable or disable Hardware antenna diversity

Value:

```
iwpriv ra0 set HwAntDiv=0
```

0: disable
1: enable

Note: RT5350 only

3.1.41 HtBw

Description: HT channel bandwidth configuration
Value:

```
iwpriv ra0 set HtBw=1
```

0: 20 MHz
1: 20/40 MHz

3.1.42 VhtBw

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

```
iwpriv ra0 set VhtBw=1
```

0: disable
1: enable
Note: 11AC chipset only

3.1.43 VhtStbc

Description: Enable/disable 11AC STBC Support
Value:

```
iwpriv ra0 set VhtStbc=1
```

0: disable
1: enable
Note: 11AC chipset only

3.1.44 VhtBwSignal

Description: Enable/disable 11 AC BandWidth signaling
Value:

```
iwpriv ra0 set VhtBwSignal=1
```

0: disable
1: enable
Note: 11AC chipset only.

3.1.45 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.

3.1.46 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.	

3.1.47 HtGi

Description: Set WLAN Guard interval support

Value:

iwpriv ra0 set HtGi=1

0: long guard interval
1: short guard interval

3.1.48 HtOpMode

Description: HT operation Mode
Value:

iwpriv ra0 set HtOpMode=0

0: HT mixed mode
1: HT Greenfield mode

3.1.49 HtStbc

Description: Enable or disable HT STBC
Value:

iwpriv ra0 set HtStbc=1

0: disable
1: enable

3.1.50 HtExtcha

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

iwpriv ra0 set HtExtcha=0

0: below
1: Above

3.1.51 HtMpduDensity

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

iwpriv ra0 set HtMpduDensity=4

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

3.1.52 HtRdg

Description: Enable or disable HT Reverse Direction Grant
Value:

```
iwpriv ra0 set HtRdg=1
```

0: disable
1: enable

3.1.53 HtAmsdu

Description: Enable or disable A-MSDU section
Value:

```
iwpriv ra0 set HtAmsdu=0
```

0: disable
1: enable

3.1.54 HtAutoBa

Description: Enable or disable automatic setup of Block Ack session with peer
Value:

```
iwpriv ra0 set HtAutoBa=1
```

0: disable
1: enable

3.1.55 BADecline

Description: Configuration of rejecting ADDBA request sent from peer
Value:

```
iwpriv ra0 set BADecline=0
```

0: disable
1: enable

3.1.56 HtBaWinSize

Description: Configuration of Block Ack receiving window size
Value:

```
iwpriv ra0 set HtBaWinSize=64
```

1~64

3.1.57 HtTxBASize

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

```
iwpriv ra0 set HtTxBASize=64
```

1~64

3.1.58 BASetup

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

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

→The six 2-digit hex-decimal numbers composes the STA MAC address
→The seventh decimal number is the TID value

3.1.59 BAOriTearDown

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

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

→The six 2-digit hex-decimal numbers composes the STA MAC address
→The seventh decimal number is the TID value

3.1.60 BARecTearDown

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

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

→The six 2-digit hex-decimal numbers composes the STA MAC address
→The seventh decimal number is the TID value

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 PktAggregate

Description: Enable or disable 11B/G packet aggregation (Piggyback)

Value:

```
iwpriv ra0 set PktAggregate=1
```

0: disable
1: enable

3.1.69 KickStaRssiLow

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

Value:

```
iwpriv ra0 set KickStaRssiLow=0
```

0: Disable
0 ~ -100

3.1.70 AssocReqRssiThres

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

Value:

```
Iwpriv ra0 set AssocReqRssiThres=0
```

0: Disable
0~ -100

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]

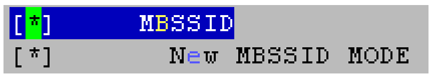
```
# iwpriv ra0 show driverinfo
Driver version: 2.7.1.6
```

5 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.

5.1 How to Setup

Please turn on MBSS_SUPPORT in driver config.



```
[*]     MBSSID
[*]       New MBSSID MODE
```

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

5.2 Parameter in RT2860AP.dat

5.2.1 BssidNum

Description: Multiple BSSID number configuration
Value:

BssidNum=1

1/2/4/8/16

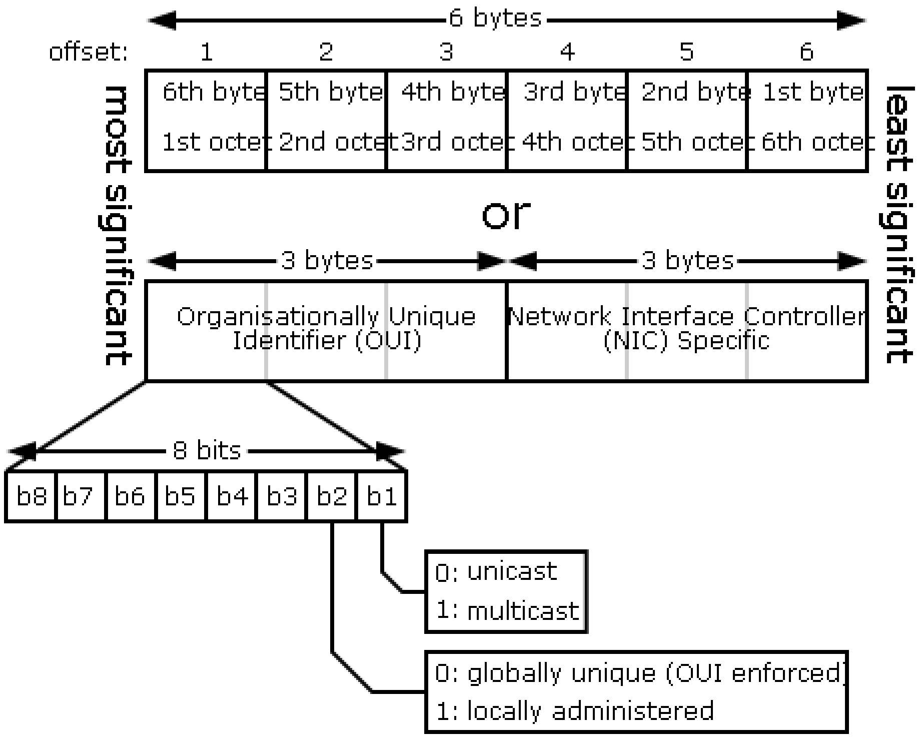
Note:

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

5.3 Important Note

5.3.1 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.



5.3.2 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:43:00:00:0**0** 00 is multiple of 4
- ra1: 00:0c:43:00:00:0**1** 01 comes from (1st byte 0x00) + 0x01
- ra2: 00:0c:43:00:00:0**2** 02 comes from (1st byte 0x00) + 0x02
- ra3: 00:0c:43:00:00:0**3** 03 comes from (1st byte 0x00) + 0x03

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.

5.3.3 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. This is supported in 5-series products.

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'0000**00**10)
● ra2: 0**6**:0c:43:00:00:00	0**6** comes from (6th byte 0x00 | b'0000**01**10)
● ra3: 0**a**:0c:43:00:00:00	0**a** comes from (6th byte 0x00 | b'0000**10**10)

5.3.4 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. The extension algorithm is **(3rd Byte & MacMSK) + (idx)**. BssidNum will affect MacMSK. This is supported only in new 7-series products and will be turned on by default.

```
if (BssidNum <= 2)            { MacMSK  = b'11111110;}
else if (BssidNum <= 4)       { MacMSK  = b'11111100;}
else if (BssidNum <= 8)       { MacMSK  = b'11111000;}
else if (BssidNum <= 16)      { MacMSK  = b'11110000;}
```

Taking BssidNum=4 for example:

● ra0: 00:0c:43:0**0**:00:00	
● ra1: 0**2**:0c:43:0**0**:00:00	00 comes from (3rd byte 0x00 & 0xfb) + 0x00
● ra2: 0**2**:0c:43:0**1**:00:00	01 comes from (3rd byte 0x00 & 0xfb) + 0x01
● ra3: 0**2**:0c:43:0**2**:00:00	02 comes from (3rd byte 0x00 & 0xfb) + 0x02

MT7603 and MT7628 take a little different policy which uses **first 4 bits** of 3rd byte to do extension. The extension algorithm is **(3rd Byte & MacMSK) + (idx << 4)**.

```
if (BssidNum <= 2)            { MacMSK  = b'11101111;}
else if (BssidNum <= 4)       { MacMSK  = b'11001111;}
else if (BssidNum <= 8)       { MacMSK  = b'10001111;}
else if (BssidNum <= 16)      { MacMSK  = b'00001111;}
```

Taking BssidNum=4 for example:

● ra0: 00:0c:43:**0**0:00:00	
● ra1: 0**2**:0c:43:**1**0:00:00	10 comes from (3rd byte 0x00 & 0xbf) + (0x01 << 4)
● ra2: 0**2**:0c:43:**2**0:00:00	20 comes from (3rd byte 0x00 & 0xbf) + (0x02 << 4)
● ra3: 0**2**:0c:43:**3**0:00:00	30 comes from (3rd byte 0x00 & 0xbf) + (0x03 << 4)

5.3.5 Address Confliction Problem

In this section, we'll explain the address conflication problem.

Suppose we have four DUTs with the following global MAC addresses.

DUT-A: 00:0c:43:10:22:33
DUT-B: 00:0c:43:11:22:33
DUT-C: 00:0c:43:12:22:33
DUT-D: 00:0c:43:13:22:33

Each DUT turns on MBSSID and its BssidNum=4. As a result, you will get the following total 16 MAC addresses.

	1st BSSID	2nd BSSID	3rd BSSID	4th BSSID
DUT-A	00:0c:43:10:22:33	02:0c:43:11:22:33	02:0c:43:12:22:33	02:0c:43:13:22:33
DUT-B	00:0c:43:11:22:33	02:0c:43:11:22:33	02:0c:43:12:22:33	02:0c:43:13:22:33
DUT-C	00:0c:43:12:22:33	02:0c:43:11:22:33	02:0c:43:12:22:33	02:0c:43:13:22:33
DUT-D	00:0c:43:13:22:33	02:0c:43:11:22:33	02:0c:43:12:22:33	02:0c:43:13:22:33

The 2nd, 3rd and 4th BSSID are exactly identical for these DUTs. So, the address conflict problem does exist but the conflicting rate is extremely low. Using local MAC address as BSSID, this problem is inevitable.

5.4 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.

5.4.1 Example

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

6 WPS

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

6.1 Architectural Overview

This section presents high-level description of the Wi-Fi Simple Configuration architecture. Most material is taken directly from the WSC specification. In the following figure, you can see that there are three logical components involved in WSC: the Registrar, the access point (AP), and the Enrollee.

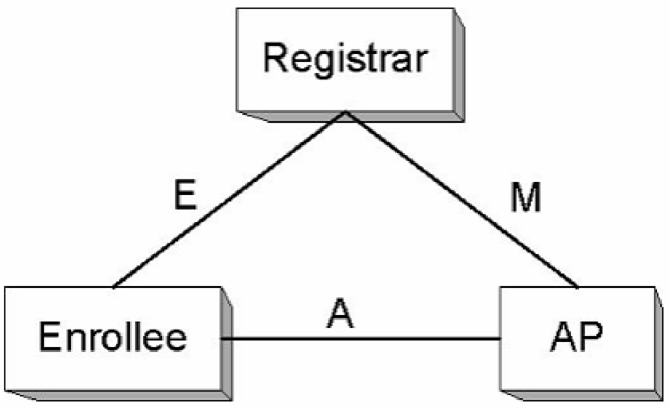


Figure 1: Components and Interfaces

[Component]

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

The **Registrar** is an entity with the authority to issue and revoke domain credentials. A registrar may be integrated into an AP, or it may be separate from the AP.

The **AP** is an infrastructure mode 802.11 Access Point. We also call it **Proxy**.

[Interface]

Interface E is logically located between the Enrollee and the Registrar and its purpose is to enable the Registrar to discover and issue WLAN credentials to the Enrollee.

Interface M is between the AP and the Registrar and it enables an external Registrar to manage a WSC AP.

Interface A is between the Enrollee and the AP and it enables discovery of the WSC WLAN and communication between the Enrollee and IP-only Registrars.

6.2 Parameters in RT2860AP.dat

6.2.1 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

6.2.2 WscConfStatus

Description: Configure WPS state
Value:

WscConfStatus=1

1: AP is unconfigured
2: AP is configured

6.2.3 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

6.2.4 WscKeyASCII

Description: Choose the format/length of a generated key for an un-configured AP (internal registrar)
Value:

```
WscKeyASCII=0
```

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

6.2.5 WscSecurityMode

Description: Configure the security mode which AP would use when being configured
Value:

```
WscSecurityMode=0
```

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

6.2.6 Wsc4digitPinCode

Description: Configure whether to use 4-digit PIN code
Value:

```
Wsc4digitPinCode=1
```

0: 8-digit PIN code
1: 4-digit PIN code

6.2.7 WscVendorPinCode

Description: Configure a fixed PIN code which AP would use as an Enrollee
Value:

```
WscVendorPinCode=[xxxx|yyyyyyyy]
```

xxxx is a 4-digit PIN code
yyyyyyyy is a 8-digit PIN code

6.2.8 WscDefaultSSID0

Description: Configure the SSID which AP would use after being configured
Value:

```
WscDefaultSSID0=SSID
```

1~32 characters

6.2.9 WscV2Support

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

    WscV2Support=1

    0: disable
    1: enable

6.2.10 WscManufacturer

Description: WPS manufacturer string
Value:

    WscManufacturer=

    Less than 64 characters

6.2.11 WscModelName

Description: WPS model name string
Value:

    WscModelName=

    Less than 32 characters

6.2.12 WscDeviceName

Description: WPS device name string
Value:

    WscDeviceName=

    Less than 32 characters

6.2.13 WscModelNumber

Description: WPS model number string
Value:

    WscModelNumber=

    Less than 32 characters

6.2.14 WscSerialNumber

Description: WPS serial number string
Value:

    WscSerialNumber=

Less than 32 characters

6.3 WPS iwpriv command

6.3.1 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

6.3.2 WscConfStatus

Description: Configure WPS state
Value:

```
iwpriv ra0 set WscConfStatus=1
```

1: AP is unconfigured
2: AP is configured

6.3.3 WscMode

Description: Configure WPS mode
Value:

```
iwpriv ra0 set WscMode=1
```

1: PIN Mode
2: PBC Mode

6.3.4 WscGetConf

Description: Trigger WPS action
Value:

```
iwpriv ra0 set WscGetConf=1
```

6.3.5 WscStop

Description: Stop WPS process
Value:

```
iwpriv ra0 set WscStop=1
```

6.3.6 WscPinCode

Description: Input Enrollee's PIN code which AP would use as a Registrar

Value:

```
iwpriv ra0 WscPinCode=[xxxx|yyyyyyyy]

xxxx is a 4-digit PIN code
yyyyyyyy is a 8-digit PIN code
```

6.3.7 WscGenPinCode

Description: Generate random PIN code which AP would use as an Enrollee

Value:

```
iwpriv ra0 set WscGenPinCode=1
```

Note:

PIN code can be either 4-digit or 8-digit depending on Wsc4digitPinCode
One of the digits in the 8-digit PIN code is used as a checksum

6.3.8 WscVendorPinCode

Description: Configure a fixed PIN code which AP would use as an Enrollee

Value:

```
iwpriv ra0 set WscVendorPinCode=[xxxx|yyyyyyyy]

xxxx is a 4-digit PIN code
yyyyyyyy is a 8-digit PIN code
```

6.3.9 WscSecurityMode

Description: Configure the security mode which AP would use when being configured

Value:

```
iwpriv ra0 set WscSecurityMode=0
```

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

6.3.10 WscOOB

Description: Reset WPS AP to the OOB (out-of-box) state

Value:

```
iwpriv ra0 set WscOOB=1
```

Note:

<OOB settings>

SSID	RalinkInitailAPxxxxxx (last 3 bytes of ra0 MAC 00:0c:43:xx:xx:xx)
AuthMode	WPA2PSK
EncrypType	AES
WPAPSK	RalinkInitialAPxx1234
WscConfStatus	1 (AP is unconfigured)

6.3.11 WscStatus

Description: Get current WPS status

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:	Received EAP-Rsp (ID)
7:	Received EAP-Req with wrong WSC SMI Vendor ID
8:	Received EAP-Req with wrong WSC Vendor Type
9:	Sending EAP-Req (WSC_START)
10:	Sending M1
11:	Received M1
12:	Sending M2
13:	Received M2
14:	Received M2D
15:	Sending M3
16:	Received M3
17:	Sending M4
18:	Received M4
19:	Sending M5
20:	Received M5
21:	Sending M6
22:	Received M6
23:	Sending M7
24:	Received M7
25:	Sending 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:	WSC configured

6.3.12 WscMultiByteCheck

Description: Enable or disable multi-byte check
Value:

```
iwpriv ra0 set WscMultiByteCheck=0
```

0: disable
1: enable

6.3.13 WscVersion

Description: Set WPS support version
Value:

```
iwpriv ra0 set WscVersion=10
```

0x10: Hexadecimal

6.3.14 WscVersion2

Description: Set WPS version of V2 support
Value:

```
iwpriv ra0 set WscVersion2=20
```

0x20: Hexadecimal

6.3.15 WscV2Support

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

```
iwpriv ra0 WscV2Support=1
```

0: disable
1: enable

6.3.16 WscFragment

Description: Enable or disable WPS fragmentation
Value:

```
iwpriv ra0 WscFragment=0
```

0: disable
1: enable

6.3.17 WscFragmentSize

Description: Configure the size of WPS fragmentation

Value:

```
iwpriv ra0 set WscFragmentSize=128

128~300
```

6.3.18 WscSetupLock

Description: Enable or disable WPS setup lock
Value:

```
iwpriv ra0 set WscSetupLock=1

0: disable
1: enable
```

6.3.19 WscSetupLockTime

Description: Configure WPS setup lock time
Value:

```
iwpriv ra0 set WscSetupLockTime=0

0: lock forever
Unit: minute
```

6.3.20 WscMaxPinAttack

Description: Configure WPS PIN attack MAX time
Value:

```
iwpirv ra0 set WscMaxPinAttack=10

0: disable
1-10
```

6.3.21 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
```

6.3.22 WscExtraTlvType

Description: Define data format of extra TLV value
Value:

```
iwpriv ra0 set WscExtraTlvType=1
```

0: ASCII string
1: Hex string

6.3.23 WscExtraTlvData

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

iwpriv ra0 set WscExtraTlvData=

ASCII string or Hex string

6.4 WPS Scenario

The following scenarios are currently supported:

- Initial WLAN Setup
  - Standalone AP with a built-in Registrar
  - AP with an external Registrar
    - EAP-based setup of External Wireless Registrar
      - [AP] --- EAP --- [Wireless Registrar]
    - UPnP-based setup of External Wired Registrar
      - [AP] --- UPnP --- [Wired Registrar]

- Adding Member Devices
  - In-band setup using a standalone AP/Registrar
    - [Enrollee] --- EAP --- [AP/Registrar]
  - In-band setup using an External Wired Registrar
    - [Enrollee] --- EAP --- [AP] --- UPnP --- [Wired Registrar]

6.4.1 Running WPS

First, run UPnP deamon.
`# wscd -w /etc/xml -m 1 -d 3 &` (if your xml file in /etc/xml)

Note: wscd must be ported to the target platform first

You may use iwpriv command sequence to trigger WPS as 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, 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

6.4.2 Initial WLAN setup with External Registrar

[Unconfigured AP] 🡠 EAP 🡢 [Wireless Registrar]
[Unconfigured AP] 🡠 UPnP 🡢 [Wired Registrar]

Please make sure that UPnP deamon has been running. After WPS registration succeeds, the configured AP will work as a proxy forwarding EAP and UPnP messages.

- PIN
  - AP configuration (as an Enrollee)
    - iwpriv ra0 set WscConfMode=7
    - iwpriv ra0 set WscConfStatus=1
    - iwpriv ra0 set WscMode=1
    - iwpriv ra0 set WscGenPinCode=1
    - iwpriv ra0 set WscGetConf=1
- PBC
  - AP configuration (as an Enrollee)
    - iwpriv ra0 set WscConfMode=7
    - iwpriv ra0 set WscConfStatus=1
    - iwpriv ra0 set WscMode=2
    - iwpriv ra0 set WscGetConf=1

6.4.3 Adding a member device using a standalone AP/Registrar

[STA] 🡐 EAP 🡒 [AP/Registrar]

- PIN
  - AP configuration (as an Registrar)
    - iwpriv ra0 set WscConfMode=7
    - iwpriv ra0 set WscPinCode=xxxxxxxx (xxxxxxxx is Enrollee’s PIN code)
    - iwpriv ra0 set WscMode=1
    - iwpriv ra0 set WscGetConf=1
- PBC
  - AP configuration (as an Registrar)
    - iwpriv ra0 set WscConfMode=7
    - iwpriv ra0 set WscMode=2
    - iwpriv ra0 set WscGetConf=1

6.4.4 Adding a member device using an External Wired Registrar

[STA] 🡐 EAP 🡒 [AP] 🡐 UPnP 🡒 [Registrar]

- PIN
  - 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.
  - on Client (Enrollee) side
    - Select PIN process
    - The process will start, and the application will display the result of the process on completion
- PBC
  - 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.
  - on Client (Enrollee) side
    - Select PBC process
    - The registration process will start, and the application will display the result of the process on completion.

6.4.5 WPS Configuration Status

The WPS attribute “Simple Configuration (SC) State” in WPS IEs (contained in beacon and probe response) indicates whether a device is configured. If an AP is shipped from the factory in an un-configured state (SC State is 0x01), then the AP must change to the configured state (SC State is 0x02) if any of the following occurs.

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.
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 an AP is shipped from the factory in an un-configured state (SC State 0x01), then a factory reset must revert the Simple Config State to un-configured. If an AP is shipped from the factory pre-configured with WPA2-Personal mixed mode and a randomly generated key, the SC State must be set to 'configured' (0x02) to prevent an external registrar from overwriting the factory settings. A factory reset must restore the unit to the same configuration as what it was when shipped.

6.5 Basic operation of Ralink WPS AP

6.5.1 Add member devices using a external Registrar in PIN mode

1. [Ralink AP] - Turn on APUT.
2. [Ralink STA] - Push PBC button.
3. [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.
4. [Ralink AP] - Do nothing.
5. [Ralink STA] - Verify that Ralink STA successes to ping Ralink A.

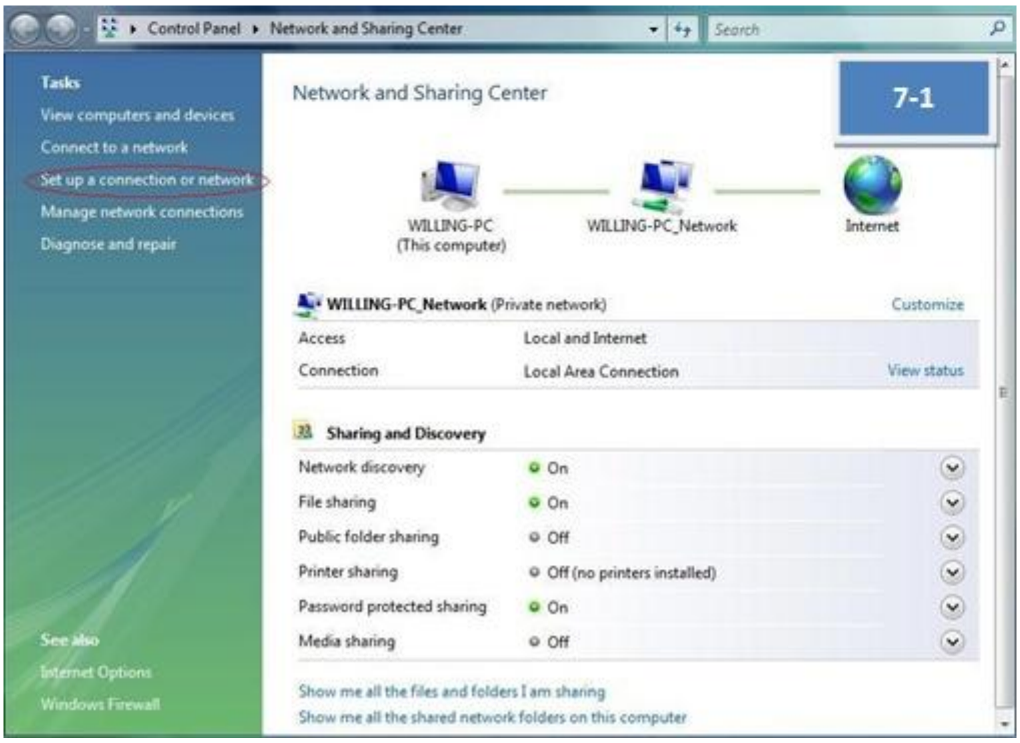
6.5.2 Configure APUT using a wireless external Registrar in PIN mode

1. [Ralink AP] - Turn on Ralink AP
2. [Ralink AP] - Change AP ability "iwpriv ra0 set WscConfMode=7"
3. [Ralink AP] - Change from configured to un-configured state: "iwpriv ra0 set WscConfStatus=1 "
4. [Ralink AP] - 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 code from console and enter it 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

6.5.3 Configure APUT using a wired external Registrar in PIN mode

1. [Ralink AP] - Turn on the Ralink AP
2. [Ralink AP] - Connect the Ethernet cable between AP and external Registrar (Windows Vista) and make sure you can ping AP from external Registrar first!
3. [Ralink AP] - Change AP ability “iwpriv ra0 set WscConfMode=7”
4. [Ralink AP] - Change from configured to un-configured state: “iwpriv ra0 set WscConfStatus=1 “
5. [Ralink AP] - 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 parameters (SSID = “scaptest4.1.3ssid” and WPA (2)-PSK=”scaptest4.1.3psk”) which should be entered when prompted
8. [Microsoft STA] - Read AP's PIN code from console and enter it at Microsoft STA
9. [Ralink STA] - Manually configure Ralink STA with the new parameters (SSID = “scaptest4.1.3ssid” and WPA (2)-PSK passphrase= “scaptest4.1.3psk”).
10. [Ralink STA] - Verify that Ralink STA successes to ping to Microsoft STA.

As to details of step-7, please refer to the following figures from [7-1] to [7-6].


Control Panel ▸ Network and Sharing Center
Search
Tasks
View computers and devices
Connect to a network
Set up a connection or network
Manage network connections
Diagnose and repair
See also
Internet Options
Windows Firewall
7-1
WILLING-PC (This computer)
WILLING-PC_Network
Internet

Network and Sharing Center

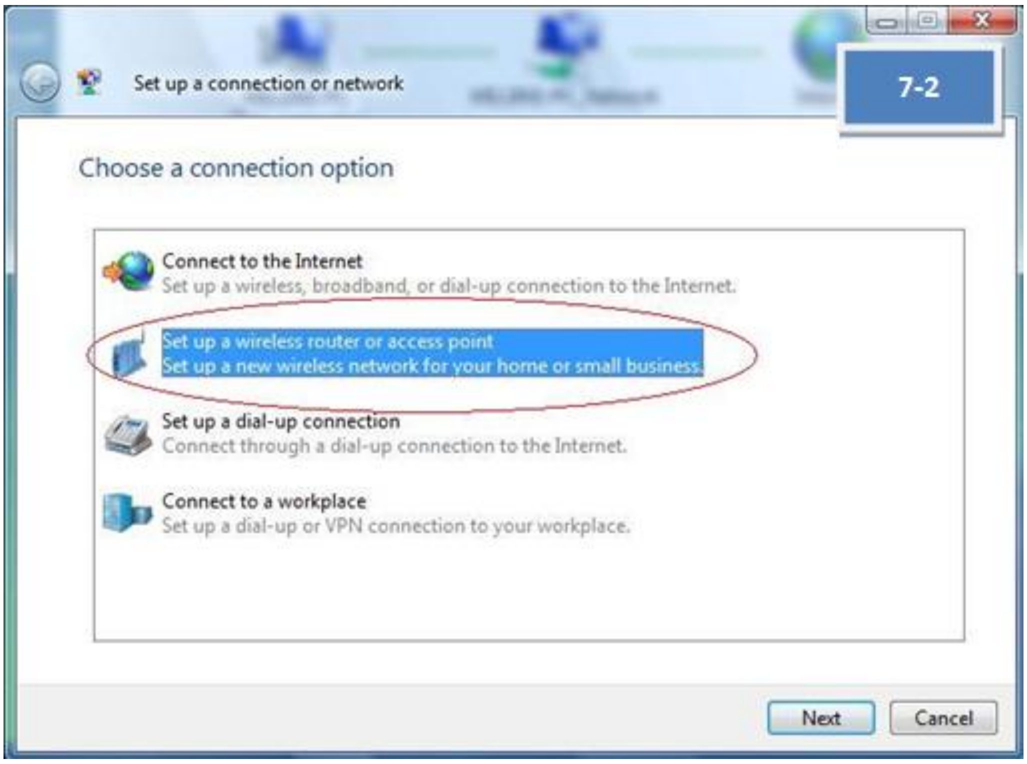
WILLING-PC_Network (Private network) Customize

Access	Local and Internet	
Connection	Local Area Connection	View status

Sharing and Discovery

Network discovery	● On
File sharing	● On
Public folder sharing	○ Off
Printer sharing	○ Off (no printers installed)
Password protected sharing	● On
Media sharing	○ Off

Show me all the files and folders I am sharing
Show me all the shared network folders on this computer



Set up a connection or network

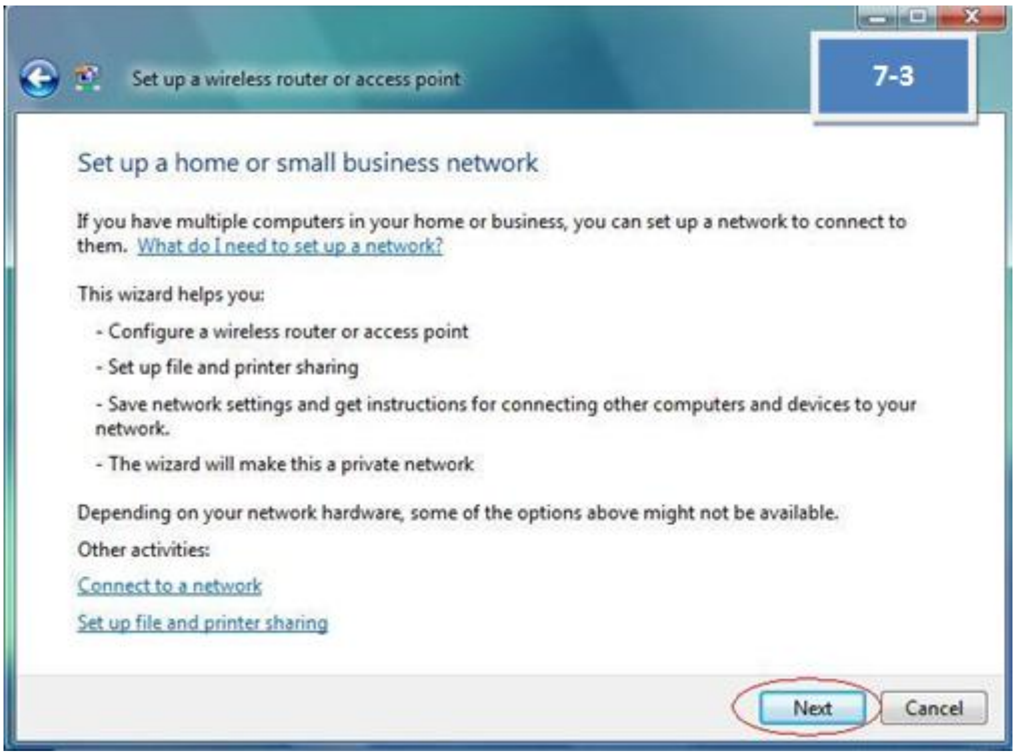
Choose a connection option

Connect to the Internet
Set up a wireless, broadband, or dial-up connection to the Internet.

Set up a wireless router or access point
Set up a new wireless network for your home or small business.

Set up a dial-up connection
Connect through a dial-up connection to the Internet.

Connect to a workplace
Set up a dial-up or VPN connection to your workplace.



Set up a wireless router or access point

Set up a home or small business network

If you have multiple computers in your home or business, you can set up a network to connect to them. What do I need to set up a network?

This wizard helps you:

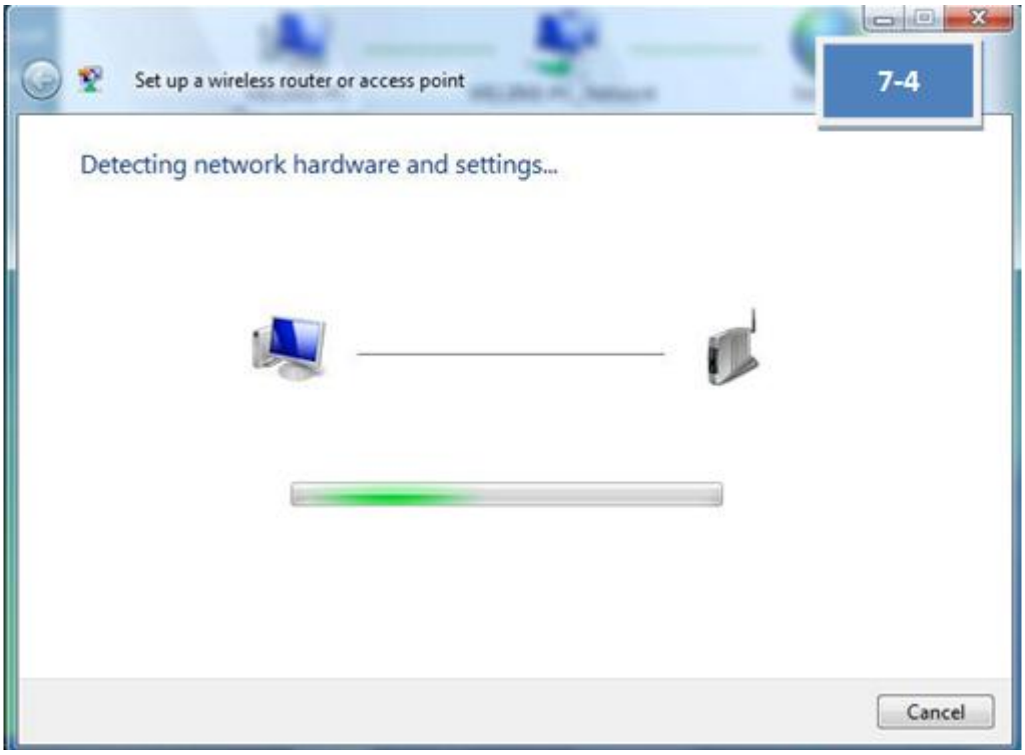
- Configure a wireless router or access point
- Set up file and printer sharing
- Save network settings and get instructions for connecting other computers and devices to your network.
- The wizard will make this a private network

Depending on your network hardware, some of the options above might not be available.

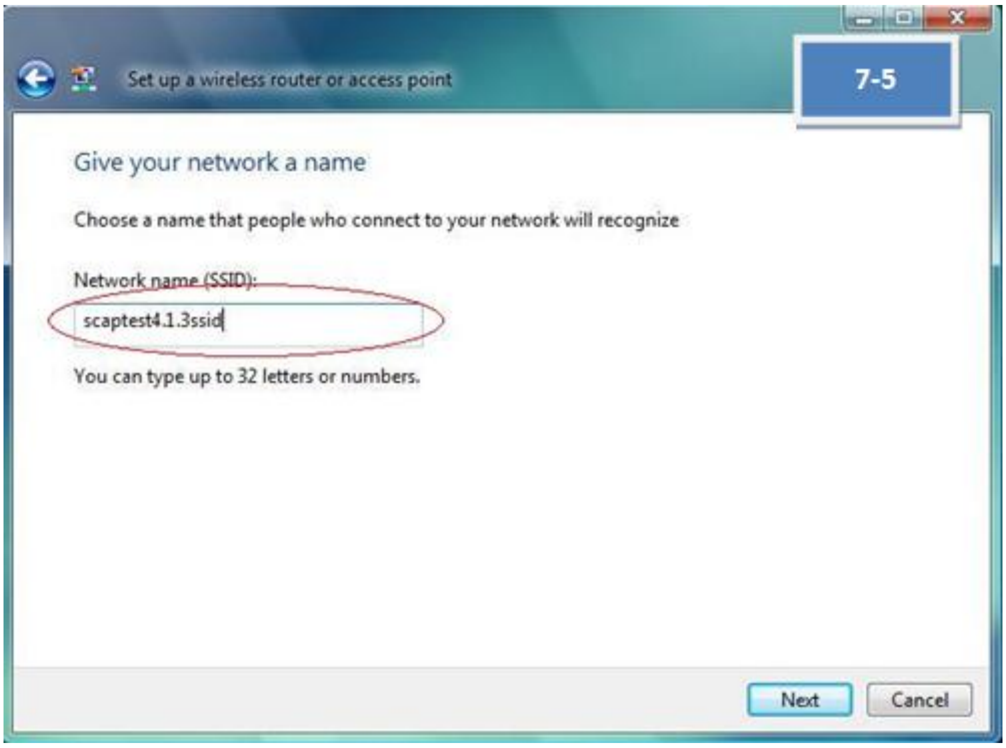
Other activities:

Connect to a network

Set up file and printer sharing



Detecting network hardware and settings...

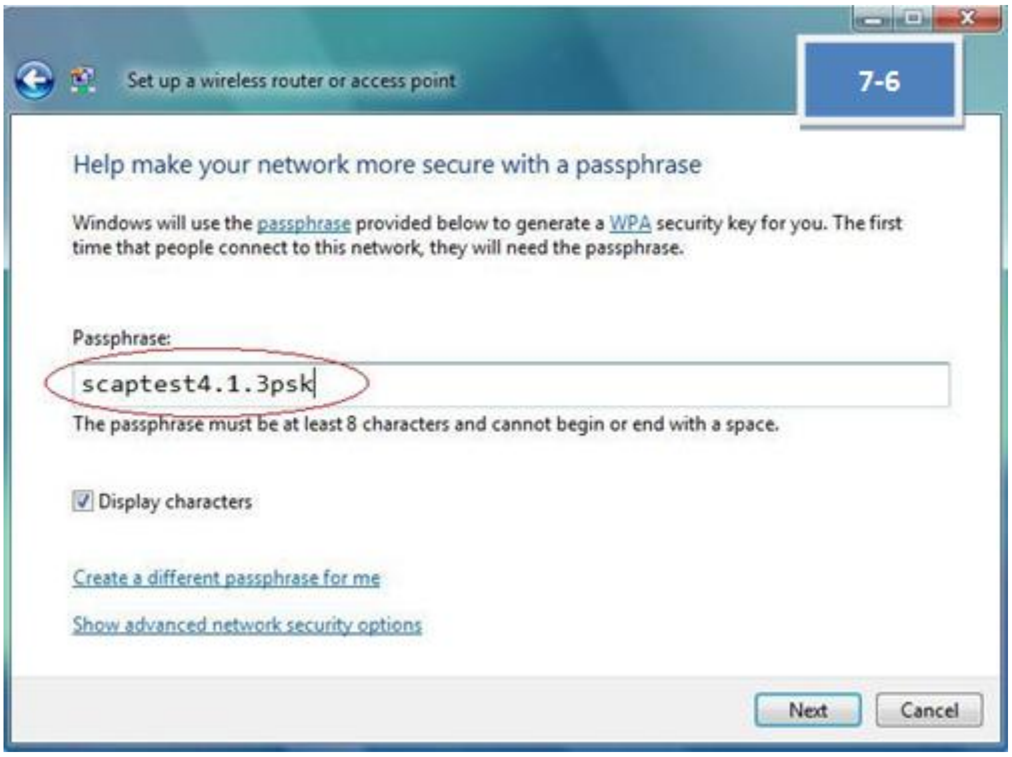


Give your network a name

Choose a name that people who connect to your network will recognize

Network name (SSID): scaptest4.1.3ssid

You can type up to 32 letters or numbers.


7-6
Set up a wireless router or access point

Help make your network more secure with a passphrase

Windows will use the passphrase provided below to generate a WPA security key for you. The first time that people connect to this network, they will need the passphrase.

Passphrase:

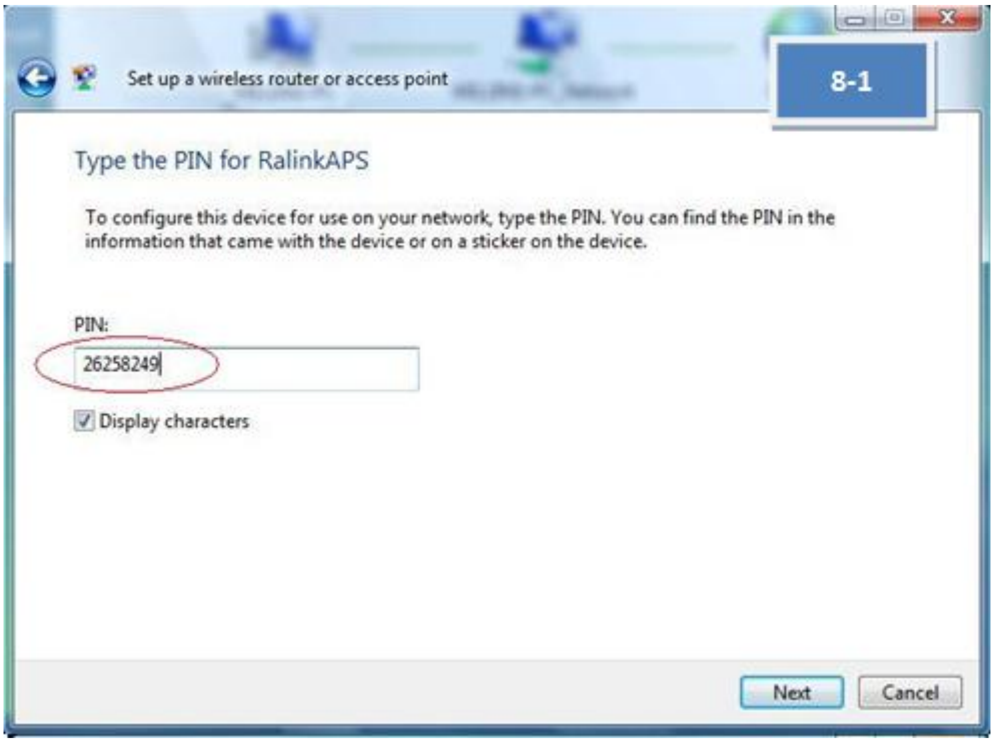
scaptest4.1.3psk

The passphrase must be at least 8 characters and cannot begin or end with a space.

☑ Display characters

Create a different passphrase for me
Show advanced network security options
Next
Cancel

As to details of step-8, please refer to the following figures from [8-1] to [8-2].



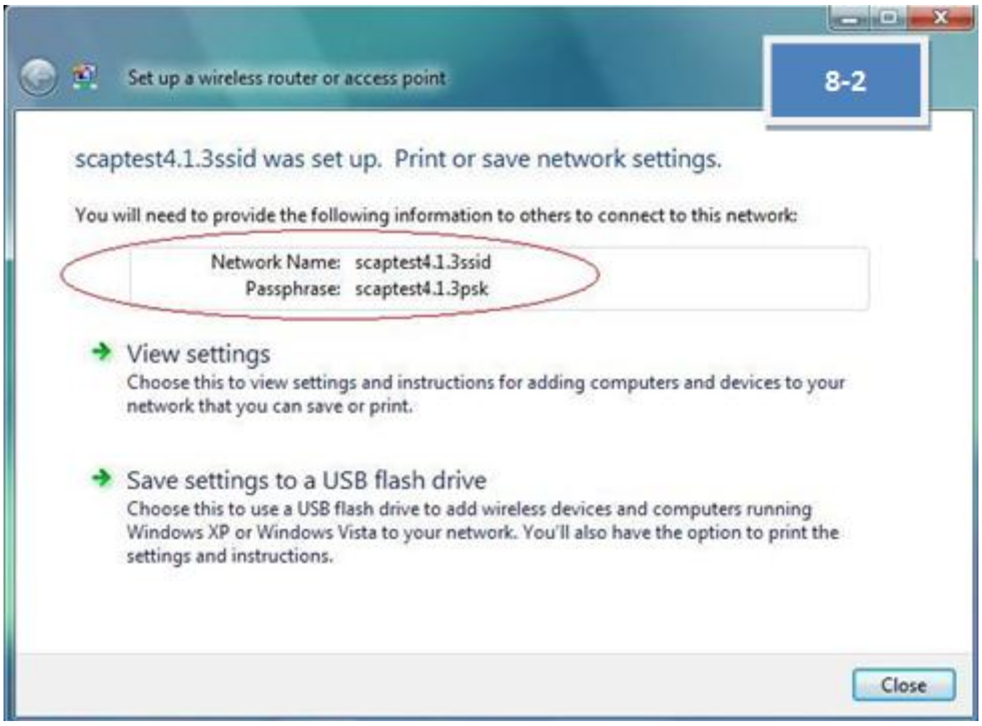
Type the PIN for RalinkAPS

To configure this device for use on your network, type the PIN. You can find the PIN in the information that came with the device or on a sticker on the device.

PIN:

26258249

☑ Display characters



scaptest4.1.3ssid was set up. Print or save network settings.

You will need to provide the following information to others to connect to this network:

Network Name: scaptest4.1.3ssid
Passphrase: scaptest4.1.3psk

View settings
Choose this to view settings and instructions for adding computers and devices to your network that you can save or print.

Save settings to a USB flash drive
Choose this to use a USB flash drive to add wireless devices and computers running Windows XP or Windows Vista to your network. You'll also have the option to print the settings and instructions.

6.5.4 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.

6.5.5 How to know WPS AP PinCode

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

6.5.6 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.

6.6 UPnP Daemon HOWTO

6.6.1 Requirements:

1. Linux platform
2. Ralink wireless driver version which support WPS
3. The UPnP library (libupnp)
   - ⇨ You can download the libupnp source code from the following URL.
     http://upnp.sourceforge.net/
   - ⇨ libupnp-1.3.1 is the preferred version.
     For other versions, you may need to patch our modification to the library by 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.

6.6.2 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

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 specification.

8 IEEE802.11h

8.1 TPC

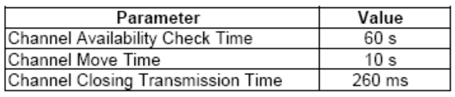
We do not support Transmission Power Control (TPC) and we provide a more flexible feature named Single SKU for fulfillment of the similar request.

8.2 DFS

Spectrum and Transmit Power Management

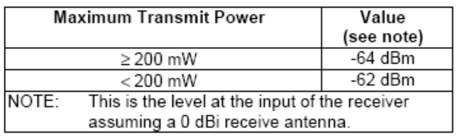
1. To turn on IEEE802.11h, just fill up the parameters of 'IEEE80211H', 'AutoChannelSelect' as 1, WirelessMode set as 3 to support A band. This parameter can work in only A band.
2. Use 'CSPeriod' to determine how many beacons before channel switch
3. Driver will turn off BBP tuning temporarily in radar 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



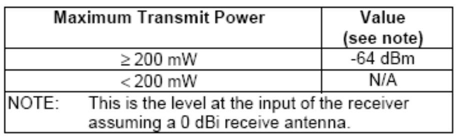
Parameter	Value
Channel Availability Check Time	60 s
Channel Move Time	10 s
Channel Closing Transmission Time	260 ms

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.	

8.3 Parameters in RT2860AP.dat

8.3.1 IEEE80211H

Description: Enable or disable IEEE 802.11h support (DFS)

Value:

    IEEE80211H=0

0: disable
1: enable

8.3.2 CSPeriod

Description: Configure how many Beacon (with Channel Switch Announcement IE) will be sent before changing to a new channel
Value:

CSPeriod=10

0 ~ 255

Note:
CS stands for Channel Switch and it default value is 10. Unit is Beacon count.

8.4 DFS iwpriv command

8.4.1 IEEE80211H

Description: Enable or disable IEEE 802.11h support (DFS)
Value:

IEEE80211H=0

0: disable
1: enable

8.5 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
```

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 RekeyMethod

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

```
RekeyMethod=DISABLE
```

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

9.3.4 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

9.3.5 PMKCachePeriod

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

```
PMKCachePeriod=10
```

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

9.3.6 WPAPSK

Description: WLAN security password for TKIP/AES
Value:

```
WPAPSK=01234567
```

8~63 ASCII characters
64 hexadecimal characters

9.3.7 DefaultKeyID

Description: Default key ID (WEP only)
Value:

DefaultKeyID=1

The ID range is 1~4

9.3.8 Key1Type

Description: Key 1 type
Value:

Key1Type=0

0: Hexadecimal
1: ASCII

9.3.9 Key1Str

Description: Key 1 string
Value:

Key1Str=

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.3.10 Key2Type

Description: Key 2 type
Value:

Key2Type=0

0: Hexadecimal
1: ASCII

9.3.11 Key2Str

Description: Key 2 string
Value:

Key2Str=

10 or 26 hexadecimal characters
5 or 13 ASCII characters

9.3.12 Key3Type

Description: Key 3 type

Value:

```
Key3Type=0

0:  Hexadecimal
1:  ASCII
```

9.3.13 Key3Str

Description: Key 3 string

Value:

```
Key3Str=

10 or 26 hexadecimal characters
5 or 13 ASCII characters
```

9.3.14 Key4Type

Description: Key 4 type

Value:

```
Key4Type=0

0:  Hexadecimal
1:  ASCII
```

9.3.15 Key4Str

Description: Key 4 string

Value:

```
Key4Str=

10 or 26 hexadecimal characters
5 or 13 ASCII characters
```

9.3.16 WpaMixPairCipher

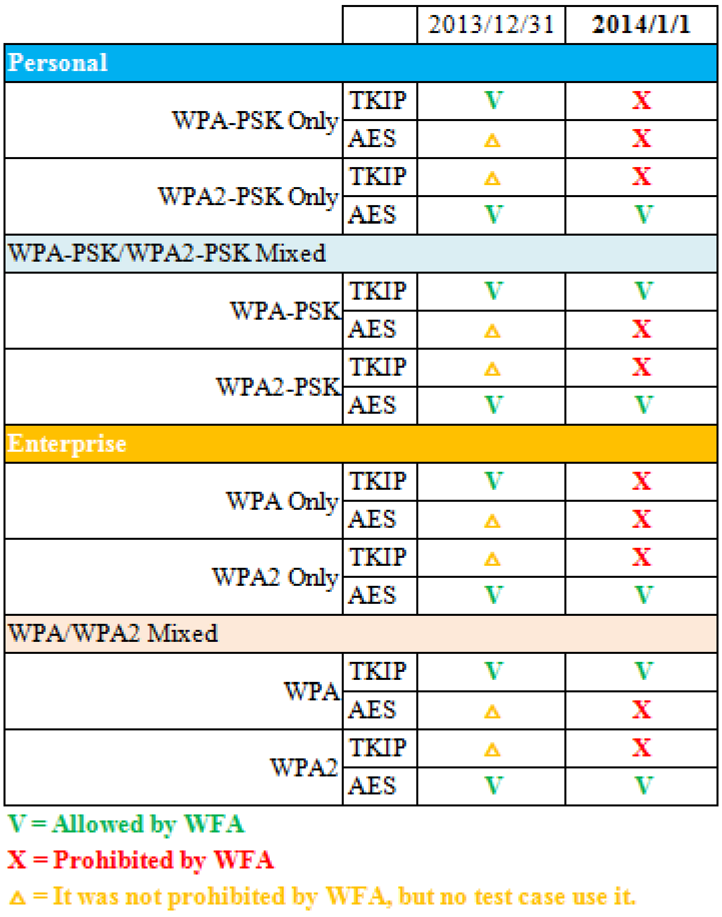
Description: Providing more flexible combination of cipher suite

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

9.4 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=WPAPSK
2. iwpriv ra0 set EncrypType=TKIP
3. iwpriv ra0 set SSID=myownssid
4. iwpriv ra0 set WPAPSK=myownpresharedkey
5. 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 SSID=MySsid
4. iwpriv ra0 set WPAPSK=MyPassword
5. iwpriv ra0 set SSID=MySsid

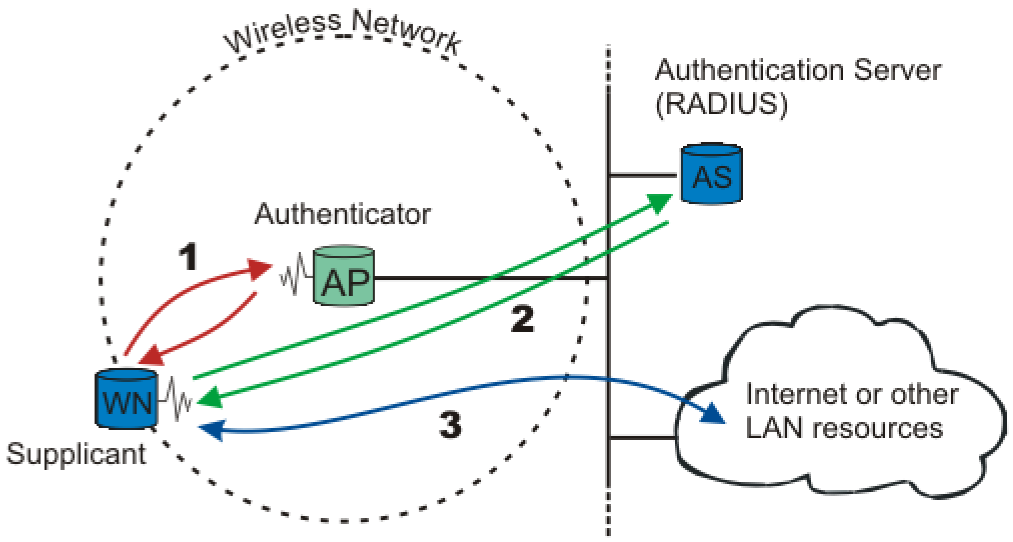
9.5.5 WPAPSKWPA2PSK/TKIPAES

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

10 Authenticator

IEEE Std. 802.1X-2001 is a standard for port-based network access control. It introduces an extensible mechanism for authenticating and authorizing users. There are 3 major components which includes **Supplicant**, **Authenticator** and **Authentication Server** (AS).

The following material is from http://tldp.org/HOWTO/html_single/8021X-HOWTO/.



Wireless Network
Authentication Server (RADIUS)
AS
Authenticator
AP
1
2
3
WN
Supplicant
Internet or other LAN resources

Figure: *A wireless node must be authenticated before it can gain access to other LAN resources.*

When a new wireless node (WN) requests access to a LAN resource, the access point (AP) asks for the WN's identity. No other traffic than EAP is allowed before the WN is authenticated.

The wireless node that requests authentication is often called **Supplicant**, although it is more correct to say that the wireless node contains a Supplicant. The Supplicant is responsible for responding to Authenticator data that will establish its credentials. The same goes for the access point; the access point contains an **Authenticator**. The Authenticator does not even need to be in the access point; it can be an external component.

After the identity has been sent, the authentication process begins.The protocol used between the Supplicant and the Authenticator is EAP, or more correctly, EAP encapsulation over LAN (**EAPoL**). The **Authenticator** re-encapsulates the EAP messages to Radius format, and passes them to the **Authentication Server**.

During authentication, the **Authenticator** just relays packets between the **Supplicant** and the **Authentication Server**. When the authentication process finishes, the Authentication Server sends a success message (or failure message if the authentication failed). The Authenticator then opens the "port" for the Supplicant. After a successful authentication, the Supplicant is granted access to other LAN resources or Internet.

10.1 Parameters in RT2860AP.dat

10.1.1 IEEE8021X

Description: Enable or disable 802.1X-WEP/802.1X-NONE mode
Value:

IEEE8021X=0

0: disable
1: enable

Note: It is enabled only when using Radius-WEP or Radius-NONE.

10.1.2 RADIUS_Server

Description: RADIUS server IP address configuration
Value:

RADIUS_Server=10.10.10.253

Note: IPv4 only

10.1.3 RADIUS_Port

Description: RADIUS server port number configuration
Value:

RADIUS_Port=1812

10.1.4 RADIUS_Key

Description: RADIUS key configuration
Value:

RADIUS_Key=password

10.1.5 own_ip_addr

Descripion: Configure SoftAP its own IP address
Value:

own_ip_addr=10.10.10.254

10.1.6 session_timeout_interval

Descripion: Configure the timeout interval for re-authentication
Value:

session_timeout_interval=120   (unit: second)

Note:

0: Disable re-authentication service

It must be larger than 60. Every session would be re-authenticated for a regular interval defined by this parameter.

10.1.7 PMKCachePeriod

Description: PMK Cache period configuration

Value:

PMKCachePeriod=10 (unit: minutes)

Note:

Default is 10 minutes.

10.1.8 EAPifname

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

Value:

EAPifname=

Example:

EAPifname=br0

Note:

Its default value is "br0”. However, if the wireless interface is not attached to the bridge interface or the name of the bridge interface is not "br0", please modify it.

10.1.9 PreAuth

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

Value:

PreAuth=0

0: disable

1: enable

10.1.10 PreAuthifname

Description: PreAuthifname is assigned as the binding interface for **WPA2** pre-authentication

Value:

PreAuthifname=

Example:

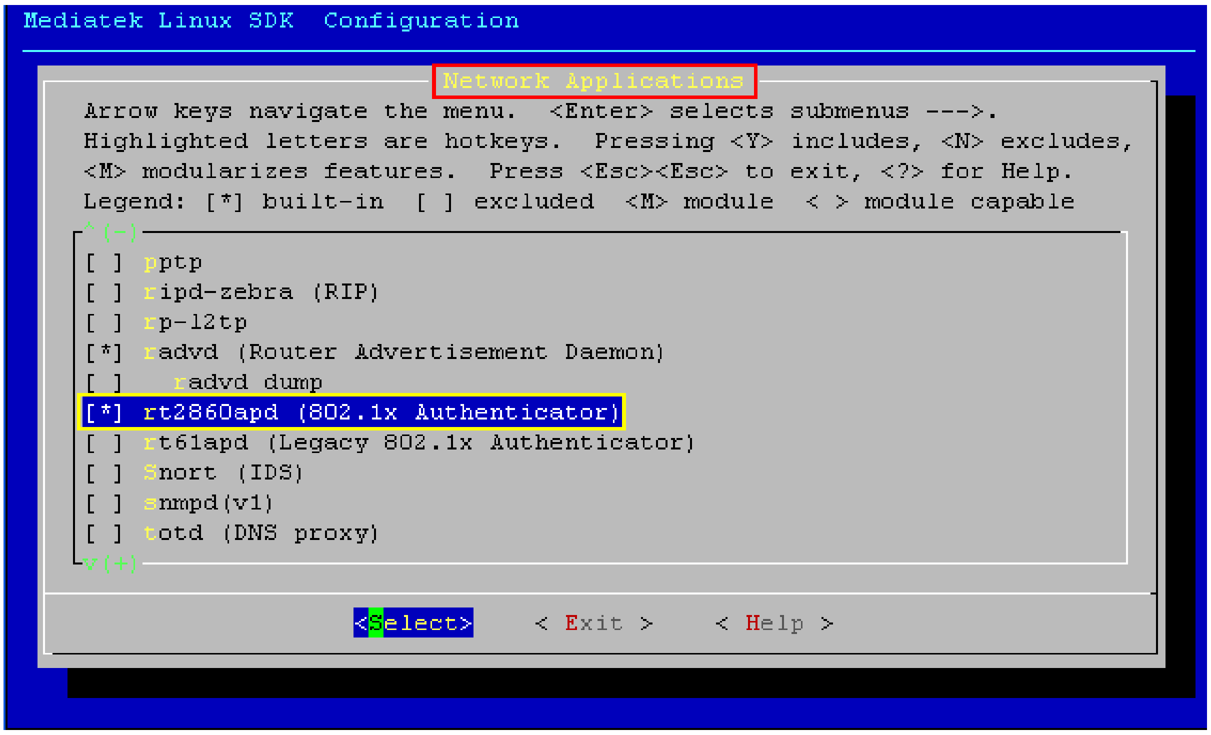
PreAuthifname=br0

10.2 rt2860apd

rt2860apd - IEEE 802.1X Authenticator (user space utility)
Source folder in reference SDK: **RT288x_SDK/source/user/802.1x**
Binary location in reference image: **/bin/rt2860apd**

rt2860apd implements part of IEEE 802.1X which helps the Authentication Server (AS) authorizing the Supplicant and also prove itself a valid Authenticator to AS. Please be noted that rt2860apd does not include the state machine for Key Management. Instead, the Key Management function is included in the wireless driver. Actually, rt2860apd relays EAP frames between the Supplicant and the AS. The port control entity is also implemented in the wireless driver.

10.2.1 How to turn on rt2860apd



```
Mediatek Linux SDK  Configuration

                         Network Applications
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help.
  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable
 ^(-)
 [ ] pptp
 [ ] ripd-zebra (RIP)
 [ ] rp-l2tp
 [*] radvd (Router Advertisement Daemon)
 [ ]   radvd dump
 [*] rt2860apd (802.1x Authenticator)
 [ ] rt61apd (Legacy 802.1x Authenticator)
 [ ] Snort (IDS)
 [ ] snmpd(v1)
 [ ] totd (DNS proxy)
 v(+)

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

10.2.2 How to configure rt2860apd

When rt2860apd starts, it will read settings from the driver profile (RT2860AP.dat). For any changes to make, you need to edit the configuration file, and then restart both the wireless interface and rt2860apd. Actually, the command “iwpriv ra0 set SSID=XXXX” would do the job.

The following four parameters in the configuration file are mandatory for rt2860apd. You should configure them correctly according to your own setup.

- RADIUS_Server='10.10.10.253'
- RADIUS_Port='1812'
- RADIUS_Key='password'
- own_ip_addr='10.10.10.254'

10.3 Multiple RADIUS Servers Support

As to MBSSID, you can use “;” to separate the settings for each BSSID. Example is as follows.

```
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
```

This implies,
The RADIUS server IP for ra0 is 192.168.2.1, its port is 1811 and its secret key is ralink_1.
The RADIUS server IP for ra1 is 192.168.2.2, its port is 1812 and its secret key is ralink_2.
The RADIUS server IP for ra2 is 192.168.2.3, its port is 1813 and its secret key is ralink_3.
The RADIUS server IP for ra3 is 192.168.2.4, its port is 1814 and its secret key is ralink_4.

Also, we have **Failover** mechanism and it means you can have a backup Radius server for each BSSID. Example is as follows. Both of them are written in the same profile.

```
<Default>
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

<Failover>
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
```

You may use iwpriv command to do the same thing for each BSSID.

```
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”
```

For backward compatibility, "RADIUS_Key" and “RADIUS_Key%d" are both accepted by the driver for key configuration. You may use either one of them but the paramter "RADIUS_Key" has higher priority.

```
<Default>
RADIUS_Key1=ralink_1          // ra0
RADIUS_Key2=ralink_2          // ra1
RADIUS_Key3=ralink_3          // ra2
RADIUS_Key4=ralink_4          // ra3

<Failover>
RADIUS_Key1=ralink_5
RADIUS_Key2=ralink_6
RADIUS_Key3=ralink_7
RADIUS_Key4=ralink_8
```

10.4 Enhanced Dynamic WEP Keying

In **Radius-WEP**, the authentication process also generates keys for both broadcast and unicast. The unicast key is unique for every individual client so it is always generated randomly by 802.1X daemon. However, the broadcast key is shared among all associated clients and it can be manually configured by User or still generated randomly by 802.1X daemon just like the unicast key does.

802.1X daemon would use the following parameters in RT2860AP.dat as material to “manually” generate the broadcast key.

- DefaultKeyID
- Key0Type, Key1Type, Key2Type, Key3Type
- Key0Str, Key1Str, Key2Str, Key3Str

The 802.1X daemon needs to read RT2860AP.dat to decide whether the broadcast key is generated randomly or not, but if the Key%dStr is empty or incorrectly configured, the broadcast key would be generated randomly by 802.1X daemon instead.

10.5 Examples

In the following examples, we all assume that DUT has IP address 192.168.1.138 and the Authentication Server has IP address 192.168.1.1. Also, we assume that Radius secret is “myownkey”.

10.5.1 Radius-None

```
RADIUS_Server=192.168.1.1
RADIUS_Port=1812
RADIUS_Key=myownkey
own_ip_addr=192.168.1.138
AuthMode=OPEN
EncrypType=NONE
IEEE8021X=1
```

10.5.2 Radius-WEP

```
RADIUS_Server=192.168.1.1
RADIUS_Port=1812
RADIUS_Key=myownkey
own_ip_addr=192.168.1.138
AuthMode=OPEN
EncrypType=WEP
IEEE8021X=1
```

10.5.3 WPA-TKIP

```
RADIUS_Server=192.168.1.1
RADIUS_Port=1812
```

```
RADIUS_Key=myownkey
own_ip_addr=192.168.1.138
AuthMode=WPA
EncrypType=TKIP
IEEE8021X=0
```

Note: Deprecated by WFA since 2014.01.01

10.5.4 WPA2-AES

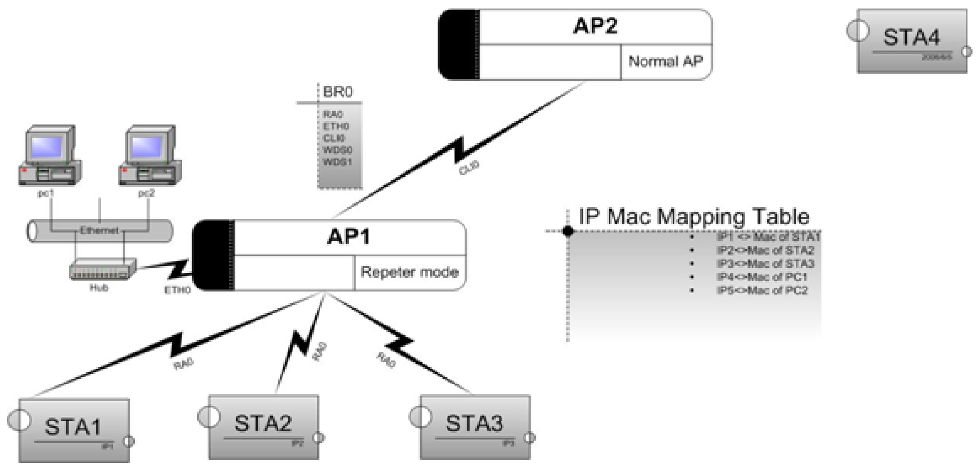
```
RADIUS_Server=192.168.1.1
RADIUS_Port=1812
RADIUS_Key=myownkey
own_ip_addr=192.168.1.138
AuthMode=WPA2
EncrypType=AES
IEEE8021X=0
```

10.5.5 WPA1WPA2-TKIPAES

```
RADIUS_Server=192.168.1.1
RADIUS_Port=1812
RADIUS_Key=myownkey
own_ip_addr=192.168.1.138
AuthMode=WPA1WPA2
EncrypType=TKIPAES
WpaMixPairCipher=WPA_TKIP_WPA2_AES
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 PIN mode WPS negotiation
Value:

```
iwpriv apcli0 set ApCliWscSsid=target_ssid
```

targer_ssid: 1~32 characters

Note: This must be configured in PIN mode and PIN mode only

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 Debug=3
iwpriv apcli0 set WscGenPinCode=1           // Generate PIN code
iwpriv apcli0 set Debug=0

iwpriv apcli0 set ApCliEnable=0
iwpriv apcli0 set WscConfMode=1             // Enrollee
iwpriv apcli0 set WscMode=1                 // PIN mode
iwpriv apcli0 set ApCliEnable=1
iwpriv apcli0 set ApCliWscSsid=<target_AP>  // SSID of the target WPS AP (must)
iwpriv apcli0 set WscGetConf=1              // Trigger
```

*Input the generated PIN code in the Registrar

11.5.2 PBC Mode

```
iwpriv apcli0 set ApCliEnable=0
iwpriv apcli0 set WscConfMode=1             // Enrollee
iwpriv apcli0 set WscMode=2                 // PBC mode
iwpriv apcli0 set ApCliEnable=1
iwpriv apcli0 set WscGetConf=1              // Trigger
```

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

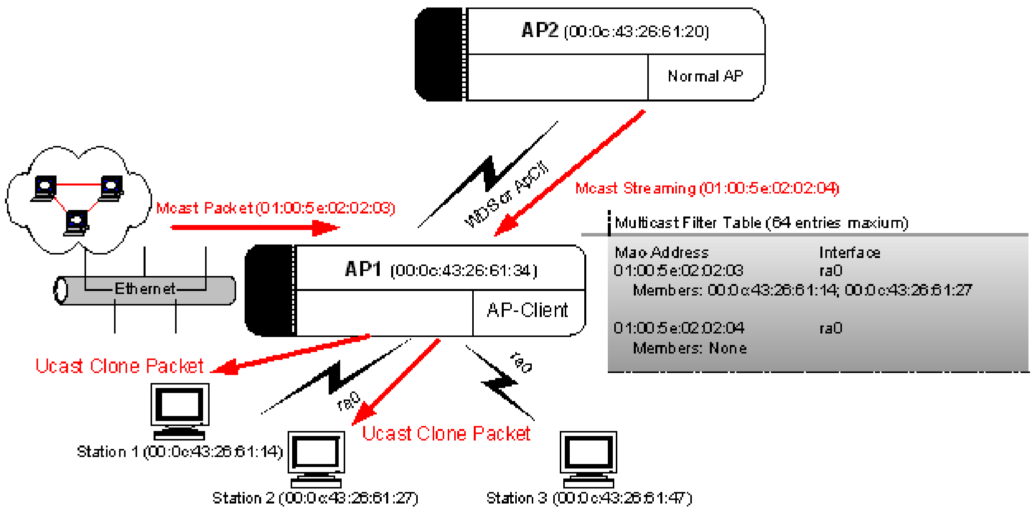
13.1 Basic

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


AP2 (00:0c:43:26:61:20)
Normal AP
WDS or APCli
Mcast Streaming (01:00:5e:02:02:04)
Mcast Packet (01:00:5e:02:02:03)
Ethernet
AP1 (00:0c:43:26:61:34)
AP-Client
Ucast Clone Packet
ra0
ra0
Ucast Clone Packet
Station 1 (00:0c:43:26:61:14)
Station 2 (00:0c:43:26:61:27)
Station 3 (00:0c:43:26:61:47)

Multicast Filter Table (64 entries maxium)

Mac Address	Interface
01:00:5e:02:02:03	ra0
Members: 00:0c:43:26:61:14; 00:0c:43:26:61:27	
01:00:5e:02:02:04	ra0
Members: None	

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

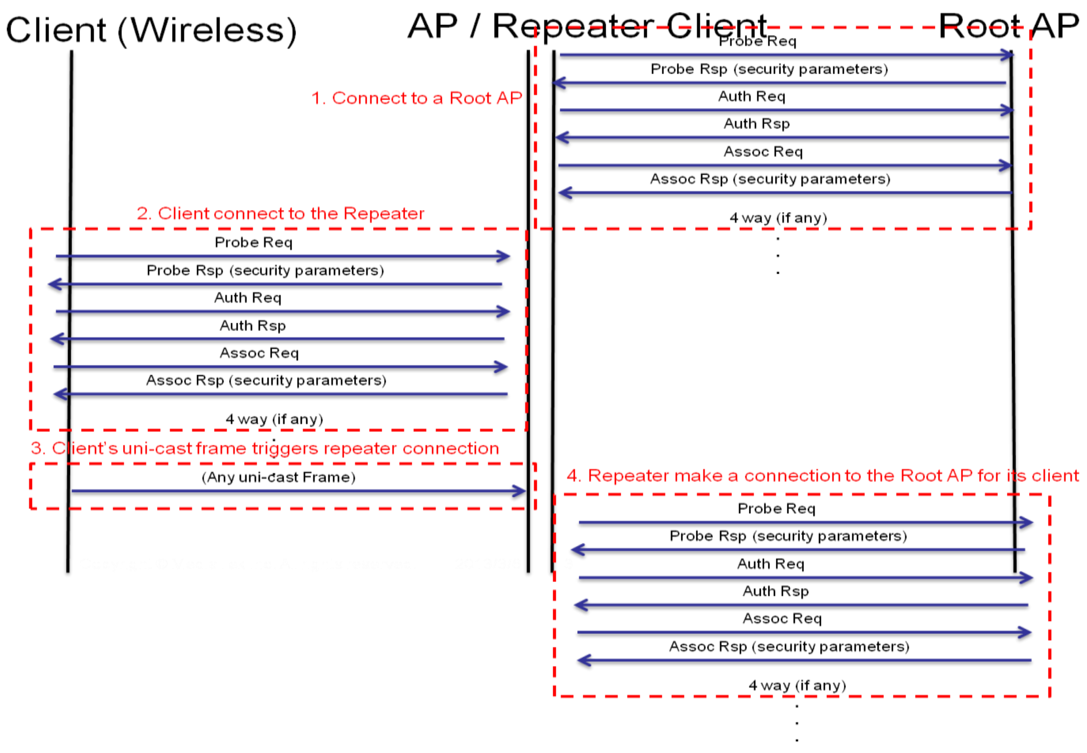
14.2.1 MACRepeaterEn

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

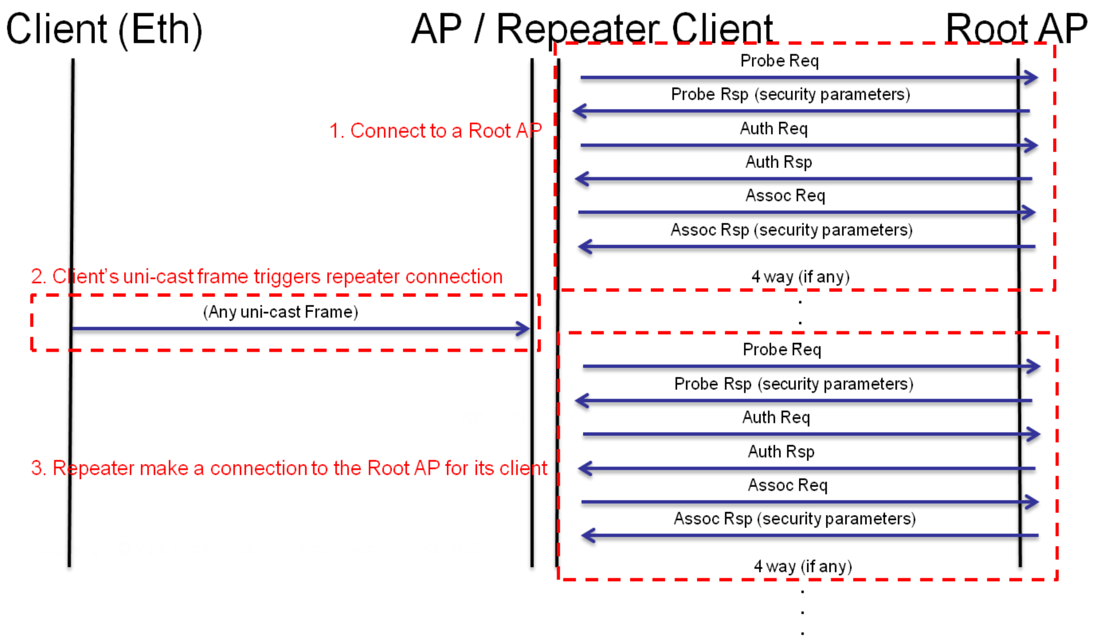
```
MACRepeaterEn=0

0: disable
1: enable
```

14.3 Management Frame Flow

14.3.1 Wireless client

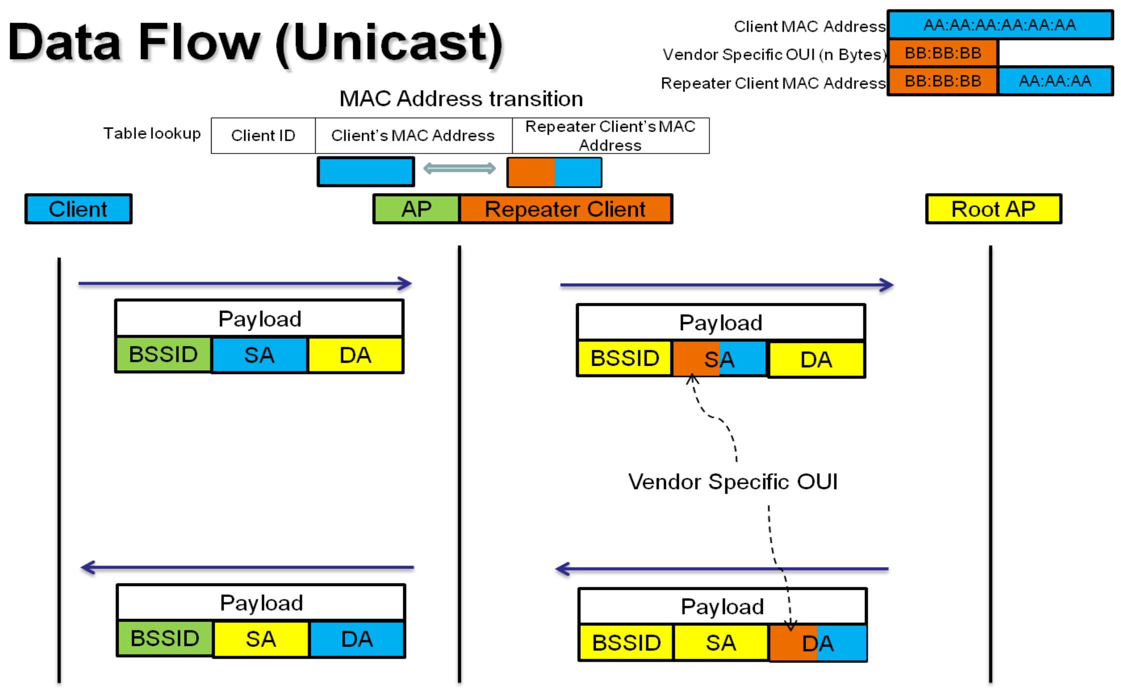

Client (Wireless)
AP / Repeater Client
Root AP
1. Connect to a Root AP
Probe Req
Probe Rsp (security parameters)
Auth Req
Auth Rsp
Assoc Req
Assoc Rsp (security parameters)
4 way (if any)
2. Client connect to the Repeater
Probe Req
Probe Rsp (security parameters)
Auth Req
Auth Rsp
Assoc Req
Assoc Rsp (security parameters)
4 way (if any)
3. Client's uni-cast frame triggers repeater connection
(Any uni-cast Frame)
4. Repeater make a connection to the Root AP for its client
Probe Req
Probe Rsp (security parameters)
Auth Req
Auth Rsp
Assoc Req
Assoc Rsp (security parameters)
4 way (if any)

14.3.2 Ethernet client

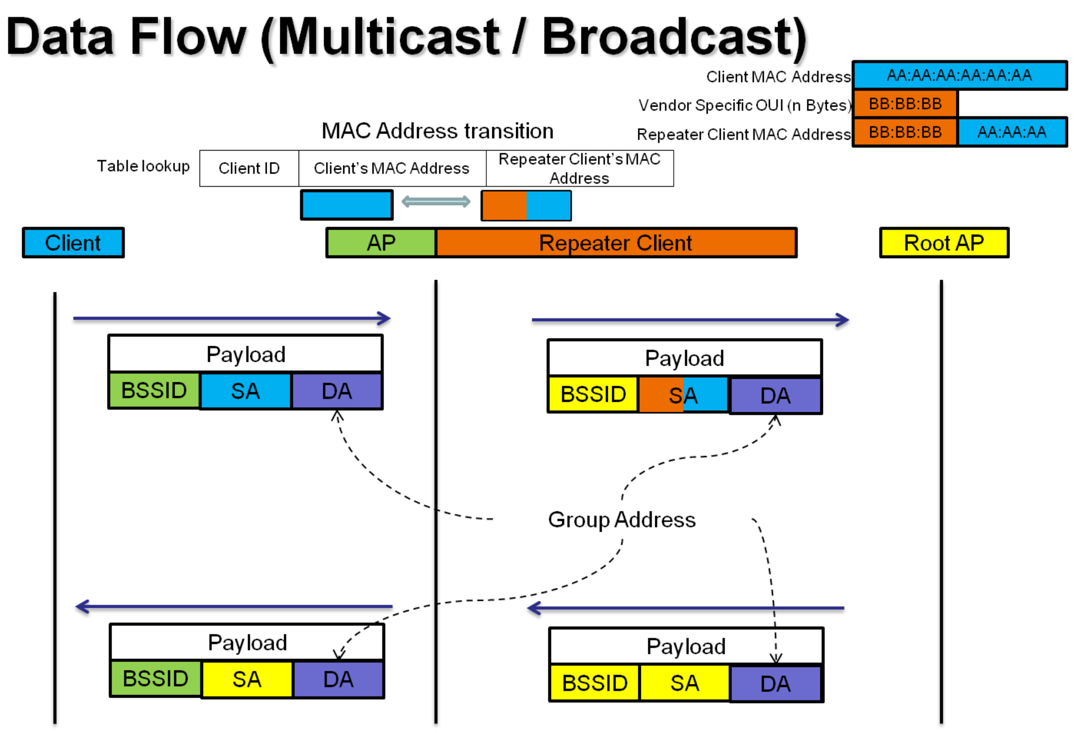

Client (Eth)
AP / Repeater Client
Root AP
1. Connect to a Root AP
Probe Req
Probe Rsp (security parameters)
Auth Req
Auth Rsp
Assoc Req
Assoc Rsp (security parameters)
4 way (if any)
2. Client's uni-cast frame triggers repeater connection
(Any uni-cast Frame)
3. Repeater make a connection to the Root AP for its client
Probe Req
Probe Rsp (security parameters)
Auth Req
Auth Rsp
Assoc Req
Assoc Rsp (security parameters)
4 way (if any)

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.

Note: Currently supported chips are MT7602E, MT7612E, MT7603E, MT7628.

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 AP.

<11ac dual band AP>

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

<11n dual band AP>

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

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**

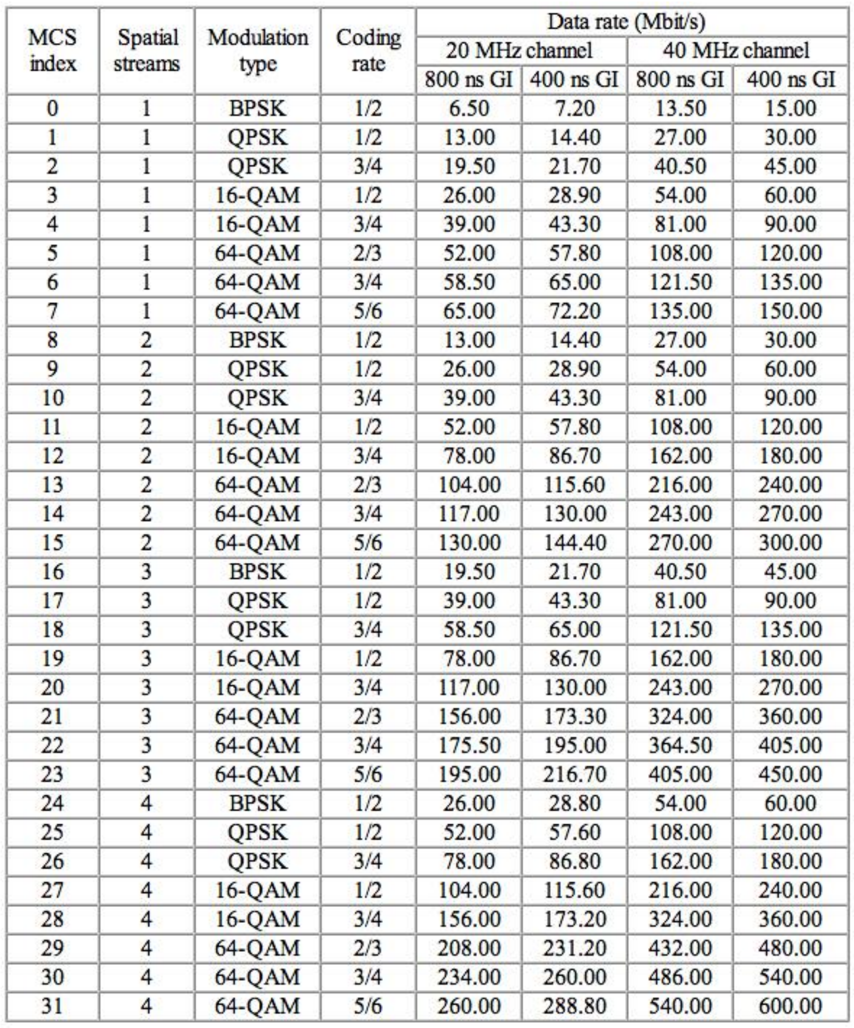
15.3.3 PMF Test Section 4.4

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 Fixed Rate

16.1 802.11n 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

16.2 2.4g

16.2.1 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

16.2.2 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

16.2.3 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.

16.2.4 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.

16.3 5g

16.3.1 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

16.3.2 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.

16.4 11ac

16.4.1 VHT Fixed Rate iwpriv command

16.4.1.1 fpga_on

Description: Turn on or off VHT fixed rate
Value:

```
iwpriv rai0 set fpga_on=6
```

0: Disable
6: Enable

16.4.1.2 dataphy

Description: PHY mode configuration
Value:

```
iwpriv rai0 set dataphy=4
```

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

16.4.1.3 databw

Description: Bandwidth configuration
Value:

```
iwpriv rai0 set databw=2

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

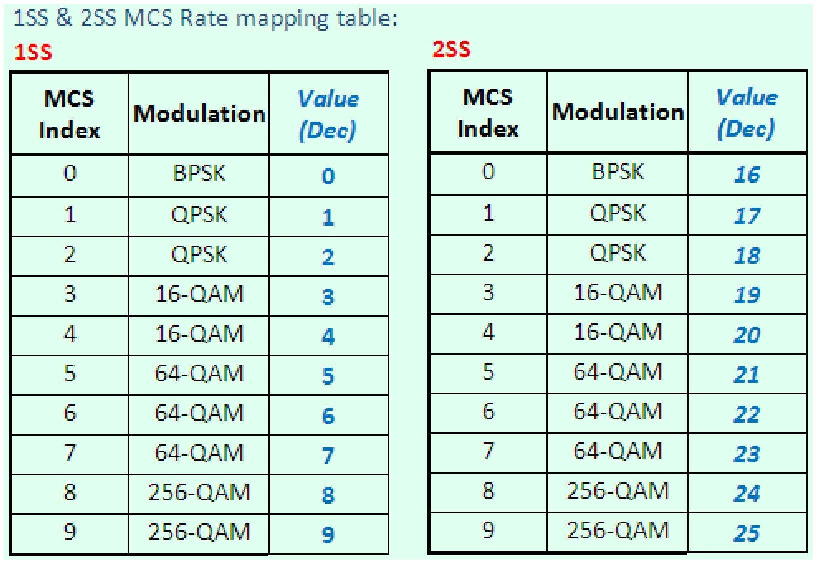
16.4.1.4 datamcs

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

16.4.1.5 datagi

Description: Guard Interval configuration

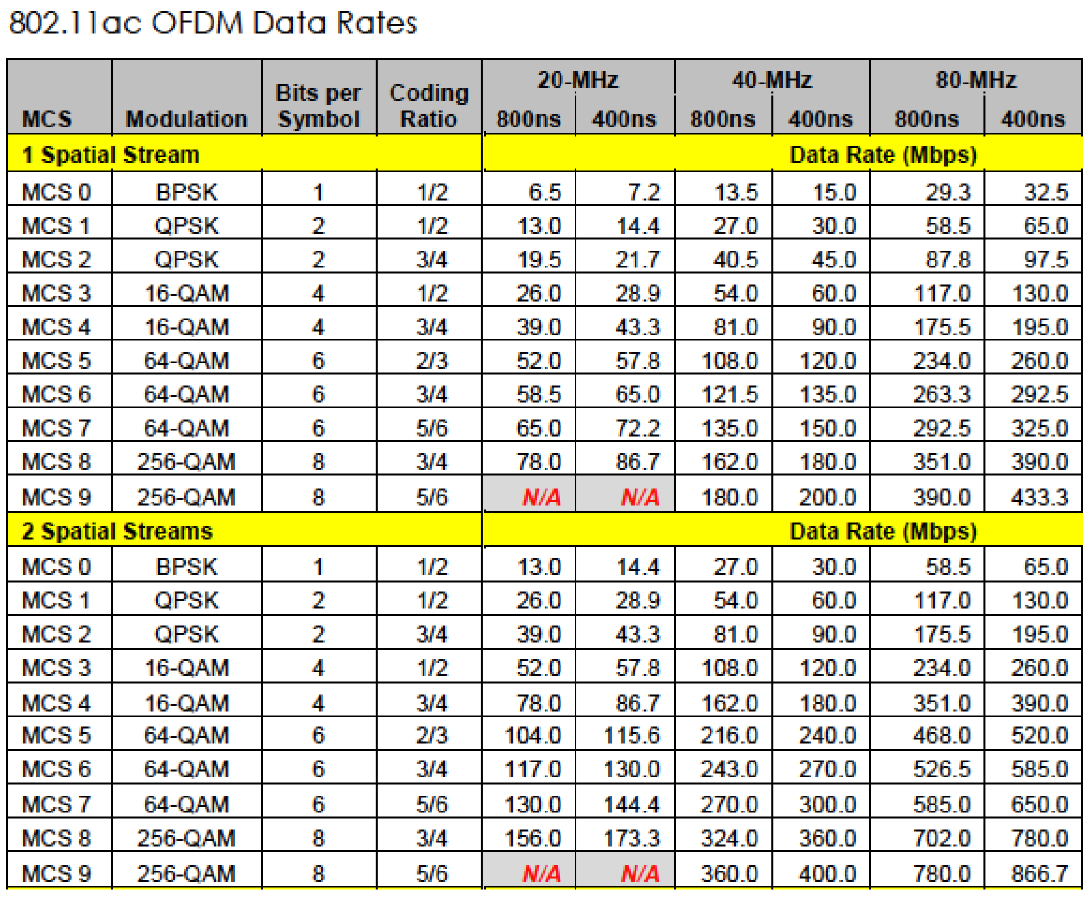
Value:

```
        iwpriv rai0 set datagi=0

        0 = Long GI
        1 = Short GI
```

16.4.2 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

				20-MHz		40-MHz		80-MHz	
MCS	Modulation	Bits per Symbol	Coding Ratio	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

17 ACL

Access Control List (ACL) provides a way to accomplish MAC address filtering. You can use this feature to implement White List or Black List.

17.1 Parameters in RT2860AP.dat

17.1.1 AccessPolicy0

Description: ACL access policy configuration for BSSID-0
Value:

```
AccessPolicy0=0

0: Disable
1: Allow all entries in the ACL table (white list)
2: Reject all entries in the ACL table (black list)
```

17.1.2 AccessControlList0

Description: ACL table entry configuration for BSSID-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: Maximum entry number is 64

17.1.3 AccessPolicy1

Description: ACL access policy configuration for BSSID-1
Value:

```
AccessPolicy1=0

0: Disable
1: Allow all entries in the ACL table (white list)
2: Reject all entries in the ACL table (black list)
```

17.1.4 AccessControlList1

Description: ACL table entry configuration for BSSID-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: Maximum entry number is 64

17.1.5 AccessPolicy2

Description: ACL access policy configuration for BSSID-2
Value:

AccessPolicy2=0

0: Disable
1: Allow all entries in the ACL table (white list)
2: Reject all entries in the ACL table (black list)

17.1.6 AccessControlList2

Description: ACL table entry configuration for BSSID-2
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: Maximum entry number is 64

17.1.7 AccessPolicy3

Description: ACL access policy configuration for BSSID-3
Value:

AccessPolicy3=0

0: Disable
1: Allow all entries in the ACL table (white list)
2: Reject all entries in the ACL table (black list)

17.1.8 AccessControlList3

Description: ACL table entry configuration for BSSID-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: Maximum entry number is 64

17.2 ACL iwpriv command

17.2.1 AccessPolicy

Description: ACL access policy configuration
Value:

```
iwpriv ra0 set AccessPolicy=0
```

0: Disable
1: Allow all entries in the ACL table (white list)
2: Reject all entries in the ACL table (black list)

17.2.2 ACLAddEntry

Description: Add new entry (MAC address) into ACL table
Value:

```
iwpriv ra0 set ACLAddEntry="xx:xx:xx:xx:xx:xx;yy:yy:yy:yy:yy:yy"
```

[MAC address];[MAC address];...;[MAC address]"

Note: Maximum entry number is 64

17.2.3 ACLDelEntry

Description: Remove old entry (MAC address) from ACL table
Value:

```
iwpriv ra0 set ACLDelEntry="xx:xx:xx:xx:xx:xx;yy:yy:yy:yy:yy:yy"
```

[MAC address];[MAC address];...;[MAC address]"

17.2.4 ACLClearAll

Description: Remove all entries from ACL table
Value:

```
iwpriv ra0 set ACLClearAll=1
```

17.2.5 ACLShowAll

Description: Dump all entries in ACL table
Value:

```
        iwpriv ra0 set ACLClearAll=1
```

17.3 ACL example

17.3.1 White List

```
iwpriv ra0 set AccessPolicy=1
iwpriv ra0 set ACLAddEntry="00:0c:43:28:aa:12;00:0c:43:28:aa:11;00:0c:43:28:aa:10"
iwpriv ra0 set ACLShowAll=1
```

17.3.2 Black List

```
iwpriv ra0 set AccessPolicy=2
iwpriv ra0 set ACLAddEntry="00:0c:43:28:aa:12;00:0c:43:28:aa:11;00:0c:43:28:aa:10"
iwpriv ra0 set ACLShowAll=1
```

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-	-		-

	access ible			
dot11AuthenticationAlgorithmsIndex	not-access ible	Y		N
dot11AuthenticationAlgorithm	read-only	Y		N
dot11AuthenticationAlgorithmsEnabl e	read-write	Y		N
dot11WEPDefaultKeysTable	not-access ible	-		-
dot11WEPDefaultKeysEntry	not-access ible	-		-
dot11WEPDefaultKeyIndex	not-access ible	Y		N
dot11WEPDefaultKeyValue	read-write	Y	OID_802_11_WEPDEFAULTKEYVALUE	Y
dot11WEPKeyMappingsTable	not-access ible	-		-
dot11WEPKeyMappingsEntry	not-access ible	-		-
dot11WEPKeyMappingIndex	not-access ible	N		N
dot11WEPKeyMappingAddress	read-create	N		N
dot11WEPKeyMappingWEPOn	read-create	N		N
dot11WEPKeyMappingValue	read-create	N		N
dot11WEPKeyMappingStatus	read-create	N		N
dot11PrivacyTable	not-access ible	-		
dot11PrivacyEntry	not-access ible	-		
dot11PrivacyInvoked	read-write	Y		N
dot11WEPDefaultKeyID	read-write	Y	OID_802_11_WEPDEFAULTKEYID	Y
dot11WEPKeyMappingLength	read-write	Y	RT_OID_802_11_WEPKEYMAPPINGLENGT H	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-access ible	-		
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_THRESHOL D	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-access ible	-		
dot11CountersEntry	not-access ible	-		
dot11TransmittedFragmentCount	read-only	Y	OID_802_11_STATISTICS	N
dot11MulticastTransmittedFrameCo unt	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-access ible	-		-
dot11GroupAddressesEntry	not-access ible	-		-

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-access ible	-		
dot11RegDomainsSupportEntry	not-access ible	-		
dot11RegDomainsSupportIndex	not-access ible	Y		N
dot11RegDomainsSupportValue	read-only	Y		N
dot11AntennasListTable	not-access ible	-		
dot11AntennasListEntry	not-access ible	-		
dot11AntennaListIndex	not-access ible	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-access ible	-		
dot11SupportedDataRatesTxEntry	not-access ible	-		
dot11SupportedDataRatesTxIndex	not-access ible	Y		N
dot11SupportedDataRatesTxValue	read-only	Y	OID_802_11_DESIRED_RATES	N
dot11SupportedDataRatesRxTable	not-access ible	-		
dot11SupportedDataRatesRxEntry	not-access ible	-		
dot11SupportedDataRatesRxIndex	not-access ible	Y	OID_802_11_DESIRED_RATES	
dot11SupportedDataRatesRxValue	read-only	Y		
dot11PhyOFDMTable	not-access ible	-		
dot11PhyOFDMEntry	not-access ible	-		
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_MANUFACTURER OUI	char oui[128];
0x0701	RT_OID_802_11_MANUFACTURER NAME	char name[128];
0x0702	RT_OID_802_11_RESOURCETYPEI DNAME	char name[128];
0x0703	RT_OID_802_11_PRIVACYOPTIONI MPLEMENTED	ULONG linfo;
0x0704	RT_OID_802_11_POWERMANAGE MENTMODE	ULONG linfo;

0x0705	OID_802_11_WEPDEFAULTKEYVAL UE	typedef struct _DefaultKeyIdxValue
		{
		UCHARKeyIdx;
		UCHARValue[16];
		}DefaultKeyIdxValue;

0x0706	OID_802_11_WEPDEFAULTKEYID	UCHARkeyid;
0x0707	RT_OID_802_11_WEPKEYMAPPIN GLENGTH	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 Q&A

20.1 Why does WPAPSK not work?

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

20.2 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.

20.3 How do I check my channel list?

Please check CountryRegion or CountryRegionABand.

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

Please use the following command.
# **iwpriv ra0 show driverinfo**

20.5 Can SoftAP support Antenna diversity?

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

20.6 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:

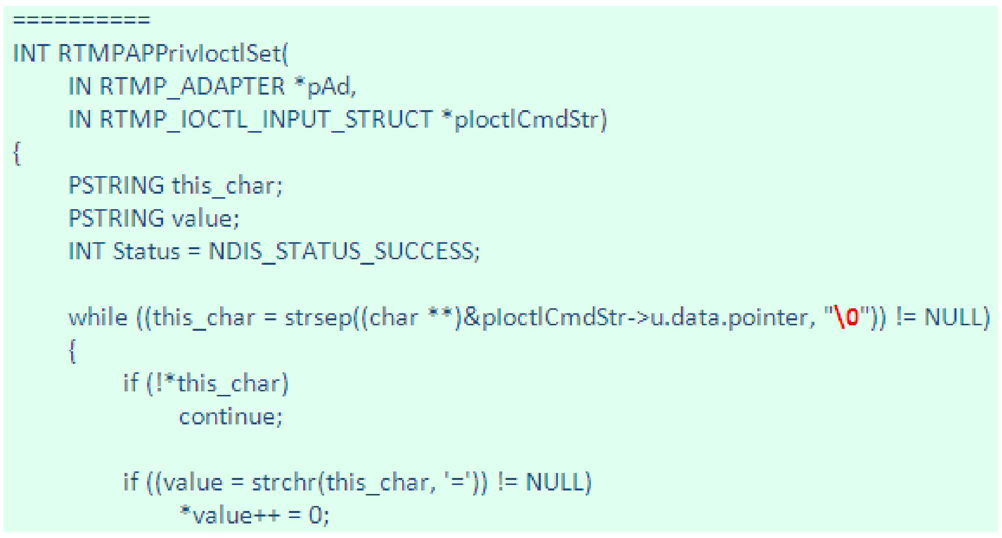
```
TxBurst=0
```

By iwpriv command:

```
iwpriv ra0 set TxBurst=0
```

20.7 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 *pIoctlCmdStr)
{
        PSTRING this_char;
        PSTRING value;
        INT Status = NDIS_STATUS_SUCCESS;

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

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

20.8 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.

20.9 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.

20.10 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.