WifiWave2 (7.12 and older)

Overview

This document applies to **7.12 and older**. The WifiWave2 package contains software for managing compatible 802.11ax and 802.11ac wave 2 wireless interfaces. New versions use the new wifi package and corresponding manual.

Builds for x86, ppc, mmips and tile architectures contain the configuration utilities needed to centrally manage interfaces (as a CAPsMAN controller). Builds for arm and arm64 also contain interface drivers and firmware.

The package can be downloaded as part of the 'Extra Packages' archive.

The WifiWave2 package in RouterOS adds certain Wave2 features, and 802.11ax devices require it. Some products which ship with the standard 'wireless' package, can replace it with wifiwave2, for more details, please see this section.

Configuration in the command line is done under /interface/wifiwave2/, when using a graphical configuration tool (WinBox or WebFig), wifiwave2 interfaces can be configured using either the 'Wireless' or 'QuickSet' tabs.

WifiWave2 Terminology

Before we move on let's familiarise ourselves with terms important for understanding the operation of the WifiWave2. These terms will be used throughout the article.

- **Profile** - refers to the configuration preset created under one of this WifiWave2 sub-menus: **aaa**, **channel**, **security**, **datapath**, or **interworking**.
- **Configuration profile** - configuration preset defined under /interface/wifiwave2/configuration, it can reference various profiles.
- **Station** - wireless client.

Basic Configuration:

Basic password-protected AP

```
/interface/wifiwave2
set wifi1 disabled=no configuration.country=Latvia configuration.ssid=MikroTik security.authentication-
types=wpa2-psk,wpa3-psk security.passphrase=8-63_characters
```

Open AP with OWE transition mode

Opportunistic wireless encryption (OWE) allows the creation of wireless networks that do not require the knowledge of a password to connect, but still offer the benefits of traffic encryption and management frame protection. It is an improvement on regular open access points.

However, since a network cannot be simultaneously encrypted and unencrypted, 2 separate interface configurations are required to offer connectivity to older devices that do not support OWE and offer the benefits of OWE to devices that do.

This configuration is referred to as OWE transition mode.

```
/interface/wifiwave2
add master-interface=wifi1 name=wifi1_owe configuration.ssid=MikroTik_OWE security.authentication-types=owe
security.owe-transition-interface=wifi1 configuration.hide-ssid=yes
set wifi1 configuration.country=Latvia configuration.ssid=MikroTik security.authentication-types="" security.
owe-transition-interface=wifi1_owe
enable wifi1,wifi1_owe
```

Client devices that support OWE will prefer the OWE interface. If you don't see any devices in your registration table that are associating with the regular open AP, you may want to move on from running a transition mode setup to a single OWE-encrypted interface.

Resetting configuration

WifiWave2 interface configurations can be reset by using the 'reset' command.

```
/interface/wifiwave2 reset wifi1
```

Configuration profiles

One of the new WifiWave2 additions is configuration profiles, you can create various presets, that can be assigned to interfaces as needed. Configuration settings for WifiWave2 are grouped in **profiles** according to the parameter sections found at end of this page - **aaa**, **channel**, **configuration**, **datapath**, **interworking**, and **security**, and can then be assigned to interfaces. **Configuration profiles** can include other profiles as well as separate parameters from other categories.

This optional flexibility is meant to allow each user to arrange their configuration in a way that makes the most sense for them, but it also means that each parameter may have different values assigned to it in different sections of the configuration.

The following priority determines, which value is used:

1. Value in interface settings
2. Value in a profile assigned to interface

3. Value in configuration profile assigned to interface
4. Value in a profile assigned to configuration profile (which in turn is assigned to interface).

If you are at any point unsure of which parameter value will be used for an interface, consult the **actual-configuration** menu. For an example of configuration profile usage, see the following example.

Example for dual-band home AP

```
# Creating a security profile, which will be common for both interfaces
/interface wifiwave2 security
add name=common-auth authentication-types=wpa2-psk,wpa3-psk passphrase="diceware makes good passwords"
wps=disable
# Creating a common configuration profile and linking the security profile to it
/interface wifiwave2 configuration
add name=common-conf ssid=MikroTik country=Latvia security=common-auth
# Creating separate channel configurations for each band
/interface wifiwave2 channel
add name=ch-2ghz frequency=2412,2432,2472 width=20mhz
add name=ch-5ghz frequency=5180,5260,5500 width=20/40/80mhz
# Assigning to each interface the common profile as well as band-specific channel profile
/interface wifiwave2
set wifi1 channel=ch-2ghz configuration=common-conf disabled=no
set wifi2 channel=ch-5ghz configuration=common-conf disabled=no

/interface/wifiwave2/actual-configuration print
 0 name="wifi1" mac-address=74:4D:28:94:22:9A arp-timeout=auto radio-mac=74:4D:28:94:22:9A
   configuration.ssid="MikroTik" .country=Latvia
   security.authentication-types=wpa2-psk,wpa3-psk .passphrase="diceware makes good passwords" .wps=disable
   channel.frequency=2412,2432,2472 .width=20mhz

 1 name="wifi2" mac-address=74:4D:28:94:22:9B arp-timeout=auto radio-mac=74:4D:28:94:22:9B
   configuration.ssid="MikroTik" .country=Latvia
   security.authentication-types=wpa2-psk,wpa3-psk .passphrase="diceware makes good passwords" .wps=disable
   channel.frequency=5180,5260,5500 .width=20/40/80mhz
```

Access List

The access list provides multiple ways of filtering and managing wireless connections.

RouterOS will check each new connection to see if its parameters match the parameters specified in any access list rule.

The rules are checked in the order they appear in the list. Only management actions specified in the first matching rule are applied to each connection.

Connections, which have been accepted by an access list rule, will be periodically checked, to see if they remain within the permitted **time** and **signal-range**. If they do not, they will be terminated.

⚠ Take care when writing access list rules which reject clients. After being repeatedly rejected by an AP, a client device may start avoiding it.

The access list has two kinds of parameters - filtering, and action. Filtering properties are only used for matching clients, to whom the access list rule should be applied to. Action parameters can change connection parameters for that specific client and potentially overriding its default connection parameters with ones specified in the access list rule.

MAC address authentication

Implemented through the **query-radius** action, MAC address authentication is a way to implement a centralized whitelist of client MAC addresses using a RADIUS server.

When a client device tries to associate with an AP, which is configured to perform MAC address authentication, the AP will send an access-request message to a RADIUS server with the device's MAC address as the user name and an empty password. If the RADIUS server answers with access-accept to such a request, the AP proceeds with whatever regular authentication procedure (passphrase or EAP authentication) is configured for the interface.

Access rule examples

Only accept connections to guest network from nearby devices during business hours

```
/interface/wifiwave2/access-list/print detail
Flags: X - disabled
 0   signal-range=-60..0 allow-signal-out-of-range=5m ssid-regexp="MikroTik Guest" time=7h-19h,mon,tue,wed,thu,
fri action=accept

 1   ssid-regexp="MikroTik Guest" action=reject
```

Reject connections from locally-administered ('anonymous'/'randomized') MAC addresses

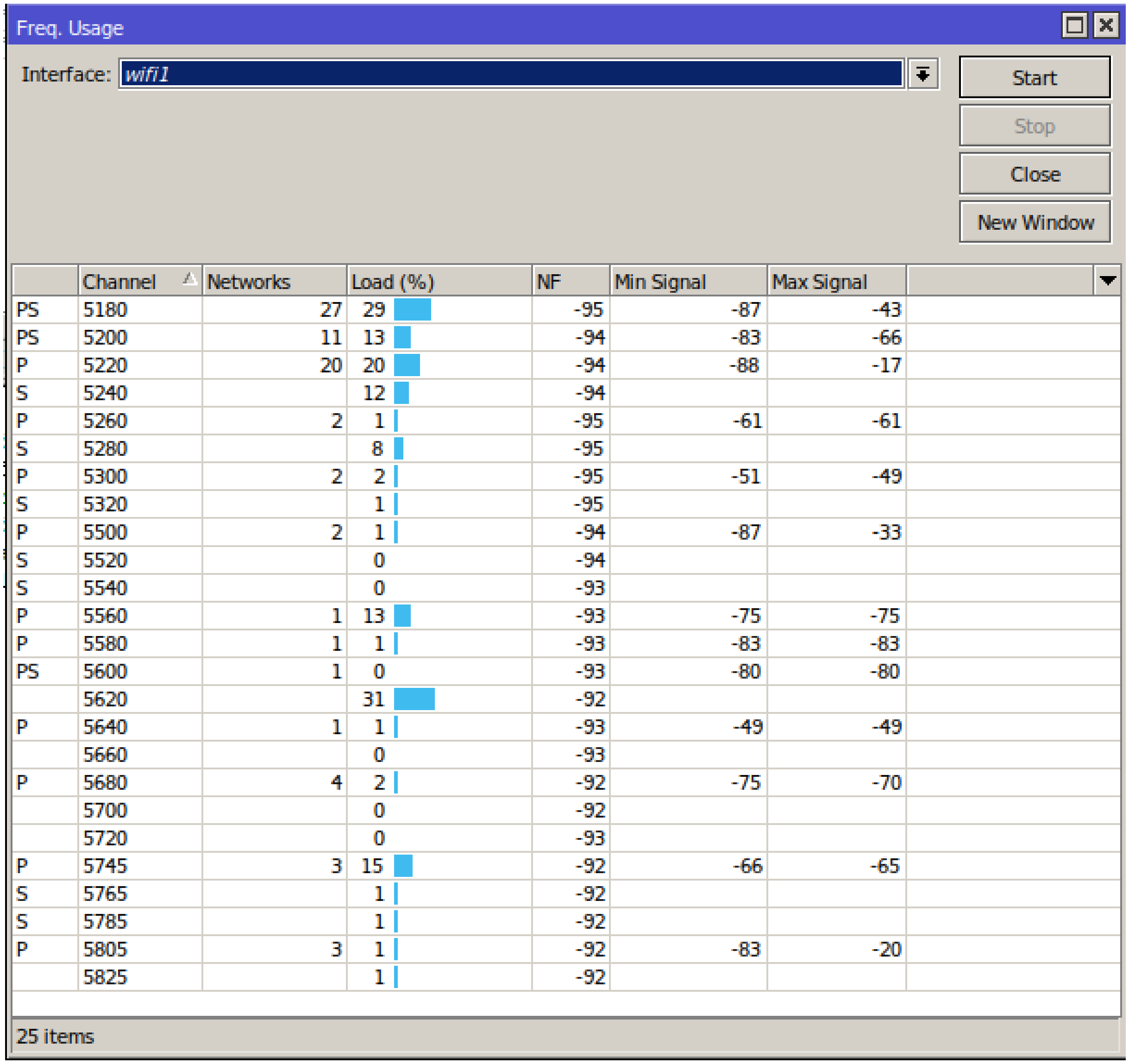
```
/interface/wifiwave2/access-list/print detail
Flags: X - disabled
 0   mac-address=02:00:00:00:00:00 mac-address-mask=02:00:00:00:00:00 action=reject
```

Assigning a different passphrase for a specific client can be useful, if you need to provide wireless access to a client, but don't want to share your wireless password, or don't want to create a separate SSID. When the matching client will connect to this network, instead of using the password defined in the interface configuration, the access list will make that client use a different password. Just make that the specific client doesn't get matched by a more generic access list rule first.

```
/interface wifiwave2 access-list
add action=accept disabled=no mac-address=22:F9:70:E5:D2:8E interface=wifi1 passphrase=StrongPassword
```

Frequency scan

The '/interface/wifiwave2/frequency-scan wifi1' command provides information about RF conditions on available channels that can be obtained by running the frequency-scan command. Used to approximate the spectrum usage, it can be useful to find less crowded frequencies.



	Channel	Networks	Load (%)	NF	Min Signal	Max Signal
PS	5180	27	29	-95	-87	-43
PS	5200	11	13	-94	-83	-66
P	5220	20	20	-94	-88	-17
S	5240		12	-94		
P	5260	2	1	-95	-61	-61
S	5280		8	-95		
P	5300	2	2	-95	-51	-49
S	5320		1	-95		
P	5500	2	1	-94	-87	-33
S	5520		0	-94		
S	5540		0	-93		
P	5560	1	13	-93	-75	-75
P	5580	1	1	-93	-83	-83
PS	5600	1	0	-93	-80	-80
	5620		31	-92		
P	5640	1	1	-93	-49	-49
	5660		0	-93		
P	5680	4	2	-92	-75	-70
	5700		0	-92		
	5720		0	-93		
P	5745	3	15	-92	-66	-65
S	5765		1	-92		
S	5785		1	-92		
P	5805	3	1	-92	-83	-20
	5825		1	-92		

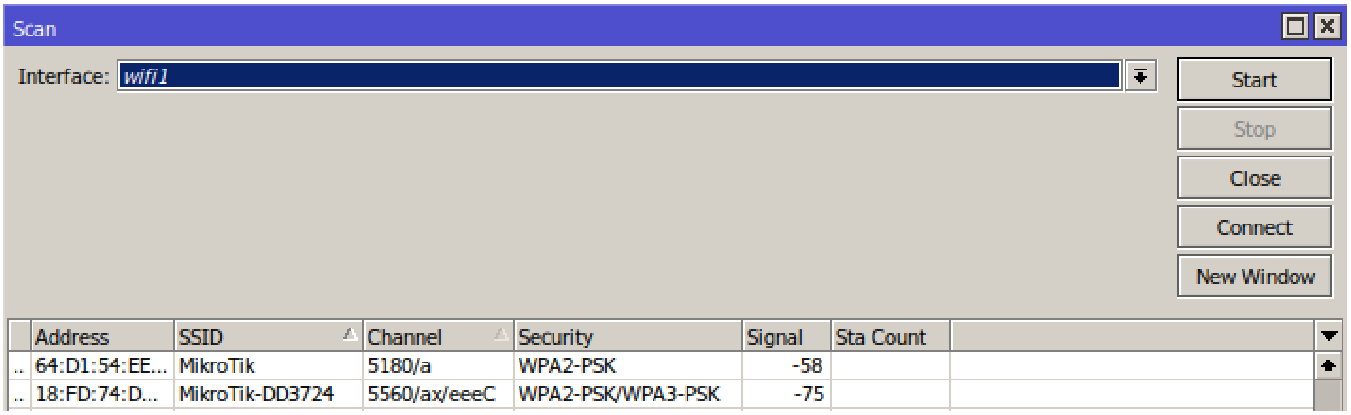
25 items

ⓘ Running a frequency scan will disconnect all connected clients, or if the interface is in station mode, it will disconnect from AP.

Scan command

The '/interface wifiwave2 scan' command will scan for access points and print out information about any APs it detects. It doesn't show the frequency usage, per channel, but it will reveal all access points that are transmitting. You can use the "connect" button, to initiate a connection to a specific AP.

The scan command takes all the same parameters as the frequency-scan command.



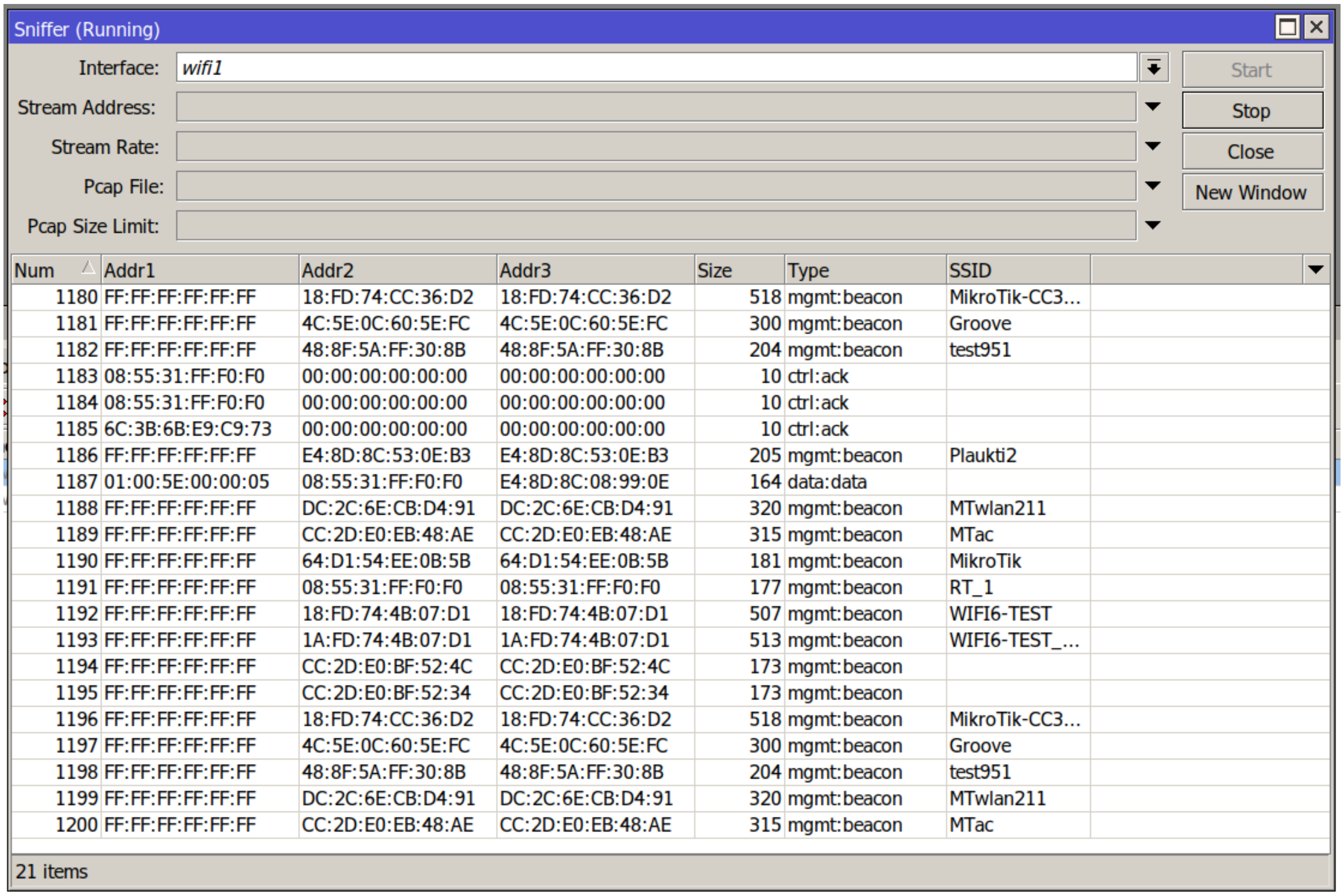
Interface: wifi1

	Address	SSID	Channel	Security	Signal	Sta Count
..	64:D1:54:EE...	MikroTik	5180/a	WPA2-PSK	-58	
..	18:FD:74:D...	MikroTik-DD3724	5560/ax/eeeC	WPA2-PSK/WPA3-PSK	-75	

Sniffer

The sniffer command enables monitor mode on a wireless interface. This turns the interface into a passive receiver for all WiFi transmissions.
The command continuously prints out information on received packets and can save them locally to a pcap file or stream them using the TZSP protocol.

The sniffer will operate on whichever channel is configured for the chosen interface.



Interface: wifi1
Stream Address:
Stream Rate:
Pcap File:
Pcap Size Limit:

Num	Addr1	Addr2	Addr3	Size	Type	SSID
1180	FF:FF:FF:FF:FF:FF	18:FD:74:CC:36:D2	18:FD:74:CC:36:D2	518	mgmt:beacon	MikroTik-CC3...
1181	FF:FF:FF:FF:FF:FF	4C:5E:0C:60:5E:FC	4C:5E:0C:60:5E:FC	300	mgmt:beacon	Groove
1182	FF:FF:FF:FF:FF:FF	48:8F:5A:FF:30:8B	48:8F:5A:FF:30:8B	204	mgmt:beacon	test951
1183	08:55:31:FF:F0:F0	00:00:00:00:00:00	00:00:00:00:00:00	10	ctrl:ack	
1184	08:55:31:FF:F0:F0	00:00:00:00:00:00	00:00:00:00:00:00	10	ctrl:ack	
1185	6C:3B:6B:E9:C9:73	00:00:00:00:00:00	00:00:00:00:00:00	10	ctrl:ack	
1186	FF:FF:FF:FF:FF:FF	E4:8D:8C:53:0E:B3	E4:8D:8C:53:0E:B3	205	mgmt:beacon	Plaukti2
1187	01:00:5E:00:00:05	08:55:31:FF:F0:F0	E4:8D:8C:08:99:0E	164	data:data	
1188	FF:FF:FF:FF:FF:FF	DC:2C:6E:CB:D4:91	DC:2C:6E:CB:D4:91	320	mgmt:beacon	MTwlan211
1189	FF:FF:FF:FF:FF:FF	CC:2D:E0:EB:48:AE	CC:2D:E0:EB:48:AE	315	mgmt:beacon	MTac
1190	FF:FF:FF:FF:FF:FF	64:D1:54:EE:0B:5B	64:D1:54:EE:0B:5B	181	mgmt:beacon	MikroTik
1191	FF:FF:FF:FF:FF:FF	08:55:31:FF:F0:F0	08:55:31:FF:F0:F0	177	mgmt:beacon	RT_1
1192	FF:FF:FF:FF:FF:FF	18:FD:74:4B:07:D1	18:FD:74:4B:07:D1	507	mgmt:beacon	WIFI6-TEST
1193	FF:FF:FF:FF:FF:FF	1A:FD:74:4B:07:D1	1A:FD:74:4B:07:D1	513	mgmt:beacon	WIFI6-TEST_...
1194	FF:FF:FF:FF:FF:FF	CC:2D:E0:BF:52:4C	CC:2D:E0:BF:52:4C	173	mgmt:beacon	
1195	FF:FF:FF:FF:FF:FF	CC:2D:E0:BF:52:34	CC:2D:E0:BF:52:34	173	mgmt:beacon	
1196	FF:FF:FF:FF:FF:FF	18:FD:74:CC:36:D2	18:FD:74:CC:36:D2	518	mgmt:beacon	MikroTik-CC3...
1197	FF:FF:FF:FF:FF:FF	4C:5E:0C:60:5E:FC	4C:5E:0C:60:5E:FC	300	mgmt:beacon	Groove
1198	FF:FF:FF:FF:FF:FF	48:8F:5A:FF:30:8B	48:8F:5A:FF:30:8B	204	mgmt:beacon	test951
1199	FF:FF:FF:FF:FF:FF	DC:2C:6E:CB:D4:91	DC:2C:6E:CB:D4:91	320	mgmt:beacon	MTwlan211
1200	FF:FF:FF:FF:FF:FF	CC:2D:E0:EB:48:AE	CC:2D:E0:EB:48:AE	315	mgmt:beacon	MTac

21 items

WPS

WPS client

The wps-client command enables obtaining authentication information from a WPS-enabled AP.

```
/interface/wifiwave2/wps-client wifi1
```

WPS server

An AP can be made to accept WPS authentication by a client device for 2 minutes by running the following command.

```
/interface/wifiwave2 wps-push-button wifi1
```

Radios

Information about the capabilities of each radio can be gained by running the `/interface/wifiwave2/radio print detail` command. It can be useful to see what bands are supported by the interface and what channels can be selected. The country profile that is applied to the interface will influence the results.

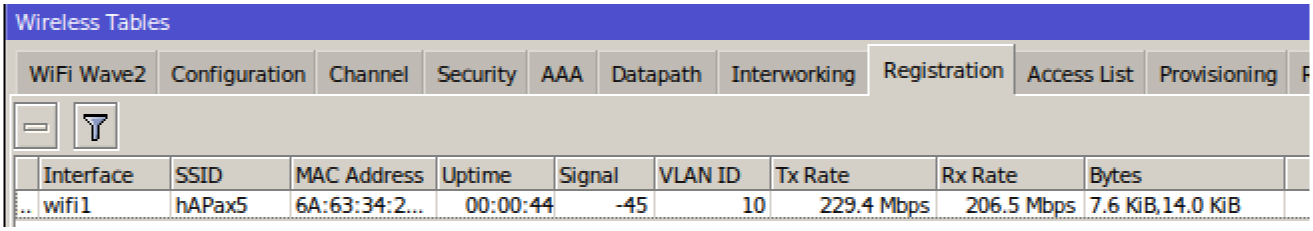
```
interface/wifiwave2/radio/print detail
Flags: L - local
 0 L radio-mac=48:A9:8A:0B:F7:4A phy-id=0 tx-chains=0,1 rx-chains=0,1
     bands=5ghz-a:20mhz,5ghz-n:20mhz,20/40mhz,5ghz-ac:20mhz,20/40mhz,20/40/80mhz,5ghz-ax:20mhz,
      20/40mhz,20/40/80mhz
     ciphers=tkip,ccmp,gcmp,ccmp-256,gcmp-256,cmac,gmac,cmac-256,gmac-256 countries=all
     5g-channels=5180,5200,5220,5240,5260,5280,5300,5320,5500,5520,5540,5560,5580,5600,5620,5640,5660,
            5680,5700,5720,5745,5765,5785,5805,5825
     max-vlans=128 max-interfaces=16 max-station-interfaces=3 max-peers=120 hw-type="QCA6018"
     hw-caps=sniffer interface=wifi1 current-country=Latvia
     current-channels=5180/a,5180/n,5180/n/Ce,5180/ac,5180/ac/Ce,5180/ac/Ceee,5180/ax,5180/ax/Ce,
                 5180/ax/Ceee,5200/a,5200/n,5200/n/eC,5200/ac,5200/ac/eC,5200/ac/eCee,5200/ax...
                 ...5680/n/eC,5680/ac,5680/ac/eC,5680/ax,5680/ax/eC,5700/a,5700/n,5700/ac,5700/ax
     current-gopclasses=115,116,128,117,118,119,120,121,122,123 current-max-reg-power=30
```

While Radio information gives us information about supported channel width, it is also possible to deduce this information from the product page, to do so you need to check the following parameters: **number of chains**, **max data rate**. Once you know these parameters, you need to check the modulation and coding scheme (MCS) table, for example, here: https://mcsindex.com/.

If we take hAP ax^2, as an example, we can see that number of chains is 2, and the max data rate is 1200 - 1201 in the MCS table. In the MCS table we need to find entry for 2 spatial streams - chains, and the respective data rate, which in this case shows us that 80MHz is the maximum supported channel width.

Registration table

'/interface/wifiwave2/registration-table/' displays a list of connected wireless clients and detailed information about them.



	Interface	SSID	MAC Address	Uptime	Signal	VLAN ID	Tx Rate	Rx Rate	Bytes
..	wifi1	hAPax5	6A:63:34:2...	00:00:44	-45	10	229.4 Mbps	206.5 Mbps	7.6 KiB,14.0 KiB

De-authentication

Wireless peers can be manually de-authenticated (forcing re-association) by removing them from the registration table.

```
/interface/wifiwave2/registration-table remove [find where mac-address=02:01:02:03:04:05]
```

WifiWave2 CAPsMAN

WifiWave2 CAPsMAN allows applying wireless settings to multiple MikroTik WifiWave2 AP devices from a central configuration interface.

More specifically, the Controlled Access Point system Manager (CAPsMAN) allows the centralization of wireless network management. When using the CAPsMAN feature, the network will consist of a number of 'Controlled Access Points' (CAP) that provide wireless connectivity and a 'system Manager' (CAPsMAN) that manages the configuration of the APs, it also takes care of client authentication.

WifiWave2 CAPsMAN only passes wireless configuration to the CAP, all forwarding decisions are left to the CAP itself - there is no CAPsMAN forwarding mode.

Requirements:

- Any RouterOS device, that supports the WifiWave2 package, can be a controlled wireless access point (CAP) as long as it has at least a Level 4 RouterOS license.
- WifiWave2 CAPsMAN server can be installed on any RouterOS device that supports the WifiWave2 package, even if the device itself does not have a wireless interface
- Unlimited CAPs (access points) supported by CAPsMAN

> ⓘ WifiWave2 CAPsMAN can only control WifiWave2 interfaces, and WifiWave2 CAPs can join only WifiWave2 CAPsMAN, similarly, regular CAPsMAN only supports non-WifiWave2 caps.

CAPsMAN - CAP simple configuration example:

CAPsMAN in WifiWave2 uses the same menu as a regular WifiWave2 interface, meaning when you pass configuration to CAPs, you have to use the same configuration, security, channel configuration, etc. as you would for regular WifiWave2 interfaces.

> ⓘ You can configure sub configuration menus, directly under "/interface/wifiwave2/configuration" or reference previously created profiles in the main configuration profile

CAPsMAN:

```
#create a security profile
/interface wifiwave2 security
add authentication-types=wpa3-psk name=sec1 passphrase=HaveAg00dDay

#create configuraiton profiles to use for provisioning
/interface wifiwave2 configuration
add country=Latvia name=5ghz security=sec1 ssid=CAPsMAN_5
add name=2ghz security=sec1 ssid=CAPsMAN2
add country=Latvia name=5ghz_v security=sec1 ssid=CAPsMAN5_v

#configure provisioning rules, configure band matching as needed
/interface wifiwave2 provisioning
add action=create-dynamic-enabled master-configuration=5ghz slave-configurations=5ghz_v supported-bands=\
    5ghz-n
add action=create-enabled master-configuration=2ghz supported-bands=2ghz-n

#enable CAPsMAN service
/interface wifiwave2 capsman
set ca-certificate=auto enabled=yes
```

CAP:

```
#enable CAP service, in this case CAPsMAN is on same LAN, but you can also specify "caps-man-addresses=x.x.x.x"
here
/interface/wifiwave2/cap set enabled=yes

#set configuration.manager= on the WifiWave2 interface that should act as CAP
/interface/wifiwave2/set wifi1,wifi2 configuration.manager=capsman-or-local
```

If the CAP is hAP ax^2 or hAP ax^3, it is strongly recommended to enable RSTP in the bridge configuration, on the CAP

configuration.manager should only be set on the CAP device itself, don't pass it to the CAP vai configuration profile that you provision.

The interface that should act as CAP needs additional configuration under "interface/wifiwave2/set wifiX configuration.manager="

CAPsMAN - CAP VLAN configuration example:

In this example, we will assign VLAN20 to our main SSID, and will add VLAN30 for the guest network, ether5 from CAPsMAN is connected to CAP.

CAPsMAN:

```
/interface bridge
add name=br vlan-filtering=yes
/interface vlan
add interface=br name=VLAN20 vlan-id=20
add interface=br name=VLAN30 vlan-id=30
#definfing channel is optional
/interface wifiwave2 channel
add frequency=5180,2412 name=CH
/interface wifiwave2 datapath
add bridge=br name=VLAN20 vlan-id=20
add bridge=br name=VLAN30 vlan-id=30
/interface wifiwave2 security
add authentication-types=wpa2-psk,wpa3-psk name=security
#make sure to change the country to one where you reside in
/interface wifiwave2 configuration
add channel=CH country=Latvia datapath=VLAN20 name=2Ghz_main security=security ssid=2G_MAIN
add channel=CH country=Latvia datapath=VLAN30 name=2Ghz_guest security=security ssid=2G_Guest
add channel=CH country=Latvia datapath=VLAN20 name=5Ghz_main security=security ssid=5G_MAIN
add channel=CH country=Latvia datapath=VLAN30 name=5Ghz_guest security=security ssid=5G_Guest
/ip pool
add name=dhcp_pool0 ranges=192.168.1.2-192.168.1.254
add name=dhcp_pool1 ranges=192.168.20.2-192.168.20.254
add name=dhcp_pool2 ranges=192.168.30.2-192.168.30.254
/ip dhcp-server
add address-pool=dhcp_pool0 interface=br name=dhcp1
add address-pool=dhcp_pool1 interface=VLAN20 name=dhcp2
add address-pool=dhcp_pool2 interface=VLAN30 name=dhcp3
/interface bridge port
add bridge=br interface=ether5
/interface bridge vlan
add bridge=br tagged=ether5,br vlan-ids=20
add bridge=br tagged=ether5,br vlan-ids=30
/interface wifiwave2 capsman
set enabled=yes interfaces=br
/interface wifiwave2 provisioning
add action=create-dynamic-enabled master-configuration=2Ghz_main name-format=2G-%I slave-
configurations=2Ghz_guest supported-bands=2ghz-ax
add action=create-dynamic-enabled master-configuration=5Ghz_main name-format=5G-%I slave-
configurations=5Ghz_guest supported-bands=5ghz-ax
/ip address
add address=192.168.1.1/24 interface=br network=192.168.1.0
add address=192.168.20.1/24 interface=VLAN20 network=192.168.20.0
add address=192.168.30.1/24 interface=VLAN30 network=192.168.30.0
/ip dhcp-client
add interface=ether1 disabled=no
/ip dhcp-server network
add address=192.168.1.0/24 gateway=192.168.1.1
add address=192.168.20.0/24 gateway=192.168.20.1
add address=192.168.30.0/24 gateway=192.168.30.1
```

CAP:

```
/interface bridge
add name=bridgeLocal
/interface wifiwave2 datapath
add bridge=bridgeLocal comment=defconf disabled=no name=capdp
/interface wifiwave2
set [ find default-name=wifi1 ] configuration.manager=capsman datapath=capdp disabled=no
set [ find default-name=wifi2 ] configuration.manager=capsman datapath=capdp disabled=no
/interface bridge port
add bridge=bridgeLocal comment=defconf interface=ether1
add bridge=bridgeLocal comment=defconf interface=ether2
add bridge=bridgeLocal comment=defconf interface=ether3
add bridge=bridgeLocal comment=defconf interface=ether4
add bridge=bridgeLocal comment=defconf interface=ether5
/interface wifiwave2 cap
set discovery-interfaces=bridgeLocal enabled=yes slaves-datapath=capdp
/ip dhcp-client
add interface=bridgeLocal disabled=no
```

Advanced examples

Enterprise wireless security with User Manager v5

Assigning VLAN tags to wireless traffic can be achieved by following the generic VLAN configuration example here.

Replacing stock wireless

The wifiwave2 package can be installed on some products, which ship with the bundled 'wireless' package, replacing it.

Installing the wifiwave2 package disables other means of configuring wireless interfaces. Before installation, make sure to back up any wireless and regular CAPsMAN configuration you may want to retain.

Compatibility

Due to storage, RAM, and architecture requirements, only the following products can replace their bundled wireless software package with wifiwave2:

- hAP ac^3 (non-LTE)
- Audience and Audience LTE6 kit
- RB4011iGS+5HacQ2HnD*

* The 2.4GHz wireless interface on the RB4011iGS+5HacQ2HnD is not compatible with the wifiwave2 package. It will not be usable with the package installed.

It is also possible to install the WifiWave2 package on other devices to use WifiWave2 CAPsMAN: builds for x86, ppc, mmips and tile architectures contain the configuration utilities needed to centrally manage interfaces (as a CAPsMAN controller). Builds for arm and arm64 also contain interface drivers and firmware.

Benefits

- WPA3 authentication and OWE (opportunistic wireless encryption)
- 802.11w standard management frame protection
- 802.11r/k/v
- MU-MIMO and beamforming
- 400Mb/s maximum data rate in the 2.4GHz band for IPQ4019 interfaces
- OFDMA

Lost features

The following notable features of the bundled wireless package do not have equivalents in the wifiwave2 package

- Nstreme and Nv2 wireless protocols

Property Reference

AAA properties

Properties in this category configure an access point's interaction with AAA (RADIUS) servers.

Certain parameters in the table below take *format-string* as their value. In a *format-string*, certain characters are interpreted in the following way:

Character	Interpretation
a	Hexadecimal character making up the MAC address of the client device in lower case
A	Hexadecimal character making up the MAC address of the client device in upper case
i	Hexadecimal character making up the MAC address of the AP's interface in lower case
I (capital 'i')	Hexadecimal character making up the MAC address of the AP's interface in upper case
N	The entire name of the AP's interface (e.g. 'wifi1')
S	The entire SSID

All other characters are used without interpreting them in any way. For examples, see default values.

Property	Description
called-format (*format-string*)	Format for the value of the Called-Station-Id RADIUS attribute, in AP's messages to RADIUS servers. Default: II-II-II-II-II-II:S
calling-format (*format-string*)	Format for the value of the Calling-Station-Id RADIUS attribute, in AP's messages to RADIUS servers. Default: AA-AA-AA-AA-AA-AA
interim-update (*time interval)*	Interval at which to send interim updates about traffic accounting to the RADIUS server. Default: 5m
mac-caching (*time interval* | *'disabled'*)	Length of time to cache RADIUS server replies, when MAC address authentication is enabled. This resolves issues with client device authentication timing out due to (comparatively high latency of RADIUS server replies. Default value: disabled.
name (*string*)	A unique name for the AAA profile. No default value.
nas-identifier (*string*)	Value of the NAS-Identifier attribute, in AP's messages to RADIUS servers. Defaults to the host name of the device (/system/identity).
password-format (*format-string*)	Format for value to use in calculating the value of the User-Password attribute in AP's messages to RADIUS servers when performing MAC address authentication. Default value: "" (an empty string).
username-format (*format-string*)	Format for the value of the User-Name attribute in APs messages to RADIUS servers when performing MAC address authentication. Default value : `AA:AA:AA:AA:AA:AA`

Channel properties

Properties in this category specify the desired radio channel.

Property	Description
band (*2ghz-g* | *2ghz-n* | *2ghz-ax* | *5ghz-a* | *5ghz-ac* | *5ghz-an* | *5ghz-ax*)	Supported frequency band and wireless standard. Defaults to newest supported standard. **Note that band support is limited by radio capabilities.**
frequency (*list of integers or integer ranges*)	For an interface in AP mode, specifies frequencies (in MHz) to consider when picking control channel center frequency. For an interface in station mode, specifies frequencies on which to scan for APs. Leave unset (default) to consider all frequencies supported by the radio and permitted by the applicable regulatory profille. The parameter can contain 1 or more comma-separated values of integers or, optionally, ranges of integers denoted using the syntax RangeBeginning-RangeEnd:RangeStep Examples of valid channel.frequency values: • 2412 • 2412,2432,2472 • 5180-5240:20,5500-5580:20
secondary-frequency (*list of integers* | 'disabled')	Frequency (in MHz) to use for the center of the secondary part of a split 80+80MHz channel. Only official 80MHz channels (5210, 5290, 5530, 5610, 5690, 5775) are supported. Leave unset (default) for automatic selection of secondary channel frequency.
skip-dfs-channels (*10min-cac* | *all* | *disabled*)	Whether to avoid using channels, on which channel availability check (listening for presence of radar signals) is required. • *10min-cac* - interface will avoid using channels, on which 10 minute long CAC is required • *all* - interface will avoid using all channels, on which CAC is required • *disabled* (default) - interface may select any supported channel, regardless of CAC requirements
width (*20mhz* | *20/40mhz* | *20/40mhz-Ce* | *20/40mhz-eC* | *20/40/80mhz* | *20/40/80+80mhz* | *20/40/80/160mhz*)	Width of radio channel. Defaults to widest channel supported by the radio hardware.

Configuration properties

This section includes properties relating to the operation of the interface and the associated radio.

Property	Description
antenna-gain (*integer 0..30*)	Overrides the default antenna gain. The *master* interface of each radio sets the antenna gain for every interface which uses the same radio. This setting cannot override the antenna gain to be lower than the minimum antenna gain of a radio. No default value.
beacon-interval (*time interval 100ms..1s*)	Interval between beacon frames of an AP. Default: 100ms. ⓘ The 802.11 standard defines beacon interval in terms of *time units* (1 TU = 1.024 ms). The actual interval between beacons will be 1 TU for every 1 ms configured. ⚠ Every AP running on the same radio (i.e. a master AP and all its 'virtual'/'slave' APs) must use the same beacon interval.

chains (*list of integer 0..7*)	Radio chains to use for receiving signals. Defaults to all chains available to the corresponding radio hardware.
country (*name of a country*)	Determines, which regulatory domain restrictions are applied to an interface. Defaults to "United States". ⚠ It is important to set this value correctly to comply with local regulations and ensure interoperability with other devices.
dtim-period (*integer 1..255*)	Period at which to transmit multicast traffic, when there are client devices in power save mode connected to the AP. Expressed as a multiple of the beacon interval. Higher values enable client devices to save more energy, but increase network latency. Default: 1
hide-ssid (*no* | *yes*)	- *yes* - AP does not include its SSID in beacon frames, and does not reply to probe requests that have broadcast SSID. - *no* - AP includes its SSID in the beacon frames, and replies to probe requests that have broadcast SSID. Default: no
manager (*capsman* | *capsman-or-local* | *local***)**	capsman - the interface will act as CAP only, this option should **not** be passed via provisioning rules to the CAP capsman-or-local - the interface will get configuration via CAPsMAN or use its own, if /interface/wifiwave2/cap is not enabled. local - interface won't contact CAPsMAN in order to get configuration. Default: local
mode (*ap* | *station*)	Interface operation mode - *ap* (default) - interface operates as an access point - *station* - interface acts as a client device, scanning for access points advertising the configured SSID - station-bridge - interface acts as a client device and enables support for a 4-address frame format, so that the interface can be used as a bridge port ⓘ The station-bridge mode, as implemented in the wifiwave2 package, is incompatible with APs running the bundled 'wireless' package and vice versa.
multicast-enhance (*enabled* | *disabled*)	With the multicast-enhance feature enabled, an AP will convert every multicast-addressed IP or IPv6 packet into multiple unicast-addressed frames for each connected station. This may improve link throughput and reliability since, unlike multicast frames, unicasts are acknowledged by stations and transmitted using a higher data rate. Default: disabled
qos-classifier (*dscp-high-3-bits* | *priority*)	- dscp-high-3-bits - interface will transmit data packets using a WMM priority equal to the value of the 3 most significant bits of the IP DSCP field - priority - interface will transmit data packets using a WMM priority equal to that set by IP firewall or bridge filter Default: priority ⓘ 802.11ac wireless chipsets do not support the dscp-high-3-bits classifier mode.
ssid (*string*)	The name of the wireless network, aka the (E)SSID. No default value.
tx-chains (*list of integer 0..7*)	Radio chains to use for transmitting signals. Defaults to all chains available to the corresponding radio hardware.

tx-power (*integer 0..40*)	A limit on the transmit power (in dBm) of the interface. Can not be used to set power above limits imposed by the regulatory profile. Unset by default.

Datapath properties

Parameters relating to forwarding packets to and from wireless client devices.

Property	Description
bridge (*bridge interface*)	Bridge interface to add interface to, as a bridge port. No default value.
bridge-cost (*integer*)	Bridge port cost to use when adding as bridge port. Default: 10
bridge-horizon (*none* | *integer)*	Bridge horizon to use when adding as bridge port Default: none.
client-isolation (*no* | *yes*)	Determines whether client devices connecting to this interface are (by default) isolated from others or not. This policy can be overridden on a per-client basis using access list rules, so a an AP can have a mixture of isolated and non-isolated clients. Traffic from an isolated client will not be forwarded to other clients and unicast traffic from a non-isolated client will not be forwarded to an isolated one. Default: no
interface-list (*interface list*)	List to which add the interface as a member. No default value.
vlan-id (*none* | *integer* 1..4095)	Default VLAN ID to assign to client devices connecting to this interface (only relevant to interfaces in AP mode). When a client is assigned a VLAN ID, traffic coming from the client is automatically tagged with the ID and only packets tagged with with this ID are forwarded to the client. Default: none ⚠ 802.11n/ac interfaces do not support this type of VLAN tagging under the wifiwave2 package, but they can be configured as VLAN access ports in bridge settings.

Security Properties

Parameters relating to authentication.

Property	Description
authentication-types (*list of wpa-psk, wpa2-psk, wpa-eap, wpa2-eap, wpa3-psk, owe, wpa3-eap, wpa3-eap-192*)	Authentication types to enable on the interface. The default value is an empty list (no authenticaion, an open network). Configuring a passphrase, adds to the default list the *wpa2-psk* authentication method (if the interface is an AP) or both *wpa-psk* and *wpa2-psk* (if the interface is a station). Configuring an *eap-username* and an *eap-password* adds to the default list *wpa-eap and wpa2-eap* authentication methods.
connect-group (*string*)	APs within the same connect group do not allow more than 1 client device with the same MAC address. This is to prevent malicious authorized users from intercepting traffic intended to other users ('MacStealer' attack) or performing a denial of service attack by spoofing the MAC address of a victim. Handling of new connections with duplicate MAC addresses depends on the connect-priority of AP interfaces involved. By default, all APs are assigned the same connect-group.

connect-priority (accept-priority/hold-priority (*integers*))	Theese parameters determine, how a connection is handled if the MAC address of the client device is the same as that of another active connection to another AP. If (accept-priority of AP2) < (hold-priority of AP1), a connection to AP2 wil cause the client to be dropped from AP1. If (accept-priority of AP2) = (hold-priority of AP1), a connection to AP2 will be allowed only if the MAC address can no longer be reached via AP1. If (accept-priority of AP2) > (hold-priority of AP1), a connection to AP2 will not be accepted. If omitted, hold-priority is the same as accept-priority. By default, APs, which perform user authentication, have higher priority (lower integer value), than open APs.
dh-groups (*list of 19, 20, 21*)	Identifiers of elliptic curve cryptography groups to use in SAE (WPA3) authentication.
disable-pmkid (*no* | *yes*)	For interfaces in AP mode, disables inclusion of a PMKID in EAPOL frames. Disabling PMKID can cause compatibility issues with client devices which make use of it. • *yes* - Do not include PMKID in EAPOL frames. • *no* (default) - include PMKID in EAPOL frames.
eap-accounting (*no* | *yes*)	Send accounting information to RADIUS server for EAP-authenticated peers. Default: no.

Properties related to EAP, are only relevant to interfaces in station mode. APs delegate (passthrough) EAP authentication to the RADIUS server.

eap-anonymous-identity (*string*)	Optional anonymous identity for EAP outer authentication. No default value.
eap-certificate-mode (*dont-verify-certificate* | *no-certificates* | *verify-certificate* | *verify-certificate-with-crl*)	Policy for handling the TLS certificate of the RADIUS server. • verify-certificate - require server to have a valid certificate. Check that it is signed by a trusted certificate authority. • dont-verify-certificate (default) - Do not perform any checks on the certificate. • no-certificates - Attempt to establish the TLS tunnel by performing anonymous Diffie-Hellman key exchange. To be used if the RADIUS server has no certificate at all. • verify-certificate-with-crl - Same as *verify-certificate,* but also checks if the certificate is valid by checking the Certificate Revocation List.
eap-methods (*list of peap, tls, ttls*)	EAP methods to consider for authentication. Defaults to all supported methods.
eap-password (*string*)	Password to use, when the chosen EAP method requires one. No default value.
eap-tls-certificate (*certificate*)	Name or id of a certificate in the device's certificate store to use, when the chosen EAP authentication method requires one. No default value.
eap-username (*string*)	Username to use when the chosen EAP method requires one. No default value.

Take care when configuring encryption ciphers.

All client devices MUST support the group encryption cipher used by the AP to connect, and some client devices (notably, Intel® 8260) will also fail to connect if the list of unicast ciphers includes any they don't support.

encryption (*list of ccmp, ccmp-256, gcmp, gcmp-256, tkip*)	A list of ciphers to support for encrypting unicast traffic. Defaults to *ccmp*.

Properties related to 802.11r fast BSS transition only apply to interfaces in AP mode. Wifiwave2 interfaces in station mode do not support 802.11r.

For a client device to successfully roam between 2 APs, the APs need to be managed by the same instance of RouterOS. For information on how to centrally manage multiple APs, see CAPsMAN

ft (*no | yes*)	Whether to enable 802.11r fast BSS transitions (roaming). Default: no.
ft-mobility-domain (*integer 0..65535*)	The fast BSS transition mobility domain ID. Default: 44484 (0xADC4).
ft-nas-identifier (string of *2..96 hex characters*)	Fast BSS transition PMK-R0 key holder identifier. Default: MAC address of the interface.
ft-over-ds (*no* | *yes*)	Whether to enable fast BSS transitions over DS (distributed system). Default: no.
ft-preserve-vlanid (*no* | *yes*)	- no - when a client connects to this AP via 802.11r fast BSS transition, it is assigned a VLAN ID according to the access and/or interface settings - yes (default) - when a client connects to this AP via 802.11r fast BSS transition, it retains the VLAN ID, which it was assigned during initial authentication The default behavior is essential when relying on a RADIUS server to assign VLAN IDs to users, since a RADIUS server is only used for initial authentication.
ft-r0-key-lifetime (*time interval 1s..6w3d12h15m*)	Lifetime of the fast BSS transition PMK-R0 encryption key. Default: 600000s (~7 days)
ft-reassociation-deadline (*time interval 0..70s*)	Fast BSS transition reassociation deadline. Default: 20s.
group-encryption (*ccmp* | *ccmp-256* | *gcmp* | *gcmp-256* | *tkip*)	Cipher to use for encrypting multicast traffic. Defaults to *ccmp*.
group-key-update (*time interval*)	Interval at which the group temporal key (key for encrypting broadcast traffic) is renewed. Defaults to 24 hours.
management-encryption (*cmac* | *cmac-256* | *gmac* | *gmac-256*)	Cipher to use for encrypting protected management frames. Defaults to *cmac*.
management-protection (*allowed* | *disabled* | *required*)	Whether to use 802.11w management frame protection. **Incompatible with management frame protection in standard wireless package**. Default value depends on value of selected authentication type. WPA2 allows use of management protection, WPA3 requires it.
owe-transition-interface (*interface*)	Name or internal id of an interface whose MAC address and SSID to advertise as the matching AP when running in OWE transition mode. Required for setting up open APs that offer OWE, but also work with older devices that don't support the standard. See configuration example below.
passphrase (*string of up to 63 characters*)	Passphrase to use for PSK authentication types. Defaults to an empty string - "". WPA-PSK and WPA2-PSK authentication requires a minimum of 8 chars, while WPA3-PSK does not have minimum passphrase length.
sae-anti-clogging-threshold (*'disabled'* | *integer*)	Due to SAE (WPA3) associations being CPU resource intensive, overwhelming an AP with bogus authentication requests makes for a feasible denial-of-service attack. This parameter provides a way to mitigate such attacks by specifying a threshold of in-progress SAE authentications, at which the AP will start requesting that client devices include a cookie bound to their MAC address in their authentication requests. It will then only process authentication requests which contain valid cookies. Default: 5.
sae-max-failure-rate (*'disabled'* | *integer*)	Rate of failed SAE (WPA3) associations per minute, at which the AP will stop processing new association requests. Default: 40.
sae-pwe (*both* | *hash-to-element* | *hunting-and-pecking*)	Methods to support for deriving SAE password element. Default: both.

wps (*disabled* | *push-button*)	- *push-button* (default) - AP will accept WPS authentication for 2 minutes after 'wps-push-button' command is called. Physical WPS button functionality not yet implemented. - *disabled* - AP will not accept WPS authentication

Steering properties

Properties in this category govern mechanisms for advertising potential roaming candidates to client devices.

Property	Description
neighbor-group (*string*)	When sending neighbor reports and BSS transition management requests, an AP will list all other APs within its neighbor group as potential roaming candidates. By default, a dynamic neighbor group is created for each set of APs with the same SSID and authentication settings. APs operating in the 5GHz band are indicated to be preferable to ones operating in the 2.4GHz band.
rrm (*no* | *yes*)	Enables sending of 802.11k neighbor reports. Default: yes
wnm (*no* | *yes*)	Enables sending of solicited 802.11v BSS transition management requests. Default: yes

Miscellaneous properties

Property	Description
arp (*disabled* | *enabled* | *local-proxy-arp* | *proxy-arp* | *reply-only)*	Address Resolution Protocol mode: - *disabled* - the interface will not use ARP - *enabled* - the interface will use ARP (default) - *local-proxy-arp* - the router performs proxy ARP on the interface and sends replies to the same interface - *proxy-arp* - the router performs proxy ARP on the interface and sends replies to other interfaces - *reply-only* - the interface will only reply to requests originated from matching IP address/MAC address combinations which are entered as static entries in the ARP table. No dynamic entries will be automatically stored in the ARP table. Therefore for communications to be successful, a valid static entry must already exist.
arp-timeout (*time interval* | *'auto'*)	Determines how long a dynamically added ARP table entry is considered valid since the last packet was received from the respective IP address. Value *auto* equals to the value of *arp-timeout* in */ip settings*, which defaults to 30s.
disable-running-check *(* *no* | *yes*)	- *yes* - interface's *running* property will be true whenever the interface is not disabled - *no* (default) - interface's *running* property will only be true when it has established a link to another device
disabled (*no* | *yes*) (X)	Hardware interfaces are disabled by default. Virtual interfaces are not.
mac-address (*MAC*)	MAC address (BSSID) to use for an interface. Hardware interfaces default to the MAC address of the associated radio interface. Default MAC addresses for virtual interfaces are generated by 1. Taking the MAC address of the associated master interface 2. Setting the second-least-significant bit of the first octet to 1, resulting in a locally administered MAC address 3. If needed, incrementing the last octet of the address to ensure it doesn't overlap with the address of another interface on the device
mtu (*integer [32..2290]*; Default: **1500**)	Layer3 Maximum transmission unit.
l2mtu (*integer [32..2290]*; Default: **2290**)	Layer2 Maximum transmission unit.

master-interface (*interface*)	Multiple interface configurations can be run simultaneously on every wireless radio. Only one of them determines the radio's state (whether it is enabled, what frequency it's using, etc). This 'master' interface, is *bound* to a radio with the corresponding *radio-mac.* To create additional ('virtual') interface configurations on a radio, they need to be *bound* to the corresponding master interface. No default value.
name (*string*)	A name for the interface. Defaults to *wifiN*, where *N* is the lowest integer that has not yet been used for naming an interface.

Read-only properties

Property	Description
bound (*boolean*) (B)	True for *master* interfaces that are currently available for WiFi manager. True for a virtual interface (configurations linked to a master interface) when both the interface itself and its master interface are not disabled and the *master* interface has a bound flag.
default-name (*string*)	The default name for an interface.
inactive (*boolean*) (I)	False for interfaces in AP mode when they've selected a channel for operation (i.e. configuration has been successfully applied). False for interfaces in station mode when they've connected to an AP (i.e. configuration has been successfully applied, an with AP with matching settings has been found). True otherwise.
master (*boolean*) (M)	True for physical interfaces on router itself or detected CAP if running as CAPsMAN. False for virtual interfaces.
radio-mac (*MAC*)	The MAC address of the associated radio.
running (*boolean*) (R)	True, when an interface has established a link to another device. If *disable-running-check* is set to 'yes', true whenever the interface is not disabled.

Access List

Filtering parameters	
Parameter	**Description**
interface (*interface* | *interface-list* | *'any'*)	Match if connection takes place on the specified interface or interface belonging to specified list. Default: any.
mac-address (*MAC address*)	Match if the client device has the specified MAC address. No default value.
mac-address-mask (*MAC address*)	Modifies the **mac-address** parameter to match if it is equal to the result of performing bit-wise AND operation on the client MAC address and the given address mask. Default: FF:FF:FF:FF:FF:FF (i.e. client's MAC address must match value of **mac-address** exactly)
signal-range (*min..max*)	Match if the strength of received signal from the client device is within the given range. Default: '-120..120'
ssid-regexp (*regex*)	Match if the given regular expression matches the SSID.
time (*start-end,days*)	Match during the specified time of day and (optionally) days of week. Default: 0s-1d

Action parameters	
Parameter	**Description**
allow-signal-out-of-range (*time period* | 'always')	The length of time which a connected peer's signal strength is allowed to be outside the range required by the **signal-range** parameter, before it is disconnected. If the value is set to 'always', peer signal strength is only checked during association. Default: 0s.
action (*accept* | *reject* | *query-radius*)	Whether to authorize a connection • *accept* - connection is allowed • *reject* - connection is not allowed • *query-radius* - connection is allowed if MAC address authentication of the client's MAC address succeeds Default: *accept*
client-isolation (*no* | *yes*)	Whether to isolate the client from others connected to the same AP. No default value.
passphrase (*string*)	Override the default passphrase with given value. No default value.
radius-accounting (*no* | *yes*)	Override the default RADIUS accounting policy with given value. No default value.
vlan-id (*none* | *integer 1..4095*)	Assign the given VLAN ID to matched clients. No default value.

Frequency scan

Information about RF conditions on available channels can be obtained by running the frequency-scan command.

Command parameters	
Parameter	**Description**
duration (*time interval)*	Length of time to perform the scan for before exiting. Useful for non-interactive use. Not set by default.
freeze-frame-interval (*time interval)*	Time interval at which to update command output. Default: 1s.
frequency (*list of frequencies /ranges)*	Frequencies to perform the scan on. See channel.frequency parameter syntax above for more detail. Defaults to all supported frequencies.
numbers (*string)*	Either the name or internal id of the interface to perform the scan with. Required. Not set by default.
rounds (*integer)*	Number of times to go through list of scannable frequencies before exiting. Useful for non-interactive use. Not set by default.
save-file (string)	Name of file to save output to. Not set by default.

Output parameters	
Parameter	**Description**
channel (*integer)*	Frequency (in MHz) of the channel scanned.
networks (*integer)*	Number of access points detected on the channel.
load (integer*)*	Percentage of time the channel was busy during the scan.
nf (integer)	Noise floor (in dBm) of the channel.
max-signal (*integer*)	Maximum signal strength (in dBm) of APs detected in the channel.
min-signal (*integer*)	Minimum signal strength (in dBm) of APs detected in the channel.

primary (*boolean*) (P)	Channel is in use as the primary (control) channel by an AP.
secondary (boolean) (S)	Channel is in use as a secondary (extension) channel by an AP.

Scan command

The '/interface wifiwave2 scan' command will scan for access points and print out information about any APs it detects.

The scan command takes all the same parameters as the frequency-scan command.

Output parameters	
Parameter	**Description**
active (*boolean*) (A)	Signifies that beacons from the AP have been received in the last 30 seconds.
address (*MAC*)	The MAC address (BSSID) of the AP.
channel (*string*)	The control channel frequency used by the AP, its supported wireless standards and control/extension channel layout.
security (*string*)	Authentication methods supported by the AP.
signal (*integer*)	Signal strength of the AP's beacons (in dBm).
ssid (*string*)	The extended service set identifier of the AP.
sta-count (*integer*)	The number of client devices associated with the AP. Only available if the AP includes this information in its beacons.

Sniffer

Command parameters	
Parameter	**Description**
duration (*time interval*)	Automatically interrupt the sniffer after the specified time has passed. No default value.
filter (*string*)	A string that specifies a filter to apply to captured frames. Only frames matched by the filter expression will be displayed, saved or streamed. This works similarly to filter strings in libpcap, for example. The filter can match • Address fields (addr1, addr2, addr3) • Wireless frame type and subtype, including shortcuts such as 'beacon' (type == 0 && subtype == 8) • Flags (to-ds, from-ds, retry, power, protected) A string can include the following operators: • == (exact match) • != (does not equal) • && (logical AND) • || (logical OR) • () (for grouping filter expressions)
number (*interface*)	Interface to use for sniffing.
pcap-file (*string)*	Save captured frames to a file with the given name. No default value (captured frames are not saved to a file by default).
pcap-size-limit (*integer*)	File size limit (in bytes) when storing captured frames locally. When this limit has been reached, no new frames are added to the capture file. No default value.
stream-address (IP address)	Stream captured packets via the TZSP protocol to the given address. No default value (captured packets are not streamed anywhere by default).

stream-rate (*integer*)	Limit on the rate (in packets per second) at which captured frames are streamed via TZSP.

WPS

interface/wifiwave2/wps-client wifi

Command parameters	
Parameter	**Description**
duration (*time interval*)	Length of time after which the command will time out if no AP is found. Unlimited by default.
interval (*time interval*)	Time interval at which to update command output. Default: 1s.
mac-address (*MAC*)	Only attempt connecting to AP with the specified MAC (BSSID). Not set by default.
numbers (*string*)	Name or internal id of the interface with which to attempt connection. Not set by default.
ssid (*string*)	Only attempt to connect to APs with the specified SSID. Not set by default.

Radios

Information about the capabilities of each radio can be gained by running the `/interface/wifiwave2/radio print detail` command.

Property	Description
2g-channels (*list of integers*)	Frequencies supported in the 2.4GHz band.
5g-channels (*list of integers*)	Frequencies supported in the 5GHz band.
bands (*list of strings*)	Supported frequency bands, wireless standards and channel widths.
ciphers (*list of strings*)	Supported encryption ciphers.
countries (*list of strings*)	Regulatory domains supported by the interface.
min-antenna-gain (*integer*)	Minimum antenna gain permitted for the interface.
phy-id (*string*)	A unique identifier.
radio-mac (*MAC*)	MAC address of the radio interface. Can be used to match radios to interface configurations.
rx-chains (*list of integers*)	IDs for radio chains available for receiving radio signals.
tx-chains (*list of integers*)	IDs for radio chains available for transmitting radio signals.

Registration table

The registration table contains read-only information about associated wireless devices.

Parameter	Description
authorized (*boolean*) (A)	True when the peer has successfully authenticated.
bytes (*list of integers*)	Number of bytes in packets transmitted to a peer and received from it.
interface (*string*)	Name of the interface, which was used to associate with the peer.
mac-address (*MAC*)	The MAC address of the peer.
packets (*list of integers*)	Number of packets transmitted to a peer and received from it.
rx-rate *(string)*	Bitrate of received transmissions from peer.
signal (*integer)*	Strength of signal received from the peer (in dBm).

tx-rate (*string)*	Bitrate used for transmitting to the peer.
uptime (*time interval)*	Time since association.

CAPsMAN Global Configuration

Menu: /interface/wifiwave2/capsman

Property	Description
ca-certificate *(auto* | *certificate name)*	Device CA certificate, CAPsMAN server requires a certificate, certificate on CAP is optional.
certificate (*auto | certificate name | none*; Default: **none**)	Device certificate
enabled *(no* | *yes*)	Disable or enable CAPsMAN functionality
package-path (*string |*; Default:)	Folder location for the RouterOS packages. For example, use "/upgrade" to specify the upgrade folder from the files section. If an empty string is set, CAPsMAN can use built-in RouterOS packages, note that in this case only CAPs with the same architecture as CAPsMAN will be upgraded.
require-peer-certificate (*yes | no*; Default: **no**)	Require all connecting CAPs to have a valid certificate
upgrade-policy (*none | require-same-version | suggest-same-upgrade*; Default: **none**)	Upgrade policy options • none - do not perform upgrade • require-same-version - CAPsMAN suggest to upgrade the CAP RouterOS version and, if it fails it will not provision the CAP. (Manual provision is still possible) • suggest-same-version - CAPsMAN suggests to upgrade the CAP RouterOS version and if it fails it will still be provisioned
interfaces *(all | interface name | none; Default:* ***all****)*	Interfaces on which CAPsMAN will listen for CAP connections

CAPsMAN Provisioning

Provisioning rules for matching radios are configured in **/interface/wifiwave2/provisioning/** menu:

Property	Description
action (*create-disabled | create-enabled | create-dynamic-enabled | none*; Default: **none**)	Action to take if rule matches are specified by the following settings: • **create-disabled** - create disabled static interfaces for radio. I.e., the interfaces will be bound to the radio, but the radio will not be operational until the interface is manually enabled; • **create-enabled** - create enabled static interfaces. I.e., the interfaces will be bound to the radio and the radio will be operational; • **create-dynamic-enabled** - create enabled dynamic interfaces. I.e., the interfaces will be bound to the radio, and the radio will be operational; • **none** - do nothing, leaves radio in the non-provisioned state;
comment (*string*; Default:)	Short description of the Provisioning rule
common-name-regexp (*string*; Default:)	Regular expression to match radios by common name. Each CAP's common name identifier can be found under "/interface/wifiwave2/radio" as value "REMOTE-CAP-NAME"
supported-bands (*2ghz-ax | 2ghz-g | 2ghz-n | 5ghz-a | 5ghz-ac | 5ghz-ax | 5ghz-n*; Default:)	Match radios by supported wireless modes.
identity-regexp (*string*; Default:)	Regular expression to match radios by router identity

address-ranges (*IpAddressRange[, IpAddressRanges] max 100x*; Default: **""**)	Match CAPs with IPs within configured address range. Will only work for CAPs that joined CAPsMAN using IP, not MAC address.
master-configuration (*string*; Default:)	If **action** specifies to create interfaces, then a new master interface with its configuration set to this configuration profile will be created
name-format (*string*)	Base string to use when constructing names of provisioned interfaces. Each new interface will be created by taking the base string and appending a number to the end of it. If included in the string, character sequence **%I** will be replaced by the system identity of the cAP. **%C** will be replaced with the cAP's TLS certificate's Common Name. Default: "cap-wifi"
radio-mac (*MAC address*)	MAC address of radio to be matched. No default value.
slave-configurations (*string*; Default:)	If **action** specifies to create interfaces, then a new slave interface for each configuration profile in this list is created.
disabled (*yes | no*; Default: **no**)	Specifies if the provision rule is disabled.

CAP configuration

Menu: /interface/wifiwave2/cap

Property	Description
caps-man-addresses *(list of IP addresses; Default:* ***empty****)*	List of Manager IP addresses that CAP will attempt to contact during discovery
caps-man-names ()	An ordered list of CAPs Manager names that the CAP will connect to, if empty - CAP does not check Manager name
discovery-interfaces (*list of interfaces*;)	List of interfaces over which CAP should attempt to discover Manager
lock-to-caps-man (*no | yes;* Default: **no**)	Sets, if CAP should lock to the first CAPsMAN it connects to
slaves-static ()	Creates Static Virtual Interfaces, allows the possibility to assign IP configuration to those interfaces. MAC address is used to remember each static-interface when applying the configuration from the CAPsMAN.
caps-man-certificate-common-names ()	List of Manager certificate CommonNames that CAP will connect to, if empty - CAP does not check Manager certificate CommonName
certificate ()	Certificate to use for authenticating
enabled (*yes | no*; Default: **no**)	Disable or enable the CAP feature
slaves-datapath ()	