OpenSync™

# OpenSync 3.4 Northbound API

Date: November 25, 2022

Document ID: EUB-020-033-101

# Table of Contents

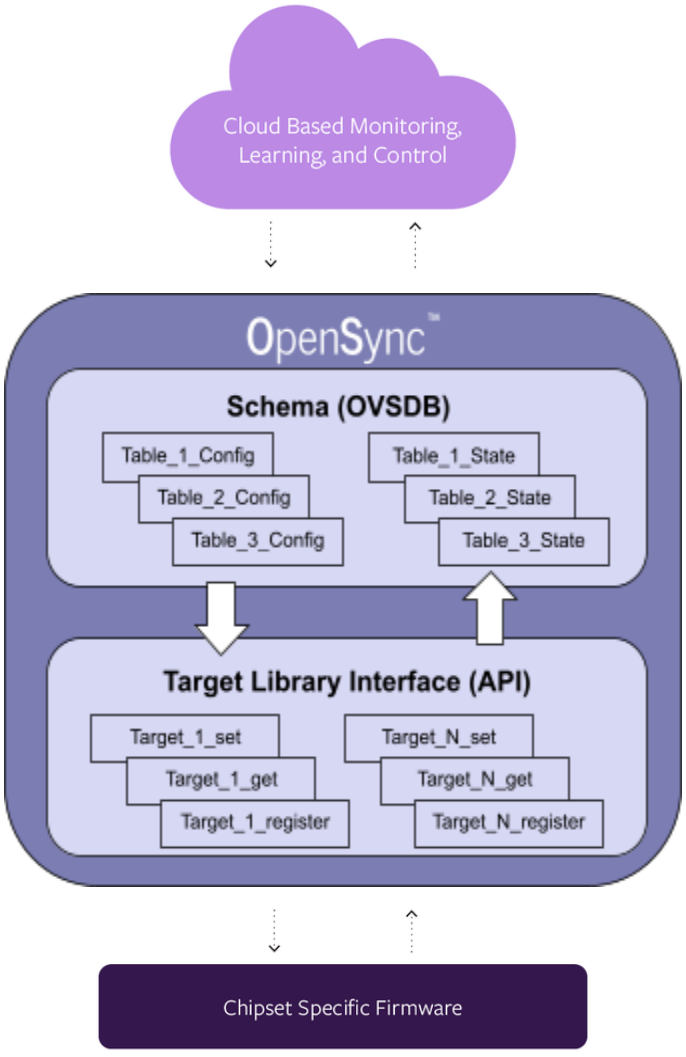
# References

*[1]* *http://docs.openvswitch.org/en/latest/ref/ovsdb.7/*

*[2]* *EUB-020-013-001_OpenSync_Overview*

# Introduction

*OpenSync*™ is a cloud-agnostic open-source software for delivery, curation, and management of home and office wireless services. *OpenSync* follows the SDN concept where the OVSDB serves as interprocess communication between services in the controller and *OpenSync* managers on the device, as illustrated in the image below.




Tables in the OVSDB schema are roughly organized into two groups - “config” and “state” tables. Usually, controller writes instructions and configuration into the config tables and the device must report its status in the state tables.

# Alarms

The *Alarms* table is used for reporting custom-defined alarms to the Cloud.

| Name | Type | Description |
|---|---|---|
| code | string | Alarm code |
| timestamp | integer | Timestamp of the alarm |
| source | string | Source of the alarm |
| add_info | string | Additional information |

# AW_Bluetooth_Config

*AW_Bluetooth_Config* is a configuration table where we can set payload in the BLE broadcast frame. CM2 uses this table to broadcast the node serial number and connection status. The frames are received by the mobile app, enabling users to claim the nodes, locate and rename the claimed nodes, and diagnose networking issues of nodes which are unable to connect to the cloud.

| Name | Type | Description |
|---|---|---|
| mode | enum (*on*, *off*) | Enable/disable sending of BLE broadcast frames |
| interval_millis | integer | Interval time in msec, how often BLE broadcast is sent<br>0 - default value (250 or 500 ms, depends on platform) |
| connectable | boolean | Enables bidirectional communication for local configuration over BLE. |
| txpower | integer | Tx power<br>0 - default value (0x005C) |
| command | enum (*on_boarding*, *diagnostic*, *locate*) | Up until *OpenSync 2.0* only *on_boarding* is used |
| payload | string | 6 bytes of broadcast frame payload in format “01:23:45:67:89:ab”. Only the first byte is used as Status byte, and only when command is set to *on_boarding*. Status bits have the following meaning: |

| | | |
|---|---|---|
| | | • 0x01 (Bit 0): Ethernet physical link (1: UP, 0: Down)<br>• 0x02 (Bit 1): Wifi physical link (1: Associated, 0: Disconnected)<br>• 0x04 (Bit 2): Backhaul over Ethernet (1: Yes, 0: No)<br>• 0x08 (Bit 3): Backhaul over Wifi (1: Yes, 0: No)<br>• 0x10 (Bit 4): Connected to Router (1: Yes, 0: No)<br>• 0x20 (Bit 5): Connected to Internet (1: Yes, 0: No)<br>• 0x40 (Bit 6): Connected to Cloud (1: Yes, 0: No) |

**Example (a node is connected via Ethernet to router, but router has no uplink connection):**

```
# ovsh s AW_Bluetooth_Config
-----------------------------------
_uuid           | 0c63~06b2         |
_version        | d2a9~62c7         |
command         | on_boarding       |
connectable     | false             |
interval_millis | 0                 |
mode            | on                |
payload         | 15:00:00:00:00:00 |
txpower         | 0                 |
-----------------------------------
```

# AWLAN_Node

*AWLAN_Node* is a configurational and informational table that stores all *device_entity* information. The table also stores information about the Cloud connection specifics. Once the *redirector_address* is specified, the Cloud configures connection information through *manager_address* (See Connection Manager - CM [2] for details), and specifies the MQTT configuration (See Queue Manager - QM [2] for details).

| Name | Type | Description |
|---|---|---|
| boot_time | integer | Number of seconds since the time the node booted up. To calculate the live uptime for a node, the Cloud subtracts current_time() - AWLAN_Node.boot_time. |

| | | |
|---|---|---|
| id | string | Device identification number - **serial_number** used by the Cloud to differentiate device management (could also be MAC address) |
| device_mode | enum (*cloud*, *monitor*, *battery*, *custom*) | Different devices operate in different modes. This requires special handling in the Cloud:<br>● **cloud** - full control including reading statistics, configuring channels, and managing backhaul POD topology changes - Adaptive Wifi<br>● **monitor** - (or read-only mode) is a mode where only certain statistics are collected and/or channels are managed - depends on customer requirements<br>● **battery** - devices running on battery need to save power, therefore, the cloud turns off most of the functionality<br>● **custom** - custom mode, individual features and responsibilities are governed via profile in the Cloud |
| factory_reset | boolean | Indicates the status of the device after factory reset has been issued |
| firmware_url | string | The descriptor in URL format indicating a new SW release download location<br>*(Only applicable when Upgrade Manager is used)* |
| firmware_pass | string | Password for firmware-encrypted images |
| firmware_version | string | *OpenSync* firmware package version |
| platform_version | string | Joined platform firmware version containing *OpenSync* |
| serial_number | string | Device serial number as seen on the BAR code |
| model | string | Device-friendly name used in the network |
| revision | string | Device HW revision |
| sku_number | string | Device SKU number/revision |
| version_matrix | key/value map | Detailed firmware version description including<br>● *OpenSync* version<br>● Date of generation<br>● Firmware package version<br>● Build number<br>● Source code repository version details<br>● other details |
| upgrade_dl_timer | integer | The "upgrade_" fields specify the node upgrading behavior. |

| | | |
|---|---|---|
| | | **Note:** The settings are only applicable when the Upgrade Manager is used.<br><br>Time in seconds that is given to the device to complete downloading of the image. Actual success of the operation is indicated in upgrade_status. |
| upgrade_status | integer | Status code that indicates status of upgrade process. Can have negative values indicating various types of errors, or positive values which indicate successful upgrade steps. |
| upgrade_timer | integer | Delay in seconds after the upgrade process is started from the time the field is set. |
| redirector_addr | string | The cloud redirector address used to differentiate between different deployments (production or development). The syntax is : <protocol>:<cloud>:<port> |
| manager_addr | string | Manager Address field is populated by the redirector and tells to which Controller Address should the OVSD connection be established. The syntax is: <protocol>:<controller_hostname\|ip>:<port> |
| mqtt_headers | key/value map | Contains location and node ID. |
| mqtt_settings | key/value map | MQTT server settings. QM uses the mqtt_settings field to get all the required parameters for establishing an MQTT connection.<br><br>Parameters:<br>● broker: hostname<br>● port: port number<br>● compress: none, zlib<br>● topics: topic name<br>● QoS: 0, 1, 2 |
| mqtt_topics | key/value map | A key/value map that defines which MQTT topics are used for certain features.<br><br>Example: aggregatedStats (wifi statistics) MQTT topic |
| led_config | key/value map | Specifies LED behavior<br>*(Only applicable when the LED Manager is used)* |
| min_backoff | integer | CM reconnect backoff timer configuration |
| max_backoff | integer | |
| vendor_name | string | Device vendor name. Up to 64 characters can be used. |

| | | |
|---|---|---|
| vendor_part_number | string | Device model as defined by the vendor. Up to 64 characters can be used. |
| vendor_manufacturer | string | Name of the device manufacturer. Up to 64 characters can be used. |
| vendor_factory | string | Factory at which the device has been produced. Up to 64 characters can be used. |
| vendor_mfg_date | string | Manufacturing date. Up to 16 characters can be used. |
| redirector_addr | string | Redirector URL. The ovsdb-server supports IPv4 and IPv6 connections. |

**Example:**

```
# ovsh s AWLAN_Node
-------------------------------------------------------------------------------------------------------
_uuid                | d120~9bc2                                                                      |
_version             | 6ae6~76d7                                                                      |
boot_time            | 1641893054                                                                     |
device_mode          | ["set",[]]                                                                     |
factory_reset        | ["set",[]]                                                                     |
firmware_pass        |                                                                                |
firmware_url         |                                                                                |
firmware_version     | 4.4.0-63-abcd123-dev                                                           |
id                   | E123456789                                                                     |
led_config           | ["map",[["mode","off"]]]                                                       |
manager_addr         | ssl:opensync.us-west-2.aws.opensync.io:443                                     |
max_backoff          | 60                                                                             |
min_backoff          | 30                                                                             |
model                | PP123Z                                                                         |
mqtt_headers         | ["map",[["locationId","61dd495d782ac93123456"],["nodeId","E123456789"]]]       |
mqtt_settings        | ["map",[["broker","emqtt-opensync.us-west-2.aws.opensync.io"],                 |
                     : ["compress","zlib"],["port","443"],["qos",""],["topics",                      :
                     : "s1/opensync/61dd495d782ac93123456/01"]]]                                     :
mqtt_topics          | ["map",[["Crash.Reports",                                                      |
                     : "Crash/Reports/opensync/E123456789/61dd495d782ac93123456"]]]                  |
platform_version     | 1                                                                              |
redirector_addr      | ssl:wildfire-dualstack.plume.tech:443                                          |
revision             | EVT.0                                                                          |
serial_number        | E123456789                                                                     |
sku_number           | 912345678                                                                      |
upgrade_dl_timer     | 0                                                                              |
upgrade_status       | 0                                                                              |
upgrade_timer        | 0                                                                              |
vendor_factory       | MyFactory                                                                      |
vendor_manufacturer  | MyManufactor                                                                   |
vendor_mfg_date      | 2020/16                                                                        |
vendor_name          | MyCompany                                                                      |
vendor_part_number   | PP123Z                                                                         |
Version_matrix       | ["map",[["OPENSYNC","3.2.0.0"],["core","3.2.0.0/=5725/abcd123"],["device",     :
                     : "0.0/=7810/g430b123"],["platform/bcm","0.0/=892/gfd38123"],                   :
                     : ["vendor/opensync","4.3.0/=3721/gd1ae123"]]]                                  :
-------------------------------------------------------------------------------------------------------
```

# Open_vSwitch

When *Open_vSwitch* is used, this table contains the reference to the networking elements. Details are available in EUB-020-013-001_OpenSync_Overview, section Network Manager - NM [2].

| Name | Type | Description |
|---|---|---|
| bridges | uuid | Reference to the table: Bridge |
| manager_options | uuid | Reference to the table: Manager |
| ssl | uuid | Reference to the table: SSL |
| other_config | string | Additional configuration |
| external_ids | string | *Not used by CM2* |
| next_cfg | integer | *Not used by CM2* |
| cur_cfg | integer | Version of the current configuration |
| statistics | string | *Not used by CM2* |
| ovs_version | string | Version of OVS |
| db_version | string | Version of database |
| dpdk_initialized | boolean | If True, the DPDK EAL initialization has succeeded. |
| dpdk_version | string | Library version linked to ovs-vswitchd |
| system_type | string | *Not used by CM2* |
| system_version | string | Identifies the platform to the controller |
| datapath_types | string | *Not used by CM2* |
| iface_types | string | Supported type of interfaces |

**Example:**

```
# ovsh s Open_vSwitch
------------------------------------------------------------------
_uuid            | f5eb~58cc                                                                                 |
_version         | 84c9~97d5                                                                                 |
bridges          | [ca0e~2356,ca66~5670]                                                                     |
cur_cfg          | 4                                                                                         |
datapath_types   | ["set",["netdev","system"]]                                                               |
db_version       | ["set",[]]                                                                                |
dpdk_initialized | false                                                                                     |
dpdk_version     | ["set",[]]                                                                                |
external_ids     | ["map",[]]                                                                                |
iface_types      | ["set",["geneve","gre","internal","lisp","patch","stt","system","tap","vxlan"]]           |
```

```
manager_options  | 569e~bbda                                                                |
next_cfg         | 4                                                                        |
other_config     | ["map",[["stats-update-interval","3600000"]]]                            |
ovs_version      | ["set",[]]                                                               |
ssl              | fc24~5e95                                                                |
statistics       | ["map",[]]                                                               |
system_type      | ["set",[]]                                                               |
system_version   | ["set",[]]                                                               |
-------------------------------------------------------------------
```

# SSL

SSL table contains SSL-related information, including the path to the currently used certificates.

| Name | Type | Description |
|---|---|---|
| private_key | string | Path to private key |
| certificate | string | Path to certificate |
| ca_cert | string | Path to CA certificate |
| bootstrap_ca_cert | boolean | Enable/disable bootstrap certificate |
| external_ids | key/value map | *Not used by CM2* |

**Example:**

```
# ovsh s SSL
-------------------------------------------------------------
_uuid             | 8e9a~8be8                                |
_version          | c61b~d046                                |
bootstrap_ca_cert | false                                    |
ca_cert           | /opt/opensync/certs/ca.pem               |
certificate       | /opt/opensync/certs/certs/client.pem     |
external_ids      | ["map",[]]                               |
private_key       | /opt/opensync/certs/certs/client_dec.key |
-------------------------------------------------------------
```

# Manager

The table contains information required to establish and keep the Cloud connection. The Manager table status field is a useful source when debugging the Cloud connection.

| Name | Type | Description |
|---|---|---|
| target | string | Direct SSL address to the Cloud |

| | | |
|---|---|---|
| max_backoff | integer | Reconnect backoff timer configuration<br>*Not used by CM2* |
| inactivity_probe | integer | Inactive timer, default value: 30000 ms |
| connection_mode | enum<br>(*in-band*, *out-of-band*) | *Not used by CM2* |
| other_config | string | *Not used by CM2* |
| external_ids | string | *Not used by CM2* |
| is_connected | boolean | Connection state |
| status | string | Connection status |

**Example:**

```
# ovsh s Manager
------------------------------------------------------------------------------
_uuid            | 0d1e~0c85                                                  |
_version         | ba1a~efba                                                  |
connection_mode  | ["set",[]]                                                 |
external_ids     | ["map",[]]                                                 |
inactivity_probe | 30000                                                      |
is_connected     | true                                                       |
max_backoff      | ["set",[]]                                                 |
other_config     | ["map",[]]                                                 |
status           | ["map",[["sec_since_connect","16001"],["state","ACTIVE"]]] |
target           | ssl:10.1.1.1:443                                           |
------------------------------------------------------------------------------
```

# Connection_Manager_Uplink

The *Connection_Manager_Uplink* table provides info and status for the uplink connection.

| Name | Type | Description |
|---|---|---|
| bridge | string | Manages WAN bridges for the main active link. |
| has_L2 | boolean | Physical link state |
| has_L3 | boolean | Uplink availability. “True” if the interface obtained an IP address. |
| if_name | string | Interface name |
| if_type | enum<br>(*bridge*, *eth*, *vif*, *gre*, *gre6*, *pppoe*, *softwds*, *tap*, *vlan, lte, unmanaged*) | Interface type |
| is_used | boolean | Link is used as a main link. |

| | | |
|---|---|---|
| loop | boolean | Used for ethernet link, notify potential loop if link will be used in LAN bridge |
| ntp_state | boolean | System clock valid (NTP success) |
| priority | integer | Link priority, link with the highest value should be used as a main link. |
| unreachable_cloud_counter | integer | Counter that increments each time the Cloud connection is lost. Resets to zero when connection is re-established. |
| unreachable_internet_counter | integer | Counter that increments each time the internet connection is lost. Resets to zero when connection is re-established. |
| unreachable_link_counter | integer | stats |
| unreachable_router_counter | integer | stats |
| eth_client | boolean | Notifies when an ethernet client is detected on the interface. |
| ipv4 | enum | Information about link state.<br>Available options:<br>● "ready": IP address assigned<br>● "active": link validated for use<br>● "inactive": not ready<br>● "blocked": link blocked due to poor quality<br>● "unblocking": moving from blocked to active state |
| ipv6 | enum | Information about link state.<br>Available options:<br>● "ready": IP address assigned<br>● "active": link validated for use<br>● "inactive": not ready<br>● "blocked": link blocked due to poor quality<br>● "unblocking": moving from blocked to active state |
| unblock_ts | string | Timestamp of the moment when the link is unblocked. |

**Example:**

```
# ovsh s Connection_Manager_Uplink
-------------------------------------------------------
_uuid                         | 4910~4011  | e22c~1143  |
_version                      | d522~0e4c  | a338~2451  |
bridge                        | ["set",[]] | br-home    |
eth_client                    | ["set",[]] | ["set",[]] |
has_L2                        | true       | true       |
has_L3                        | false      | true       |
if_name                       | eth0       | eth1       |
if_type                       | eth        | eth        |
ipv4                          | ["set",[]] | ready      |
ipv6                          | ["set",[]] | ready      |
is_used                       | ["set",[]] | true       |
loop                          | false      | ["set",[]] |
ntp_state                     | ["set",[]] | true       |
priority                      | 11         | 11         |
unblock_ts                    | ["set",[]] | ["set",[]] |
unreachable_cloud_counter     | ["set",[]] | 0          |
unreachable_internet_counter  | ["set",[]] | 0          |
unreachable_link_counter      | ["set",[]] | 0          |
unreachable_router_counter    | ["set",[]] | 0          |
-------------------------------------------------------
```

# Wifi_Radio_Config

*Wifi_Radio_Config* table stores the system-wide wireless radio (physical) configuration.

| Name | Type | Description |
|---|---|---|
| if_name | string | Interface name |
| freq_band | enum (*2.4G*, *5G*, *5GL*, *5GU, 6G*) | Frequency band of the specified interface. If two separate 5 GHz radios are available on the device, these radios should be specified as “lower”: **5GL** or “upper”: **5GU**. |
| enabled | boolean | The desired interface state. “true” if the interface should be configured. |
| dfs_demo | boolean | Controls the demo mode for DFS. |
| hw_type | string | The descriptor of the hardware type for a specific wifi radio. For example this is useful e.g. for differentiating the configurations for different hardware chips. |
| hw_config | string | Configuration type, complementing the “hw_type” parameter |
| country | string | Two-letter country code descriptor for setting the wireless regulatory domain. Examples: “AT”, “AU”, “BE”, etc. |

| | | |
|---|---|---|
| channel | integer | Channel number of the specified radio. The available channels list depends on the frequency band, regulatory domain, and channel width. |
| channel_sync | integer | *Deprecated* |
| channel_mode | enum (*auto*, *manual*, *cloud,* *acs*) | *Not used.* |
| hw_mode | enum (*11a*, *11b*, *11g*, *11n*, *11ab*, *11ac,* *11ax*) | Specifies the IEEE 802.11 standard used for the specified radio. |
| ht_mode | enum (*HT20*, *HT2040*, *HT40*, *HT40+*, *HT40-*, *HT80*, *HT160*, *HT80+80*) | Stands for “high throughput mode”; specifies the bandwidth of the selected channel, in MHz:<br>● **HT20**: single 20 MHz channel<br>● **HT2040**: *deprecated*<br>● **HT40:** dual 20 MHz channels, automatic selection of upper or lower secondary channel<br>● **HT40-:** dual 20 MHz channels, upper channel is primary or control<br>● **HT40+:** dual 20 MHz channels, lower channel is primary or control<br>● **HT80:** dual 40 MHz channels<br>● **HT160:** dual 80 MHz channels<br>● **HT80+80:** *deprecated* |
| thermal_shutdown | integer | Temperature in degrees (℃) at which the unit shuts down. |
| thermal_downgrade_temp | integer | Temperature at which the tx_chainmask is reduced, effectively using less antennas. |
| thermal_upgrade_temp | integer | Temperature at which the tx_chainmask is increased, effectively using more antennas. |
| thermal_integration | integer | Time required for the device to upgrade or downgrade chainmask after exceeding the temperature threshold |
| temperature_control | integer | *deprecated* |
| vif_configs | set if uuids | Array of VIF uuids linked to entries (columns) in the *Wifi_VIF_Config* table |
| tx_power | integer | Transmission power of the specified radio (in dBm) |
| bcn_int | integer | Beacon interval in TU (time units), which depend on the system, e.g., TU = 1.024 ms |

| | | |
|---|---|---|
| tx_chainmask | integer | Transmission chainmask: bitmask that specifies which antennas are used for transmission. Each WiFi radio can be connected to many antennas (2, 3, 4), so disabling one or more of the antennas affects the temperature of the WiFi chip. Value is a decimal value of bitmask which controls the antennas used. |
| zero_wait_dfs (available in *OpenSync 2.0* and later) | enum | Adds zero-wait DFS options.<br>● "disable": default value. Executes the channel switch, switches channel and runs CAC.<br>● "enable": If the cloud enables this and chooses a DFS channel in Wifi_Radio_Config::channel the device runs background CAC. In case of success, the device switches to a new DFS channel.<br>● "precac": extends the “enable” mode. Runs background CAC on all DFS channels. |

**Example:**

```
 # ovsh s Wifi_Radio_Config
-------------------------------------------------------------------------------------------------
_uuid                  | 5074~e243             | c9aa~f7f0             | 38fc~4d0b             |
_version               | f0ee~c713             | a33a~b519             | 0a39~809d             |
bcn_int                | ["set",[]]            | ["set",[]]            | ["set",[]]            |
channel                | 44                    | 6                     | 108                   |
channel_mode           | cloud                 | cloud                 | cloud                 |
channel_sync           | 0                     | 0                     | 0                     |
country                | ["set",[]]            | ["set",[]]            | ["set",[]]            |
dfs_demo               | false                 | ["set",[]]            | false                 |
enabled                | true                  | true                  | true                  |
fallback_parents       | ["map",[]]            | ["map",[]]            | ["map",[]]            |
freq_band              | 5GL                   | 2.4G                  | 5GU                   |
ht_mode                | HT160                 | HT20                  | HT80                  |
hw_config              | ["map",[]]            | ["map",[]]            | ["map",[]]            |
hw_mode                | 11ax                  | 11ax                  | 11ax                  |
hw_type                | bcm43684              | bcm6755               | bcm6755               |
if_name                | wl0                   | wl1                   | wl2                   |
temperature_control    | ["map",[]]            | ["map",[]]            | ["map",[]]            |
thermal_downgrade_temp | ["set",[]]            | ["set",[]]            | ["set",[]]            |
thermal_integration    | ["set",[]]            | ["set",[]]            | ["set",[]]            |
thermal_shutdown       | ["set",[]]            | ["set",[]]            | ["set",[]]            |
thermal_tx_chainmask   | ["set",[]]            | ["set",[]]            | ["set",[]]            |
thermal_upgrade_temp   | ["set",[]]            | ["set",[]]            | ["set",[]]            |
tx_chainmask           | 15                    | 3                     | 3                     |
tx_power               | ["set",[]]            | ["set",[]]            | ["set",[]]            |
vif_configs            | [a064~1f13,c5f1~9011, | [1800~ea0d,71fd~6156, | [5d9f~bdfa,900b~9770] |
                       : fcbe~2c12]            : 7c46~1289]            :                       :
zero_wait_dfs          | disable               | ["set",[]]            | disable               |
-------------------------------------------------------------------------------------------------
```

# Wifi_Radio_State

*Wifi_Radio_State* table represents the actual state of the device and is therefore read by the Cloud. The table is pushed by WM2 and reflects the current system-wide wireless radio (physical) configuration.

Entries marked in gray differ from the *Wifi_Radio_Config* table, while the other rows are equal.

| Name | Type | Description |
|---|---|---|
| if_name | string | Interface name |
| radio_config | uuid | The uuid linked to the entity (column) in the *Wifi_Radio_Config* table |
| freq_band | Enum (*2.4G*, *5G*, *5GL*, *5GU, 6G*) | The frequency band of the specified interface. If two separate 5GHz radios are available on the device, they should be specified as “lower”: **5GL** or “upper”: **5GU**. |
| enabled | boolean | The desired interface state. “true” if the interface should be in the UP state; otherwise the interface should be in the DOWN state (the equivalent of ifconfig up/down) |
| dfs_demo | boolean | Controls the demo mode for DFS |
| hw_type | string | Hardware type for a specific wifi radio. For example, use this value for differentiating the configurations between different hardware chips. |
| hw_params | string | Field for hardware-specific configuration |
| radar | string | Field for DFS-specific information |
| hw_config | string | Configuration type, complementing the “hw_type” parameter |
| country | string | Two-letter country code descriptor for setting wireless regulatory domain. Examples: “AT”, “AU”, “BE”, etc. |
| channel | integer | Channel number of the specified radio. The available channels list depends on the frequency band, regulatory domain, and channel width. |
| channel_sync | integer | *Deprecated* |
| channel_mode | enum (*auto*, | *Not used* |

| | | |
|---|---|---|
| | *manual*, *cloud*) | |
| mac | string | MAC address of the specified radio |
| hw_mode | enum (*11a*, *11b*, *11g*, *11n*, *11ab*, *11ac, 11ax*) | Specifies the IEEE 802.11 standard used for the specified radio |
| ht_mode | enum (*HT20*, *HT2040*, *HT40*, *HT40+*, *HT40-*, *HT80*, *HT160*, *HT80+80*) | Stands for “high throughput mode”; specifies the bandwidth of the selected channel, in MHz:<br>● **HT20**: single 20 MHz channel<br>● **HT2040**: *deprecated*<br>● **HT40:** dual 20 MHz channels, automatic selection of upper or lower secondary channel<br>● **HT40-:** dual 20 MHz channels, upper channel is primary or control<br>● **HT40+:** dual 20 MHz channels, lower channel is primary or control<br>● **HT80:** dual 40 MHz channels<br>● **HT160:** dual 80 MHz channels<br>● **HT80+80:** *deprecated* |
| thermal_shutdown | integer | Temperature in degrees (℃), at which the unit shuts down |
| thermal_downgrade_temp | integer | Temperature at which the tx_chainmask is reduced |
| thermal_upgrade_temp | integer | Temperature at which the tx_chainmask is increased |
| thermal_integration | integer | Time required for the device to upgrade or downgrade the chainmask after exceeding the temperature threshold |
| thermal_downgraded | boolean | Information if device chainmask is currently downgraded due to the exceeded thermal threshold |
| temperature_control | integer | *Deprecated* |
| vif_states | set of uuids | Array of VIF uuids linked to the entries (columns) in the Wifi_VIF_State table |
| tx_power | integer | Transmission power of the specified radio (in dBm). |
| bcn_int | integer | Beacon interval in TU (time units), which depend on the system, e.g. TU = 1.024 ms |
| tx_chainmask | integer | Transmission chainmask: bitmask to specify which radio chains are used for transmission |
| thermal_tx_chainmask | integer | Value used by thermal manager |

| | | |
|---|---|---|
| allowed_channels | integer list | The list of allowed channels for the specified interface. The list depends on several other parameters, like frequency and regulatory domain. |
| channels | map | The map is keyed with channel numbers to json state descriptions. The json is expected to be a dictionary and should include “state” keys with one of the 4 values:<br>● allowed: non-dfs channel<br>● nop_finished: dfs channel, requires cac before using<br>● cac_completed: dfs channel, cac completed, usable<br>● nop_started: dfs channel, radar was detected and it must not be used |
| zero_wait_dfs | enum | Adds zero-wait DFS options.<br>● "disable": default value. Executes the channel switch, switches channel and runs CAC.<br>● "enable": If the cloud enables this and chooses a DFS channel in Wifi_Radio_Config::channel the device runs background CAC. In case of success, the device switches to a new DFS channel.<br>● "precac": extends the “enable” mode. Runs background CAC on all DFS channels. |

# Wifi_Route_Config

This table contains static routes for IPTV traffic that are pushed from Cloud to the nodes.

| Name | Type | Description |
|---|---|---|
| dest_addr | string | IPv4 destination address. |
| dest_mask | string | IPv4 destination mask. |
| gateway | string | IPv4 address of the next hop router. |
| metric | integer | 32-bit integer value to help the router choose the best route among the multiple feasible routes. Lower metric routes are preferred. Optional field. |
| if_name | string | The name of the output interface device for the route. The name should be specified unless the decision on which interface to select is delegated to the kernel routing logic (based on the gateway address |

| | | |
|---|---|---|
| | | specification). The if_name field value is required if the gateway address is not provided. |

# Wifi_Route_State

This table reflects the current state of the routing tables in the system.

| Name | Type | Description |
|---|---|---|
| dest_addr | string | IPv4 destination address. |
| dest_mask | string | IPv4 destination mask. |
| gateway | string | IPv4 address of the next hop router. |
| if_name | string | The name of the output interface device for the route. |
| gateway_hwaddr | string | MAC address of the gateway. |

# Wifi_Speedtest_Config

An update to this table configures and initiates a speedtest, either using Ookla, iPerf or SamKnows.

| Name | Type | Description |
|---|---|---|
| test_type | enum (OOKLA, IPERF3_S, IPERF3_C, SAMKNOWS) | Type of speedtest to be run. |
| traffic_cap | real | If the current amount of traffic exceeds this value, speedtest does not start. |
| delay | integer | Delay the test start in seconds (currently not in use). |
| testid | integer | Arbitrary test ID for your reference. Copied to Wifi_Speedtest_Status. |
| select_server_id | integer | Ookla: Initiate Ookla speedtest with a specific server ID. |
| preferred_list | integer list | Ookla: a list of Ookla server IDs that will be preferred for the speedtest. |
| st_bw | integer | iPerf: target bandwidth (0 for unlimited) for UDP tests. [bits/s] |
| st_dir | enum (DL, UL, | iPerf: Specify direction (download, upload, both) |

| | | |
|---|---|---|
| | DL_UL) | |
| st_len | integer | iPerf: Time in seconds to transmit - test length (one way) |
| st_parallel | integer | iPerf: number of parallel client streams to run |
| st_pkt_len | integer | iPerf: MTU - 40 bytes |
| st_port | integer | iPerf: server port to listen on/connect to |
| st_server | string | iPerf: server host or IP to connect to or bind to |
| st_udp | boolean | iPerf: use UDP rather than TCP |
| st_endpoint_url | string | Ookla endpoint URL |

# Wifi_Speedtest_Status

This table serves to report speedtest results.

| Name | Type | Description |
|---|---|---|
| UL | real | Download speed (MB/s) |
| DL | real | Upload speed (MB/s) |
| server_name | string | Test server FQDN or test server ID (Ookla). |
| server_IP | string | Test server IP. |
| ISP | string | ISP name. |
| RTT | real | Round trip time. |
| jitter | real | Ookla: measured jitter (in ms) |
| duration | real | Test duration in seconds |
| timestamp | integer | Unix timestamp |
| status | integer | Test status. 0 if successful, negative values signal various errors. |
| DL_bytes | integer | The amount of downloaded bytes. |
| UL_bytes | integer | The amount of uploaded bytes. |
| DL_duration | real | Duration of download part of speedtest (in s) |
| UL_duration | real | Duration of upload part of speedtest (in s) |
| test_type | enum (OOKLA, IPERF_S, IPERF_C, | The type of speed test that was run. |

| | | |
|---|---|---|
| | SAMKNOWS.<br>) | |
| testid | integer | The test ID used when running this speedtest. |
| pref_selected | boolean | Ookla: If true, the preferred server was used for speedtest |
| hranked_offere<br>d | boolean | Ookla: If true, the preferred server was chosen even if a non-preferred server was ranked higher due to better latency. |
| DL_pkt_loss | real | iPerf: packet loss [percent] - download direction |
| UL_pkt_loss | real | iPerf: packet loss (percentage) - upload direction |
| DL_jitter | real | iPerf: measured jitter (in ms) - download direction |
| UL_jitter | real | iPerf: measured jitter (in ms) - upload direction |
| host_remote | string | iPerf: remote host IP |

# Wifi_VIF_Config

The *Wifi_VIF_Config* table is used for the system-wide wireless (logical) interface - AP and STA configuration.

| Name | Type | Description |
|---|---|---|
| if_name | string | Interface name |
| enabled | boolean | Defines if the specified interface is enabled or not |
| mode | enum<br>(*ap*,<br>*monitor*,<br>*sta*) | Interface mode:<br>• **ap:** access point<br>• **ap_vlan:** interface used by WDS with multi-AP functionality<br>• **monitor:** monitor (promiscuous) mode - *not used*<br>• **sta:** station (client) |
| parent | string | Parent BSSID to which the STA needs to be connected (*Only for extenders*). If empty, any BSSID is allowed to match. |
| vif_radio_idx | integer | Consecutive radio index per VIF type, such as:<br>• 1=bhaul-ap,<br>• 2=home-ap, etc.<br><br>**Note:** Radio index values are model-specific. For example, bhaul-ap can be 0 on platforms where the first (primary) VIF of a radio is acting as the main BSS. |
| vif_dbg_lvl | integer | Introduces Wi-Fi debug config switch. |

| | | |
|---|---|---|
| wds | boolean | *Deprecated* |
| ssid | string | Interface SSID |
| ssid_broadcast | enum (*enabled*, *disabled*, *disabled_null*) | Determines if the interface SSID is broadcast or not. |
| security | key/value map | *Deprecated. Should remain unset.* |
| credential_configs | uuid | Wifi_Credential_Config. This is for extender only. This column is respected only if “ssid” is empty. |
| bridge | string | Specifies the bridge interface on which the authentication listens for EAPOL. |
| mac_list | string | ACL MAC list for layer 2 packet filtering |
| mac_list_type | enum (*whitelist*, *blacklist*, *none*) | How to filter the MAC addresses:<br>● **whitelist:** only forward traffic from addresses in *mac_list*, block all other traffic<br>● **blacklist:** only block traffic from addresses in *mac_list*, forward all other traffic<br>● **none:** no filtering |
| vlan_id | integer | Identification number for VLAN-tagged traffic (min 1, max 4096). |
| min_hw_mode | enum (*11b*, *11g*, *11a*, *11n*, *11ac*) | Minimum required IEEE 802.11 standard for the specified interface |
| uapsd_enable | boolean | Enables uAPSD power saving capability for the specified interface |
| group_rekey | integer | Time in seconds between the GTK rekeying. Valid for AP mode only. Empty or -1 means default, 0 means disabled, >0 is interval in seconds. |
| ap_bridge | boolean | AP isolation (intra-bss communication) |
| ft_psk | integer | “Fast transition” (802.11R) pre-shared key |
| ft_mobility_domain | integer | “Fast transition” (802.11R) mobility domain |
| btm | integer (0, 1) | Enable (1) or disable (0) WNM BSS Transition support. |
| rrm | integer (0, 1) | Enable (1) or disable (0) RRM support. Current minimum requirement is to enable RRM Neighbor Report support. |

| | | |
|---|---|---|
| dynamic_beacon | boolean | Only applicable if *ssid_broadcast* is disabled. When enabled, the AP stops beaconing until a non-ap STA attempts to associate. |
| mcast2ucast | boolean | Enables multicast-to-unicast packet conversion:<br>● 0: disabled<br>● 1: enabled |
| multi_ap | string | Device type as defined by the Multi AP specification:<br>● Backhaul STA<br>● Backhaul BSS<br>● Fronthaul BSS<br>● Fronthaul Backhaul BSS (available with *OpenSync* 2.4 and later) |
| wpa | boolean | Enables support for WPA on AP:<br>● “true”: Enable WPA<br>● “false”: AP uses OPEN mode (unencrypted mode; the AP does not request client password) |
| wpa_key_mgmt | enum list | Configures the WiFi security mode:<br>● "dpp"<br>● "wpa-psk"<br>● "wpa2-psk"<br>● "wpa2-eap"<br>● "sae"<br>● "ft-wpa2-psk"<br>● "ft-sae" |
| wpa_psks | map | List of passwords used by WPA1/2 and SAE. |
| wpa_oftags | map | WPA1/2 password’s oftags |
| radius_srv_addr | string | Remote RADIUS server address (IP or domain name) |
| radius_srv_port | integer | Remote RADIUS server port number |
| radius_srv_secret | string | Remote RADIUS server secret |
| default_oftag | string | An oftag used for OPEN, WPA-EAP or as a fallback oftag. |
| dpp_connector | string | Signed public part of the client dpp_netaccesskey, with extra info, such as role, expiry time, etc. |
| dpp_csign_hex | string | Public part of the Configurator end-user client key. |
| dpp_netaccesskey_hex | string | Exposes the DPP Configuration Object’s WPA/SAE PSK when acting as Enrollee. |
| dpp_cc | boolean | A marker that can be added by the AP<br>to its Beacon and Probe Resp frames. The marker is then used by chirping enrollees to<br>build their scan lists on which they will chirp |

| | | |
|---|---|---|
| | | on. |
| min_rssi | integer | RSSI threshold to prevent low RSSI clients from associating<br>to specified VAPs. |
| max_sta | integer | Limits the number of allowed public clients to be associated with each VAP. |
| passpoint_hessid | string | Configures a Homogenous Extended Service Set Identifier (HESSID) for a Hotspot 2.0 network. |
| airtime_precedence | string | Sets percentage for the WiFi air time per each VAP on the same radio. Available values:<br>● "low"<br>● "medium"<br>● "high" |
| wps | boolean | Enables support for WPS on AP (i.e., broadcast support for WPS in beacons):<br>● “true”: Enable WPS<br>● “false”: Disable WPS |
| wps_pbc | boolean | Initiate WPS single session:<br>● “true”: Start WPS session<br>● “false”: Cancel ongoing WPS session |
| wps_pbc_key_id | string | The value must match one of the password key IDs stored in the security field of the AP's configuration. |

**Example:**

```
# ovsh s Wifi_VIF_Config -w if_name==wl0.1
-------------------------------------------------------------------------------------------
_uuid                | a064~1f13                                                          |
_version             | 3571~1dc9                                                          |
airtime_precedence   | ["set",[]]                                                         |
ap_bridge            | ["set",[]]                                                         |
bridge               | ["set",[]]                                                         |
btm                  | ["set",[]]                                                         |
credential_configs   | ["set",[]]                                                         |
default_oftag        | ["set",[]]                                                         |
dpp_cc               | ["set",[]]                                                         |
dpp_connector        | ["set",[]]                                                         |
dpp_csign_hex        | ["set",[]]                                                         |
dpp_netaccesskey_hex | ["set",[]]                                                         |
dynamic_beacon       | false                                                              |
enabled              | true                                                               |
ft_mobility_domain   | ["set",[]]                                                         |
ft_psk               | ["set",[]]                                                         |
group_rekey          | ["set",[]]                                                         |
if_name              | wl0.1                                                              |
mac_list             | ["set",["e4:26:86:12:34:56","e4:26:86:12:34:58"]]                  |
mac_list_type        | whitelist                                                          |
max_sta              | ["set",[]]                                                         |
mcast2ucast          | ["set",[]]                                                         |
min_hw_mode          | ["set",[]]                                                         |
min_rssi             | ["set",[]]                                                         |
mode                 | ap                                                                 |
multi_ap             | none                                                               |
parent               | ["set",[]]                                                         |
passpoint_hessid     | ["set",[]]                                                         |
radius_srv_addr      | ["set",[]]                                                         |
radius_srv_port      | ["set",[]]                                                         |
radius_srv_secret    | ["set",[]]                                                         |
rrm                  | ["set",[]]                                                         |
security             | ["map",[]]                                                         |
ssid                 | os.backhaul                                                        |
ssid_broadcast       | disabled                                                           |
uapsd_enable         | ["set",[]]                                                         |
vif_dbg_lvl          | ["set",[]]                                                         |
vif_radio_idx        | 1                                                                  |
vlan_id              | ["set",[]]                                                         |
wds                  | ["set",[]]                                                         |
wpa                  | true                                                               |
wpa_key_mgmt         | wpa2-psk                                                           |
wpa_oftags           | ["map",[["key--1","bhaul--1"]]]                                    |
wpa_psks             | ["map",[["key--1",                                                 |
                     : "33070BD49FE4970FC28F6BC2ABCE079684BA2729168AC272FD20886C8882EC4"]]] :
wps                  | ["set",[]]                                                         |
wps_pbc              | ["set",[]]                                                         |
wps_pbc_key_id       |
-------------------------------------------------------------------------------------------
```

# Wifi_VIF_State

*Wifi_VIF_State* table represents the actual state on the device, and is therefore read by the Cloud. The table is pushed by WM2 and reflects the current system-wide wireless (logical) interface configuration.

Entries marked in gray are different to the *Wifi_VIF_Config* table, while the other rows are equal.

| Name | Type | Description |
|---|---|---|
| vif_config | uuid | The associated Wifi_VIF_Config entry |
| if_name | string | Interface name |
| enabled | boolean | Shows if the specified interface is enabled or not. |
| mode | enum (*ap*, ap_vlan *monitor*, *sta*) | Interface mode:<br>• **ap:** access point<br>• **ap_vlan:** interface used by WDS with multi-AP functionality<br>• **monitor:** monitor (promiscuous) mode - *not used*<br>• **sta:** station (client) |
| state | string | Client state |
| channel | integer | Channel of the radio interface is attached (some platforms support multiple channel configurations). |
| mac | string | MAC address of the interface |
| vif_radio_idx | integer | VIF index of the radio interface |
| wds | boolean | Defines if the specified interface support supports WDS |
| parent | string | Parent BSSID to which the STA needs to be connected (*Only for extenders*) |
| ssid | string | Interface SSID |
| ssid_broadcast | enum (*enabled*, *disabled*, *disabled_null*) | Determines if the interface SSID is broadcasted or not. |
| security | key/value map | *Deprecated. Should remain unset.* |
| bridge | string | Specifies the bridge interface to which the VIF is added |

| | | |
|---|---|---|
| mac_list | string | ACL MAC list for layer 2 packet filtering |
| mac_list_type | enum (*whitelist*, *blacklist*, *none*) | How to filter MAC addresses:<br>● **whitelist:** only forward traffic from addresses in *mac_list*, block all other traffic<br>● **blacklist:** only block traffic from addresses in *mac_list*, forward all other traffic<br>● **none:** no filtering |
| associated_clients | uuid | MAC list of all associated clients |
| vlan_id | integer | Identification number for VLAN-tagged traffic |
| min_hw_mode | enum (*11a*, *11b*, *11g*, *11n*, *11ab*, *11ac*) | Minimal required IEEE 802.11 standard for the specified interface |
| uapsd_enable | boolean | Indicates uAPSD power saving capability for the specified interface |
| group_rekey | integer | Time in seconds between the GTK rekeying. Valid for AP mode only. Empty or -1 means default, 0 means disabled, >0 is interval in seconds. |
| ap_bridge | boolean | AP isolation (infra-bss communication) |
| ap_vlan_sta_addr (available in *OpenSync* 2.0 and later) | string | MAC address of connected WDS station (Backhaul STA) |
| ft_psk | integer | *Deprecated. “Fast transition” (802.11R) pre-shared key.* |
| ft_mobility_domain | integer | “Fast transition” (802.11R) mobility domain |
| rrm | integer (0, 1) | Indicates if the RRM support is enabled. If RRM Neighbor Report isn’t available, 0 must be reported. |
| btm | integer (0, 1) | ● Indicates whether WNM BSS Transition support is enabled or not. |
| dynamic_beacon | boolean | Indicates if the dynamic_beacon configuration is enabled. It does not reflect if the feature is actively quiescing beacons at the moment or not. |
| mcast2ucast | boolean | Indicates multicast-to-unicast packet conversion:<br>● 0: disabled<br>● 1: enabled |
| multi_ap | string | Device type as defined by the Multi AP specification:<br>● Backhaul STA<br>● Backhaul BSS<br>● Fronthaul BSS |

| | | |
|---|---|---|
| | | ● Fronthaul Backhaul BSS (available with *OpenSync* 2.4 and later) |
| wps | boolean | Indicates support for WPS on AP (i.e., broadcast support for WPS in beacons):<br>● “true”: Enable WPS<br>● “false”: Disable WPS |
| wps_pbc | boolean | Indicates WPS single session:<br>● “true”: Start WPS session<br>● “false”: Cancel ongoing WPS session |
| wps_pbc_key_id | string | The value must match one of the password key IDs stored in the security field of the AP's configuration. |
| wpa | boolean | Enables support for WPA on AP:<br>● “true”: Enable WPA<br>● “false”: AP uses OPEN mode (unencrypted mode; the AP does not request client password) |
| wpa_key_mgmt | enum | WiFi security modes in use:<br>● "dpp"<br>● "wpa-psk"<br>● "wpa2-psk"<br>● "wpa2-eap"<br>● "sae"<br>● "ft-wpa2-psk"<br>● "ft-sae" |
| wpa_psks | map | List of passwords used by WPA1/2 and SAE. |
| radius_srv_port | integer | Remote RADIUS server port number |
| radius_srv_secret | string | Remote RADIUS server secret |
| dpp_connector | string | Signed public part of the client dpp_netaccesskey with extra info, such as role, expiry time, etc. |
| dpp_csign_hex | string | Public part of the Configurator end-user client key. |
| dpp_netaccesskey_hex | string | Exposes the DPP Configuration Object’s WPA/SAE PSK when acting as Enrollee. |
| dpp_cc | boolean | A marker that can be added by the AP to its Beacon and Probe Resp frames. The marker is then used by chirping enrollees to build their scan lists on which they will chirp<br>on. |
| min_rssi | integer | RSSI threshold to prevent low RSSI clients from associating<br>to specified VAPs. |
| max_sta | integer | Limits the number of stations associated with each |

| | | |
|---|---|---|
| | | VAP. |
| airtime_precedence | string | Displays percentage for the WiFi air time per each VAP on the same radio. Available values:<br>• "low"<br>• "medium"<br>• "high" |

# DPP_Config

The DPP_Config table stores the parameters that are necessary for enrolling devices without a user interface in a secure WiFi network.

| Name | Type | Description |
|---|---|---|
| configurator_key_hex | string | Private key for signing the end-user clients |
| configurator_key_curve | enum | End-user client key parameters alias:<br>• prime256v1 (default)<br>• secp384r1<br>• secp521r1<br>• brainpoolP256r1<br>• brainpoolP384r1<br>• brainpoolP512r1 |
| configurator_conf_role | enum | Set of configuration parameters given out to the Enrollee upon DPP Authentication:<br>• "sta-psk"<br>• "sta-sae"<br>• "sta-psk-sae"<br>• "sta-dpp"<br>• "sta-dpp-sae"<br>• "sta-dpp-psk-sae"<br>• "ap-psk"<br>• "ap-sae"<br>• "ap-psk-sae"<br>• "ap-dpp"<br>• "ap-dpp-sae"<br>• "ap-dpp-psk-sae" |
| configurator_conf_ssid_hex | string | The hex-encoded SSID to be given out to the Enrollee. |
| configurator_conf_psk_hex | string | The hex-encodepassphrase to be given out to the Enrollee. |
| peer_bi_uri | string | The DPP URI of an Enrollee. |

| | | |
|---|---|---|
| own_bi_key_hex | string | The local DPP key to be used. Currently planned only to be used for node onboarding. |
| own_bi_key_curve | enum | The own key parameters to be used in own_bi_key_hexr:<br>● prime256v1 (default)<br>● secp384r1<br>● secp521r1<br>● brainpoolP256r1<br>● brainpoolP384r1<br>● brainpoolP512r1 |
| timeout_seconds | integer | Once a device queue arrives at the particular DPP_Config row (ie. it moves it into in_progress status), the device will spend up to the timeout_seconds value (in seconds) waiting for a success/failure. |
| auth | enum | The DPP pairing intent type that implies which other parameters need to be set:<br>● initiate_on_announce (for zero touch, and pod-to-pod onboarding AP side)<br>● initiate_now (for non-zero touch qr-code scanning)<br>● respond_only (when providing qr code for another device to connect to the AP)<br>● chirp_and_respond (for pod-to-pod STA side) |
| ifnames | string | The  Wifi_VIF_Config ifnames this DPP pairing intent should be run at. These interfaces' statuses are then used to infer if a given DPP_Config row can be started or not, or should be restarted. |
| config_uuid | uuid | Points the result rows to their associated job config. |
| renew | boolean | When enabled, the node will provide results in a separate child-like DPP_Config row entry. This entry will be created while holding onto the given job config and without stopping the job. |
| status | enum | Current device status:<br>● requested (set by the Cloud only, can be also set on the onboarding nodes)<br>● in_progress (set by the device when it picks up the given DPP_Config row and processes it; the device processes only 1 DPP_Config row at any given time) |

| | | |
|---|---|---|
| | | • succeeded (DPP Auth completed, and the below sta_* or dpp_* columns are filled accordingly as much as possible)<br>• timed_out (timeout_seconds have elapsed with nothing else happening)<br>• failed (DPP Auth failed for whatever reason, as far as the target goes, could sometimes mean "internal timeout" if there is one, specific to target implementation) |
| sta_mac_addr | string | The Enrollee MAC address. Populated upon DPP successful completion. |
| sta_netaccesskey_sha256_hex | string | The public part of the end-user client key of the enrollee. These keys are ephemeral and bound to the given DPP Authentication. This column is intended for the cloud controller to later use the value to the populate DPP_Oftag table.<br><br>**Note:** These are not bootstrap keys. |
| akm | string | Exposes the DPP Configuration Object's AKM that was received when acting as an Enrollee. The DPP Configuration Object is handed upon DPP Configuration Request after the DPP Authentication has been completed.<br><br>**Note:** This is valid only for node onboarding and does not belong to the OpenSync 3.0 release. The options are:<br>• "psk"<br>• "sae"<br>• "psk-sae"<br>• "dpp"<br>• "dpp-sae"<br>• "dpp-psk-sae"<br><br>**Note:** AKM support is from OpenSync 3.2 release forward. |
| psk_hex | string | Exposes the DPP Configuration Object's WPA/SAE PSK when acting as Enrollee. This is expected only when the *akm* column includes "psk" or "sae". |
| ssid_hex | string | Exposes the DPP Configuration Object's SSID when acting as Enrollee. The value is expected only when auth=chirp_and_respond. |

| | | |
|---|---|---|
| pmk_hex | string | Same as psk_hex, but instead, the WPA PMK can be exposed. Reserved for possible future use. |
| dpp_netaccesskey_hex | string | Same as psk_hex, but expected for "dpp" akm. |
| dpp_connector | string | Same as psk_hex, but expected for "dpp" akm. |
| dpp_csign_hex | string | Same as psk_hex, but expected for "dpp" akm. |

# DPP_Announcement

This table stores information about the onboarded client devices. The table rows are aged out automatically by the OpenSync node itself.

| Name | Type | Description |
|---|---|---|
| sta_mac_addr | string | Chirping Enrollee MAC address observed on the air. |
| chirp_sha256_hex | string | The hash of "chirp" public key. |

# DPP_Oftag

This table enables handling of HomePass zoning for the non-PSK (DPP AKM) client devices.

**Note:** AKM support is supported OpenSync 3.2 release forward.

| Name | Type | Description |
|---|---|---|
| sta_netaccesskey_sha256_hex | string | The public part of the Enrollee's ephemeral client key. This key is generated during the DPP Auth, is signed by the Configurator/AP, and handed over back to the Enrollee as a DPP Connector. This is the same key hash as the one reported in the DPP_Config table. |
| oftag | string | The device reads and applies the oftags to non-PSK (DPP AKM) STA connections. |

# Wifi_VIF_Neighbors

The *Wifi_VIF_Neighbors* table contains the neighboring AP information for steering and other features.

| Name | Type | Description |
|---|---|---|
| bssid | string | BSSID of the neighbor |
| if_name | string | Associated interface name |
| channel | integer | Channel used by the neighbor |
| ht_mode | enum (*HT20*, *HT2040*, *HT40*, *HT40+*, *HT40-*, *HT80*, *HT160*, *HT80+80*) | Bandwidth used |
| op_class | integer | Builds a neighbor list to use with a BTM request sent to the client. |
| priority | integer | *Not used (always set to 1)* |

# Wifi_Channels

This table is DEPRECATED.

# Public_Wifi_Config

This table enables Home-as-a-hotspot feature management.

| Name | Type | Description |
|---|---|---|
| vlan_id | integer | Defines which VLAN is used for traffic. |
| gre_of_port | integer | Defines the GRE port number. |
| gre_endpoint | string | GRE endpoint FQDN. |
| keepalive_interval | integer | GRE tunnel keepalive parameter. |
| vif_ifnames | string | Defines which VIFs are to be connected to the GRE tunnel. |
| tunnel_ifname | string | Tunnel interface name |

# Lte_Config

In case of an internet outage over primary Ethernet WAN connection, the nodes switch to backup LTE link. The Lte_Config table stores the necessary parameters to enable LTE to act as a primary link, and Ethernet as a backup link.

| Name | Type | Description |
|---|---|---|
| if_name | string | Interface name |
| lte_failover_enable | boolean | Enable/disable failover on WAN outage |
| manager_enable | boolean | Enable/disable the LTE manager. |
| ipv4_enable | boolean | Enable/disable IPv4 addresses on LTE interface |
| ipv6_enable | boolean | Enable/disable IPv6 addresses on LTE interface |
| force_use_lte | boolean | Test mode to force LTE as the primary WAN interface |
| esim_download | string | Name of eSIM profile being downloaded |
| esim_active | string | Name of active eSIM profile |
| active_sim_card_slot | string | Active SIM card slot (slot0 or slot1) |
| modem_enable | boolean | Enable/disable LTE modem |
| report_interval | integer | MQTT report interval (in seconds) |
| apn | string | Access Point Name for the LTE network |
| lte_bands_enable | string | Enables/disables the LTE bands |
| os_persist | boolean | Once set to true, the LTE interface configuration remains persistent after a reboot. |
| enable_persist | boolean | Setting “enable_persist” to “true” makes the LTE interface configuration persistent and brings up the LTE interface during a wired Ethernet outage |
| esim_activation_code | string | eSIM activation code |
| esim_download_start | boolean | Triggers the LTE Manager to pass the esim_activation_code to the LPA code to download a |

| | | |
|---|---|---|
| | | profile to the eSIM. |
| esim_profile_activate | boolean | Triggers the LTE Manager to reset the modem and to start using the downloaded profile. |

# Lte_State

The Lte_state table represents the actual state on the device, and is therefore read by the Cloud.

| Name | Type | Description |
|---|---|---|
| if_name | string | Interface name |
| lte_failover_enable | boolean | Enable/disable failover on WAN outage |
| manager_enable | boolean | Enable/disable the LTE manager. |
| ipv4_enable | boolean | Enable/disable IPv4 addresses on LTE interface |
| ipv6_enable | boolean | Enable/disable IPv6 addresses on LTE interface |
| force_use_lte | boolean | Test mode to force LTE as the primary WAN interface |
| active_simcard_slot | string | Active SIM card slot (slot0 or slot1) |
| modem_enable | boolean | Enable/disable LTE modem |
| report_interval | integer | MQTT report interval (in seconds) |
| apn | string | Access Point Name for the LTE network |
| lte_bands_enable | string | LTE bands state |
| modem_present | boolean | Presence of LTE modem |
| iccid | string | Integrated circuit card ID |
| imei | string | International Mobile Equipment Identifier |
| imsi | string | International Mobile Subscriber Identity |
| chip_serial | string | Same as IMEI |
| sim_status | string | “Inserted”, “Removed”, "Bad", “Unknown” |

| | | |
|---|---|---|
| service_provider_name | string | Carrier name (AT&T, T-Mobile, etc.) |
| mcc | integer | Mobile Country Code |
| mnc | integer | Mobile Network Code |
| tac | integer | Tracking Area Code |
| lte_net_state | string | LTE network state. The options are:<br>● Not Registered<br>● Registered, Home Network<br>● Not Registered but searching<br>● Registration Denied<br>● Unknown<br>● Roaming |
| enable_persist | boolean | State of the “enable_persist” field |
| esim_activation_code | string | eSIM activation code |
| esim_download_start | boolean | Triggers the LTE Manager to pass the esim_activation_code to the LPA code to download a profile to the eSIM |
| esim_profile_activate | boolean | Triggers the LTE Manager to reset the modem and to start using the downloaded profile |
| esim_download_in_progress | boolean | Once the download starts, LTE Manager sets esim_download_in_progress to True |
| esim_download_complete | boolean | When the download is complete, LTE Manager sets esim_download_complete to True |
| esim_active_profile | boolean | Reports the active eSIM profile |
| modem_firmware_version | string | Displays the modem FW version |

# Wifi_Associated_Clients

*Wifi_Associated_Clients* lists all connected clients and their MAC addresses. *OpenSync* also updates the reference index in the proper *Wifi_VIF_State* row.

| Name | Type | Description |
|---|---|---|
| mac | string | MAC address of the associated client |

| | | |
|---|---|---|
| state | enum (*power save*, *idle*, *active*) | The state of client connection |
| capabilities | enum (*11b*, *11g*, *11a*, *11n*, *11ac*) | Client wireless capabilities, described by the conformity to one of the specified IEEE 802.11 standards |
| key_id | string | *(Not used)* |
| oftag | string | OpenFlow tag associated with the client |
| uapsd | integer | TBD |
| kick | string | The cloud uses this field to instruct the WM to kick a connected client. |
| dpp_netaccesskey_sha256_hex | string | Hash of the DPP Enrollee's public dpp_netaccesskey |

The *Wifi_Associated_Clients* table needs to show all clients on any interface: backhaul, home, guest, etc.

**Example:**

```
 # ovsh s Wifi_Associated_Clients
-------------------------------------------------------------------------
_uuid                       | 5bba~c937         | 8652~69d0         |
_version                    | 51b9~1523         | c771~6db9         |
capabilities                | ["set",[]]        | ["set",[]]        |
dpp_netaccesskey_sha256_hex | ["set",[]]        | ["set",[]]        |
key_id                      | key--1            | key--1            |
kick                        | ["map",[]]        | ["map",[]]        |
mac                         | e4:26:86:12:34:56 | e4:26:86:12:34:58 |
oftag                       | bhaul--1          | bhaul--1          |
state                       | active            | active            |
uapsd                       | ["set",[]]        | ["set",[]]        |
-------------------------------------------------------------------------
```

# Wifi_Credential_Config

*Wifi_Credential_Config* provides authentication details for the STA connection.

| Name | Type | Description |
|---|---|---|
| ssid | string | SSID for which the credentials apply |
| security | key/value | credentials |

| | | |
|---|---|---|
| | map | encryption: encryption type (e.g., WPA-PSK )<br>key: authentication key value |
| onboard_type | string | gre, no-gre |

**Example:**

```
# ovsh s Wifi_Credential_Config
-----------------------------------------------------
_uuid        | 03f2~d792                              |
_version     | f9b9~5a4a                              |
onboard_type | gre                                    |
security     | ["map",[["encryption","WPA-PSK"],      |
             : ["key","sample1234"]]]                    :
ssid         | sample                                 |
-----------------------------------------------------
```

# Wifi_Inet_Config

The Wifi_Inet_Config is a configuration table and is therefore read by the NM. The table is pushed by the Cloud or CM. The table stores system-wide network configuration:

- Interface address assignment method (DHCP, static, PPPoE, etc.)
- Interface MTU
- Interface creation for certain interface types (most notably, for GRE tunnels)
- DNS services (dnsmasq)

| Name | Type | Description |
|---|---|---|
| if_name | string | Interface name. Ethernet and VIF type interfaces must already exist. If they do not, an error should be reported. GRE, VLAN and possibly other types of interfaces will be created with the name specified by this field. |
| if_type | enum (*bridge*, *eth*, *vif*, *gre, gre6*, *vlan*, *pppoe*, *softwds*, *tap, lte*) | Interface type. Specifies the type of the selected interface. Some interface types are fixed (exist at boot), while the others must be created on the fly. Interface creation typically occurs when a row with the appropriate if_type is inserted into OVS.<br>• **bridge** - Specifies the bridge interface. The interface is created on the fly.<br>• **eth** - Contains the specific configuration for an Ethernet interface. The interface must already exist.<br>• **vif** - Specifies a wifi interface. It is assumed that the interface is created by the WM.<br>• **gre** - GRE tunneling interface. The interface is |

| | | |
|---|---|---|
| | | created on-the-fly when this row is inserted.The gre_ifname, gre_remote_inet_addr, and gre_local_inet_addr fields must not be empty.<br>● **gre6** - Specifies a GREv6 tunneling interface. Enables setting up GRE tunnels with IPv6 endpoint addresses.<br>● **vlan** - Specifies a VLAN interface. The interface is created on the fly when a row of this type is inserted. The parent_ifname field must not be empty.<br>● **pppoe** - PPP over Ethernet interface *(Not supported in OpenSync 2.0)*<br>● **softwds** - SoftWDS interface; deprecated<br>● **tap -** Specifies the TAP interface.<br>● **lte -** Specifies the LTE interface. |
| if_uuid | uuid | |
| enabled | boolean | The desired interface state. “True” if the interface should be in the UP state, otherwise the interface is in the DOWN state (the equivalent of ifconfig up/down). |
| network | boolean | In certain conditions, the interfaces must be in the UP state, but should not have any configuration applied. This field is “True” if the network configuration should be applied to the interface, and “False” if it should not. |
| NAT | boolean | “True” if NAT/Masquerading should be effective for the outgoing traffic on this interface. |
| ip_assign_scheme | enum (*none*, *dhcp*, *static*) | ● **none** - the interface has no address configuration. This is the equivalent of setting the *network* field to false<br>● **dhcp** - dynamic address configuration using the DHCP protocol. The dhcp or some other variant of the DHCP client should be started on this interface.<br>● **static** - The interface has static IP configuration. The address present in the “inet_addr” and “netmask” fields should be used for the IP address configuration. |
| inet_addr | string | The IP address when “ip_assign_scheme” is “static” |
| netmask | string | Interface netmask when “ip_assign_scheme” is “static” |
| gateway | string | The default gateway when “ip_assign_scheme” is “static” |

| | | |
|---|---|---|
| broadcast | string | The broadcast address when “ip_assign_scheme” is “static” |
| gre_remote_inet_addr | string | *Only applicable when “if_type” is “gre”*. Specifies the remote tunnel IP address. |
| gre_local_inet_addr | string | *Only applicable when “if_type” is “gre”*. Specifies the local tunnel IP address. |
| gre_remote_mac_addr | string | *Only applicable when “if_type” is “gre”*. Specifies the remote tunnel MAC address. |
| gre_ifname | string | *Only applicable when “if_type” is “gre”*. Specifies the parent interface for the GRE tunnel. |
| mtu | integer | Desired MTU |
| dns | key/value map | *Only applicable when “ip_assign_scheme” is “static”.* DNS server list. |
| dhcpd | key/value map | If populated, the DHCP server should be enabled on this interface. This field specifies a list of DHCP options that should be used to configure the DHCP server. See DHCP Server Configuration. |
| upnp_mode | enum (*disabled*, *internal*, *external*) | If populated and not set to “disabled”, UPnP is enabled on this interface. For more information, see the UPnP Configuration.<br>• **disabled** - UPnP is disabled, this is the default.<br>• **internal** - UPnP is enabled. Only one interface with this option value must exist. This specifies that a UPnP service must be started on this interface and that this is the interface facing the internal network.<br>• **external** - UPnP is enabled. Only one interface with this option value must exist. This specifies that a UPnP service must be started on this interface and that this is the interface facing the “external” network (internet).<br>• **internal_iptv** - Enabled secondary UPnP server for IPTV. This specifies that a UPnP service must be started on this interface and that this is the interface facing the internal network. Only one interface with this option value must exist.<br>• **external_iptv** - Enabled secondary UPnP server for IPTV. This specifies that a UPnP service must be started on this interface and that this is |

| | | |
|---|---|---|
| | | the interface facing the external network. Only one interface with this option value must exist. |
| dhcp_sniff | boolean | True if DHCP sniffing is enabled on this interface |
| vlan_id | integer | *Only applicable when "if_type" is "vlan"*. This is the VLAN ID. |
| vlan_egress_qos_map | string | Enables mapping of ethernet packet-assigned linux internal PRI (QoS) value to any other pri value represented in the PCP field of the 802.1Q header tag. Mapping is applied for egress traffic (i.e. output packets). Format of this string must follow TO:FROM convention, where TO defines input priority in 0..7 range, FROM defines output priority in 0..7 range. Up to 8 pairs separated by space are allowed.<br><br>E.g., "0:4 3:7" modifies the priority field of output VLAN packets, so that pri 0 is changed to 4 and priority 3 is changed to 7 in the PCP field of 802.1Q header tag. |
| parent_ifname | string | *Only applicable when "if_type" is "vlan"*. Parent interface name. |
| ppp_options | key/value map | *Only applicable when "if_type" is "pppoe"*. PPPoE options. Not supported in *OpenSync* 2.0. |
| softwds_mac_addr | string | *Only applicable when "if_type" is "softwds"*. Deprecated. |
| softwds_wrap | boolean | *Only applicable when "if_type" is "softwds"*. Deprecated. |
| igmp | boolean | *Enables IGMP multicast snooping on the interface. Applicable to OVS bridges only (mcast_snooping_enable).* |
| igmp_age | integer | *IGMP multicast snooping aging time in seconds (mcast-snooping-aging-time)* |
| igmp_tsize | integer | *IGMP multicast snooping table size (mcast-snooping-table-size)* |
| igmp_proxy | enum (*disabled,* IGMPv1, IGMPv2, IGMPv3) | Enables or disables IPv4 multicast proxy, i.e. IGMP on this interface. When a version is specified the proxy is enabled with that version. |
| mld_proxy | enum | Enables or disables ipv6 multicast proxy i.e MLD on this |

| | | |
|---|---|---|
| | (*disabled,* MLDv1, MLDv2) | interface. When a version is specified the proxy is enabled with that version. |
| role | string | Identifies the role of the interface against its name. The value is defined by the controller and used for reporting to the data pipeline. |
| mac_reporting | boolean | Enables or disables MAC reporting for the interface. |
| no_flood | boolean | Enables or disables port-flooding for the specified interface.<br><br>● “min": 0 (disabled)<br>● "max": 1 (enabled)<br><br>**Note:** Enabling this flag on non-OVS interfaces issues a warning. |
| collect_stats | boolean | Enables or disables stats monitoring for each interface. |
| os_persist | boolean | Once set to true, the LTE interface configuration remains persistent after a reboot. |

**Note:** IP forwarding must always be enabled on the device, regardless of the Wifi_Inet_Config table settings.

**Example:**

```
# ovsh s Wifi_Inet_Config

NAT                  : false
_uuid                : 0ebc~239c
_version             : e6f5~ca31
broadcast            : 192.168.40.255
dhcp_sniff           : ["set",[]]
dhcpd                :
["map",[["dhcp_option","3,192.168.40.1;6,192.168.40.1"],["force","false"],["lease_time","12h"],["start","1
92.168.40.50"],["stop","192.168.40.254"]]]
dns                  : ["map",[]]
enabled              : true
gateway              : ["set",[]]
gre_ifname           : ["set",[]]
gre_local_inet_addr  : ["set",[]]
gre_remote_inet_addr : ["set",[]]
gre_remote_mac_addr  : ["set",[]]
if_name              : br-home
if_type              : bridge
if_uuid              :
igmp                 : true
igmp_age             : ["set",[]]
igmp_tsize           : ["set",[]]
igmp_proxy           : IGMPv2
inet_addr            : 192.168.40.1
mld_proxy            : MLDv2
ip_assign_scheme     : static
mtu                  : ["set",[]]
netmask              : 255.255.255.0
network              : true
no_flood             : true
parent_ifname        : ["set",[]]
ppp_options          : ["map",[]]
softwds_mac_addr     : ["set",[]]
softwds_wrap         : ["set",[]]
upnp_mode            : internal
vlan_id              : ["set",[]]
```

# Wifi_Inet_State

This table reflects the current system status and is updated by the NM. The Cloud/CM reads this table to determine the current network status. If some data is not available on the platform, this data can be taken from the homonymous values that are available in the *Wifi_Inet_Config* table ***after*** they are applied to the system.

| Name | Type | Description |
|---|---|---|
| if_name | string | Interface name |
| inet_config | uuid reference | Reference to the corresponding row in Wifi_Inet_Config |
| if_type | enum (*bridge*, *eth*, *vif*, *gre*, *gre6*, *vlan*, *pppoe*, *softwds*, *tap, lte*) | Interface type. Can be derived from the homonymous field in Wifi_Inet_Config. |
| if_uuid | uuid | |
| enabled | boolean | “True” if the interface is administratively enabled. Can be derived from the homonymous field in Wifi_Inet_Config. |
| network | boolean | “True” if the network configuration is applied. Can be derived from the homonymous field in Wifi_Inet_Config. |
| NAT | boolean | “True” if the Network Address Translation is enabled for the outgoing traffic on this interface. Can be derived from the homonymous field in Wifi_Inet_Config. |
| ip_assign_scheme | string | The IP address assignment scheme. Can be derived from the homonymous field in Wifi_Inet_Config. |
| inet_addr | string | Currently configured IP address. If ip_assign_scheme is “dhcp”, this field must contain the assigned IP address; if ip_assign_scheme is “static”, the value can be derived from the homonymous field in Wifi_Inet_Config. |

| | | |
|---|---|---|
| netmask | string | Currently configured IP netmask. If ip_assign_scheme == “dhcp”, this field must contain the assigned IP netmask; if ip_assign_scheme == “static”, the value can be derived from the homonymous field in Wifi_Inet_Config. |
| gateway | string | Currently configured default gateway associated with this interface. The value can be derived from the homonymous field in Wifi_Inet_Config. |
| broadcast | string | Currently configured IP broadcast address. If ip_assign_scheme is “dhcp”, this field should contain the assigned IP broadcast address; if ip_assign_scheme is “static”, the value can be derived from the homonymous field in Wifi_Inet_Config. |
| gre_remote_inet_addr | string | *Only applicable if if_type == “gre”.* Currently configured GRE remote tunnel address. Can be derived from the homonymous field in Wifi_Inet_Config. |
| gre_local_inet_addr | string | *Only applicable if if_type == “gre”.* Currently configured GRE local tunnel address. Can be derived from the homonymous field in Wifi_Inet_Config. |
| gre_ifname | string | *Only applicable if if_type == “gre”.* GRE parent interface. Can be derived from the homonymous field in Wifi_Inet_Config. |
| mtu | integer | Currently configured MTU. Can be derived from the homonymous field in Wifi_Inet_Config. |
| dns | key/value map | Currently applied DNS settings related to this interface; can be derived from the homonymous field in Wifi_Inet_Config |
| dhcpd | key/value map | Currently applied DHCP server settings pertaining to this interface. Can be derived from the homonymous field in Wifi_Inet_Config. |
| hwaddr | string | Interface hardware address |
| dhcpc | key/value map | *Only applicable if ip_assign_scheme == “dhcp”.* This field contains the list of received DHCP client options. |
| upnp_mode | string | Currently configured UPnP mode of operation. Can be derived from the homonymous field in |

| | | |
|---|---|---|
| | | Wifi_Inet_Config. |
| vlan_id | integer | *Only applicable if if_type* is *“vlan”.* The VLAN tag associated with the current interface. can be derived from the homonymous field in Wifi_Inet_Config. |
| vlan_egress_qos_map | string | *Enables mapping of ethernet packet-assigned linux internal PRI (QOS) value to any other pri value represented in the PCP field of the 802.1Q header tag.* |
| parent_ifname | string | *Only applicable if if_type* is *“vlan”*, The parent interface. Can be derived from the homonymous field in Wifi_Inet_Config. |
| softwds_mac_addr | string | *Only applicable if if_type* is *“softwds”*. Not supported in *OpenSync* 2.0. |
| softwds_wrap | boolean | *Only applicable if if_type* is *“softwds”*. Not supported in *OpenSync* 2.0. |

# Wifi_Master_State

The *Wifi_Master_State* table enables synchronization between WM, NM, and CM. CM uses this table for determining the preferred backhaul connection (read-only). WM and NM update the values in this table. WM updates interfaces of type vif, while NM updates the rest.

| Name | Type | Description |
|---|---|---|
| dhcpc | map | DHCP client parameters |
| if_name | string | Interface name |
| if_type | enum | Interface type. Specifies the type of the selected interface. Some interface types are fixed (exist at boot), while the others must be created on the fly. Interface creation typically occurs when a row with the appropriate if_type is inserted into OVS.<br>• **bridge** - this row specifies the bridge interface. The interface is created on the fly.<br>• **eth** - this row contains the specific configuration for an Ethernet interface. The interface must already exist.<br>• **vif** - this row specifies a wifi interface. It is assumed that the interface is created by the |

| | | |
|---|---|---|
| | | WM.<br>● **gre** - GRE tunneling interface. The interface is created on-the-fly when this row is inserted.The gre_ifname, gre_remote_inet_addr, and gre_local_inet_addr fields must not be empty.<br>● **gre6** - specifies a GREv6 tunneling interface. Enables setting up GRE tunnels with IPv6 endpoint addresses.<br>● **vlan** - this row specifies a VLAN interface. The interface is created on the fly when a row of this type is inserted. The parent_ifname field must not be empty.<br>● **pppoe** - PPP over Ethernet interface *(Not supported in OpenSync 2.0)*<br>● **softwds** - SoftWDS interface; deprecated<br>● **tap -** TAP interface<br>● **lte -** LTE interface. Created on devices with LTE capabilities for the WAN over LTE feature. |
| if_uuid | uuid | Pointer to the Wifi_Inet_Config record |
| inet_addr | string | IP address |
| netmask | string | Netmask |
| network_state | enum | Network state<br>● up<br>● down |
| onboard_type | string | Onboard type |
| port_state | enum | Physical port state:<br>● active<br>● inactive |
| uplink_priority | integer | Uplink priority *(deprecated)* |

**Example:**

```
# ovsh s Wifi_Master_State
-------------------------------------------------------------------------------
_uuid           | 995f~e506  | 9a72~ea4c  | c69e~dab5       | 14df~c3a3       |
_version        | 0491~5aa8  | 13a7~017f  | 1ce1~9b2e       | 2849~b720       |
dhcpc           | ["map",[]] | ["map",[]] | ["map",[]]      | ["map",[]]      |
                :            :            :                 :                 :
                :            :            :                 :                 :
                :            :            :                 :                 :
                :            :            :                 :                 :
                :            :            :                 :                 :
if_name         | eth1       | pgd3-169   | wl2.1           | wl0.3           |
if_type         | eth        | gre        | vif             | vif             |
if_uuid         | 0000~0000  | 0000~0000  | 0000~0000       | 0000~0000       |
inet_addr       | 0.0.0.0    | 0.0.0.0    | 169.254.171.129 | 169.254.3.1     |
netmask         | 0.0.0.0    | 0.0.0.0    | 255.255.255.128 | 255.255.255.128 |
network_state   | up         | up         | up              | up              |
onboard_type    | ["set",[]] | ["set",[]] | ["set",[]]      | ["set",[]]      |
port_state      | active     | active     | active          | active          |
uplink_priority | ["set",[]] | ["set",[]] | ["set",[]]      | ["set",[]]      |
-------------------------------------------------------------------------------
```

# IP_Port_Forward

The *IP_Port_Forward* table contains the user-defined port forwarding rules.
This feature is only used when a device is configured to run in bridge mode.

| Name | Type | Description |
|---|---|---|
| protocol | enum (*tcp*, *udp*) | Protocol type |
| src_ifname | string | Source interface for the rule |
| src_port | integer | Source port |
| dst_port | integer | Destination port |
| dst_ipaddr | string | Destination IP address |

# OVS_MAC_Learning

The *OVS_MAC_Learning* table reports the current bridge MAC learning table to the Cloud. In the case of OVS bridges, the contents of this table are derived from various OVS tables.

| Name | Type | Description |
|---|---|---|
| brname | string | Bridge name |
| ifname | string | Interface from which the learned MAC address originates |
| hwaddr | string | Learned MAC address |
| vlan | integer | VLAN ID |

**Example:**

```
# ovsh s OVS_MAC_Learning
------------------------------
_uuid    | 631e~2674         |
_version | a321~1956         |
brname   | br-home           |
hwaddr   | 00:11:22:33:44:55 |
ifname   | eth0              |
vlan     | 0                 |
------------------------------
```

# DHCP_leased_IP

The *DHCP_leased_IP* table presents all device-specific information retrieved through DHCP fingerprinting, such as IP (only IPv4 in *OpenSync* 2.0) or MAC address.

| Name | Type | Description |
|---|---|---|
| hwaddr | string | MAC address of the associated client |
| inet_addr | string | IP address |
| hostname | string | Client hostname |
| fingerprint | string | DHCP option 55 |
| vendor_class | string | DHCP option 60 |
| lease_time | string | Lease time of DHCP in seconds |

**Example:**

```
# ovsh s DHCP_leased_IP
-------------------------------------------------------
_uuid        : ce82~d167
_version     : 8217~4250
fingerprint  : 1,3,6,15,31,33,43,44,46,47,119,121,249,252
hostname     : MasardaTvPC
hwaddr       : 54:8d:5a:56:ca:84
inet_addr    : 192.168.1.52
lease_time   : 83115
vendor_class : MSFT 5.0
-------------------------------------------------------
```

# DHCP_reserved_IP

The *DHCP_reserved_IP* table assigns a specific IP address to a known device MAC address.

| Name | Type | Description |
|---|---|---|
| hostname | string | Client hostname |
| hw_addr | string | MAC address of associated client |
| ip_addr | string | IP address |

**Example:**

```
# ovsh s DHCP_reserved_IP
------------------------------
_uuid    | ff76~1ece         |
_version | a18f~b288         |
hostname | ["set",[]]        |
hw_addr  | 11:22:33:44:55:66 |
ip_addr  | 192.168.40.77     |
------------------------------
```

# DHCP Server Configuration

The DHCP server configuration is represented as a map of strings (array of key, value pairs) in the *dhcpd* column of the *Wifi_Inet_Config* table. Since a map is a very flexible structure, the format used by the dhcpd column requires some explanation.

The format that is understood by the NM is as follows:

| Key | Value / Description |
|---|---|
| dhcp_option | “option_id,value;option_id,value”<br>Specifies a list of DHCP options to be provided by the DHCP server to the DHCP client.<br>Example: “3,192.168.40.1;6,192.168.40.1”<br>Note: DHCP option 3 = router; dhcp option 6 = DNS server |
| force | “true” or “false”<br>*Not used in OpenSync 2.0* |
| lease_time | Lease time in the N[mhs] format. This format must be understood by the dnsmasq.<br>Example: “12h” |
| start | IP pool start address |
| stop | IP pool end address |

## UPnP Configuration

The UPnP configuration requires exactly two interfaces to work properly: one WAN and one LAN interface. These interfaces are referred to as “external” and “internal” respectively.

These two interfaces  create dynamic port forwards from the WAN interface to the IP address on the LAN interface.

## DHCP Sniffing

The DHCP sniffing feature allows NM to capture DHCP packets on the configured interface, and extract the assigned hostname and DHCP fingerprint. This feature is mainly used when the device is configured to run in bridge mode.

# Wifi_Stats_Config

The *Wifi_Stats_Config* table is used to configure the stats collected on the device.

| Name | Type | Description |
|---|---|---|
| stats_type | enum (*neighbor*, *survey*, *client*, *capacity*, *radio*, *essid*, *quality*, *device*, *rssi*, *steering*) | The Cloud selects the stats type when it wants to fetch certain statistics from the device.<br><br>NOTE: Only *neighbor*, *survey*, *client*, *device*, *rssi*, and *steering* are used in *OpenSync* 2.0. |
| report_type | enum (*raw*, *average*, *histogram*, *percentile*, *diff*) | Cloud can specify different reporting formats:<br>• **raw:** bins of samples collected during the *sampling_interval* time<br>• **average:** value of samples or specific parameters collected during the *reporting_interval* time<br>• **histogram:** distributed sample values through the *reproting_interval*<br>• **percentile:** of sampled values inside the *reporting_interval*<br>• **diff:** special reporting when only diff values between samples are sent to the Cloud |
| radio_type | enum (*2.4G*, *5G*, *5GL*, *5GU, 6G*) | The selection of *radio_type* configuration depends on the device wireless capabilities listed inside the Wifi_VIF/Radio_State tables. |
| survey_type | enum (*on-chan*, *off-chan*, *full*) | For surveying (scan and utilization), the Cloud specifies the following types:<br>• **on-chan:** measurements on the home channel can be done frequently and without user interruption. They are done periodically.<br>• **off-chan:** measurements on the foreign channel must be done with care, since they involve switching to off-channel. The channel measurements listed in *channel_list* are periodically chosen using the round robin |

| | | |
|---|---|---|
| | | fashion (“1,6,11” -> t0=1, t1=6, t2=11, t3=1) to minimize user impact.<br>● **full:** the measurements are done on all channels at once and as specified in the *channel_list* (“1,6,11” -> t0=1,6,11 t1=1,6,11) which is very intrusive. |
| reporting_interval | integer | Interval specifying the time after which the report is sent |
| reporting_count | integer | Max number of consecutive reports. 0 means periodic. |
| sampling_interval | integer | Interval specifying the time between sample collections |
| survey_interval_ms | integer | Scan DWEL time used for surveying |
| channel_list | set of integers | Channel list that needs to be surveyed |
| threshold | key/value map | Threshold specifies the intrusiveness behavior of the scan, while it can also contain the diff thresholds:<br>● **max_delay:** max delay of measurement when the threshold is reached - each sampling interval the threshold delta is used<br>● **util:** utilization percentage that still allows measurements |

**Example:**

```
# ovsh s Wifi_Stats_Config
-------------------------------------------------------------------------------------------------...
_uuid              | b5db~37e6              | de8e~6de0  | 006d~c137         | 88e8~9d00  | 0d50~4d65  |
_version           | 0572~d01b              | 37ff~8d4e  | 4a66~909b         | 735f~01af  | 82a9~ea8f  |
channel_list       | ["set",[40,153]]       | ["set",[]] | ["set",[1,6,11]] | ["set",[]] | ["set",[]] |
radio_type         | 5G                     | 2.4G       | 2.4G              | 5G         | 2.4G       |
reporting_count    | 0                      | 0          | 0                 | 0          | 0          |
reporting_interval | 0                      | 60         | 120               | 60         | 900        |
sampling_interval  | 0                      | 0          | 0                 | 10         | 0          |
stats_type         | survey                 | neighbor   | neighbor          | client     | device     |
survey_interval_ms | 10                     | 0          | 0                 | ["set",[]] | ["set",[]] |
survey_type        | off-chan               | on-chan    | off-chan          | ["set",[]] | ["set",[]] |
threshold          | ["map",[["max_delay",  | ["map",[]] | ["map",[]]        | ["map",[]] | ["map",[]] |
                   : 600],["util",10]]]    :            :                   :            :            :
-------------------------------------------------------------------------------------------------...
```

```
...-------------------------------------------------------------------------------------------------
    | 27f5~40e1              | 3c28~cbb1  | 52e5~0480          | d943~6425  | ebe4~aa07  | 9205~441b  |
    | a271~1855              | a4aa~db93  | ad9a~918c          | cf24~f006  | da27~99f0  | ff08~5577  |
    | ["set",[1,6,11]]       | ["set",[]] | ["set",[40,153]] | ["set",[]] | ["set",[]] | ["set",[]] |
    | 2.4G                   | 5G         | 5G                 | 2.4G       | 5G         | 2.4G       |
    | 0                      | 0          | 0                  | 0          | 0          | 0          |
    | 120                    | 60         | 0                  | 60         | 60         | 60         |
    | 10                     | 10         | 0                  | 10         | 0          | 10         |
    | survey                 | survey     | neighbor           | client     | neighbor   | survey     |
    | 50                     | 0          | 10                 | ["set",[]] | 0          | 0          |
    | off-chan               | on-chan    | off-chan           | ["set",[]] | on-chan    | on-chan    |
    | ["map",[["max_delay",  | ["map",[]] | ["map",[]]         | ["map",[]] | ["map",[]] | ["map",[]] |
    : 600],["util",10]]]     :            :                    :            :            :            :
...-------------------------------------------------------------------------------------------------
```

# Band_Steering_Config

The table enables general configuration of band steering and client steering parameters.

| Name | Type | Description |
|---|---|---|
| chan_util_avg_count | integer | Channel utilization average count |
| chan_util_check_sec | integer | Channel utilization sampling period for pre-association band steering (in seconds) |
| chan_util_hwm | integer | Channel utilization high water mark |
| chan_util_lwm | integer | Channel utilization low watermark |
| dbg_2g_raw_chan_util | boolean | Enables channel utilization logging on the 2.4G interface |
| dbg_2g_raw_rssi | boolean | Enables raw RSSI logging on the 2.4G interface |
| dbg_5g_raw_chan_util | boolean | Enables channel utilization logging on the 2.4G interface |
| dbg_5g_raw_rssi | boolean | Enables raw RSSI logging on the 5G interface |
| debug_level | integer | Sets overall logging severity level |
| def_rssi_inact_xing | integer | Inactive RSSI threshold |
| def_rssi_low_xing | integer | Inactive RSSI low threshold |
| def_rssi_xing | integer | Inactive RSSI high threshold |
| gw_only | boolean | Indicates if this is the only *OpenSync* device (gateway) in this location |

| | | |
|---|---|---|
| if_name_2g | string | 2.4G interface name |
| if_name_5g | string | 5G interface name |
| inact_check_sec | integer | Client inactivity check period |
| inact_tmout_sec_normal | integer | Client inactivity timeout in normal mode |
| inact_tmout_sec_overload | integer | Client inactivity timeout in overload mode |
| kick_debounce_period | integer | Time for which client kick is blocked due to failed attempts |
| kick_debounce_thresh | integer | Number of failed attempts before client kicks are disabled for the time period defined by *kick_debounce_period* |
| stats_report_interval | integer | Time period in seconds which defines band steering related stats reporting to the Cloud |
| success_threshold_secs | integer | Time period in seconds which defines the time of the successful kick |
| ifnames | string | A map replacing the names stored in if_name_2g and if_name_5g columns. Can be used for 6g interfaces. |

**Example:**

```
# ovsh s Band_Steering_Config
----------------------------------------
_uuid                     | 48f4~5ae6   |
_version                  | 9153~edce   |
chan_util_avg_count       | 0           |
chan_util_check_sec       | 0           |
chan_util_hwm             | 80          |
chan_util_lwm             | 50          |
dbg_2g_raw_chan_util      | false       |
dbg_2g_raw_rssi           | false       |
dbg_5g_raw_chan_util      | false       |
dbg_5g_raw_rssi           | false       |
debug_level               | 0           |
def_rssi_inact_xing       | 0           |
def_rssi_low_xing         | 0           |
def_rssi_xing             | 0           |
gw_only                   | false       |
if_name_2g                | home-ap-24  |
if_name_5g                | home-ap-u50 |
inact_check_sec           | 10          |
inact_tmout_sec_normal    | 60          |
inact_tmout_sec_overload  | 30          |
kick_debounce_period      | 0           |
kick_debounce_thresh      | 0           |
stats_report_interval     | 1           |
success_threshold_secs    | 15          |
----------------------------------------
```

# Band_Steering_Clients

This table contains per client band steering configuration and states.

| Name | Type | Description |
|---|---|---|
| backoff_exp_base | integer | Exponential duration after pre-assoc steering failures |
| backoff_secs | integer | Backoff timer before another steering event is processed |
| cs_mode | enum (*off*, *home*, *away*) | Client steering mode:<br>● off<br>● home<br>● away |
| cs_params | string | Client steering parameters |
| cs_state | enum (*none*, | Client steering state:<br>● none |

| | | |
|---|---|---|
| | *steering*, *expired*, *failed*, *xing_low*, *xing_high*, *xing_disabled*) | • steering<br>• expired<br>• failed<br>• xing_low<br>• xing_high<br>• xing_disabled |
| force_kick | enum (*none*, *speculative*, *directed*, *ghost_device*) | Trigger force client kick |
| pref_bs_allowed | string | Configures how BM should report probe requests events to the controller.<br>When not set, the default BM value is used (3 dB). When preq_snr_thr equals 0, BM reports each probe request event to the controller. In other cases, BM reports probe request events to the controller only if the difference between the last reported probe request event SNR and current SNR is higher or equal to preq_snr_thr. |
| preq_snr_thr | integer | See pref_bs_allowed |
| | | |
| hwm | integer | High watermark |
| kick_debounce_period | integer | When a kick has failed, another kick will not be attempted for this amount of seconds (and once client's RSSI crosses the HWM/LWM threshold again). |
| kick_reason | integer | Client kick reason |
| kick_type | enum (*none*, *deauth*, *disassoc*, *bss_tm_req*, *rrm_br_req*, *btm_deauth*, *btm_disassoc*) | Different kick types:<br>**deauth** - sending deauthentication frame to the client<br>**disassoc** - sending disassociation frame to the client<br>**bss_tm_req** - sending BSS Transition Management(802.11v) frame to the client<br>**btm_deauth** - If a client is 11v capable, send a 802.11v frame, else send a deauthentication frame.<br>**btm_disassoc** - If the client is 11v capable, |

| | | |
|---|---|---|
| | | send a 802.11v frame, else send a disassociation frame. |
| kick_upon_idle | boolean | If kicking is enabled, attempt to “kick” the client only if it is not ‘busy’ (doing data transfer). If the client is busy, wait until it becomes idle. |
| lwm | integer | Low watermark. If kicking is enabled and if the client's active RSSI goes below this value, the node attempts to kick the client for being a “sticky client". |
| mac | string | Client MAC address |
| max_rejects | integer | If we get this number of rejects within the time period (*rejects_tmout_secs*), we consider steering as failed. |
| pre_assoc_auth_block | boolean | Block responses to authorization requests from a client during pre-association band steering. |
| pref_5g | enum (*hwm*, *never*, *always*) | Prefer connecting clients on the 5 GHz radio based on these settings:<br>● **hwm** - Depending on the signal strength on 5 GHz, the radio blocks 2.4 GHz.<br>● **never**<br>● **always**<br>● **nonDFS -** Certain clients might not prefer DFS channels. In such a case, BM does not block the 2.4G interface when a DFS channel is used for 5G/5GL/5GU. |
| reject_detection | string | Why was the client rejected? Possible values:<br>● none<br>● probe_all<br>● probe_null<br>● probe_direcet<br>● auth_blocked |
| rejects_tmout_secs | integer | Number of rejects (*max_rejects*) within the time period (*rejects_tmout_secs*) to be considered a steering failure, and to not go into backoff for 300 seconds |

| | | |
|---|---|---|
| | | (*backoff_secs*).<br>**Note:** Deprecated field. A timer-based pre-assoc 5 GHz steering is used instead. |
| pref_5g_pre_assoc_block_timeout_msecs | integer | This field specifies how long the 2.4 GHz band should be blocked for an ST. The value is valid only when pref_5g == always policy. |
| rrm_bcm_rpt_params | map | 802.11k beacon report request parameters |
| sc_kick_debounce_period | integer | Steering from Cloud(sc) client kick debounce period |
| sc_kick_reason | integer | Steering from Cloud(sc) client kick reason |
| sc_kick_type | enum<br>(*none*,<br>*deauth*,<br>*disassoc*,<br>*bss_tm_req*,<br>*rrm_br_req*,<br>*btm_deauth,*<br>*btm_disassoc*,<br>*rrm_deauth,*<br>*rrm_disassoc*) | Possible kick types:<br>● none<br>● disassoc<br>● deauth<br>● 802.11v BSS Transition Request<br>● 802.11k Beacon Report Request<br>● 802.11v request or deauth<br>● 802.11v request or disassoc |
| sc_btm_params | map | 802.11v BSS Transition request parameters |
| stats_2g | map | *Not used* |
| stats_5g | map | *Not used* |
| steer_during_backoff | bool | Allow/disallow steering during backoff |
| sticky_kick_guard_time | integer | Guard time (in seconds) describe how long BM will monitor connection(s) after steering/sticky kick. |
| steering_kick_guard_time | integer | Guard time (in seconds) describe how long BM will monitor connection(s) after steering/sticky kick. |
| sticky_kick_backoff_time | integer | Defines the post association steering/sticky kick backoff time. |
| steering_kick_backoff_time | integer | Defines the post association steering/sticky |

| | | |
|---|---|---|
| | | kick backoff time. |
| settling_backoff_time | integer | Defines the settling time between steering after successful attempts. |
| send_rrm_after_assoc | boolean | Enables sending of measurement beacon request (active mode) to the client after client connection. |
| send_rrm_after_xing | boolean | Enables direct measurement beacon request after a crossing low event (XING_LOW). |
| rrm_better_factor | integer | Set in dB. Allow checking the RRM results - how a client sees current and other nodes. If the difference is less than a factor, a client could connect to a better node. A BTM request should be prepared. |
| rrm_age_time | integer | Each RRM single result has an age (in seconds). When we check if a BTM request has been created, only fresh RRM results (skip older than rrm_age_time seconds) are used. |
| active_treshold_bps | integer | Enables detection of client status: active/inactive. |
| steering_btm_params | map | BSS Transition Management (802.11v) configuration. |
| steering_fail_cnt | integer | Client steering failed attempt count |
| steering_kick_cnt | integer | Client steering kick count |
| steering_success_cnt | integer | Client steering successful attempt count |
| sticky_kick_cnt | integer | Sticky client kick count |
| sticky_kick_debounce_period | integer | Sticky client kick debounce period |
| sticky_kick_reason | integer | Sticky client kick reason |
| sticky_kick_type | enum (*none*, *deauth*, *disassoc*, *bss_tm_req*, *rrm_br_req*, | Possible kick types:<br>● none<br>● disassoc<br>● deauth<br>● 802.11v BSS Transition Request<br>● 80211k Beacon Report Request |

| | | |
|---|---|---|
| | *btm_deauth*,<br><br>*btm_disassoc*) | ● 802.11v request or deauth<br>● 80211v request or disassoc |
| pref_6g | string | Prefer connecting clients on 6 GHz radio based on these settings:<br>● never<br>● always |
| pref_6g_pre_assoc_block_timeout_msecs | integer | This field specifies for how long an STA should be blocked. The value is valid only when pref_6g == hwm policy. |
| neighbor_list_filter_by_beacon_report | boolean | Enables/disables filtering of neighbors per STA using the 11k/v neighbor report. |
| bottom_lwm | integer | Bottom low watermark - If kicking is enabled and if the legacy (i.e., doesn’t support IEEE 802.11k/v) client's active RSSI drops below this value, the client is forcibly kicked. |

**Example:**

```
# ovsh s Band_Steering_Clients
-------------------------------------------------------------------------------------------------
_uuid                       | de8a~5d9e                                                         |
_version                    | b09a~1360                                                         |
backoff_exp_base            | 2                                                                 |
backoff_secs                | 120                                                               |
cs_mode                     | ["set",[]]                                                        |
cs_params                   | ["map",[]]                                                        |
cs_state                    | ["set",[]]                                                        |
force_kick                  | ["set",[]]                                                        |
hwm                         | 35                                                                |
kick_debounce_period        | 60                                                                |
kick_reason                 | 1                                                                 |
kick_type                   | btm_deauth                                                        |
kick_upon_idle              | true                                                              |
lwm                         | 20                                                                |
mac                         | 33:07:4d:33:22:33                                                 |
max_rejects                 | 7                                                                 |
pre_assoc_auth_block        | true                                                              |
pref_5g                     | always                                                            |
reject_detection            | probe_all                                                         |
rejects_tmout_secs          | 120                                                               |
rrm_bcn_rpt_params          | ["map",[]]                                                        |
sc_btm_params               | ["map",[]]                                                        |
Sc_kick_debounce_period     | 0                                                                 |
sc_kick_reason              | 0                                                                 |
sc_kick_type                | ["set",[]]                                                        |
stats_2g                    | ["map",[]]                                                        |
stats_5g                    | ["map",[]]                                                        |
steer_during_backoff        | false                                                             |
steering_btm_params         | ["map",[["abridged","1"],["bss_term","0"],["btm_max_retries","3"],  |
                            : ["btm_retry_interval","10"],["disassoc_imminent","1"],["pref","1"], :
                            : ["valid_interval","255"]]]                                        :
steering_fail_cnt           | 0                                                                 |
steering_kick_cnt           | 0                                                                 |
steering_success_cnt        | 0                                                                 |
sticky_btm_params           | ["map",[["abridged","1"],["bss_term","0"],["btm_max_retries","3"],  |
                            : ["btm_retry_interval","10"],["disassoc_imminent","1"],["inc_neigh", :
                            : "true"],["pref","0"],["valid_interval","255"]]]                   :
sticky_kick_cnt             | 0                                                                 |
sticky_kick_debounce_period | 60                                                                |
sticky_kick_reason          | 1                                                                 |
sticky_kick_type            | btm_deauth                                                        |
-------------------------------------------------------------------------------------------------
```

# AW_LM_Config

This table enables the logpull functionality.

| Name | Type | Description |
|---|---|---|
| upload_location | string | URL for uploading the collected files |
| upload_token | string | Security token for uploading the collected files |
| name | string | *(Deprecated)* |
| periodicity | string | *(Deprecated)* |

# AW_LM_State

This table is DEPRECATED.

# AW_Debug

The table is currently used for setting the log levels to the registered modules. Note that this table is used only as a local interface.

| Name | Type | Description |
|---|---|---|
| name | string | Module name used for registration to the *OpenSync* log library |
| log_severity | string | Dynamic log severity, which can be one of these values:<br>• **EMERG** - System is unusable<br>• **ALERT** - Action must be taken immediately<br>• **CRIT** - Critical conditions<br>• **ERR** - Error conditions<br>• **WARNING** - Warning conditions<br>• **NOTICE** - Normal but significant condition<br>• **INFO** - Informational message<br>• **DEBUG** - Debug message<br>• **TRACE** - Trace messages |

# Openflow_Config

This table contains all the flows that need to be applied to the system. Flows can be set as a constant or defined as a template by using variables described in Openflow_Tag or Openflow_Tag_Group table.

| Name | Type | Description |
|---|---|---|
| action | string | Flow action (drop, normal, resubmit, etc.) |
| bridge | string | Bridge name |
| priority | integer | Flow priority |
| rule | string | Flow rule or flow rule template |
| table | integer | Flow table |
| token | string | Name of the flow |

# Openflow_State

This table reflects the status of all currently applied flows.

| Name | Type | Description |
|---|---|---|
| bridge | string | Bridge name |
| openflow_config | string | NOT USED |
| success | boolean | Marks successfully applied flow |
| token | string | Name of the flow |

# Openflow_Tag

This table enables expansion of the packet flow rules. Defines variables used to construct a flow in combination with a template rule from Openflow_Config table

| Name | Type | Description |
|---|---|---|
| cloud_value | string | Cloud-only tag value |

| | | |
|---|---|---|
| device_value | string | Device-only tag value |
| name | string | Tag value name |

# Openflow_Local_Tag

If Captive Portal is used at a location, this table stores the white-listed domains for resolving the IP addresses. These are used subsequently to create openflow rules to allow traffic to the stored white-listed domains.

| Name | Type | Description |
|---|---|---|
| values | string | White-listed domain address |
| name | string | Descriptive name of the white-listed domain |

# Openflow_Tag_Group

This table combines a list of tags defined in the Openflow_Tag table to create a new group tag. Group tag can be used to construct a flow in combination with a template rule from the Openflow_Config table.

| Name | Type | Description |
|---|---|---|
| tags | string | List of tags from Openflow_Tag table |
| name | string | Tag group name |

# Client_Nickname_Config (not used)

Used for nickname synchronization between the device and the Cloud.

| Name | Type | Description |
|---|---|---|
| mac | string | Client MAC address |
| nickname | string | Client nickname |

**Example:**

```
# ovsh s Client_Nickname_Config
------------------------------
_uuid    | c431~a7a0         |
_version | 4913~5531         |
mac      | 00:11:22:33:44:55 |
nickname | device_test_name  |
------------------------------
```

# Client_Freeze_Config (not used)

The device freeze feature restricts client access to the internet or local network. Users set the rules over mobile app/cloud or local GUI if present. The user can perform instant freeze or may arrange multiple scheduled freezes (Bedtime, School nights, custom).

| Name | Type | Description |
|---|---|---|
| blocked | bool | Block or unblock the client |
| mac | string | Client MAC address |
| source | enum (*init*, *cloud*, *gw*) | Mark the source of truth:<br>● **init** - unknown owner after reboot<br>● **cloud** - cloud managed client<br>● **gw** - gateway or device is the owner |
| type | enum (*schedule*, *always*) | The cloud controller only sets "*always*" value here. However, some devices might set the values to "*schedule*" if the device supports a schedule mode that is configured outside the scope of the cloud controller. |

**Example:**

```
# ovsh s Client_Freeze_Config
------------------------------
_uuid    | 5382~faee         |
_version | f039~bd24         |
blocked  | false             |
mac      | 69:78:65:66:76:69 |
source   | gw                |
type     | schedule          |
------------------------------
```

# Node_Config

This table serves as a key-value storage. As a generic interface, Node_Config allows determining the state of the node (including overrides) and control applications or services

| Name | Type | Description |
|---|---|---|
| persist | boolean | Makes the functionality setting persistent after a node reboot. |
| module | string | The name of the module. This could be an OpenSync manager, or an add-on service. |
| key | string | The name of the key. Can be the name of the functionality that belongs to a module, or enable/disable toggle. |
| value | string | Setting that is relevant for the given key. |

**Example:**

Enabling SIP-ALG on a node:

```
# ovsh s Node_Config -T
------------------------------
_uuid    | b9cf~30b5         |
_version | dbd9~0941         |
module   | sipalg            |
key      | enable            |
value    | true              |
------------------------------
```

# Node_State

This table displays the current key-value storage state.

| Name | Type | Description |
|---|---|---|
| persist | boolean | Makes the functionality setting persistent after a node reboot. |
| module | string | The name of the module. This could be an OpenSync manager, or an add-on service. |
| key | string | The name of the key. Can be the name of the functionality that belongs to a module, or enable/disable toggle. |
| value | string | Setting that is relevant for the given key. |

# Flow_Service_Manager_Config

This table provisions the plugins to the FSM service. The table informs the FSM about the plugin name, and provides additional information: location of the shared library instantiating the plugin, and plugin type (parser dedicated to a specific traffic type, DPI plugin, or web categorization backend plugin).

| Name | Type | Description |
|---|---|---|
| handler | string | Unique FSM plugin identifier |
| type | enum | Plugin type:<br>- parser: Dedicated to a specific traffic type<br>- web_cat_provider: Dedicated to categorize the web access type<br>- dpi: legacy plugin for general data traffic signature<br>- dpi_plugin: plugins for general data traffic signature<br>- dpi_dispatcher:<br>  - binds to a given tap interface<br>  - processes received packets and manages the relevant flows<br>  - dispatches received packets to the dpi plugins<br>- dpi_client: brokers between dpi plugins, web |

| | | |
|---|---|---|
| | | category provides and policy enforcers<br>Read more about the FSM plugin types in EDE-021-030-501 OpenSync FSM Plugins |
| if_name | string | Name of the tap interface, from which the FSM gets the packets and forwards them to the plugin packet handler |
| pkt_capt_filter | string | An optional BPF filter applied to the incoming traffic. Originally used to filter out the multicast/broadcast traffic; now optional. |
| plugin | string | The DSO file instantiating the plugin. |
| other_config | key/value map | A (key, value) map allowing the opaque parameters to pass to the plugin. |

**Example:**

```
# ovsh s Flow_Service_Manager_Config -w handler==http
-------------------------------------------------------------------------------------------------------
_uuid           | 63d6~3fc4                                                                           |
_version        | d3a1~0f61                                                                           |
handler         | http                                                                                |
if_name         | br-home.http                                                                        |
other_config    | ["map",[["mqtt_v","HTTP/Requests/opensync/4C77701123/59efd33d2c9383202533aaaa"]]] |
pkt_capt_filter | tcp dst port 80                                                                     |
plugin          |                                                                                     |
type            | ["set",[]]                                                                          |
-------------------------------------------------------------------------------------------------------
```

# FSM_Policy

The FSM_Policy table instructs the requesting FSM plugins which action to take in case of a specific event. Its main usage targets advanced device typing, IP threat detection, and content filtering.

The organized policy names are contained in the rules columns. A rule is a set made of filters, an action, and a report policy. Should an event pass the filters, the plugin applies the rules's action, and reports the event according to the report policy.

| Name | Type | Description |
|---|---|---|
| name | string | Policy rule name |
| idx | integer | Within a policy name space; the sorted index of |

| | | |
|---|---|---|
| | | the current rule |
| mac_op | enum<br>(*in*, *out*) | {event mac} (mac_op) [set of macs] filter. IE is mac X in/out the given set of macs |
| macs | set of strings | The set of macs to compute an event against |
| fqdn_op | enum<br>(*in*, *out*, *sfr_in*, *sfr_out*, *sfl_in*, *sfl_out*) | {event fqdn} (fqdn_op) [set of fqdns] filter<br>● sfr denotes “start from right”<br>● sfl denotes “start from left”<br>These prefixes allow wildcard matching. |
| fqdns | string | The set of FQDNs to compute an event against |
| fqdncat_op | enum<br>(*in*, *out*) | {event fqdn category} (fqdncat_op) [set of fqdn categories] filter.<br>The FQDN category is determined by the web_cat_provider plugin. |
| fqdncats | set of integers | The set of FQDN categories to compute an event against. |
| ipaddr_op | enum<br>(*in*, *out*) | {ip addr} (ipaddr_op) [set of ip addr] filter. |
| ipaddrs | set of strings | The set of ip addresses to compute an event against. |
| risk_level | integer | The web remote resource risk level to compare against. |
| risk_op | enum<br>(*eq*, *neq*, *gt*, *lt*, *gte*, *lte*) | The risk operation to use in the risk assessment. |
| app_op | string | Allows the controller to request invalidation of Gatekeeper cache entries. |
| apps | string | Allows the controller to request invalidation of Gatekeeper cache entries. |
| log | enum<br>(*none*, *all*, *blocked*) | **none** - do not log the event<br>**all** - log the event regardless of the action<br>**blocked** - log only blocked events (web categorization) |
| action | enum<br>(*allow*, *drop,* | Applies to web categorization events. |

| | | |
|---|---|---|
| | *update_tag, gatekeeper, flush, flush all*) | |
| next | key:integer | Enables jumping to the specific rule of the requested table, chaining up the event processing across multiple tables. |
| policy | string | Policy rule name |
| redirect[0..2] | string | Redirects a FQDN resolution to the provided Ipv4/IPv6/cname |
| other_config | key/value map | A (key, value) map allowing the opaque parameters to pass the rule. |

**Example:**

```
# ovsh  s FSM_Policy -w idx==2
------------------------------
_uuid        | 6333~af57      |
_version     | b5be~006c      |
action       | ["set",[]]     |
fqdn_op      | ["set",[]]     |
fqdncat_op   | ["set",[]]     |
fqdncats     | ["set",[]]     |
fqdns        | ["set",[]]     |
idx          | 2              |
ipaddr_op    | ["set",[]]     |
ipaddrs      | ["set",[]]     |
log          | all            |
mac_op       | out            |
macs         | $[dns-exclude] |
name         | dt_dns         |
next         | ["map",[]]     |
other_config | ["map",[]]     |
policy       | my_policy      |
redirect     | ["set",[]]     |
risk_level   | ["set",[]]     |
risk_op      | ["set",[]]     |
------------------------------
```

# FCM_Collector_Config

This table contains configurations for the FCM plugin. This plugin collects and reports the network flow statistics.

| Name | Type | Description |
|---|---|---|
| name | string | Name of the plugin used for network flow collection. **Important!** Do not modify the name of the plugin. |
| interval | integer | Collection interval for network flow samples in seconds. |
| filter_name | string | Name of FCM_Filter to be applied on network flows during the collection time |
| report_name | string | Name of FCM_Report_Config to be used for sending reports about the collected flows. |
| other_config | key/value map | Contains plugin-specific configurations such as shared library path, entry function name of the plugin, and conntrack zone to be used for flow collection. |

**Example:**

```
# ovsh s FCM_Collector_Config
-------------------------------------------------------------------------------------------
_uuid        | 7315~98ec                                                                    |
_version     | 7fe0~97b1                                                                    |
filter_name  | ip_filter                                                                    |
interval     | 10                                                                           |
name         | ct_stats                                                                     |
other_config | ["map",[["ct_zone","1"],["dso_init","ct_stats_plugin_init"],["dso_path", |
             : "/usr/opensync/lib/"]]]                                                        :
report_name  | ip_flow_report                                                               |
-------------------------------------------------------------------------------------------
```

# FCM_Filter

This table contains various FCM filter attributes that can be applied to the network flows.

| Name | Type | Description |
|---|---|---|
| name | string | Name of the filter |
| index | integer | Priority of the filter. Index 0 is applied first, and so on. |
| smac | set of strings | Array of source mac addresses. Accepts smac Tags. |
| dmac | set of strings | Array of destination mac addresses. Accepts dmac tags. |
| vlanid | set of integers | Array of VLAN ID values |
| src_ip | set of strings | Array of source IP addresses |
| dst_ip | set of strings | Array of destination IP addresses |
| src_port | set of strings | Array of source port values |
| dst_port | set of strings | Array of destination port values |
| proto | set of integers | Array of protocol number values |
| smac_op | enum<br>(*in*, *out*, *none*) | **in** - Flows matching *'smac'* values will be considered for final *'action'*<br>**out** - Flows not matching *'smac'* values will be considered for final *'action*' |
| dmac_op | enum<br>(*in*, *out*, *none*) | **in** - Flows matching *'dmac'* values will be considered for final *'action'*<br>**out** - Flows not matching *'dmac'* values will be considered for final *'action*' |
| vlanid_op | enum<br>(*in*, *out*, *none*) | **in** - Flows matching '*vlanid'* values will be considered for final *'action'*<br>**out** - Flows not matching '*vlanid'* values will be considered for final *'action'* |
| src_ip_op | enum<br>(*in*, *out*, *none*) | **in** - Flows matching '*src_ip'* values will be considered for final *'action'*<br>**out** - Flows not matching '*src_ip'* values will be considered for final *'action'* |

| | | |
|---|---|---|
| dst_ip_op | enum<br>(*in*, *out*, *none*) | **in** - Flows matching '*dst_ip'* values will be considered for final *'action'*<br>**out** - Flows not matching '*dst_ip'* values will be considered for final *'action'* |
| src_port_op | enum<br>(*in*, *out*, *none*) | **in** - Flows matching *'src_port'* values will be considered for final *'action'*<br>**out** - Flows not matching *'src_port'* values will be considered for final *'action'* |
| dst_port_op | enum<br>(*in*, *out*, *none*) | **in** - Flows matching '*dst_port'* values will be considered for final *'action'*<br>**out** - Flows not matching *'dst_port*' values will be considered for final *'action'* |
| proto_op | enum<br>(*in*, *out*, *none*) | **in** - Flows matching '*proto'* values will be considered for final *'action'*<br>**out** - Flows not matching *'proto'* values will be considered for final *'action'* |
| pktcnt | integer | Number of packets in a flow |
| pktcnt_op | enum<br>(*lt*, *leq*, *gt*, *geq*, *eq*, *neq*) | **le** - Flows having packet count less than *'pktcnt'* will be considered for final *'action'*<br>**leq** - Flows having packet count less than or equal to ` *pktcnt'* will be considered for final *'action'*<br>**gt** - Flows having packet count greater than *'pktcnt'* will be considered for final *'action'*<br>**geq** - Flows having packet count greater than or equal to *'pktcnt'* will be considered for final *'action'*<br>**eq** - Flows having packet count equal to *'pktcnt'* will be considered for final *'action'*<br>**neq** - Flows having packet count not equal to *'pktcnt'* will be considered for final *'action'* |
| action | enum<br>(*include*, *exclude*) | Final action of the considered flows based on the above said options:<br>**include** - Flow will be included for further processing in FCM<br>**exclude** - Flow will be excluded from further processing in FCM |
| other_config | set of key:value | Not used |

**Example:**

```
# ovsh s FCM_Filter
-------------------------------------------------
_uuid        | ba66~04a1      | fe11~6498      |
_version     | 6227~3ad0      | 7b57~780b      |
action       | include        | include        |
dmac         | ${iot_devices} | ["set",[]]     |
dmac_op      | in             | ["set",[]]     |
dst_ip       | ["set",[]]     | ["set",[]]     |
dst_ip_op    | ["set",[]]     | ["set",[]]     |
dst_port     | ["set",[]]     | ["set",[]]     |
dst_port_op  | ["set",[]]     | ["set",[]]     |
index        | 1              | 2              |
name         | ip_filter      | ip_filter      |
other_config | ["map",[]]     | ["map",[]]     |
pktcnt       | ["set",[]]     | ["set",[]]     |
pktcnt_op    | ["set",[]]     | ["set",[]]     |
proto        | ["set",[]]     | ["set",[]]     |
proto_op     | ["set",[]]     | ["set",[]]     |
smac         | ["set",[]]     | ${iot_devices} |
smac_op      | ["set",[]]     | in             |
src_ip       | ["set",[]]     | ["set",[]]     |
src_ip_op    | ["set",[]]     | ["set",[]]     |
src_port     | ["set",[]]     | ["set",[]]     |
src_port_op  | ["set",[]]     | ["set",[]]     |
vlanid       | ["set",[]]     | ["set",[]]     |
vlanid_op    | ["set",[]]     | ["set",[]]     |
-------------------------------------------------
```

# FCM_Report_Config

This table is used for configuring statistics reports from FCM to the Cloud.

| Name | Type | Description |
|---|---|---|
| name | string | Report config name |
| interval | integer | Reporting interval of collected flows to MQTT server in seconds |
| format | enum (*cumulative*, *delta*, *raw*) | Specifies format of reporting of flows. **cumulative** - Reports consolidated bytes/packets of flows from the start of flow at each report interval **delta** - Reports difference in bytes/packets of flows between each consecutive report interval. |

| | | |
|---|---|---|
| hist_interval | integer | Histogram interval of reports. Not used. |
| hist_filter | string | Name of FCM_Filter to be used for histogram reports. Not used. |
| report_filter | string | Name of FCM_Filter to be used when reporting |
| mqtt_topic | string | MQTT topic to be used for reporting |
| other_config | set of key:value | Not used. |

**Example:**

```
# ovsh s FCM_Report_Config
------------------------------------------------------------------------------
_uuid         | bfbe~c686                                                      |
_version      | 1689~4b6d                                                      |
format        | cumulative                                                     |
hist_filter   | ["set",[]]                                                     |
hist_interval | 0                                                              |
interval      | 60                                                             |
mqtt_topic    | IP/Flows/IoT/opensync/4C77701234/5c58ae5050d44e0b8df6aaaa      |
name          | ip_flow_report                                                 |
other_config  | ["map",[]]                                                     |
report_filter | ["set",[]]                                                     |
------------------------------------------------------------------------------
```

# IP_Interface

This table includes information for L3 configuration and status reporting. The IPv6 tables refer to or are referenced by this table.

| Name | Type | Description |
|---|---|---|
| name | string | Name of the IP interface |
| enable | boolean | Enable or disable the interface |
| status | enum | Current operational state of the interface.<br>Enumeration of:<br>• up<br>• down<br>• unknown<br>• dormant<br>• notpresent<br>• lowerlayerdown |

| | | |
|---|---|---|
| | | • error |
| interfaces | set of uuids | Reference to the table:Interface |
| if_name | string | Interface name as configured in the operating system |
| ipv4_addr | set of uuids | Reference to the table: IPv4_Address |
| ipv6_addr | set of uuids | Reference to the table: IPv6_Address |
| ipv6_prefix | set of uuids | Reference to the table: IPv6_Prefix |
| qos | set of uuids | Reference to the table: Interface_QoS |

# IPv4_Address (not used)

This table is used for IPv4 address configuration and reporting. Tables that directly require an IPv4 address (either for configuration or reporting) strongly reference the rows in this table.

| Name | Type | Description |
|---|---|---|
| enable | boolean | Enables or disables this IPv4 address |
| status | enum (*disabled*, *enabled*, *error*) | The status of this IPv4 address entry. Enumeration of:<br>● disabled<br>● enabled<br>● error |
| address | string | IPv4 address |
| subnet_mask | string | Subnet mask (CIDR notation) |
| type | enum | Addressing method used to assign the IPv4 address. Enumeration of:<br>● dhcp<br>ikev2<br>● auto_ip<br>● ipcp<br>● static |

# IPv6_Address

This table enables IPv6 address configuration and reporting. Tables that directly require an IPv6 address (either for configuration or reporting) refer to the rows in this table.

| Name | Type | Description |
|---|---|---|
| enable | bool | Enables or disables this IPv6 address |
| status | enum (*disabled*, *enabled*, *error*) | The status of this IPv6 address entry. Enumeration of:<br>● disabled<br>● enabled<br>● error |
| address_status | enum | The status of the IPv6 address, indicating whether it can be used for communication. Enumeration of:<br>● preferred<br>● deprecated |

| | | |
|---|---|---|
| | | • invalid<br>• inaccessible<br>• unknown<br>• tentative<br>• duplicate<br>• optimistic |
| address | string | IPv6 address |
| origin | enum | Mechanism using which the IPv6 address is assigned. Enumeration of:<br>• auto_configured<br>• dhcp<br>• ikev2<br>• map<br>• well_known<br>• static |
| prefix | string | Prefix of the IPv6 address (CIDR notation) |
| preferred_lifetime | string | The time at which this address will cease to be preferred (i.e. will become deprecated), or empty if not known. For an infinite lifetime, the parameter value MUST be infinite. |
| valid_lifetime | string | The time at which this address will cease to be valid (i.e. will become invalid), or empty if unknown. For an infinite lifetime, the parameter value MUST be infinite. |

# IPv6_Prefix

This table enables IPv6 prefix configuration and reporting. Tables that directly require an IPv6 prefix (either for configuration or reporting) refer to the rows in this table.

| Name | Type | Description |
|---|---|---|
| enable | boolean | Enables or disables this IPv6 prefix |
| status | enum (*disabled*, *enabled*, *error*) | The status of this IPv6 prefix entry. Enumeration of:<br>• disabled<br>• enabled<br>• error |
| prefix_status | enum | The status of the IPv6 prefix, indicating whether it can be used for communication. Enumeration of: |

| | | |
|---|---|---|
| | | • preferred<br>• deprecated<br>• invalid<br>• inaccessible<br>• unknown |
| address | string | IPv6 address prefix |
| origin | enum | Mechanism using which the IPv6 prefix was assigned or most recently updated. Enumeration of:<br>• auto_configured<br>• prefix_delegation<br>• ra<br>• well_known<br>• static<br>• child |
| static_type | enum | Static prefix sub-type. For a Static prefix, this can be set to PrefixDelegation or Child, thereby creating an unconfigured prefix of the specified type that will be populated with preference to creating a new instance. This allows the controller to pre-create the "prefix slots" with known path names that can be referred to from elsewhere in the data model before they are populated. Enumeration of:<br>• static<br>• inapplicable<br>• prefix_delegation<br>• child |
| parent_prefix | uuid | The value is a reference in the table:IPv6_Prefix. If the referenced object is deleted, the parameter value MUST be set to an empty string. Indicates the parent prefix from which this prefix was derived. The parent prefix is relevant only for Child prefixes and for Static Child prefixes (both of which will always be on downstream interfaces), i.e. for Origin=Child and for (Origin,StaticType) = (Static,Child) prefixes. |
| child_prefix_bits | string | A prefix that specifies the length of Static Child prefixes and how they are derived from their ParentPrefix. It will be used if, and only if, it is not an empty string and is longer than the parent prefix (if it is not used, derivation of such prefixes is implementation-specific). Any bits to the right of the parent prefix are set to the bits in this prefix. |

| | | |
|---|---|---|
| | | For example, for a parent prefix of fedc::/56, if this parameter had the value 123:4567:89ab:cdef::/64, the child /64 would be fedc:0:0:ef::/64. For a parent prefix of fedc::/60, the child /64 would be fedc:0:0:f::/64. |
| on_link | boolean | On-link flag [Section 4.6.2/RFC4861] as received (in the RA) for RouterAdvertisement. Indicates whether this prefix can be used for on-link determination. |
| autonomous | boolean | Autonomous address configuration flag [Section 4.6.2/RFC4861] as received (in the RA) for RouterAdvertisement. Indicates whether this prefix can be used for generating global addresses as specified by SLAAC [RFC4862]. |
| preferred_lifetime | string | This parameter is based on ipAddressPrefixAdvPreferredLifetime from [RFC4293]. The time at which this prefix will cease to be preferred (i.e. will become deprecated), or empty if not known. For an infinite lifetime, the parameter value MUST be infinite. |
| valid_lifetime | string | This parameter is based on ipAddressPrefixAdvValidLifetime from [RFC4293]. The time at which this prefix will cease to be valid (i.e. will become invalid), or empty if not known. For an infinite lifetime, the parameter value MUST be infinite. |

# DHCPv4_Client (not used)

This table enables DHCPv4 client configuration and status reporting.

| Name | Type | Description |
|---|---|---|
| enable | boolean | Enable or disable the DHCPv4 Client |
| ip_interface | uuid | Reference to the table:IP_interface |
| request_options[0..32] | set of integers | DHCPv4 options requested by the client |
| received_options[0..32] | set of uuids | Reference to table:DHCP_Option. DHCPv4 options are received from the server. |
| send_options[0..32] | set of uuids | Reference to table:DHCP_Option. DHCPv4 options sent to the server. |

# DHCPv6_Client

This table enables DHCPv6 client configuration and status reporting.

| Name | Type | Description |
|---|---|---|
| enable | boolean | Enable or disable the DHCPv6 Client |
| ip_interface | uuid | Reference to the table:IP_interface |
| request_address | boolean | Enables or disables inclusion of the Identity Association for Non-Temporary Address. |
| request_prefixes | boolean | Enables or disables inclusion of the Identity Association for Prefix Delegation. |
| rapid_commit | boolean | Enables or disables inclusion of the Rapid Commit. |
| renew | boolean | When set to true, the Client renews its DHCPv6-supplied information. |
| request_options | set of integers | DHCPv6 options requested by the client |
| received_options | set of uuids | Reference to table:DHCP_Option. DHCPv4 options received from the server. |
| send_options | set of uuids | Reference to table:DHCP_Option. DHCPv4 options sent to the server. |

# DHCP_Option

This table stores various DHCP options related to DHCPv4 and DHCPv6 configuration, and status reporting. IPv6 and IPv4 DHCP configuration tables refer to the rows in this table.

| Name | Type | Description |
|---|---|---|
| enable | bool | True if this option is active, or false if ignored |
| version | enum (*v4*, *v6*) | DHCP version for this option, used only to help read the table. It should be consistent with what is configured in the table:DHCPv4_Client or table:DHCPv6_Client. Enumeration of:<br>● v4<br>● v6 |
| type | enum (*rx*, *tx*) | Type of the DHCP option, used only to help read the table. It should be consistent with what is configured in the table:DHCPv4_Client or table:DHCPv6_Client. Enumeration of:<br>● rx<br>● tx |
| tag | integer | Option tag (code) |
| value | string | The DHCP Client option value. Base64-encoded. The maximum size of a 255-byte binary string encoded to base64 is 340. |

# Netfilter

The Netfilter table defines firewall (iptables) rules. Each row in the table maps to a single iptables rule. This table is managed by the Netfilter Manager (NFM).

| Name | Type | Description |
|---|---|---|
| enable | boolean | True whether the rule is enabled. |
| status | enum | Rule status, populated by NFM:<br>● "disabled" - Rule is disabled.<br>● "enabled" - Rule is enabled and was successfully configured.<br>● "error" - Rule is enabled, but there was an error while applying the rule configuration to the system. |
| protocol | string | Rule protocol family:<br>● "ipv4"<br>● "ipv6" |
| table | enum | Iptables table. This translates to the -t parameter of the iptables command.<br>● "filter",<br>● "nat",<br>● "mangle",<br>● "raw",<br>● "security |
| chain | string | Iptables chain name.<br>● "minLength": 1,<br>● "maxLength": 64 |
| priority | integer | Rule priority. Rules with lower priority will be applied first.<br>**Note:** This affects the rule number, but it is a 1:1 mapping to the rule numbers. |
| rule | string | Rule specification. The rule specification can reference tags in the Openflow_Tag table to create dynamically expanding rules.<br>● "minLength": 0,<br>● "maxLength": 512 |

| | | |
|---|---|---|
| target | string | The iptables target name. This roughly translates to the parameter of the -j command line switch.<br>● "minLength": 1,<br>● "maxLength": 64 |

# Netfilter_Ipset

Ipset is a companion application for the iptables Linux firewall. Ipset allows you to create rules to manage sets of IP addresses. The Netfilter_Ipset table stores various ipset options.

| Name | Type | Description |
|---|---|---|
| name | string | This value directly correlates with the name given to the ipset command. The name must be unique. |
| type | enum | The ipset command type:<br>["bitmap:ip", "bitmap:ip,mac", "bitmap:port", "hash:ip","hash:mac", "hash:ip,mac", "hash"net", "hash:net,net", "hash:ip,port", "hash:net,port", "hash:ip,port,ip", "hash:ip,port,net", "hash:ip,mark", "hash:net,port,net", "hash:net,iface", "list:set", "local"] |
| options | string | Ipset creation options. These are the options that are issued on top of the ipset command when creating an ipset. |
| values | key | Ipset entries. Max number of strings: 64. |
| origin | string | Rule origin. |
| status | string | Ipset command status:<br>● success: active<br>● error: error when configuring<br>● unset: not processed yet<br><br>**Note:** The status value is ephemeral and updated by the Net Filter Manager (NFM). |

# DHCPv4_Server (not used)

Used for DHCPv4 server configuration and status reporting.

| Name | Type | Description |
|---|---|---|
| interface | uuid | Reference to table:IP_Interface. Parent interface |
| status | enum (*disabled*, *enabled*, *error*) | Indicates the status of this entry. Enumeration of:<br>- disabled (origin: cloud)<br>- enabled (origin: cloud)<br>- error (origin: device) |
| min_address | string | Specifies the first IPv4 address in the pool to be assigned by the DHCP server on the interface. |
| max_address | string | Specifies the last IPv4 address in the pool to be assigned by the DHCP server on the interface. |
| lease_time | integer | Lease time in seconds. -1 indicates infinite. |
| options[0..256] | set of uuids | Reference to table:DHCP_Option. DHCP options offered to clients. |
| static_address[0..64] | set of uuids | Reference to table:DHCPv4_Lease. List of statically assigned IP addresses. |
| leased_address[0..256] | set of uuids | Reference to table:DHCPv4_Lease. List of leased IP addresses. |

# DHCPv4_Lease (not used)

Used for configuration and reporting of DHCPv4 leases. The DHCPv4_Server table refers to this table.

| Name | Type | Description |
|---|---|---|
| status | enum (*leased*, *static*, *error*) | Indicates the status of this entry. Enumeration of:<br>**leased** - This is a lease, reported by manager (origin: device)<br>**static** - Static lease entry (origin: cloud)<br>**error** - Error applying static entry (origin: device) |
| address | string | IPv4 Address of lease/static entry |
| hwaddr | string | Hardware address (MAC) of lease/static entry |
| hostname | string | Hostname |
| leased_time | integer | Lease time. Only applicable if status==leased. |
| leased_fingerprint | string | DHCP fingerprint. Only applicable if status==leased. |

# DHCPv6_Server

This table enables DHCPv6 server configuration and status reporting.

| Name | Type | Description |
|---|---|---|
| interface | uuid | Reference to IP_Interface |
| status | enum (*disabled*, *enabled*, *error*) | Indicate the status of this entry. Enumeration of:<br>- disabled (origin: cloud)<br>- enabled (origin: cloud)<br>- error (origin: device) |
| prefixes | set of uuids | Reference to table:IPv6_Prefix. List of prefixes that the DHCPv6 server will be offering (stateful DHCP). |
| prefix_delegation | boolean | If “True”, this Server row is used for prefix delegation |
| options | set of uuids | Reference to table:DHCP_Option. DHCP options offered to clients. |

| | | |
|---|---|---|
| lease_prefix | set of uuids | Reference to table:DHCPv6_Lease. List of leased IP prefixes. |
| static_prefix | set of uuids | Reference to table:DHCPv6_Lease. List of statically assigned IP prefixes/addresses. |

# DHCPv6_Lease

Used for configuration and reporting of DHCPv6 leases. The DHCPv6_Server table refers to this table.

| Name | Type | Description |
|---|---|---|
| status | enum (*leased*, *static*, *error*) | Indicates the status of this entry.<br>**leased** - This is a lease, reported by the manager (origin: device)<br>**static** - Static lease entry (origin: cloud)<br>**error** - Error applying static entry (origin: device) |
| prefix | string | IPv6 address of prefix (/XX notation) |
| duid | string | Client unique identifier (DUID) (2 bytes header + 128 bytes max data in hex notation without separators) |
| hwaddr | string | Hardware address (MAC) of lease/static entry in hex notation |
| hostname | string | Assigned/Requested hostname |
| leased_time | integer | Leased time |

# IPv6_RouteAdv

This table enables IPv6 Router Advertisement configuration and status reporting.

| Name | Type | Description |
|---|---|---|
| interface | set of uuids | Reference to table:IP_Interface. Parent interface. |
| status | enum (*disabled*, | Router advertisement status:<br>**disabled** - disable this entry (origin: cloud) |

| | | |
|---|---|---|
| | *enabled*, *error*) | **enabled** - enable Router Advertisement on interface (origin: cloud)<br>**error** - error configuring Router Advertisement (origin: device) |
| prefixes | set of uuids | Reference to the table:IPv6_Prefix. Prefixes advertised through RA. |
| managed | boolean | Managed address configuration -- the (M) flag |
| other_config | boolean | Other configuration -- the (O) flag |
| home_agent | boolean | Home Agent flag -- the (H) flag |
| max_adv_interval | integer | Maximum time allowed between sending unsolicited multicast Router Advertisements from the interface; in seconds. |
| min_adv_interval | integer | Minimum time allowed between sending unsolicited multicast Router Advertisements from the interface; in seconds.<br><br>The value must be greater than 3/4 * max_adv_interval. |
| default_lifetime | integer | Router lifetime. 0 indicates that the router should not be used as the default router. |
| preferred_router | enum (*low*, *medium*, *high*) | reference associated with the default router, as either "low", "medium", or "high". |
| mtu | integer | The MTU option is used in router advertisement messages to ensure that all nodes on a link use the same MTU value in those cases where the link MTU is not a well- known value. |
| reachable_time | integer | The time, in milliseconds, that a node assumes a neighbor is reachable after having received a reachability confirmation. Used by the Neighbor Unreachability Detection algorithm (see Section 7.3 of RFC 4861). A value of zero means unspecified (by this router). |
| retrans_timer | integer | The time, in milliseconds, between retransmitted Neighbor Solicitation messages. Used by address resolution and the Neighbor Unreachability Detection |

| | | |
|---|---|---|
| | | algorithm (see Sections 7.2 and 7.3 of RFC 4861). A value of zero means unspecified (by this router). |
| current_hop_limit | integer | The default value that should be placed in the Hop Count field of the IP header for outgoing (unicast) IP packets. The value should be set to the current diameter of the Internet. The value zero means unspecified (by this router). |
| rdnss | set of uuids | Reference to the table:IPv6_Address. Recursive DNS servers advertised via RA. |
| dnssl | set of strings | List of DNS search domains that will be advertised via RA |

# IPv6_Neighbors

This table enables IPv6 neighbor reporting. This is a tentative list of neighbors that are participating in an IPv6-enabled network.

| Name | Type | Description |
|---|---|---|
| address | string | Client device IPv6 address |
| hwaddr | string | Client device MAC address |
| if_name | string | Interface name |

**Example:**

```
$ ovsh -T s IPv6_Neighbors -w if_name==br-home.tndp
+----------+----------+---------------------------------------+-------------------+--------------+
| _uuid    | _version | address                               | hwaddr            | if_name      |
+----------+----------+---------------------------------------+-------------------+--------------+
| 3626~9727 | fd69~fc9b | fe80::cd1:6a3f:29a1:35f9              | 00:05:1B:D1:A5:7B | br-home.tndp |
| 9961~6ce5 | e5fa~7cd4 | fe80::225:90ff:fe87:175d              | 00:25:90:87:17:5D | br-home.tndp |
| 9895~3737 | 81dc~d033 | 2601:647:4900:1f63:115a:5cc2:f08f:7d68 | 26:16:29:D8:CF:99 | br-home.tndp |
| e754~0b49 | 5b84~decb | 2601:647:4900:1f63:cd32:d093:dd9e:dda1 | 26:16:29:D8:CF:99 | br-home.tndp |
+----------+----------+---------------------------------------+-------------------+--------------+
```

# IPv4_Neighbors

ARP protocol provides IPv4-to-MAC mappings to the devices. This table stores the ARP mappings exchange. Collecting the ARP messages exchanged on the LAN enhances the OpenSync IPv4-to-MAC mapping performance.

| Name | Type | Description |
|---|---|---|
| address | string | Client device IPv4 address. |
| hwaddr | string | Client device MAC address. |
| if_name | string | Incoming interface of the ARP packet which triggered the entry. |
| source | string | Source of ARP mapping - either FSM or the system ARP table. |

# IGMP_Config

This table sets various IGMP parameters in the system.

| Name | Type | Description |
|---|---|---|
| igmp_version | string | Default IGMP version selection. Available options:<br>• "IGMPv1"<br>• "IGMPv2"<br>• "IGMPv3" |
| snooping_enabled | boolean | Enable or disable snooping behavior on the specified bridge. |
| snooping_bridge | string | Bridge on which snooping should be used. |
| static_mrouter_port | string | Specifies the port to which IGMP reports should be sent. |
| mcast_group_exceptions | string | Multicast groups that should bypass snooping behavior. A comma-separated string of CIDR IP addresses. |
| unknown_mcast_group_behavior | string | Default forwarding behavior for multicast groups that are not stored in the snooping table. |

| | | |
|---|---|---|
| query_robustness_value | integer | Controls the IGMP query robustness variable. When the system receives an IGMP leave message on a shared network running IGMPv2, the query router must send an IGMP group query message for a specified number of times. The number of IGMP group query messages sent is determined by the robust count. |
| maximum_groups | integer | Maximum number of groups that can be learned by the system. |
| fast_leave_enable | boolean | If True, the option instructs to immediately drop the stream when an IGMP leave message is received. |
| proxy_enabled | boolean | Enables or disables IGMP Proxy behavior. |
| proxy_upstream_if | string | The interface on which IGMP reports are proxied, and where multicast stream is expected. |
| proxy_dowstream_if | string | The interface on which Proxy is listening for IGMP reports. |
| proxy_group_exceptions | string | This option instructs not to proxy the specified multicast groups. |
| proxy_allowed_subnets | string | Proxy only multicast groups for the specified source IP subnet. |
| querier_enabled | boolean | Enables or disables the IGMP Querier. |
| query_interval | integer | Interval duration between queries. |
| query_response_interval | integer | The interval to wait for a response after sending out a query. |
| last_member_query_interval | integer | The interval defines the maximum response time advertised in IGMP group-specific queries. |
| maximum_sources | integer | Maximum number of addresses allowed in the source filter list for a multicast group. |
| other_config | string | Platform-specific options represented as a JSON string. |

**Example:**

```
ovsh s IGMP_Config
---------------------------------------
_uuid                       | 3896~0ed6 |
_version                    | 1ef6~09ff |
fast_leave_enable           | false     |
last_member_query_interval  | 150       |
maximum_groups              | 150       |
maximum_members             | 10        |
maximum_sources             | 30        |
query_interval              | 125       |
query_response_interval     | 20        |
query_robustness_value      | 10        |
---------------------------------------
```

# MLD_Config

This table sets various MLD parameters in the system.

| Name | Type | Description |
|---|---|---|
| snooping_enabled | boolean | Enables or disables snooping behavior on the specified bridge. |
| snooping_bridge | string | The bridge on which snooping should be used. |
| static_mrouter_port | string | The port to which the IGMP reports should be sent explicitly – even if querier is not learned on that port. |
| mcast_group_exceptions | string | Multicast groups that should bypass snooping behavior. |
| unknown_mcast_group_behavior | string | Default forwarding behavior for the mcast groups that are not stored in the snooping table. |
| proxy_enabled | boolean | Enables or disables IGMP Proxy behavior. |
| proxy_upstream_if | string | Interface on which IGMP reports are proxied, and where multicast stream is expected. |
| proxy_downstream_if | string | Interface on which Proxy is listening for IGMP reports. |
| proxy_group_exceptions | string | This option instructs not to proxy the specified multicast groups. |
| proxy_allowed_subnets | string | This option instructs to only proxy the multicast groups for the specified source IP subnet. |

| | | |
|---|---|---|
| fast_leave_enable | boolean | This option instructs to immediately drop the stream when IGMP leave is received. |
| last_member_query_interval | integer | The interval defines the maximum response time advertised in the MLD group-specific queries. |
| maximum_groups | integer | Maximum number of multicast groups that can be learned. |
| maximum_members | integer | Maximum number of members in a group that can be supported. |
| maximum_sources | integer | Maximum number of addresses allowed in the source filter list for a multicast group. |
| querier_enabled | boolean | Enables or disables the MLD Querier. |
| query_interval | integer | The interval between General Queries sent by the Querier. |
| query_response_interval | integer | The interval to wait for a response after sending out a query. |
| query_robustness_value | integer | Controls the MLD query robustness variable. When the system receives an MLD leave message on a shared network running MLDv2, the query router must send an MLD group query message for a specified number of times. The number of MLD group query messages sent is determined by the robust count. |
| other_config | string | Platform-specific options. |

**Example:**

```
---------------------------------------
_uuid                       | fd46~7352 |
_version                    | 1d59~bee9 |
fast_leave_enable           | false     |
last_member_query_interval  | 160       |
maximum_groups              | 40        |
maximum_members             | 20        |
maximum_sources             | 30        |
query_interval              | 125       |
query_response_interval     | 20        |
query_robustness_value      | 4         |
---------------------------------------
```

# Node_Services

This table lists all *OpenSync* services (managers) that are available on the device, their configuration and current status. Services can be dynamically started or stopped by modifying the **enable** column of this table.

| Name | Type | Description |
|---|---|---|
| service | string | The name of the service name (executable name) |
| enable | bool | “True” if the service is enabled |
| status | enum | One of:<br>● enabled: If the service has been started<br>● disabled: If the service has been stopped<br>● error: If there was an error during starting/stopping of the service |
| other_config | map | Additional service configuration. This is the current list of values as interpreted by the DM:<br>● needs_plan_b (boolean): Must be stopped (“True”) whether a service crash requires restart of OpenSync<br>● restart_delay (integer): Restart delay in seconds<br>● always_restart (boolean): If true, services should always be restarted even if they are killed by the signals that usually do not trigger a restart. |

**Example:**

```
+----------+----------+--------+-------------------------------------------------------------+---------+---------+
| _uuid    | _version | enable | other_config                                                | service | status  |
+----------+----------+--------+-------------------------------------------------------------+---------+---------+
| 5afe~e959 | 3952~a6b8 | true   | ["map",[]]                                                  | bm      | enabled |
| 80d5~cec1 | 204e~399e | true   | ["map",[["needs_plan_b","true"]]]                           | cm      | enabled |
| 3582~7096 | e4a1~e3c3 | true   | ["map",[["always_restart","true"],["restart_delay","-1"]]] | fcm     | enabled |
| 432c~9d74 | 4bc4~f874 | true   | ["map",[["always_restart","true"],["restart_delay","-1"]]] | fsm     | enabled |
| e178~f0fc | 7bd4~3fa5 | true   | ["map",[["needs_plan_b","true"]]]                           | nm      | enabled |
| d617~4bb9 | bf2b~2e44 | true   | ["map",[["needs_plan_b","true"]]]                           | om      | enabled |
| a4b5~5096 | be53~296c | true   | ["map",[]]                                                  | pm      | enabled |
| de54~48da | 9b1e~e683 | true   | ["map",[]]                                                  | qm      | enabled |
| 4fbe~ba7e | 8c23~f270 | true   | ["map",[]]                                                  | sm      | enabled |
| 3f51~74e2 | d955~f38a | true   | ["map",[["needs_plan_b","true"]]]                           | wm      | enabled |
+----------+----------+--------+-------------------------------------------------------------+---------+---------
```

# OMS_Config

Normally, the device firmware upgrades require upgrading the entire firmware image. However, it is also possible to upgrade only specific firmware modules. Examples of such upgrades are signatures for third party components, or adding new modules or managers. OMS_Config is the table that the receiving modules are monitoring for the object updates. This table lists the available versions for a given object name. In case of multiple versions for the same object name, it is up to the receiving module to decide which one to use.

| Name | Type | Description |
|---|---|---|
| object_name | string | Name of the object to be upgraded |
| version | string | Version of the object (set by the Cloud) |
| other_config | string | |

# Object_Store_Config

The Object_Store_Config table enables the Cloud-to-device communication when upgrading only specific firmware modules.

| Name | Type | Description |
|---|---|---|
| dl_url | string | Download url:<br>● Set by the Cloud when a new object is available<br>● Cleared by device when download is done (even if download fails) |
| dl_timeout | integer | Download timeout:<br>● Set by the Cloud for download timeout<br>● Cleared by device when download is done (even if download fails) |
| name | string | Name of the object:<br>● Set by the Cloud when a new download is triggered<br>● Set by the device at boot (from the already installed objects) |
| version | string | Version of the object:<br>● Set by the Cloud when a new download is triggered<br>● Set by device at boot (from already installed objects) |

# Object_Store_State

This table enables the device-to-Cloud communication when upgrading only specific firmware modules.

| Name | Type | Description |
|---|---|---|
| fw_integrated | boolean | Object pre integrated in the firmware image:<br>● True if the object is pre integrated in the image - it is installed by default.<br>● False if the object is installed from the Cloud (Object_Store_Config table).<br><br>The pre- integrated objects can not be removed. |
| status | enum | Status of object:<br>● always set only by the device<br>● enum:<br>○ ACTIVE<br>○ ERROR<br>○ DOWNLOAD_STARTED<br>○ DOWNLOAD_COMPLETED<br>○ DOWNALOD_ERROR<br>○ INSTAL_FAILED<br>○ INSTAL_COMPLETED<br>○ VER_MISMATCH |
| name | string | Name of the object:<br>● Set by the device |
| version | string | Version of object:<br>● Set by the device at boot |

# Captive_Portal

This table configures Captive Portal authentication parameters on the devices.

| Name | Type | Description |
|---|---|---|
| name | string | This column is the name of the Captive_Portal config table entry. |
| proxy_method | string | This column is for configuring the proxy either in “reverse” or “forward” proxy method. |
| uam_url | string | URL of the NAS server that will be accessed using the device and location id |
| additional_headers | string | Map of key-value pairs for any tinyproxy config |
| other_config | string | |

# Interface_QoS

This table defines the QoS schema.

| Name | Type | Description |
|---|---|---|
| queues | set of uuids | Reference table to Interface_Queue. |
| other_config | map of strings | Key/value pairs representing auxiliary and/or QoS implementation specific configuration. |
| status | enum | QoS status as reported by the device:<br>● “success”<br>● “error”<br>● “ephemeral” |

# Interface_Queue

| Name | Type | Description |
|---|---|---|
| priority | integer | The queue priority (lower value indicates higher priority). |
| bandwidth | integer | The queue maximum bandwidth expressed in Kbit/s. |
| tag | string | Name of the entry that will be populated in the Openflow_Tag table:<br>● The `tag` entry will be populated with the firewall mark value.<br>● The `tag_class` entry will be populated with the queue class value.<br>This can be used to create dynamic Netfilter rules that expand according to the `tag` value. |
| other_config | map of strings | Key/value pairs representing auxiliary and/or Queue implementation specific configuration |
| mark | integer | The firewall (iptables) mark that shall be used to categorize packets for this queue. |

# Reboot_Status

This table enables reporting of device reboot/boot reason. Entries in this table are automatically populated by the device upon system startup. Table rows are persistent until deleted.

| Name | Type | Description |
|---|---|---|
| count | int | This column indicates the reboot sequence number. Each time the system boots, this counter increases by 1. |
| reason | string | Optional: Information provided in addition to the type. This string may include, for example, a snippet from the kernel stack trace upon crash, reason why the health-check failed, etc. |
| type | enum | This field indicates the reboot type and is an enum of:<br>● UNKNOWN - Unable to determine reboot reason<br>● COLD_BOOT - First boot after device power on<br>● POWER_CYCLE - Device was power cycled<br>● WATCHDOG - Reboot due to watchdog timeout<br>● CRASH - Reboot due to system crash<br>● USER - Reboot issued by user (reboot command)<br>● DEVICE - Firmware initiated reboot<br>● HEALTH_CHECK - Reboot due to health check failure<br>● UPGRADE - Reboot due to system upgrade<br>● THERMAL - Reboot due to critical thermal event<br>● CLOUD - Cloud-initiated reboot (for example, reboot from NOC) |

**Example:**

```
|+-----------+-----------+-------+-------------------------+-------------+
| _uuid      | _version  | count | reason                  | type        |
|+-----------+-----------+-------+-------------------------+-------------+
|  a66d~d8f5 | 0277~bbc7 | 23    | delayed-reboot          | CLOUD       |
|  1455~5c54 | 1189~6df6 | 74    |                         | COLD_BOOT   |
|  8a66~c0aa | 118c~1c84 | 69    |                         | COLD_BOOT   |
|  9fe1~4367 | 11c3~ae43 | 61    | delayed-reboot          | CLOUD       |
|  dbdf~9a35 | 1a79~2b54 | 40    | delayed-reboot          | CLOUD       |
+  0676~345e | 21d5~831f | 43    | delayed-reboot          | CLOUD       |
|  3ac8~f9c2 | 26fe~d218 | 32    | delayed-reboot          | CLOUD       |
|  ef64~976b | 2e36~3c13 | 24    | delayed-reboot          | CLOUD       |
|  3ece~6f2a | 348d~6a46 | 50    | delayed-reboot          | CLOUD       |
|  37cd~5fba | 374f~3e3c | 84    | (unknown)               | POWER_CYCLE |
|  0569~53c5 | 3887~3617 | 67    | delayed-reboot          | CLOUD       |
|  dac1~7cfb | 3bf6~cd03 | 54    | delayed-reboot          | CLOUD       |
|+-----------+-----------+-------+-------------------------+-------------+
```

# Service_Announcement

This table contains service name, metadata, and interval for mDNS service advertising.

| Name | Type | Description |
|---|---|---|
| name | string | Name of the service being announced. |
| protocol | string | This is the type, i.e http or ftp. For example: _http._tcp or _smb._udp |
| port | integer | Port number of the service, for example: 80 or 443. |
| txt | string | Text records of the service. This could be any arbitrary text with the format “type=value” **Example**: "v1","https://bd.opensync.com/api/v1/" |

**Example:**

```
-------------------------------------------------------------
_uuid    | aa8d~f70f                                        |
_version | 1d4c~787e                                        |
name     | bw.opensync                                        |
port     | 3000                                             |
protocol | _smb._udp                                        |
txt      | ["map",[["v1","https://bd.opensync.com/api/v1/"]]] |
-------------------------------------------------------------
```

# WAN_Config

With the introduction of persistent OVSDB rows (OpenSync 3.2), the WAN settings were moved from persistent storage to OVSDB. The WAN_Config table includes the new WAN settings.

| Name | Type | Description |
|---|---|---|
| enable | boolean | True if the configuration in this row is active. |
| type | enum | WAN configuration type. Available options:<br>• dhcp<br>• "pppoe"<br>• "vlan"<br>• "static_ipv4" |
| priority | integer | WANO always uses the row with the highest priority (according to its type). If multiple entries exist, the entries with lower priorities are ignored. The value can be a timestamp. |
| other_config | map of strings | WAN type-specific configuration. The field depends on the "type" parameter. The supported parameters are defined in the table other_config supported parameters below. |
| os_persist | boolean | True if the row is set as persistent data.[1] |
| status | enum | Enum of "success" or "error". Unset if WAN provisioning with the current settings is still in progress or is not being used. Value "success" if WANO was able to use the settings in this row to establish a WAN connection. Value "error" if provisioning was unsuccessful. |

## other_config supported parameters

| Type | Setting | Description |
|---|---|---|
| pppoe | username | PPPoE username |

[1] The os_persist boolean column is a generic column that can be defined in any OVSDB table. When detecting a change in the OVSDB data, and if the os_persist column value is set to true, Persistent Storage Manager (PSM) serializes the data and adds the data to persistent storage.

| | | |
|---|---|---|
| pppoe | password | PPPoE password |
| vlan | vlan_id | The VLAN ID |
| static_ipv4 | gateway | The default IPv4 gateway |
| static_ipv4 | ip | The static IP address of the device |
| static_ipv4 | subnet | The static IP subnet of the device |
| static_ipv4 | primary_dns | The primary DNS IP address |
| static_ipv4 | secondary_dns (optional) | The secondary DNS IP address |

# TELOG_Config

This table enables configuration of the time-event logging system. The time-event logging system generates and transports the time-event reports from target to the specified MQTT topic destination.

| Name | Type | Description |
|---|---|---|
| mqtt_topic | string | MQTT topic for the transmitted reports. Reports won't be generated when this field is cleared. |
| mqtt_qos | integer | MQTT QoS for the transmitted reports. The recommended value is 1 (deliver at least once). |
| report_interval | integer | Time-event reports generation interval in the 1–3600 s range. Recommended value: 60 sec. |
| log_severity | string | Controls local time-event logs verbosity in the syslog. Optional field to be used for development only. |

# Passpoint_Config

The Passpoint_Config table stores the "Home as a Hotspot - Public Wifi" network parameters.

| Name | Type | Description |
|---|---|---|
| enabled | boolean | Enables or disables Hotspot |

| | | |
|---|---|---|
| hessid | string | The Homogenous ESSID (HESSID) is an MAC address field that is the same for all APs belonging to the same network. |
| list_3gpp | list of strings | List of 3GPP Cellular Network ANQP-elements. Contains the cellular network identity based on public land mobile network (PLMN) information. |
| roaming_consortium | list of strings | The Roaming Consortium ANQP-element and beacon elements provide a list of identifiers of roaming consortiums and SPs that are roaming partners of the Passpoint hotspot service provider and that are accessible from the Passpoint AP. OI values assigned by the IEEE1 identify the roaming consortium (i.e., a group of SPs with an inter-SP roaming agreements) or a single SP. |
| domain_name | string | Domain name of the entity operating the hotspot network. |
| nairealm_list | list of strings | A list of NAI realms corresponding to the Home SPs that can authenticate a mobile device with username/password or certificate credentials. |
| adv_wan_status | boolean | Advertise WAN link status. |
| adv_wan_symmetric | boolean | Advertise WAN link symmetry (whether link speed is the same in the uplink and downlink). |
| adv_wan_at_capacity | boolean | In this condition the AP won't allow additional mobile devices to associate. |