API Reference Document

M4300 REST API

API Version: 2.0.0.59

M4300 REST API with ConfigAgent Documentation.

INDEX

Security and Authentication

SECURITY SCHEMES

KEY	TYPE	DESCRIPTION
bearerAuth	http, bearer	

API
1. AUTHENTICATION

1.1 POST /login

Logging into device

REQUEST

REQUEST BODY - application/json

```
{
  login {
    username* string  Admin username
    password* string  Admin user's password
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  login {
    token    string   Auth token string used to for subsequent requests at `Authorization` header.
    expires  integer  Duration for token to expire in seconds
  }
}
```

1.2 POST /logout

Logging into device

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
```

```
}
```

2. DEVICE SETTINGS

2.1 GET /device_info

Get the device information

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  device_info {
    name              string   Switch display name
    serialNumber      string   Switch Serial Number
    macAddr           string   Switch MAC Address
    model             string   Switch Model Number
    lanIpAddress      string   LAN IP Address
    swVer             string   Active firmware version
    lastReboot        string   Time of last reboot with time zone information
    numOfPorts        integer  Total number of switch ports available
    numOfActivePorts  integer  Total number of currently active switch ports
    rstpState         boolean  RSTP State
    memoryUsed        string   Amount of RAM used in KBs
    memoryUsage       string   % of memory usage
    cpuUsage          string   % of CPU usage
    fanState          string   Fan status
    poeState          boolean  PoE enabled status
    upTime            string   Up time of device
    temperatureSensors [{
      Array of object:
        sensorNum    integer  Temperature sensor SKU
        sensorDesc   integer  Description of the temperature sensor
        sensorTemp   string   Temperature sensor temperature in Celcius
        sensorState  enum     ALLOWED:0, 1, 2, 3, 4, 5, 6
                              Temperature sensor state:
                              * `0` = NONE
                              * `1` = NORMAL
                              * `2` = WARNING
                              * `3` = CRITICAL
                              * `4` = SHUTDOWN
                              * `5` = NOT PRESENT
                              * `6` = NOT OPERATIONAL
    }]
    bootVersion       string   Bootcode version of the Switch.
    rxData            integer  Total number of bytes received
    txData            integer  Total number of bytes transmitted
    adminPoePower     integer  Admin PoE power as selected from Web UI (unit is mW)
```

```
    }
  }
```

2.2 GET /device_name

Get the device name

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  deviceName {
    name*     string
    location  string
  }
}
```

2.3 POST /device_name

Set the device name

REQUEST

REQUEST BODY - application/json

```
{
  deviceName {
    name*     string
    location  string
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

2.4 POST /device_reboot

Reboot switch

REQUEST

REQUEST BODY - application/json

```
{
  deviceReboot {
    afterSecs* integer between 0 and 10
                       DEFAULT:2
                       Delay reboot for t seconds
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status   enum    ALLOWED:success, failure
    respCode integer
    respMsg  string
  }
}
```

2.5 GET /bonjour

Get the device bonjour status

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  bonjour {
    status* string DEFAULT:enabled
                   Set the bonjour status
  }
}
```

2.6 POST /bonjour

Set the device bonjour status

REQUEST

REQUEST BODY - application/json

```
{
  bonjour {
    status* string DEFAULT:enabled
                   Set the bonjour status
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

2.7 GET /lldp_remote_devices

Get remote device lldp information

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  lldp_remote_devices {
    id                      integer  LLDP ID Number
    ifIndex                 integer  Internal interface number
    remoteId                integer  Identifier for device on remote system
    chassisId               string   Remote device hardware platform
    chassisIdSubtype        enum     ALLOWED:1, 2, 3, 4, 5, 6, 7
                                     Chassis ID field subtype:
                                     * `1` = Chassis component
                                     * `2` = Interface alias
                                     * `3` = Port component
                                     * `4` = MAC address
                                     * `5` = Network address
                                     * `6` = Interface name
                                     * `7` = Local
    remotePortId            string   Device port that transmitted LLDP data
    remotePortIdSubtype     enum     ALLOWED:1, 2, 3, 4, 5, 6, 7
                                     Remote port field subtype:
                                     * `1` = Chassis component
                                     * `2` = Interface alias
                                     * `3` = Port component
                                     * `4` = MAC address
                                     * `5` = Network address
                                     * `6` = Interface name
                                     * `7` = Local
    remotePortDesc          string   Remote system port description
    remoteSysName           string   Name assigned to the device in the remote system
    remoteSysDesc           string   Description assigned to the device in the remote system
```

```
      sysCapabilitiesSupported  string   List of primary functions supported by the remote system
      sysCapabilitiesEnabled    string   List of primary functions enabled on the remote system
      mgmtAddresses [{
      Array of object:
        type     string  List of types
        address  string  List of IP addresses
      }]
      remoteTTL                 integer  Remote device Time To Live information offset
    }
  }
```

2.8 GET /dual_image_status

Get the device flash image status

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  dualImageStatus {
    image1Label        string  DEFAULT:image1
                               image1 label
    image1Descr        string  image1 description
    image1Version      string  image1 version
    image2Label        string  DEFAULT:image2
                               image2 label
    image2Descr        string  image2 description
    image2Version      string  image2 version
    activatedImgLabel  string  active image label
  }
}
```

2.9 GET /active_image

Get device activate flash image

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  active_image {
    label      string  Active image label
    imageDescr string  Active image description
  }
}
```

2.10 POST /active_image

Set device active flash image

REQUEST

REQUEST BODY - application/json

```
{
  active_image {
    label* enum  ALLOWED:image1, image2
                 Set the activated image:
                 * `image1`
                 * `image2`
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

2.11 POST /config_copy

Copy configuration within switch

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*directive	enum ALLOWED: rtos, stob, btos, rtof		

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

2.12 GET /config_file_compare

Get configuration comparison

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*directive	enum ALLOWED: rtos, stob		

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  config_file_compare {
    diff [string] Diff of configurations compared
  }
}
```

2.13 GET /system_config

Get device console and telnet settings

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
```

```
    respCode integer
    respMsg  string
  }
  system_config {
    sysAccessLine             enum     ALLOWED:console, telnet, ssh
                                       System access type setting
    sysLineTerminalLen        integer  Terminal length of an access line.
    sysSerialTimeOut          integer  Serial timeout.
    sysTelnetServerAdminMode  enum     ALLOWED:enabled, disabled
                                       Telnet server admin mode.
  }
}
```

2.14 POST /system_config

Set device console and telnet settings

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*access_line	enum ALLOWED: console, telnet, ssh		

REQUEST BODY - application/json

```
{
  system_config {
    sysLineTerminalDefaultLen*  integer  System access type setting
    sysLineTerminalLen*         integer  Set terminal length of an access line to default.
    sysSerialTimeOutDefault*    integer  Set serial timeout to default value.
    sysSerialTimeOut*           integer  Serial timeout.
    sysTelnetServerAdminMode*   enum     ALLOWED:enabled, disabled
                                         Telnet server admin mode
    sysTransferBytesCompleted*  integer  Bytes transferred for the file.
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status   enum     ALLOWED:success, failure
    respCode integer
    respMsg  string
  }
}
```

2.15 GET /system_rfc1213

Get device name, description, location and contact

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  system_rfc1213 {
    sysDescr     string  Description of the system
    sysName      string  Name of the system
    sysLocation  string  Physical location of the system
    sysContact   string  System administrator contact information
  }
}
```

2.16 POST /system_rfc1213

Set device name, location and contact

REQUEST

REQUEST BODY - application/json

```
{
  system_rfc1213 {
    sysName*     string  max:255 chars
                         Name of the system
    sysLocation* string  max:255 chars
                         Location of the system
    sysContact*  string  max:255 chars
                         Contact of the system
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

3. DIAGNOSTICS

3.1 POST /ping_test_start

Ping test

REQUEST

REQUEST BODY - application/json

```
{
  pingTest {
    action*    enum     ALLOWED:1, 0
                        Action to start ping:
                        * `1` = Start
                        * `0` = Stop
    ipVersion  enum     DEFAULT:4
                        ALLOWED:4, 6
                        IP address version:
                        * `4` = IPv4
                        * `6` = IPv6
    host*      string   Hostname/IP address
    count      integer  between 1 and  1024
                        DEFAULT:3
                        Number of echo requests sent
    size       integer  between 1 and  655535
                        DEFAULT:64
                        Size of ping packet
    timeout    integer  between 1 and  300
                        DEFAULT:60
                        Time out value in seconds
    interval   string   Interval between ping packets in seconds
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

3.2 GET /ping_test_status

Get ping test status

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  pingTestStatus {
    state     enum     ALLOWED:0, 1, 2
                       Ping Test State:
                       * `0` = PT_SUCCESS
                       * `1` = PT_IN_PROGRESS
                       * `2` = PT_FAILURE
    pingMsg   string   Response for Ping message.
  }
}
```

3.3 POST /traceroute_start

Traceroute start and stop

REQUEST

REQUEST BODY - application/json

```
{
  tracerouteStart {
    action*     enum     ALLOWED:1, 0
                         Traceroute action:
                         * `1` = Start
                         * `0` = Stop
    host*       string   DEFAULT:www.netgear.com
                         Traceroute host or IP
    size        integer  between 38 and  32768
                         DEFAULT:38
                         Size of probe packets
    ipVersion   enum     DEFAULT:4
                         ALLOWED:4, 6
                         IP address version:
                         * `4` = IPv4
                         * `6` = IPv6
    initTTL     integer  between 1 and  255
                         DEFAULT:1
                         Initial Time To Live to be used
    maxTTL      integer  between 1 and  255
                         DEFAULT:30
                         Maximum Time To Live for the destination
    port        integer  between 1 and  65535
                         DEFAULT:33434
                         UDP destination port for probe packets
    nQueries    integer  between 1 and  10
                         DEFAULT:3
                         Number of probes per hop
    wait        integer  between 1 and  60
                         DEFAULT:3
                         Time between probes in seconds
```

```
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

3.4 GET /traceroute_status

Traceroute results

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  tracerouteInfo {
    state          enum    ALLOWED:0, 1, 2
                           Traceroute Test State:
                           * `0` = PT_SUCCESS
                           * `1` = PT_IN_PROGRESS
                           * `2` = PT_FAILURE
    tracerouteMsg  string  Response message for traceroute
  }
}
```

3.5 GET /sw_portmirroring

Get Port Mirroring Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*sessionNum	integer	1	Port mirroring session number

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  switchRstpPortConfig {
    sessionNum*   integer  between 1 and  4
                           Port mirroring session number
    sessionMode*  boolean  Port mirroring admin mode configuration
    destPort*     integer  Destination or probe port ID. No ports selected when set to `0`.
    srcPort [{
    Array of object:
      intfType*   enum     ALLOWED:0, 1, 2, 3, 4, 5, 6, 7, 8
                           Source port capture type:
                           * `0` = INTF_TYPE_PHY
                           * `1` = INTF_TYPE_CPU
                           * `2` = INTF_TYPE_LAG
                           * `3` = INTF_TYPE_VLAN
                           * `4` = INTF_TYPE_LOOPBACK
                           * `5` = INTF_TYPE_TUNNEL
                           * `6` = INTF_TYPE_SERVICE_PORT
                           * `7` = INTF_TYPE_OTHER
                           * `8` = INTF_TYPE_ANY
      intfNum*    integer  Source port interface number
      direction*  enum     ALLOWED:1, 2, 3, 4
                           Source port capture direction:
                           * `1` = BIDIRECTIONAL
                           * `2` = INGRESS
                           * `3` = EGRESS
                           * `4` = SFLOW
    }]
  }
}
```

3.6 POST /sw_portmirroring

Set Port Mirroring Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*sessionNum	integer	1	Port mirroring session number

REQUEST BODY - application/json

```
{
  switchRstpPortConfig {
    sessionNum*   integer  between 1 and  4
                           Port mirroring session number
    sessionMode*  boolean  Port mirroring admin mode configuration
    destPort*     integer  Destination or probe port ID. No ports selected when set to `0`.
    srcPort [{
```

Array of object:

```
      intfType*   enum      ALLOWED:0, 1, 2, 3, 4, 5, 6, 7, 8
                            Source port capture type:
                            * `0` = INTF_TYPE_PHY
                            * `1` = INTF_TYPE_CPU
                            * `2` = INTF_TYPE_LAG
                            * `3` = INTF_TYPE_VLAN
                            * `4` = INTF_TYPE_LOOPBACK
                            * `5` = INTF_TYPE_TUNNEL
                            * `6` = INTF_TYPE_SERVICE_PORT
                            * `7` = INTF_TYPE_OTHER
                            * `8` = INTF_TYPE_ANY
      intfNum*    integer   Source port interface number
      direction*  enum      ALLOWED:1, 2, 3, 4
                            Source port capture direction:
                            * `1` = BIDIRECTIONAL
                            * `2` = INGRESS
                            * `3` = EGRESS
                            * `4` = SFLOW
    }]
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

3.7 DELETE /sw_portmirroring

Delete Port Mirroring Configuration Session

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*sessionNum	integer	1	Port mirring session number

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

3.8 GET /device_cable_test

Get device cable test results

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	integer	1	Port ID

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  cableTestStatus {
    status          enum     ALLOWED:0, 1, 2, 3, 4, 5, 6, 7
                             Status of the cable test:
                             * `0` = Untested
                             * `1` = Fail
                             * `2` = Normal
                             * `3` = Open
                             * `4` = Short
                             * `5` = Open Short
                             * `6` = Cross Talk
                             * `7` = No Cable
    lenKnown        integer  Length of Cable in meters. (0 if not known)
    shortestLen     integer  Cable length range shorter limit in meters.
    longestLen      integer  Cable length range longer limit in meters.
    cableFailureLen integer  Distance along cable to detected fault.
  }
}
```

4. LINK AGGREGRATION GROUP SETTINGS

4.1 GET /sw_lag_cfg

Get Link Aggregration Group settings

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*lag_group	undefined	1	LAG Group ID# or `ALL`

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  switchConfigLagGroup {
    lag_group   integer    between 0 and  128
                           LAG Group:
                           * `0` = Create a new LAG group
                           * Non-zero values will modify an existing LAB group.
    name*       string     LAG description
    groupId*    integer    LAG Group ID
    adminMode*  boolean    LAG enabled state
    type*       enum       ALLOWED:0, 1
                           LAG Type:
                           * `0` = Dynamic or Static LAG
                           * `1` = Static LAG
    members*    [integer]
  }
}
```

4.2 POST /sw_lag_cfg

Set Link Aggregration Group settings

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*lag_group	integer	1	LAG Group ID

REQUEST BODY - application/json

```
{
```

```
  switchConfigLagGroup {
    lag_group   integer    between 0 and  128
                           LAG Group:
                           * `0` = Create a new LAG group
                           * Non-zero values will modify an existing LAB group.
    name*       string     LAG description
    groupId*    integer    LAG Group ID
    adminMode*  boolean    LAG enabled state
    type*       enum       ALLOWED:0, 1
                           LAG Type:
                           * `0` = Dynamic or Static LAG
                           * `1` = Static LAG
    members*    [integer]
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

5. LOGGING

5.1 GET /device_log_reader

Get device log reader

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*num_logs	integer	2	Number of logs pulled

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  logReader {
    logs [string]
  }
}
```

6. MULTICAST

6.1 GET /snooping_vlan

Get Snooping VLAN Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*family	enum ALLOWED: mdl, igmp	igmp	Snooping family

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  snooping_vlans [{
  Array of object:
    vlanId                   integer  VLAN ID
    family                   enum     ALLOWED:mld, igmp
                                      Snooping family
    fastLeaveMode            string   Snooping fast leave mode for the specified VLAN
    vlanMode                 string   Snooping mode for the specified VLAN
    reportSuppMode           string   Snooping report suppression mode for the specified VLAN
    proxyQuerierMode         string   Proxy Querier Admin mode for the specified VLAN.
    groupMembershipInterval  integer  IGMP/MLD snooping group membership interval for the specified VLAN.
    maxResponseTime          integer  IGMP/MLD snooping maximum response time for the specified VLAN.
  }]
}
```

6.2 GET /snooping_config

Get Snooping Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*family	enum ALLOWED: mdl, igmp	igmp	Snooping family

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  snooping_config {
    family                 enum     ALLOWED:mdl, igmp
                                    Snooping family
    adminMode              enum     ALLOWED:enabled, disabled
                                    IGMP/MLD Admin mode
    proxyQuerierAdminMode  enum     ALLOWED:enabled, disabled
                                    IGMP/MLD Proxy Querier Admin Mode
    floodAllUnknownPort    enum     ALLOWED:enabled, disabled
                                    Flood unknown multicast traffic to all ports.
    controlFrames          integer  Number of multicast control frames processed by the CPU.
    forwardedFrames        integer  Number of multicast data frames forwarded by the CPU.
  }
}
```

6.3 POST /snooping_config

Set Snooping Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*family	enum ALLOWED: mdl, igmp	igmp	Snooping family

REQUEST BODY - application/json

```
{
  snooping_config {
    family*                enum  ALLOWED:mdl, igmp
                                 Snooping family
    adminMode*             enum  ALLOWED:enabled, disabled
                                 IGMP/MLD Admin mode
    proxyQuerierAdminMode* enum  ALLOWED:enabled, disabled
                                 IGMP/MLD Proxy Querier Admin Mode
    floodAllUnknownPort*   enum  ALLOWED:enabled, disabled
                                 Flood unknown multicast traffic to all ports.
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

6.4 GET /snooping_interfaces

Get Snooping Interface Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*family	enum ALLOWED: mdl, igmp	igmp	Snooping family

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  snooping_interfaces [{
  Array of object:
    interface                integer  Port interface ID
    fastLeaveAdminMode       string   IGMP/MLD Snooping Fast leave admin mode
    groupMembershipInterval  integer  IGMP/MLD group membership interval
    intfMode                 string   Snooping mode
    proxyQuerierMode         string   Proxy Querier Admin mode for the specified interface
    responseTime             integer  Query response time for the specified interface
    family                   enum     ALLOWED:mdl, igmp
                                      Snooping family
  }]
}
```

6.5 POST /snooping_interfaces

Set Snooping Interface Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	integer	1	Port interface ID

REQUEST BODY - application/json

```
{
  snooping_interfaces {
    interface*                integer  Port interface ID
    family*                   enum     ALLOWED:mdl, igmp
                                       Snooping family
    fastLeaveAdminMode*       string   IGMP/MLD Snooping Fast leave admin mode
    groupMembershipInterval*  integer  IGMP/MLD group membership interval
    intfMode*                 string   Snooping mode
    proxyQuerierMode*         string   Proxy Querier Admin mode for the specified interface
    responseTime*             integer  Query response time for the specified interface
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

6.6 GET /snooping_queriers

Get Snooping Querier Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*family	enum ALLOWED: mdl, igmp	igmp	Snooping family
*vlanid	integer	1	VLAN ID

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  snooping_queriers {
    address              string   Configured IP address of querier
    adminMode            enum     ALLOWED:enabled, disabled
                                  Querier enabled status
```

```
      expiryInterval      integer  between 60 and  300
                                   Expiry interval of a snoop instance in seconds
      lastQuerierAddress  string   Last IP address detected for querier
      lastQuerierVersion  integer  Last version detected for querier
      operMaxRespTime     integer  Operational value of max response time in seconds
      operState           string   Operational state of querier
      operVersion         integer  Operational version of querier
      queryInterval       integer  between 1 and  1800
                                   Snooping query interval in seconds
      querierVersion      integer  between 1 and  2
                                   Configured version for querier
      vlanAddress         string   IP address configured for the querier.
      vlanElectionMode    enum     ALLOWED:enabled, disabled
                                   Configured snooping querier election mode for the VLAN ID
      vlanMode            enum     ALLOWED:enabled, disabled
                                   Configured snooping querier mode for the VLAN ID
    }
  }
```

6.7 POST /snooping_queriers

Set Snooping Querier Configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*family	enum ALLOWED: mdl, igmp	igmp	Snooping family
*vlanid	integer	1	VLAN ID

REQUEST BODY - application/json

```
{
  snooping_queriers {
    address*         string   Configured IP address of querier
    adminMode*       enum     ALLOWED:enabled, disabled
                              Enable or disable querier
    expiryInterval*  integer  between 60 and  300
                              Expiry interval of a snoop instance in seconds
    queryInterval*   integer  between 1 and  1800
                              Snooping query interval in seconds
    querierVersion*  integer  between 1 and  2
                              Configured version for the querier
    vlanAddress*     string   IP address configured for the querier
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
```

```
    respCode integer
    respMsg  string
  }
}
```

---

7. PORT INFORMATION AND SETTINGS

7.1 GET /swcfg_port

Get port configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	integer	1	Port ID Number

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  switchPortConfig {
    ID*          integer  Port Number
    description* string   Port description label
    portType*    enum     DEFAULT:1
                          ALLOWED:0, 1, 2, 3, 4, 5
                          Port configuration type:
                          * `0` = MODE_NONE
                          * `1` = MODE_GENERAL
                          * `2` = MODE_ACCESS
                          * `3` = MODE_TRUNK
                          * `4` = MODE_PRIVATE_HOST
                          * `5` = MODE_PRIVATE_PROMISC
    adminMode*   boolean  DEFAULT:true
                          Enable physical port interface
    portSpeed    enum     DEFAULT:0
                          ALLOWED:0, 1, 2, 3, 4
                          Port link speed:
                          * `0` = auto
                          * `1` = SP10
                          * `2` = SP100
                          * `3` = SP1000
                          * `4` = SP10G
    duplexMode   enum     DEFAULT:2
                          ALLOWED:2, 1, 0
                          Duplex Mode:
                          * `2` = auto
                          * `1` = full
                          * `0` = half
    linkStatus   enum     DEFAULT:1
                          ALLOWED:0, 1
                          Link up or down Status (read-only):
                          * `0` = Up
                          * `1` = Down
    linkTrap     boolean  DEFAULT:true
                          Enable link trap
    maxFrameSize integer  between 1500 and  9198
                          DEFAULT:1518
                          Max frame size
    isPoE*       boolean  DEFAULT:true
```

```
                                Port is PoE capable
    txRate*       integer  between 0 and  100
                           DEFAULT:0
                           Traffic shaping rate for the interface as a percentage. 0% means traffic is Unlimited. 100% means
                           traffic is Blocked/Limited.
    rtlimitUcast* {
    Rate limit for unicast
      status*     boolean  Rate limit for unicast enabled or disabled
      threshold*  integer  DEFAULT:5
                           Rate limiting value for unicast as a percentage
    }
    rtlimitMcast* {
    Rate limit for multicast
      status*     boolean  Rate limit for multicast enabled or disabled
      threshold*  integer  DEFAULT:5
                           Rate limiting value for multicast as a percentage
    }
    rtlimitBcast* {
    Rate limit for broadcast
      status*     boolean  Rate limit for broadcast enabled or disabled
      threshold*  integer  DEFAULT:5
                           Rate limiting value for broadcast as a percentage
    }
    portVlanId*   integer  between 1 and  4096
                           DEFAULT:1
                           Port's VLAN ID
    defVlanPrio*  integer  between 0 and  7
                           DEFAULT:0
                           Default VLAN priority
    scheduleName  string   Name of the schedule
  }
}
```

7.2 POST /swcfg_port

Set port configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	integer	1	Port ID Number

REQUEST BODY - application/json

```
{
  switchPortConfig {
    ID*          integer  Port Number
    description* string   Port description label
    portType*    enum     DEFAULT:1
                          ALLOWED:0, 1, 2, 3, 4, 5
                          Port configuration type:
                          * `0` = MODE_NONE
                          * `1` = MODE_GENERAL
                          * `2` = MODE_ACCESS
                          * `3` = MODE_TRUNK
                          * `4` = MODE_PRIVATE_HOST
                          * `5` = MODE_PRIVATE_PROMISC
```

```
    adminMode*    boolean  DEFAULT:true
                           Enable physical port interface
    portSpeed     enum     DEFAULT:0
                           ALLOWED:0, 1, 2, 3, 4
                           Port link speed:
                           * `0` = auto
                           * `1` = SP10
                           * `2` = SP100
                           * `3` = SP1000
                           * `4` = SP10G
    duplexMode    enum     DEFAULT:2
                           ALLOWED:2, 1, 0
                           Duplex Mode:
                           * `2` = auto
                           * `1` = full
                           * `0` = half
    linkStatus    enum     DEFAULT:1
                           ALLOWED:0, 1
                           Link up or down Status (read-only):
                           * `0` = Up
                           * `1` = Down
    linkTrap      boolean  DEFAULT:true
                           Enable link trap
    maxFrameSize  integer  between 1500 and  9198
                           DEFAULT:1518
                           Max frame size
    isPoE*        boolean  DEFAULT:true
                           Port is PoE capable
    txRate*       integer  between 0 and  100
                           DEFAULT:0
                           Traffic shaping rate for the interface as a percentage. 0% means traffic is Unlimited. 100% means traffic is Blocked/Limited.
    rtlimitUcast* {
    Rate limit for unicast
      status*     boolean  Rate limit for unicast enabled or disabled
      threshold*  integer  DEFAULT:5
                           Rate limiting value for unicast as a percentage
    }
    rtlimitMcast* {
    Rate limit for multicast
      status*     boolean  Rate limit for multicast enabled or disabled
      threshold*  integer  DEFAULT:5
                           Rate limiting value for multicast as a percentage
    }
    rtlimitBcast* {
    Rate limit for broadcast
      status*     boolean  Rate limit for broadcast enabled or disabled
      threshold*  integer  DEFAULT:5
                           Rate limiting value for broadcast as a percentage
    }
    portVlanId*   integer  between 1 and  4096
                           DEFAULT:1
                           Port's VLAN ID
    defVlanPrio*  integer  between 0 and  7
                           DEFAULT:0
                           Default VLAN priority
    scheduleName  string   Name of the schedule
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

7.3 GET /sw_portstats

Get port statistics

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	undefined	ALL	Port ID Number by `<port#>` or `ALL`

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  switchStatsPort {
    portId          integer   Port Number
    switchName      string    Name of Switch
    myDesc          string    Port description
    adminMode       boolean   Admin Mode of port
    status          enum      ALLOWED:0, 1
                              Link Status:
                              * `0` = LINK_UP
                              * `1` = LINK_DOWN
    poeStatus       enum      ALLOWED:-1, 0, 1, 2, 3, 4, 5, 6, 7
                              PoE Status:
                              * `-1` = STATUS_INVALID
                              * `0` = STATUS_DISABLED
                              * `1` = STATUS_SEARCHING
                              * `2` = STATUS_DELIVERING_POWER
                              * `3` = STATUS_TEST
                              * `4` = STATUS_FAULT
                              * `5` = STATUS_OTHER_FAULT
                              * `6` = STATUS_REQUESTING_POWER
                              * `7` = STATUS_OVERLOAD
    mode            enum      ALLOWED:0, 1, 2, 3, 4, 5
                              Port Mode:
                              * `0` = MODE_NONE
                              * `1` = MODE_GENERAL
                              * `2` = MODE_ACCESS
                              * `3` = MODE_TRUNK
                              * `4` = MODE_PRIVATE_HOST
```

		* `5` = MODE_PRIVATE_PROMISC
vlans	[integer]	VLAN ID membership
trafficRx	integer	Total number of bytes received
trafficTx	integer	Total number of bytes transmitted
rxMbps	string	Current receive bit rate in Mbps
txMbps	string	Current transmit bit rate in Mbps
crcErrorsRx	integer	Total numver of packets with CRC errors received
errorsRxTx	integer	Number of error packets
dropsRxTx	integer	Number of packets dropped
portMacAddress	string	Port MAC Address
speed	enum	ALLOWED:1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 128, 129, 130 Interface Speed: * `1` = SPEED_AUTO_NEG * `2` = SPEED_HALF_100TX * `3` = SPEED_FULL_100TX * `4` = SPEED_HALF_10T * `5` = SPEED_FULL_10T * `6` = SPEED_FULL_100FX * `7` = SPEED_FULL_1000SX * `8` = SPEED_FULL_10GSX * `9` = SPEED_FULL_20GSX * `10` = SPEED_FULL_40GSX * `11` = SPEED_FULL_25GSX * `12` = SPEED_FULL_50GSX * `13` = SPEED_FULL_100GSX * `14` = SPEED_AAL5_155 * `15` = SPEED_FULL_5FX * `128` = SPEED_FULL_2P5FX * `129` = SPEED_LAG * `130` = SPEED_UNKNOWN
duplex	enum	ALLOWED:0, 1, 65535 Interface Duplex mode: * `0` = half * `1` = full * `65535` = auto
frameSize	integer	Packets size measured in Maximum Transmission Units
flowControl	boolean	Flow control enabled
lacpMode	boolean	LACP enabled
mirrored	boolean	Port mirror enabled
stpStatus	boolean	STP admin mode enabled on the port
portState	enum	ALLOWED:1, 2, 3, 4, 5, 6 STP port state: * `1` = Discarding * `2` = Learning * `3` = Forwarding * `4` = Disabled * `5` = Manual forwarding * `6` = Not participate
oprState	enum	ALLOWED:0, 1, 2 Port operational state: * `0` = DIAG_PORT_DISABLE * `1` = DIAG_PORT_ENABLE * `2` DIAG_PORT_D_DISABLE
powerLimitClass	enum	ALLOWED:0, 1, 2, 3, 4, 5 PoE port power class: * `0` = INVALID * `1` = CLASS0 * `2` = CLASS1 * `3` = CLASS2 * `4` = CLASS3 * `5` = CLASS4

neighborInfo {
Neighbor connected device Information

name	string	Neighbor name
description	string	Neighbor description
capabilities	string	Neighbor capabilites
chassisId	string	Neighbor chassis ID

```
      chassisIdSubtype  enum    ALLOWED:1, 2, 3, 4, 5, 6, 7
                                Neighbor chassis subtype:
                                * `1` = SUBTYPE_CHASSIS_COMP
                                * `2` = SUBTYPE_INTF_ALIAS
                                * `3` = SUBTYPE_PORT_COMP
                                * `4` = SUBTYPE_MAC_ADDR
                                * `5` = SUBTYPE_NET_ADDR
                                * `6` = SUBTYPE_INTF_NAME
                                * `7` = SUBTYPE_LOCAL
      portId            string  Neighbor port ID
      portIdSubtype     enum    ALLOWED:1, 2, 3, 4, 5, 6, 7
                                Neighbor port subtype:
                                * `1` = SUBTYPE_INTF_ALIAS
                                * `2` = SUBTYPE_PORT_COMP
                                * `3` = SUBTYPE_MAC_ADDR
                                * `4` = SUBTYPE_NET_ADDR
                                * `5` = SUBTYPE_INTF_NAME
                                * `6` = SUBTYPE_AGENT_ID
                                * `7` = SUBTYPE_LOCAL
      portDescription   string  Neighbor port description
      mgmtIpAddress     string  Neighbor management IP Address
    }
    portAuthState    enum       ALLOWED:1, 2, 3
                                Port authorization state:
                                * `1` = Authorised
                                * `2` = Unauthorised
                                * `3` = N/A
  }
}
```

7.4 POST /device_reset_counters

Reset interface coutnters of device

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	undefined	ALL	Port ID Number by `<port#>` or `ALL`

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

7.5 GET /fdb_stats

Get forwarding database (fdb) statistics

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  fdb_stats {
    staticEntries         integer  Count of the static entries in the FDB table.
    dynamicEntries        integer  Count of the dynamic entries in the FDB table.
    maxTableEntries       integer  Maximum number of entries FDB table can hold.
    currentTableEntries   integer  Current number of entries in the FDB table.
    greatestTableEntries  integer  Greatest number of entries the FDB table held.
  }
}
```

7.6 POST /fdb_stats

Reset forwarding database (fdb) table entries

REQUEST

REQUEST BODY - application/json

```
{
  fdb_stats {
    greatestTableEntriesReset* boolean  Reset the greatest number of entries in the forwarding database to zero.
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

7.7 GET /fdbs

Get forwarding database (fdb) information

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  fdb_stats [{
  Array of object:
    interface  integer  Interface entry (slot/port)
    vlanId     integer  VLAN ID of the entry
    mac        string   MAC Address of the entry
    entryType  enum     ALLOWED:0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12
                        Fdb entry type:
                        * `0` = Static
                        * `1` = Learned
                        * `2` = Management
                        * `3` = GMRP Learned
                        * `4` = Self
                        * `5` = Dot1x Static
                        * `6` = Dot1ag Static
                        * `7` = Routing Intf address
                        * `8` = Address is learned, but not guaranteed to be in HW (relevant for SW learning)
                        * `9` = FIP Snooping Learned
                        * `10` = CP client MAC Address
                        * `11` = ethcfm Static
                        * `12` = Y.1731 Static
  }]
}
```

7.8 DELETE /fdbs

Delete forwarding database (fdb) MAC address entry

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*mac	string		Delete all learned MAC entries in the forwarding database

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

7.9 GET /ptpv2_global

Get switch's PTPv2 status configuration

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  ptpv2_global {
    adminMode* string  Switch's PTPv2 service status
  }
}
```

7.10 POST /ptpv2_global

Set switch's PTPv2 status configuration

REQUEST

REQUEST BODY - application/json

```
{
  ptpv2_global {
    adminMode* string  Switch's PTPv2 service status
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

7.11 GET /ptpv2

Get switch's PTPv2 status configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	string		Port interface ID number by `<port#>` or `All`

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  ptpv2 {
    adminMode* string  Port PTPv2 configuration status
    operaMode* string  Port PTPv2 operational status
  }
}
```

7.12 POST /ptpv2

Set switch's PTPv2 status configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	string		Port interface ID

REQUEST BODY - application/json

```
{
  ptpv2 {
    adminMode* string  Toggle port PTPv2
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

7.13 GET /fiber_optics

Get bridge base configuration

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  fiber_optics {
    port                   string  Port interface
    temp                   string  Temperature of the module in celsius
    voltage                string  Voltage usage of the module
    current                string  Current usage of the module in milliamps
    outputPower            string  Power output of the module in decibel-milliwatts
    inputPower             string  Power input of the module in decibel-milliwatts
    txFault                string  Transmitter fault
    los                    string  Loss of signal
    faultStatus            string  Fault status
    vendorName             string  Vender name of the module
    linkLength_50_um       string  Link length in meters
    linkLength_62.5_um     string  Link length in meters
    linkLength             string  Link length in meters
    serialNumber           string  Serial number of the module
    partNumber             string  Part number of the module
    nominalBitRate         string  Nominal bit rate in Mbps
    rev                    string  Vendor revision
    compliance             string  Module compliance type
    Supported              string  Support status
    possibleSpeedDetected  string  Possible speed detected
  }
}
```

7.14 GET /dot1d_base_config

Get Bridge Base Configuration

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
```

```
    respCode integer
    respMsg  string
  }
  dot1d_base_config {
    baseBridgeAddress string  Base bridge address
    baseNumPorts      integer Base number ports
    baseType          integer Base type
  }
}
```

7.15 GET /dot1d_tp_config

Get bridge timeout period for aging out dynamically learned forwarding information

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status   enum    ALLOWED:success, failure
    respCode integer
    respMsg  string
  }
  dot1d_tp_config {
    learnedEntryDiscards integer Learned entry
    agingTime            integer DEFAULT:300
                                 Aging time in seconds.
  }
}
```

7.16 POST /dot1d_tp_config

Set bridge timeout period for aging out dynamically learned forwarding information

REQUEST

REQUEST BODY - application/json

```
{
  dot1d_tp_config {
    agingTime* integer DEFAULT:300
                       Aging time in seconds.
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
```

```
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

7.17 GET /dot1d_tp_port_entries

Get bridge timeout period port entries

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  dot1d_tp_port_entries [{
```

Array of object:

```
    port        integer  Port Interface Number
    maxInfo     integer  Maximum size of the information field this port can receive or transmit.
    inFrames    integer  Number of frames received by this port from its segment.
    outFrames   integer  Number of frames transmitted by this port to its segment.
    inDiscards  integer  Count of valid frames received which were discarded by the forwarding process.
  }]
}
```

8. POWER OVER ETHERNET INFORMATION AND SETTINGS

8.1 GET /swcfg_poe

Get port PoE configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	undefined	ALL	Port ID Number by `<port#>` or `ALL`

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  poePortConfig {
    portid           integer  Port ID
    enable*          boolean  Enable PoE power
    powerLimitMode*  enum     ALLOWED:0, 1, 2, 3, 4
                              Power limit mode:
                              * `0` = Invalid
                              * `1` = DOT3AF
                              * `2` = USER
                              * `3` = NONE
                              * `4` = COUNT
    classification*  enum     ALLOWED:0, 1, 2, 3, 4
                              PoE power classification:
                              * `0` = Invalid
                              * `1` = Class 0
                              * `2` = Class 1
                              * `3` = Class 2
                              * `4` = Class 3
    currentPower     integer  between 0 and  30000
                              Current power used in milliwatts
    powerLimit*      integer  between 0 and  30000
                              Power limit in milliwatts
    status           enum     ALLOWED:-1, 0, 1, 2, 3, 4, 5, 6, 7
                              PoE status:
                              * `-1` = Invalid
                              * `0` = Disabled
                              * `1` = Searching
                              * `2` = Delivering Power
                              * `3` = Test
                              * `4` = Fault
                              * `5` = Other Fault
                              * `6` = Requesting Power
                              * `7` = Overload
    reset            boolean  PoE port power cycle
  }
}
```

8.2 POST /swcfg_poe

Set port PoE configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*portid	integer	ALL	Port ID Number by `<port#>` or `ALL`

REQUEST BODY - application/json

```
{
  poePortConfig {
    portid           integer  Port ID
    enable*          boolean  Enable PoE power
    powerLimitMode*  enum     ALLOWED:0, 1, 2, 3, 4
                              Power limit mode:
                              * `0` = Invalid
                              * `1` = DOT3AF
                              * `2` = USER
                              * `3` = NONE
                              * `4` = COUNT
    classification*  enum     ALLOWED:0, 1, 2, 3, 4
                              PoE power classification:
                              * `0` = Invalid
                              * `1` = Class 0
                              * `2` = Class 1
                              * `3` = Class 2
                              * `4` = Class 3
    currentPower     integer  between 0 and  30000
                              Current power used in milliwatts
    powerLimit*      integer  between 0 and  30000
                              Power limit in milliwatts
    status           enum     ALLOWED:-1, 0, 1, 2, 3, 4, 5, 6, 7
                              PoE status:
                              * `-1` = Invalid
                              * `0` = Disabled
                              * `1` = Searching
                              * `2` = Delivering Power
                              * `3` = Test
                              * `4` = Fault
                              * `5` = Other Fault
                              * `6` = Requesting Power
                              * `7` = Overload
    reset            boolean  PoE port power cycle
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status   enum     ALLOWED:success, failure
    respCode integer
    respMsg  string
  }
}
```

8.3 GET /poe_config

Get switch PoE settings

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  poe_config {
    firmwareVersion          string   PoE Firmware Version
    pseMainOperationStatus   enum     ALLOWED:ON, OFF
                                      Power Sourcing Equipment main operation status
    totalPowerConsumedWatts  string   Total power consumed in watts
    powerManagmentMode       enum     ALLOWED:dynamic, static
                                      PoE power management mode
    traps                    enum     ALLOWED:enable, disable
    powerDetectionMode       integer  PoE Detection Mode:
                                      * `0` = Invalid
                                      * `1` = Legacy
                                      * `2` = 4pt 802.3af
                                      * `3` = 4pt 802.3af and legacy
                                      * `4` = 2pt 802.3af
                                      * `5` = 2pt 802.3af and legacy
                                      * `6` = None
                                      * `7` = Count
  }
}
```

8.4 POST /poe_config

Set switch PoE settings

REQUEST

REQUEST BODY - application/json

```
{
  poe_config {
    pseMainOperationStatus*  enum     ALLOWED:enabled, disabled
                                      Main PoE Status
    usageThreshold*          integer  between 1 and  99
                                      DEFAULT:95
                                      Limit PoE usage with a threshold in percentage
    powerManagmentMode*      enum     ALLOWED:dynamic, static
                                      PoE power management mode
    powerDetectionMode*      integer  PoE Detection Mode:
                                      * `0` = Invalid
                                      * `1` = Legacy
                                      * `2` = 4pt 802.3af
                                      * `3` = 4pt 802.3af and legacy
```

```
                                                * `4` = 2pt 802.3af
                                                * `5` = 2pt 802.3af and legacy
                                                * `6` = None
                                                * `7` = Count
    traps*                      enum        ALLOWED:enabled, disabled
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

---

9. QUALITY OF SERVICE

9.1 GET /costrust

Get Class of Service (CoS) trust settings

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	undefined	ALL	Port interface ID Number by `<port#>`, `ALL`, or `Global`

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  costrust {
    mode* enum ALLOWED:dot1p, untrusted, ip-dscp
                  Trust mode of COS - Global/ALL/Per Interface:
                  * dot1p
                  * untrusted
                  * ip-dscp
  }
}
```

9.2 POST /costrust

Set Class of Service (CoS) trust settings

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	integer	ALL	Port interface ID Number by `<port#>`, `ALL`, or `Global`

REQUEST BODY - application/json

```
{
  costrust {
    mode* enum ALLOWED:dot1p, untrusted, ip-dscp
                  Trust mode of COS - Global/ALL/Per Interface:
                  * dot1p
                  * untrusted
                  * ip-dscp
  }
```

```
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

9.3 GET /dot1p_queue_map

Get Class of Service (CoS) 802.1p queue mapping

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	undefined	ALL	Port interface ID Number by `<port#>`, `ALL`, or `Global`

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  dot1p_queue_map [{
  Array of object:
    priority  integer  between 0 and  7
                       Priority assigned to this class
    queuemap  integer  between 0 and  7
                       Assigned queue number
  }]
}
```

9.4 POST /dot1p_queue_map

Set Class of Service (CoS) 802.1p queue mapping

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	undefined	ALL	Port interface ID Number by `<port#>` or `Global`

REQUEST BODY - application/json

```
{
  dot1p_queue_map [{
  Array of object:
    priority integer between 0 and 7
                     Priority assigned to this class
    queuemap integer between 0 and 7
                     Assigned queue number
  }]
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status   enum    ALLOWED:success, failure
    respCode integer
    respMsg  string
  }
}
```

9.5 GET /ipdscp_queue_map

Get mapping from the Differentiated Services Code Point (DSCP) to the outgoing traffic forwarding queue

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status   enum    ALLOWED:success, failure
    respCode integer
    respMsg  string
  }
  ipdscp_queue_map [{
  Array of object:
    dscpid  integer between 0 and 63
                     Class identifier for this DSCP
    dscpmap integer between 0 and 7
                     Assigned queue number
  }]
}
```

9.6 POST /ipdscp_queue_map

Set mapping from the Differentiated Services Code Point (DSCP) to the outgoing traffic forwarding queue

REQUEST

REQUEST BODY - application/json

```
{
  ipdscp_queue_map [{
  Array of object:
    dscpid   integer  between 0 and  63
                      Class identifier for this DSCP
    dscpmap  integer  between 0 and  7
                      Assigned queue number
  }]
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

9.7 GET /cos_queue_config

Get Class of Service (CoS) queue configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	integer		Port interface ID by `<port#>`, `Global`, or `ALL`

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  cos_queue_config [{
  Array of object:
```

```
      id             integer  between 0 and  7
                              Queue ID
      min_bw         integer  between 0 and  100
                              Minimum bandwidth percentage
      mgmt_type      enum     ALLOWED:TailDrop, Wred
                              CoS Management Type
      schedule_type  enum     ALLOWED:Weighted, Strict
                              CoS Schedule Type
    }]
  }
```

9.8 POST /cos_queue_config

Set Class of Service (CoS) queue configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	integer		Port interface ID by `<port#>` or `Global`

REQUEST BODY - application/json

```
{
  cos_queue_config [{
  Array of object:
    id             integer  between 0 and  7
                            Queue ID
    min_bw         integer  between 0 and  100
                            Minimum bandwidth percentage
    mgmt_type      enum     ALLOWED:TailDrop, Wred
                            CoS Management Type
    schedule_type  enum     ALLOWED:Weighted, Strict
                            CoS Schedule Type
  }]
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

10. ROUTING SETTINGS

10.1 GET /ip_route_table

Get IP Routing Table

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  ip_route_table [{
  Array of object:
    nextHopIntf  string   Name of interface for next hop
    routeMask    string   Route mask
    nextHopAddr  string   IP address for next hop
    routeType    string   Type of route
    routeProto   string   Learned route protocol
    routeDest    string   Next destination
    metric       integer  Routing metric
    routePref    integer  between 1 and  255
                          Preference of route
  }]
}
```

10.2 GET /host_table

Get Switch's Host Table

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
```

```
  hostTable [{
  Array of object:
    ipAddr   string   IP Address
    macAddr  string   MAC Address
    vlanId   integer  between 1 and  4096
                      VLAN ID
  }]
}
```

11. SPANNING TREE PROTOCOL

11.1 GET /stp

Get STP information

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  spanningTree {
    status*              boolean    DEFAULT:true
                                    STP/RSTP/MST enabled status
    rootBridgePriority*  integer    DEFAULT:32768
                                    Bridge priority
    stpMode*             enum       DEFAULT:2
                                    ALLOWED:0, 1, 2, 3
                                    Selection STP state:
                                    * `0` = STP
                                    * `1` = Unused
                                    * `2` = RSTP
                                    * `3` = MST
    rootBridgeId*        string     Root bridge MAC address
    ports*               [integer]
    lagGroupID*          [integer]
  }
}
```

11.2 POST /stp

Set STP information

REQUEST

REQUEST BODY - application/json

```
{
  spanningTree {
    status*              boolean    DEFAULT:true
                                    STP/RSTP/MST enabled status
    rootBridgePriority*  integer    DEFAULT:32768
                                    Bridge priority
    stpMode*             enum       DEFAULT:2
                                    ALLOWED:0, 1, 2, 3
                                    Selection STP state:
```

```
                                        * `0` = STP
                                        * `1` = Unused
                                        * `2` = RSTP
                                        * `3` = MST
    rootBridgeId*       string      Root bridge MAC address
    ports*              [integer]
    lagGroupID*         [integer]
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

11.3 GET /dot1d_stp_entries

Get Spanning Tree Protocol (STP) entries

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  dot1d_stp_entries [{
    Array of object:
    port              integer  Port interface number
    priority          integer  Spanning Tree port priority
    state             string   Spanning Tree port state
    pathCost          integer  Spanning Tree path cost for the port.
    designatedRoot    string   Spanning Tree designated root for the switch.
    designatedCost    integer  Spanning Tree designated cost for the port
    designatedBridge  string   Spanning Tree designated bridge for the port
    designatedPort    string   Spanning Tree designated port ID
  }]
}
```

11.4 GET /dot1d_stp_config

Get Spanning Tree Protocol (STP) configuration

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  dot1d_stp_config {
    protocolSpecification    enum     ALLOWED:dot1d, unknown
                                      STP protocol
    priority                 integer  between 0 and  61440
                                      Bridge priority and displayed in multiples of 4096
    timeSinceTopologyChange  integer  Time passed since topology change (seconds)
    topChanges               integer  Number of times topology changed.
    designatedRoot           string   Bridge identifier of the root bridge
    rootCost                 integer  Value of the Root Path Cost for the common and internal spanning tree
    rootPort                 integer  Root port identifier
    maxAge                   integer  Maximum age
    helloTime                integer  Hello time
    holdTime                 integer  Hold time
    forwardDelay             integer  Forward delay
    bridgeMaxAge             integer  Bridge maximum age
    bridgeHelloTime          integer  Bridge hello time
    bridgeForwardDelay       integer  Bridge forward delay
  }
}
```

11.5 GET /dot1s_interfaces

Get Multiple Spanning Tree Protocol (MSTP) interface configuration

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  dot1s_interfaces [{
```

Array of object:

```
    interface          integer  Port interface number
    bpduFilterMode     boolean  BPDU filter mode
    bpduFloodMode      boolean  BPDU flood mode
    intfEdgePortMode   boolean  Interface edge port mode
    intfGuardMode      enum     ALLOWED:0, 1, 2
                                STP Guard Mode:
                                * `0` = Loop
                                * `1` = Root
                                * `2` = None
    intfMode           integer  DEFAULT:true
                                Interface mode
  }]
}
```

11.6 POST /dot1s_interfaces

Set Multiple Spanning Tree Protocol (MSTP) interface configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	integer	1	Port Interface Number

REQUEST BODY - application/json

```
{
  dot1s_interfaces {
    interface          integer  Port interface number
    bpduFilterMode     boolean  BPDU filter mode
    bpduFloodMode      boolean  BPDU flood mode
    intfEdgePortMode   boolean  Interface edge port mode
    intfGuardMode      enum     ALLOWED:0, 1, 2
                                STP Gaurd Mode:
                                * `0` = Loop
                                * `1` = Root
                                * `2` = None
    intfMode           boolean  Interface Mode
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

11.7 GET /msti

Get Multiple Spanning Tree (MST) ID

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  dot1s_msti_entries [{
  Array of object:
    mstId      integer   MST Instance
    priority   integer   Instance priority
    vlans [{
    Array of object:
      id    integer  VLAN ID WRT Instance
      type  string   VLAN Type
      name  string   VLAN Name
    }]
  }]
}
```

11.8 POST /msti

Set Multiple Spanning Tree (MST) ID

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
mstid	integer	1	Multiple Spanning Tree (MST) ID

REQUEST BODY - application/json

```
{
  dot1s_msti_entries {
    vlanid*    integer  between 1 and  4093
                        VLAN ID
    priority*  integer  between 0 and  240
                        DEFAULT:128
                        msti priority
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

11.9 DELETE /msti

Delete MST ID

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*mstid	integer	1	MST ID

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

12. VIRTUAL LOCAL AREA NETWORKS

12.1 GET /swcfg_vlan

Get VLAN configuration settings

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*vlanid	integer between 1 and 4093	1	VLAN ID

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  switchConfigVlan {
    vlanId*         integer  between 1 and  4096
                             VLAN ID
    name*           string   VLAN Name
    voiceVlanState  boolean  Voice VLAN status
    autoVoipState   boolean  AutoVoIP
    autoVideoState  boolean  Auto Video
    igmpConfig {
      igmpState  boolean  IGMP state for the VLAN
    }
  }
}
```

12.2 POST /swcfg_vlan

Set VLAN configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*vlanid	integer between 1 and 4093	1	VLAN ID

REQUEST BODY - application/json

```
{
```

```
  switchConfigVlan {
    vlanId*          integer  between 1 and  4096
                              VLAN ID
    name*            string   VLAN Name
    voiceVlanState   boolean  Voice VLAN status
    autoVoipState    boolean  AutoVoIP
    autoVideoState   boolean  Auto Video
    igmpConfig {
      igmpState  boolean  IGMP state for the VLAN
    }
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

12.3 DELETE /swcfg_vlan

Delete VLAN configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*vlanid	integer between 1 and 4093	1	VLAN ID

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

12.4 GET /swcfg_vlan_membership

Get VLAN port membership list

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*vlanid	integer between 1 and 4093	1	VLAN ID

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  vlanMembership {
    vlanid*             integer       between 1 and  4096
                                      VLAN ID
    portMembers* {
      port    integer  Physical ports belonging to the VLAN
      tagged  boolean  VLAN tagged membership
    }
    lagMembers* {
      portid  integer  LAG ports belonging to the VLAN
      tagged  boolean  VLAN tagged membership
    }
    trafficPrio*        integer       Traffic Priority of VLAN
    trafficPrioPortMem* [{
    Array of object:
      portid  integer  Traffic priority and Port VLAN IDs (PVID) for these physical ports
    }]
    trafficPrioLagMem* [{
    Array of object:
      portid  integer  Traffic priority and Port VLAN IDs (PVID) for these LAG ports.
    }]
    pvidMembers [{
    Array of object:
      portid  integer  between 1 and  4093
                       Port VLAN IDs (PVID) assignments for these port interfaces.
    }]
  }
}
```

12.5 POST /swcfg_vlan_membership

Set list of VLAN port membership

REQUEST

REQUEST BODY - application/json

```
{
  vlanMembership {
```

```
    vlanid*            integer       between 1 and  4096
                                     VLAN ID
    portMembers* {
      port    integer  Physical ports belonging to the VLAN
      tagged  boolean  VLAN tagged membership
    }
    lagMembers* {
      portid  integer  LAG ports belonging to the VLAN
      tagged  boolean  VLAN tagged membership
    }
    trafficPrio*       integer       Traffic Priority of VLAN
    trafficPrioPortMem* [{
    Array of object:
      portid  integer  Traffic priority and Port VLAN IDs (PVID) for these physical ports
    }]
    trafficPrioLagMem* [{
    Array of object:
      portid  integer  Traffic priority and Port VLAN IDs (PVID) for these LAG ports.
    }]
    pvidMembers [{
    Array of object:
      portid  integer  between 1 and  4093
                       Port VLAN IDs (PVID) assignments for these port interfaces.
    }]
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

12.6 GET /dot1q_sw_port_config

Get VLAN switchport interface configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	integer	1	Port Interface ID

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  dot1q_sw_port_config {
    interface                 string    Physical or logical interface in slot/port format
    accessVlan                integer   Access VLAN ID for the interface
    allowedVlanList           [string]  For a given interface get VLAN membership for a range of VLANs
    dynamicallyAddedVlanList  string    Dynamically Added VLANs for the interface
    forbiddenVlanList         string    Forbidden VLANs for the interface
    configMode                enum      ALLOWED:none, general, access, trunk,
                                        privateHost, privatePromisc
                                        Switchport Configuration Mode for the interface
    nativeVlan                integer   Native VLAN ID for the interface
    taggedVlanList            [string]  Tagged VLANs for the interface
    untaggedVlanList          [string]  Untagged VLANs for the interface
  }
}
```

12.7 POST /dot1q_sw_port_config

Set VLAN switchport interface configuration

REQUEST

QUERY PARAMETERS

NAME	TYPE	EXAMPLE	DESCRIPTION
*interface	integer	1	Port Interface ID

REQUEST BODY - application/json

```
{
  dot1q_sw_port_config {
    accessVlan*       integer   Set access VLAN ID for the interface
    allowedVlanList*  [string]
    configMode*       enum      ALLOWED:none, general, access, trunk, privateHost,
                                privatePromisc
                                Switchport Configuration Mode for a port
    nativeVlan*       string    Native VLAN ID for a port
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
}
```

12.8 GET /vlan_ip

Get VLAN IP configuration

REQUEST

No request parameters

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
    respCode  integer
    respMsg   string
  }
  vlan_ip [{
  Array of object:
    vlanId       integer  VLAN ID
    dhcpStatus   boolean  Enable VLAN DHCP client
    ipAddr       string   VLAN IP address
    ipMask       string   VLAN subnet mask
    ipMtu        integer  VLAN Maximum Transmission Unit (MTU) size
    vlanRouting  boolean  Enable VLAN routing
  }]
}
```

12.9 POST /vlan_ip

Set VLAN IP configuration

REQUEST

REQUEST BODY - application/json

```
{
  vlan_ip {
    vlanId       integer  VLAN ID
    dhcpStatus   boolean  Enable VLAN DHCP client
    ipAddr       string   VLAN IP address
    ipMask       string   VLAN subnet mask
    ipMtu        integer  VLAN Maximum Transmission Unit (MTU) size
    vlanRouting  boolean  Enable VLAN routing
  }
}
```

RESPONSE

STATUS CODE - 200: successful operation

RESPONSE MODEL - application/json

```
{
  resp {
    status    enum     ALLOWED:success, failure
```

```
    respCode  integer
    respMsg   string
  }
}
```
