



WiNG REST API Getting Started Guide

For release 5.9.3



Copyright © 2018 Extreme Networks, Inc. All rights reserved.

Legal Notice

Extreme Networks, Inc. reserves the right to make changes in specifications and other information contained in this document and its website without prior notice. The reader should in all cases consult representatives of Extreme Networks to determine whether any such changes have been made.

The hardware, firmware, software or any specifications described or referred to in this document are subject to change without notice.

Trademarks

Extreme Networks and the Extreme Networks logo are trademarks or registered trademarks of Extreme Networks, Inc. in the United States and/or other countries.

All other names (including any product names) mentioned in this document are the property of their respective owners and may be trademarks or registered trademarks of their respective companies/owners.

For additional information on Extreme Networks trademarks, please see:

www.extremenetworks.com/company/legal/trademarks

Software Licensing

Some software files have been licensed under certain open source or third-party licenses. End-user license agreements and open source declarations can be found at:

www.extremenetworks.com/support/policies/software-licensing

Support

For product support, phone the Global Technical Assistance Center (GTAC) at 1-800-998-2408 (toll-free in U.S. and Canada) or +1-408-579-2826. For the support phone number in other countries, visit: <http://www.extremenetworks.com/support/contact/>

For product documentation online, visit: <https://www.extremenetworks.com/documentation/>

Table of Contents

Preface.....	4
Text Conventions.....	4
Platform-Dependent Conventions.....	4
Providing Feedback to Us.....	5
Getting Help.....	5
Documentation and Training.....	6
Chapter 1: WiNG REST API.....	7
Accessing the WiNG API.....	8
API Request Components.....	8
API Response Codes.....	9
Authentication and Authorization.....	10
Making Your First API Call.....	12
List of Available URIs.....	14
Chapter 2: API Usage Examples.....	18
Change the WLAN SSID.....	18
Create a New WLAN and Assign it to a Profile.....	19
Delete a VLAN.....	20
Clear the Device Upgrade History.....	21
Update the Software Image.....	22
Disassociate Wireless Clients.....	22
Get Wireless Radio Information for RF Domain.....	23

Preface

This section discusses the conventions used in this guide, ways to provide feedback, additional help, and other Extreme Networks® publications.

Text Conventions

The following tables list text conventions that are used throughout this guide.

Table 1: Notice Icons

Icon	Notice Type	Alerts you to...
	General Notice	Helpful tips and notices for using the product.
	Note	Important features or instructions.
	Caution	Risk of personal injury, system damage, or loss of data.
	Warning	Risk of severe personal injury.
<i>New!</i>	New Content	Displayed next to new content. This is searchable text within the PDF.

Table 2: Text Conventions

Convention	Description
Screen displays	This typeface indicates command syntax, or represents information as it appears on the screen.
The words enter and type	When you see the word “enter” in this guide, you must type something, and then press the Return or Enter key. Do not press the Return or Enter key when an instruction simply says “type.”
[Key] names	Key names are written with brackets, such as [Return] or [Esc] . If you must press two or more keys simultaneously, the key names are linked with a plus sign (+). Example: Press [Ctrl]+[Alt]+[Del]
<i>Words in italicized type</i>	Italics emphasize a point or denote new terms at the place where they are defined in the text. Italics are also used when referring to publication titles.

Platform-Dependent Conventions

Unless otherwise noted, all information applies to all platforms supported by software, which are the following:

- ExtremeSwitching® switches
- Summit® switches
- SummitStack™

When a feature or feature implementation applies to specific platforms, the specific platform is noted in the heading for the section describing that implementation in the ExtremeXOS command documentation (see the Extreme Documentation page at www.extremenetworks.com/documentation/). In many cases, although the command is available on all platforms, each platform uses specific keywords. These keywords specific to each platform are shown in the Syntax Description and discussed in the Usage Guidelines sections.

Providing Feedback to Us

Quality is our first concern at Extreme Networks, and we have made every effort to ensure the accuracy and completeness of this document. We are always striving to improve our documentation and help you work better, so we want to hear from you! We welcome all feedback but especially want to know about:

- Content errors or confusing or conflicting information.
- Ideas for improvements to our documentation so you can find the information you need faster.
- Broken links or usability issues.

If you would like to provide feedback to the Extreme Networks Information Development team, you can do so in two ways:

- Use our short online feedback form at <https://www.extremenetworks.com/documentation-feedback/>.
- Email us at documentation@extremenetworks.com.

Please provide the publication title, part number, and as much detail as possible, including the topic heading and page number if applicable, as well as your suggestions for improvement.

Getting Help

If you require assistance, contact Extreme Networks using one of the following methods:

- **GTAC (Global Technical Assistance Center) for Immediate Support**
 - **Phone:** 1-800-998-2408 (toll-free in U.S. and Canada) or +1 408-579-2826. For the support phone number in your country, visit: www.extremenetworks.com/support/contact
 - **Email:** support@extremenetworks.com. To expedite your message, enter the product name or model number in the subject line.
- **Extreme Portal** — Search the GTAC knowledge base, manage support cases and service contracts, download software, and obtain product licensing, training, and certifications.
- **The Hub** — A forum for Extreme Networks customers to connect with one another, answer questions, and share ideas and feedback. This community is monitored by Extreme Networks employees, but is not intended to replace specific guidance from GTAC.

Before contacting Extreme Networks for technical support, have the following information ready:

- Your Extreme Networks service contract number and/or serial numbers for all involved Extreme Networks products
- A description of the failure
- A description of any action(s) already taken to resolve the problem
- A description of your network environment (such as layout, cable type, other relevant environmental information)

- Network load at the time of trouble (if known)
- The device history (for example, if you have returned the device before, or if this is a recurring problem)
- Any related RMA (Return Material Authorization) numbers

Subscribing to Service Notifications

You can subscribe to email notifications for product and software release announcements, Vulnerability Notices, and Service Notifications.

- 1 Go to www.extremenetworks.com/support/service-notification-form.
- 2 Complete the form with your information (all fields are required).
- 3 Select the products for which you would like to receive notifications.



Note

You can modify your product selections or unsubscribe at any time.

- 4 Click **Submit**.

Documentation and Training

To find Extreme Networks product guides, visit our documentation pages at:

Current Product Documentation	www.extremenetworks.com/documentation/
Archived Documentation (for earlier versions and legacy products)	www.extremenetworks.com/support/documentation-archives/
Release Notes	www.extremenetworks.com/support/release-notes
Hardware/Software Compatibility Matrices	https://www.extremenetworks.com/support/compatibility-matrices/
White papers, data sheets, case studies, and other product resources	https://www.extremenetworks.com/resources/

Open Source Declarations

Some software files have been licensed under certain open source licenses. More information is available at: www.extremenetworks.com/support/policies/open-source-declaration/.

Training

Extreme Networks offers product training courses, both online and in person, as well as specialized certifications. For more information, visit www.extremenetworks.com/education/.

1 WiNG REST API

[Accessing the WiNG API](#)
[API Request Components](#)
[API Response Codes](#)
[Authentication and Authorization](#)
[Making Your First API Call](#)
[List of Available URIs](#)

The WiNG REST API provides a programmatic interface to access configuration, statistics, and runtime information and issue additional configuration parameters. It creates a single management framework for application developers and IT services to work together and share data and services through it. Using an API to consolidate multiple management systems allows businesses to create a custom management system that is tailored to their business needs.

The WiNG API is based on RESTful principles, and uses a combination of configuration daemon and an HTTP/HTTPS front-end. API request and response bodies are formatted in JavaScript Object Notation (JSON). The front-end receives REST requests via standard HTTP/HTTPS methods, and forwards the request to the WiNG operating system after converting it into internal configuration daemon format. The response generated by the WiNG configuration daemon is converted to JSON and sent back to the client.

To submit API calls and build custom applications to manage and monitor an ExtremeWireless WiNG deployment, your RESTful API consuming program needs to have logged in using credentials granting at least read permissions. Any administrator account can be used with the REST API, but only fully privileged accounts can be used to make configuration changes through the REST API.



Note

The WiNG REST API is currently supported on all NX/VX WiNG controllers running release 5.9.2 and later.

This guide provides information about accessing the API, structure of the API request and response bodies, error codes, and usage examples.



Note

You cannot run the sample requests in this guide as-is. Replace call-specific parameters such as host IP address, user credentials, and authorization tokens with your own values.

Related Links

[Accessing the WiNG API](#) on page 8

[API Request Components](#) on page 8

[API Response Codes](#) on page 9

[Making Your First API Call](#) on page 12

[List of Available URIs](#) on page 14

[API Usage Examples](#) on page 18

Accessing the WiNG API

You can use any language or library that can submit REST API requests and process JSON to query the WiNG API. Examples of languages and libraries that can build REST API clients include:

- For Java, the Jersey library provides the reference implementation of JAX-RS, a Java standard for RESTful web services. The implementation includes a client library that can run directly on the JVM.
- For Python, the Requests and JSON libraries facilitate REST API applications.
- For .Net, the core language provides facilities for submitting HTTP requests, and .Net libraries include a serializer for JSON.
- For the Linux shell, Wget and cURL can execute REST API calls. Linux shell utilities, like awk and grep, can parse and process JSON.

You can also use tools like Postman, an easy-to-use [Chrome extension](#) for making HTTP requests.

Note



The examples in this guide use [cURL](#), a standard command line tool. All you need to do is replace call-specific parameters such as host IP address, user credentials, and authorization token with your own values and you can test the calls from the command line.

API Request Components

To construct a REST API request, combine the following components:

Component	Description
The HTTP method	<ul style="list-style-type: none"> • GET: Return data from the server • DELETE: Delete a resource from the server • POST: Create a new resource or update an existing resource on the server • PUT: Create or update a resource on the server <p>Note: The WiNG REST API implements the POST and PUT methods in the same manner.</p>
The base URL of the API	<code>http(s):// wiNG_host_name_or_IP_address/rest</code>
The URI to the resource	The resource to create, update, query, or delete. For example, <code>/v1/stats/wireless/clients</code> .
Path parameters	These variables are part of the full URL path and are used to point to a specific resource within a collection. For example, <code>/v1/cfg/wlan/{test}</code> , where <code>{test}</code> is the path parameter and points to the specific WLAN{test} within the <code>cfg</code> resource. Substitute the path parameters with your actual values when you construct your API request.

Component	Description
HTTP request headers	The following HTTP headers are supported: <ul style="list-style-type: none"> • Accept: Required for operations with a response body, syntax is <code>Accept: application/json</code>. • Content-Type: Required for operations with a request body, syntax is <code>Content-Type: application/json</code>. • Authorization: Required to get an access token or make API calls.
JSON request body	Required for most POST and PUT requests.

API Response Codes

The WiNG API returns standard HTTP status codes in addition to JSON-based error codes and messages in the response body.

Table 3: HTTP Response Status Codes

Code	Description
200 OK	The request was successful
201 Created	The resource was created successfully
204 No Content	Success with no response body
400 Bad Request	The operation failed because the request is syntactically incorrect or violated schema
401 Unauthorized	The authentication credentials are invalid or the user is not authorized to use the API
404 Not Found	The server did not find the specified resource that matches the request URL
405 Method Not Allowed	The API does not support the requested HTTP method

Error Codes and Messages

If an API request is successful, the response looks similar to the following example:

```
{
  "success" : true
  "return_code" : 0
  "data" : <some data> - if the call returns data
}
```

If an API request cannot be completed or results in an error, the response looks similar to the following example:

```
{
  "success" : false
  "errors": "get [wlan/rest-1/vlans/102] is not valid - vlans[102] does not exist",
}
```

```
"return_code": 1
}
```

Encrypted Data

WiNG can encrypt configuration parameters containing confidential information, e.g. wireless keys, SNMP community strings, etc. Encryption is enabled by the `password-encryption` CLI command. When enabled, REST calls return encrypted data instead of clear text values for these parameters.

Sample Response with `password-encryption` Enabled

```
{
  "data": [
    {
      "access": "ro",
      "ip_snmp_accesslist": "default",
      "name": "CRbtqHNZDOjKTEv4+uQ/CQAAAVzEVuqRcuiQmysSVMKJ3vx"
      // Values are encrypted
    },
    {
      "access": "rw",
      "ip_snmp_accesslist": "default",
      "name": "$ES1" // Alias names are not encrypted
    }
  ],
  "return_code": 0
}
```

Authentication and Authorization

You must start a valid REST session by sending a basic authentication request to the WiNG API server before you can start making API calls. The request header should include a valid management user name and password in `username:password` format. The username and password are encoded with Base64, which is an encoding technique that converts the username and password into a set of 64 characters to ensure safe transmission as part of the Authorization header. The WiNG server can use authentication mechanisms such as local database, RADIUS, etc, but the actual authentication mechanism(s) used depends on the management policy of the WiNG device.

Note



WiNG's REST API is protected by the same access restrictions which are provided via the WiNG command line or graphical user interface. For example, if a user role does not allow write access to a resource, then an attempt to configure/update this resource via REST will fail. For more information on WiNG user roles, see the WiNG 5.9.3 System Reference Guide located at: <https://www.extremenetworks.com/support/documentation/>.

Sample Login Request

```
curl -X GET -u <mgmt-username>:<mgmt-user-password> -k https://10.190.50.43/rest/v1/act/login
```

Sample Login Response

```
{
  "data": {
    "auth_token": "e5c6c3bd73057b5252d683ced64897ef"
  },
  "return_code": 0
}
```



Note

Save the `auth_token` and forward it as a cookie in the request header in subsequent API calls.

Example: Including `auth_token` in subsequent API calls.

```
cookie = e5c6c3bd73057b5252d683ced64897ef
```

```
curl -X GET --cookie auth_token=$cookie -k
https://10.190.50.43/rest/v1/cfg/management_policy/default/snmp/community_string
```

You can send a logout request to the WiNG API server to close a session. Include the `auth_token` in the request header to indicate which session you wish to close.



Note

An idle REST session is terminated automatically by the WiNG device after the duration exceeds the `idle-session-timeout` value in the management policy. The default interval is 30 minutes.

Sample Logout Request

```
curl -X GET --cookie auth_token=$cookie -k https://10.190.50.43/rest/v1/act/logout
```

Successful Logout Response

```
{
  "return_code": 0
}
```



Note

After you log out or if the session expires, you need to log in and start a new session to continue making API calls. You will see an invalid authentication token error message if your session is no longer active.

Sample Invalid Authentication Token Error Message

```
{
  "errors": [
    "Unable to find the session for auth_token: [e5c6c3bd73057b5252d683ced64897ef]"
  ],
  "return_code": 1
}
```

Making Your First API Call

This sample API call demonstrates how to read a specific WLAN configuration `test-1` within the `cfg` resource.

- 1 Download [cURL](#) for your environment.



Tip

If you use Windows, use a Bash shell to make cURL calls.

- 2 [Log in to the REST API server](#) using valid management user credentials.



Note

You must forward the `auth_token` as a cookie with each API call.

- 3 Use the GET method to access the `cfg` resource and fetch the specific WLAN configuration.

Sample Request

```
curl -X GET --cookie auth_token=$cookie https://10.190.50.43/rest/v1/cfg/wlan/test-1
```

Sample Response (200 OK)

```
{
  "data": {
    "aaa_policy": "",
    "accounting_radius": false,
    "accounting_syslog": {
      "enable": false,
      "host": "",
      "mac_case": "upper",
      "mac_format": "pair-hyphen",
      "port": 514,
      "proxy_mode": "none"
    },
    "wpa_wpa2_handshake_timeout": [
      500,
      500,
      500,
      500
    ],
    "wpa_wpa2_opmk_caching": true,
    "wpa_wpa2_pmk_caching": true,
    "wpa_wpa2_preauthentication": false,
    "wpa_wpa2_psk": "Z+eRSxI9kwENQ6svDyOBawAAAC58Tyzp/
VuvHsLVbmmuTkMatQ4pmt3YgTNij1W6Q2eQ9Am2KnYW9WZ14jJrxnSS5G8=",
    "wpa_wpa2_tkip_ctrmeas_hold_time": 60,
    "wpa_wpa2_use_sha256": false
  },
  "return_code": 0
}
```

Table 4: Response Definitions

Response Item	Description	Data Type
aaa_policy	The AAA policy name	String (1 to 32 characters)
accounting_radius	A flag to enable/disable support for RADIUS accounting messages	Boolean
accounting_syslog	Configuration of Syslog accounting messages	Object
enable	A flag to enable/disable support for Syslog accounting messages	Boolean
host	Syslog destination host name or IP address for accounting records	String
mac_case	The case in which the MAC address is to be specified. Default is upper case.	Enumeration [lower,upper]
mac_format	The format in which the MAC address must be filled in the syslog messages. Default is pair-hyphen.	Enumeration [no-delim, air-hyphen, quad-dot, middle-hyphen]
port	UDP port number of the Syslog server	Integer (1 to 65535, default is 514)
proxy_mode	Specifies whether the request is transmitted directly to the server or proxied through the controller or the RF Domain Manager. Default is none.	Enumeration [none, through-controller, through-rf-domain-manager]
wpa_wpa2_handshake_timeout	The timeout (in milliseconds) on a handshake message, before it is retried	Integer (10 to 5000)
wpa_wpa2_opmk_caching	A flag to enable/disable the use of Opportunistic Key Caching (same PMK across APs for fast roaming with 802.1X EAP)	Boolean
wpa_wpa2_pm_k_caching	A flag to enable/disable the use of cached pairwise master keys (fast roaming with 802.1X EAP)	Boolean
wpa_wpa2_preauthentication	A flag to enable/disable the use of preauthentication (802.11i fast roaming)	Boolean
wpa_wpa2_psk	Pre-shared key	String (8 to 63 characters)
wpa_wpa2_tkip_ctrmeas_hold_time	The amount of time (in seconds) for which a WLAN is disabled when TKIP countermeasures are invoked	Integer (0 to 65535, default is 60)
wpa_wpa2_use_sha256	A flag to enable/disable use of SHA256 authentication key management	Boolean

For other common WiNG API use cases, see [API Usage Examples](#).

List of Available URIs

The WiNG API provides the following top level paths to fetch data and configure various parameters.

Table 5: Configuration Resource URIs: Read and modify device configuration

Path	Description
/v1/cfg/wlan/	This path serves all resource requests and initiates all configuration operations on the WLAN entity.
/v1/cfg/device/	This path serves all resource requests and initiates all configuration operations on the device entity.
/v1/cfg/profile/	This path serves all resource requests and initiates all configuration operations on the profile entity.
/v1/cfg/management_policy/	This path serves all resource requests and initiates all configuration operations on the management policy entity.
/v1/cfg/smart_rf_policy/	This path serves all resource requests and initiates all configuration operations on the smart-rf policy entity.
/v1/cfg/captive_portal/	This path serves all resource requests and initiates all configuration operations on the captive portal entity.
/v1/cfg/dns_whitelist/	This path serves all resource requests and initiates all configuration operations on the DNS whitelist entity.
/v1/cfg/wips_policy/	This path serves all resource requests and initiates all configuration operations on the WIPS policy entity.
/v1/cfg/aaa_policy/	This path serves all resource requests and initiates all configuration operations on the AAA policy entity.
/v1/cfg/ip_acl/	This path serves all resource requests and initiates all configuration operations on the IPv4 access list entity.
/v1/cfg/app_policy	This path serves all resource requests and initiates all configuration operations on the application policy entity.
/v1/cfg/mac_acl/	This path serves all resource requests and initiates all configuration operations on the MAC access list entity.

Table 6: Action URIs: Perform Actions (Such as: login/logout, commit, save, revert, upgrade, and more)

Path	Description
/v1/act/login	This path allows you to create a new session.
/v1/act/logout	This path allows you to close the session identified by <code>auth_token</code> .
/v1/act/commit	This path allows you to commit the configuration changes made in the session.
/v1/act/revert	This path allows you to revert the configuration changes made in the session.

Table 6: Action URIs: Perform Actions (Such as: login/logout, commit, save, revert, upgrade, and more) (continued)

Path	Description
/v1/act/wrmem	This path allows you to write the committed configuration to persistent storage.
/v1/act/upgrade	This path allows you to upgrade the software image.
/v1/act/disassociate-client	This path allows you to disassociate the wireless client(s).
/v1/act/device-upgrade	This path allows you to upgrade adopted devices.
/v1/act/device-upgrade-cancel	This path allows you to cancel upgrading a device.
/v1/act/load-device-upgrade-image	This path allows you to load the device images to controller for device upgrades.
/v1/act/clear-device-upgrade-history	This path allows you to clear the device upgrade history.
/v1/act/file-read	This path allows you to read files from flash directory; startup-config from nvram or copy running-config.
/v1/act/file-copy	This path allows you to copy files from flash directory; startup-config from nvram or copy running-config.
/v1/act/clone	This path allows you to clone an existing top-level configuration object.
/v1/act/rename	This path allows you to rename an existing top-level configuration object.
/v1/act/replace	This path allows you to replace a device configuration object.

Table 7: Statistics URIs: Retrieve Statistics and Runtime Information

Path	Description
/v1/stats/wireless/radio	This path allows you to retrieve information about wireless radios for the device executing the REST call, a rf-domain, or another specified device.
/v1/stats/wireless/client	This path allows you to retrieve information about wireless clients for the device executing the REST call, a rf-domain, or another specified device.
/v1/stats/wireless/client-stats	This path allows you to retrieve wireless client statistics for the device executing the REST call, a rf-domain, or another specified device.
/v1/stats/wireless/radio-stats	This path allows you to retrieve wireless radio statistics for the device executing the REST call, a rf-domain, or another specified device.
/v1/stats/wireless/ap-info	This path allows you to retrieve information about wireless-managed access points for the device executing the REST call, a rf-domain, or another specified device.

Table 7: Statistics URIs: Retrieve Statistics and Runtime Information (continued)

Path	Description
/v1/stats/mint/links	This path allows you to retrieve information about mint links for the device executing the REST call or another specified device.
/v1/stats/mint/neighbors	This path allows you to retrieve information about mint neighbors for the device executing the REST call or another specified device.
/v1/stats/noc/domains	This path allows you to retrieve information about all domains for the device executing the REST call.
/v1/stats/noc/devices	This path allows you to retrieve information about devices in the network for the device executing the REST call.
/v1/stats/adoption/status	This path allows you to retrieve adoption status information for the device executing the REST call or another specified device.
/v1/stats/device	This path allows you to retrieve information about the device executing the REST call.
/v1/stats/event-history	This path allows you to retrieve information about event history for the device executing the REST call, a rf-domain, or another specified device.
/v1/stats/cdp-neighbors	This path allows you to retrieve information about CDP neighbors for the device executing the REST call or another specified device.
/v1/stats/lldp-neighbors	This path allows you to retrieve information about LLDP neighbors for the device executing the REST call or another specified device.
/v1/stats/dhcp-vendor-opts	This path allows you to retrieve information about DHCP options for the device executing the REST call or another specified device.
/v1/stats/captive-portal/sessions	This path allows you to retrieve information about captive portal sessions for the device executing the REST call, a rf-domain, or another specified device.
/v1/stats/gre/info	This path allows you to retrieve general information about GRE tunnels for the device executing the REST call or another specified device.
/v1/stats/gre/detail	This path allows you to retrieve detailed information about GRE tunnels for the device executing the REST call or another specified device.
/v1/stats/l2tpv3_stats/ l2tpv3_tunnels	This path allows you to retrieve information about L2TPv3 tunnels for the device executing the REST call or another specified device.
/v1/stats/l2tpv3_stats/ l2tpv3_tunnel_summary	This path allows you to retrieve a summary of L2TPv3 tunnels for the device executing the REST call or another specified device.

Table 7: Statistics URIs: Retrieve Statistics and Runtime Information (continued)

Path	Description
<code>/v1/stats/System/upgrade-status</code>	This path allows you retrieve information about last upgrade for the device executing the REST call or another specified device.
<code>/v1/stats/Device-upgrade/load-image-status</code>	This path allows you retrieve status of the firmware file download for the device executing the REST call or another specified device.
<code>/v1/stats/Device-upgrade/status</code>	This path allows you retrieve status of the device upgrade for the device executing the REST call, a rf-domain, or another specified device.
<code>/v1/stats/Device-upgrade/history</code>	This path allows you retrieve history of the device upgrade for the device executing the REST call, a rf-domain, or another specified device.
<code>/v1/stats/Device-upgrade/versions</code>	This path allows you retrieve versions of device upgrade images for the device executing the REST call, a rf-domain, or another specified device.

2 API Usage Examples

Change the WLAN SSID
Create a New WLAN and Assign it to a Profile
Delete a VLAN
Clear the Device Upgrade History
Update the Software Image
Disassociate Wireless Clients
Get Wireless Radio Information for RF Domain

This section provides information on how to accomplish some common tasks using the WiNG REST API methods and endpoints.

Related Links

- [Change the WLAN SSID](#) on page 18
- [Create a New WLAN and Assign it to a Profile](#) on page 19
- [Delete a VLAN](#) on page 20
- [Clear the Device Upgrade History](#) on page 21
- [Update the Software Image](#) on page 22
- [Disassociate Wireless Clients](#) on page 22
- [Get Wireless Radio Information for RF Domain](#) on page 23

Change the WLAN SSID

To change SSID of the WLAN named `test-1` within the `cfg` resource:

- 1 [Log in to the REST API server](#) using valid management user credentials.



Note

You must forward the `auth_token` as a cookie with each API call.

- 2 Use the PUT method to access the `/cfg/wlan/` URI and change WLAN configuration.



Note

The WiNG API, in its current form does not differentiate between PUT and POST methods. The API behaves the same way for both requests.

Sample Request

```
curl -X PUT --cookie auth_token=$cookie -k https://10.190.50.43/rest/v1/cfg/wlan/test-1/ssid -d "newssid"
```

Sample Response (200 OK)

```
{
  "return_code": 0
}
```

Create a New WLAN and Assign it to a Profile

To create a new WLAN named `New-Event` and assign it an access point profile:

- 1 [Log in to the REST API server](#) using valid management user credentials.



Note

You must forward the `auth_token` as a cookie with each API call.

- 2 Use the PUT method to access the `/cfg/wlan/` URI and create a new WLAN configuration.



Note

The WiNG API, in its current form does not differentiate between PUT and POST methods. The API behaves the same way for both requests.

Sample Request

```
curl -X PUT --cookie auth_token=$cookie https://10.190.50.43/rest/v1/cfg/wlan
-d '{
  "ssid" : "New-Event",
  "name" : "New-Event",
  "encryption_type" : "ccmp",
  "wpa_wpa2_psk" : "accessneweventkey",
  "client_client_communication" : false,
  "vlans" : [
    {
      "limit" : 0,
      "vlan" : 23
    }
  ]
}
```

Table 8: Request Body Parameters

Parameter	Description	Required/Optional	Data Type
ssid	Service Set Identifier for this WLAN	Optional. Default value is the name of the WLAN truncated to 32 characters.	String (1 to 32 characters)
name	Name of this WLAN	Required	String (1 to 32 characters)
encryption_type	Encryption to use on this WLAN	Optional. Default value is none.	Enumeration[none,wep64 , wep128-keyguard, keyguard, tkip-ccmp, ccmp]
wpa_wpa2_psk	Pre-shared key	Optional	String (8 to 63 characters)

Table 8: Request Body Parameters (continued)

Parameter	Description	Required/Optional	Data Type
client_client_communication	A flag to allow switching of frames from one wireless client to another on this WLAN	Optional. Default value is true	Boolean
vlan	The VLAN where traffic from this WLAN is mapped.	Optional	Integer (1 to 4094) of String (1 to 32 characters)

Sample Response (200 OK)

```
{
  "return_code": 0
}
```

- Use the PUT method to access the `/cfg/profile/` URI and assign the new WLAN to the 'Conference-Room' access point profile with the 2.4 GHz radio interface.

Sample Request

```
curl -X PUT --cookie auth_token=$cookie https://10.190.50.43/rest/v1/cfg/profile/Conference-Room/interface/radio1/wlanbssmap -d '{
  "wlan" : "New-Event",
  "bss" : 2,
  "primary" : true
}
```

Table 9: Request Body Parameters

Parameter	Description	Required/Optional	Data Type
wlan	The name of the WLAN	Required	String (1 to 32 characters)
bss	BSS number on the radio where this WLAN is to be mapped.	Optional	Integer (1 to 16)
primary	A flag to indicate if this WLAN is primary if there are multiple WLANs on this BSS.	Optional	Boolean

Sample Response (200 OK)

```
{
  "return_code": 0
}
```

Delete a VLAN

To delete a specific VLAN from a WLAN:

- 1 [Log in to the REST API server](#) using valid management user credentials.

**Note**

You must forward the `auth_token` as a cookie with each API call.

- 2 Use the DELETE method to access the `/cfg/wlan/` URI and delete VLAN 101 from WLAN `test-1`.

Sample Request

```
curl -X DELETE --cookie auth_token=$cookie https://10.190.50.43/rest/v1/
cfg/wlan/test-1/vlans/101
```

Sample Response (200 OK)

```
{
  "return_code": 0
}
```

Clear the Device Upgrade History

To clear the device upgrade history on the device making the REST API calls, a remote device, or rf-domain:

- 1 [Log in to the REST API server](#) using valid management user credentials.

**Note**

You must forward the `auth_token` as a cookie with each API call.

- 2 Use the GET and POST methods to access the `/act/clear-device-upgrade-history/` URI and clear the upgrade history.

Sample Requests with Different Possible Input Parameters

```
curl -X GET --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/clear-device-
upgrade-history
```

```
curl -X POST --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/clear-device-
upgrade-history
-d
  '{"rf-domain":"guest-domain"}
```

```
curl -X POST --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/clear-device-
upgrade-history
-d
  '{"device":"08-00-27-96-5F-EA"}
```

```
curl -X POST --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/clear-device-
upgrade-history
-d
  '{"scope":"noc"}
```

Sample Response (200 OK)

```
{
  "return_code": 0
}
```

Update the Software Image

To upgrade the software image on the device making the REST API calls, a remote device, or rf-domain:

- 1 [Log in to the REST API server](#) using valid management user credentials.



Note

You must forward the `auth_token` as a cookie with each API call.

- 2 Use the POST method to access the `/act/upgrade/` URI and update the software image.



Note

The WiNG API, in its current form does not differentiate between PUT and POST methods. The API behaves the same way for both requests.

Sample Request

```
curl -X POST --cookie auth_token=$cookie http://10.1.1.1/rest/v1/act/upgrade
-d
'{"rf-domain":"guest-domain","url":"tftp://10.1.1.101/VX9000.img"}'
```

```
curl -X POST --cookie auth_token=$cookie http://10.1.1.1/rest/v1/act/upgrade
-d
'{"device":"08-00-27-96-5F-EA","dhcp-vendor-options":"True"}'
```

Sample Response (200 OK)

```
{
  "return_code": 0
}
```

Disassociate Wireless Clients

To disassociate wireless clients on the device making the REST API calls, a remote device, or rf-domain:

- 1 [Log in to the REST API server](#) using valid management user credentials.



Note

You must forward the `auth_token` as a cookie with each API call.

- 2 Use the POST method to access the `/act/disassociate-client/` URI.



Note

The WiNG API, in its current form does not differentiate between PUT and POST methods. The API behaves the same way for both requests.

Sample Requests with Different Possible Input Parameters

```
curl -X POST --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/disassociate-
client
-d
'{"client_mac":"all"}'
```

Disassociate the wireless client with the given MAC address on the device

```
curl -X POST --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/disassociate-client
-d
  '{"client_mac":"D0-04-01-3B-01-70"}'
```

Disassociate the wireless clients in the given WLAN on the device

```
curl -X POST --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/disassociate-client
-d
  '{"wlan_name":"testwlan"}'
```

Disassociate all wireless clients on the remote device with the given mac address

```
curl -X POST --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/disassociate-client
-d
  '{"device":"74-67-F7-5C-21-D8","client_mac":"all"}'
```

Disassociate the wireless client with a given MAC address on the rf-domain

```
curl -X POST --cookie auth_token=$cookie http://172.16.0.16/rest/v1/act/disassociate-client
-d
  '{"rf-domain":"guest-domain","client_mac":"D0-04-01-3B-01-70"}'
```

Sample Response (200 OK)

```
{
  "return_code": 0
}
```

Get Wireless Radio Information for RF Domain

To get wireless radio information for a rf-domain:

- 1 [Log in to the REST API server](#) using valid management user credentials.



Note

You must forward the `auth_token` as a cookie with each API call.

- 2 Use the GET method to access the `/stats/wireless/` URI and retrieve the wireless radio configuration.

Sample Request

```
curl -X GET --cookie auth_token=$cookie http://10.1.1.1/rest/v1/stats/wireless/radio
-d
  '{"rf-domain":"SITE-1"}'
```

Sample Response (200 OK)

```
{
  "data": [
    {
      "adopted_to": "00-00-00-00-00-00",
      "ap_type": "ap7532",
      "config_channel": "smt",
      "config_power": "smt",
    }
  ]
}
```

```

    "current_channel": "11",
    "current_power": 18,
    "device_mac": "84-24-8D-15-E5-D8",
    "expire_at": 2769507,
    "hostname": "ap-zzz",
    "last_update": 2769417,
    "location": "",
    "max_user_rate": 216600,
    "num_clients": 0,
    "protocol": 22,
    "radio_alias": "ap-zzz:R1",
    "radio_id": "84-24-8D-15-E5-D8:R1",
    "radio_mac": "FC-0A-81-A3-1C-C0",
    "radio_num": 1,
    "radio_type": "2.4GHz-wlan",
    "rf_domain_name": "SITE-1",
    "state": "On"
  },
  {
    "adopted_to": "00-00-00-00-00-00",
    "ap_type": "ap7532",.....
    "state": "Off"
  }
],
"return_code": 0
}

```

Table 10: Response Definitions

Response Item	Description	Data Type
adopted_to	MAC address of the parent wireless controller that the access point is adopted to	mac-address
ap_type	Type of access point	String
config_channel	Configured channel for this radio	String (length 64)
config_power	Configured power for this radio	String (length 16)
current_channel	Current channel this radio is operating at	String (length 16)
current_power	Current power (in dBm) that this radio is operating at	Integer
device_mac	Device MAC address	mac-address
expire_at	Expiry time	Integer
hostname	Host name of the access point device	String (length 64)
last_update	Last Update time	Integer
location	Location description of the access point device	String (length 64)
max_user_rate	Theoretical maximum user-level data rate in kbps	Integer
num_clients	Number of clients associated with this radio	Integer

Table 10: Response Definitions (continued)

Response Item	Description	Data Type
protocol	Bit flag for supported protocol (bit4=11GN, bit3=11AN, bit2=11G, bit1=11B, bit0=11A)	Integer
radio_alias	Radio ID alias of its associated radio in the form of hostname:R%d. E.g., rfs4000-22A24E:R1	String
radio_id	Unique ID for its associated radio in the form of AP-MAC:R%d. E.g., 00-A0-F8-00-00-00:R1	String
radio_mac	MAC address of this radio interface	mac-address
radio_num	Radio number of the radio interface	Integer
radio_type	802.11 radio-type of the client	Enumeration [11a, 11b, 11g, 11an, 11bn, 11ac]
rf_domain_name	RF-domain name of the AP device	String (length 64)
state	Current radio state	String (length 64)