

# **Deployment Guide**

## **RUCKUS WAN Gateway – REST API**

June 2023 Rev. 0



### **Table of Contents**

Changes	3
INTENDED AUDIENCE	3
THE RWG REST API	4
The API Documentation	4
The API Browser	4
EXAMPLES	6
API Authentication	6
Postman Collection	6
Example 1 – Create Subnet Filter Rule	7
Example 2 – Read Subnet Filter Rules	7
Example 3 – Edit Subnet Filter Rule	8
Example 4 – Delete Subnet Filter Rule	8
CONCLUSION	9



#### Changes

• No changes – initial document version

### **Intended Audience**

The audience for this document is System Engineers and developers who needs to control RWG programmatically or develop applications using REST API. It is expected that the reader possesses a working knowledge on REST API and the RUCKUS WAN Gateway.

For more information on how to configure RUCKUS products, please refer to the appropriate RUCKUS user guide available on the RUCKUS support site at <a href="https://support.ruckuswireless.com/">https://support.ruckuswireless.com/</a>

The RWG documentation is embedded into the product. You can access the embedded documentation at <u>https://{your RWG\_IP\_address}/admin/manual/help\_online</u>



### The RWG REST API

#### **The API Documentation**

RWG offers a very rich REST API using CRUD calls to manage every operation. The RWG nodes come with a builtin API documentation. To see the API documentation, navigate to <u>https://{RWG\_FQDN}/rdoc/</u>

Click on any class at the left menu to see its details:

Home Pages Classes Methods	class Access	Point	
Search			
	0.1		
	Schema Informati	on	
Pages	Tabla anno anno aciata		
README FOR APP	Table name: access_points		
in Britting, Gright 1			
	id	:integer	not null, primary key
Class and Module Index	approved	:boolean	
Delet	channel_24	:integer	
AccessPoint	channel_5	:integer	
ccessPointProfile	client_count	:integer	
ccessPointProfilesRadioProf	color	:string	default("#ff0000")
ccessPointRadio	connection_state	:string	
ccessPointRadioProfile	country	:string	
ccessPointZone	created_by	:string	
	description	:string	
ccount	in	.string	
ccountGroup	last seen at	:datetime	
ccountTrafficRate	latitude	:decimal(8, 6)	
ccountingGraph	location	:string	
ctiveRecordModel	longitude	:decimal(9, 6)	
ctiveStorageAttachment	mac	:string	
ActiveStorageBlob	mesh	:string	
Iddrees	model	:string	
dmin	name	:string	
	note	:text	
amin::ApiController	online	:boolean	default(FALSE)
dmin::Scattolds	piti_error	text	
dmin::Scaffolds::AccountsController	serial number	etring	derautt([]), is an Array
Admin::Scaffolds::DeviceOptionsController	undated by	-string	
dmin::Scaffolds::DevicesController	untime	integer	

FIGURE 1 – API DOCUMENTATION

#### **The API Browser**

You can also browse the API directly by navigating to <u>https://{RWG\_FQDN}/api</u>. Click on any line to drill down into a class, execute an API call or see the API call schema.

rXg API							
api							
Root		OPTIONS GET					
GET /api							
HTTP 200 OK Content-Type: text/html; charset=	itf-8						
.json .xml							
<pre>{     "message": "This is the root of the rXg API     "authentication": "To authenticate, you can } Routes Raw Form</pre>	", either login via the admin m	web UI, or you can pass your admin's API token with an `api_key` URL paramete					
Path	Verb	Controller#Action					
api	GET	api/root#root					
api	OPTIONS	api/root#options					
api/register_cluster_node	POST	api/root#register_cluster_node					
api/whoami	GET	api/root#whoami					
api/access_point_profiles							
api/access_point_profiles_radio_profs							
api/access_point_radio_profiles							
api/access_point_radios							

FIGURE 2 – API BROWSER



The lines with a hyperlink indicate an API call that can be executed directly. You need to pass an API key in the URL as a parameter, or get an authentication session for the API call to work. The simplest way is to open another browser tab and authenticate using the RWG UI.

api/active_storage_blobs							
api/addresses							
api/addresses GET api/addresses#index							
api/addresses	POST	api/addresses#create					
api/addresses	OPTIONS	api/addresses#options					
api/addresses/help GET a		api/addresses#help					
api/addresses/send_garp	api/addresses#send_garp						
api/addresses/:id	GET	api/addresses#show					
api/addresses/:id	PUT	api/addresses#update					
api/addresses/:id	PATCH	api/addresses#update					
api/addresses/:id	DELETE	api/addresses#destroy					
api/admin_controller_acls							
api/admin_logs							

FIGURE 3 – HYPERLINKS INDICATE EXECUTABLE API CALLS

Once you are authenticated, you can use **GET** and **OPTIONS**. If you click GET, you will run the GET call for the class. If you click OPTIONS, you will see the API schema for all methods in the class. Here is an example using the endpoint api/addresses:

api / addresses	api / addresses
Addresses OPTIONS GET	Addresses Options Get
GET /api/addresses	OPTIONS /api/addresses
HTTP 200 OK Content-Type: text/html; charset=utf-8	HTTP 200 OK Content-Type: text/html; charset=utf-8
.json .xml	.json .xml
<pre>{     "count": 9,     "page": 1,     "page_size": 30,     "total_pages": 1,     "results": [     {         " id": 2,         "name": "WAN",         "span": 1,         "note": null,         "created_at": "2022-08-11T12:45:32.812-07:00",         "created_by": "ruckus",         "updated_at": "2022-08-11T12:45:32.812-07:00",         "created_by": "ruckus",         "updated_by": "ruckus",         "updated_by": "ruckus",         "autoincrement": 1,         "primary": true,         "scratch": null,         "created_by": "ruckus",         "autoincrement": 1,         "primary": true,         "scratch": null,         "created_by": "log.166.11.5/24",         "interface": ['d": 1, "name": "igb0"},         "nat_assignments": [], </pre>	<pre>{     "title": "Addresses",     "renders": ["text/html", "application/json", "application/xml"],     "actions": {         "index": {             "path": "",             "methods": "get",             "type": "builtin",             "methods": "[nabel": "Index"}         },         "create": {             "path": "",             "methods": "gott",             "type": "builtin",             "methods": "options",             "path": "",             "methods": "options",             "type": "builtin",             "methods": "options",             "type": "builtin",             "methods": "options",             "type": "builtin",             "methods": "options",             "type": "builtin",             "methods": "options",             "type: "builtin",             "methods": "options",             "methods": "options",</pre>





### **Examples**

#### **API Authentication**

Every API call needs to include an API key as a parameter in the URL. You can use the **/api/login** endpoint to obtain an API key. The body contains the credentials for a valid RWG administrative account.

The API keys are valid for one week.

POST		~	https://{	{host}}/api/logi	n?api_key={	(API_KEY	·}}					
Param	ne (	Autho	orization -data 🌑	Headers (7) x-www-form-u	Body •	Pre-re	equest Scrip	•t • Tests • • GraphQL	Settings			
1 2 3 4	{ · · }	use pas	rname": sword":	"mmolinari", "password"								
Body Prett	Cook y	ies H Raw	leaders (16 Previe	6) Test Result w Visualiz	ts re <b>JSO</b> I	N ~	위			æ	Status: 200 OK	Time: 354 ms
1 2 3 4 5	{ }	"api "nam "exp	_key": " e": "9ac iration"	hNMoYUvVPorY a5c9b-2f8a-4 : "2023-04-2	TPxuyQe6Z 995-9c2c- 1T16:45:0	LB2HHx9 37e5579 7.874-0	pwMgzru1R cf5c6", 7:00"	6atgBQ4aSxYf	NgyWQ4wG8`	Yvh	UnG93PhiS9N3q	kE8Zov",

FIGURE 5 – GET THE API KEY

#### **Postman Collection**

A postman collection with a sample of API calls is available at <u>https://github.com/commscope-ruckus/RUCKUS-RWG-Postman</u>

✓ RWG ★	
> 🗎 Administration	POST v https://{{host}}/api/subnets_filters?api_key={{API_KEY}}
> 🗎 VLAN interfaces	Params   Authorization Headers (7) Body   Pre-request Script   Tests Settings
Hetwork addresses     Hetwork DHCP pools	🌑 none 🌑 form-data 🌑 x-www-form-urlencoded 🕘 raw 🌑 binary 🔘 GraphQL JSON 🗸
✓  ☐ Packet filter	1 (
GET get wan targets	2 ····"name": "block-CNN", 3 ····"wan target ids": 1
GET get wan target by ID	4 29
GET get wan target by name	5 ····],
GET get policies	7
GET get subnet filter rules	81 9 }
POST create wan target - subnet 40	Both Cooking Headers (15) Test Desults (P. Status 20
POST create wan target - subnet 50	
POST create subnet filter rule	Pretty Raw Preview Visualize JSON V 🛱
PATCH update subnet subnet rule	1 🕅
DEL delete subnet subnet rule	2 "id": 10, 3 "name": "block CNN".
POST create wan target using XML - ol	4 "note": null,
> 🗎 IPsec VPN	<pre>5 "created_by": "mmolinari", 6 "updated_by": "mmolinari",</pre>
POST login	7 "created_at": "2023-04-14T16:59:32.100-07:00",

FIGURE 6 – POSTMAN COLLECTION FOR RWG



### **Example 1 – Create Subnet Filter Rule**

arams  Authorization Headers (7) Body  Pre-request Script  Tests Settings	Pretty Raw Preview Visualize JSON V
<pre>none torm-data *-www-torm-unencoded raw binary GraphQL JSON *  1 { 2 *** "name": "block CNN", 3 *** "wan_target_ids": [ 4 *** *** 29 5 *** ], 6 **** "policy_ids": -[ 7 **** *** 1 8 *** ] 9 } </pre>	<pre>2 "id": 10, 3 "name": "block CNN", 4 "note": null, 5 "created_by": "mmolinari", 7 "updated_by": "molinari", 7 "created_at": "2023-04-14T16:59:32.100-07:00", 8 "updated_at": "2023-04-14T16:59:32.100-07:00", 9 "scratch": null, 10 "wan_targets: [ 11 { 12 { 13   14 } 15 ], 16 "wan_targets.count": 1, 17 "policies": [ 18 { 19 { 10 "id": 1, 20   10 "id": 1, 21 } 22 ], 23 "policies.count": 1</pre>

FIGURE 7 – CREATE SUBNET FILTER RULE

### **Example 2 – Read Subnet Filter Rules**



FIGURE 8 - READ SUBNET FILTER RULE



#### **Example 3 – Edit Subnet Filter Rule**

The endpoint includes the filter ID, and the body includes the policy ID.

PATCH v https://{{host}}/api/subnets_filters	Body Cookies Headers (16) Test Results
PATCH       https://{{host}}/api/subnets_filters       Image: Subnets_filters         Params       Authorization       Headers (7)       Body       Pre-request Script       Tests       Settings         Image: Subnet Su	Body       Cookies       Headers (16)       Test Results         Pretty       Raw       Preview       Visualize       JSON ~ =>         1       "iname": "block CNN",       "note": null,       "created_by": "mmolinari",         5       "created_by": "mmolinari",       "updated_by": "mmolinari",         7       "created_at": "2023-04-14T16:59:32.100-07:00",         8       "updated_at": "2023-04-14T16:59:32.100-07:00",         9       "scratch": null,         10       "wan_targets": [         12       { "id": 29,         13       "name": "CNN"
	15     ],       16     "wan_targets.count": 1,       17     "policies": [       18     {       19     ["id":7,]       20     ["name": "ICX 7150-C-12"]       21     ]       22     ],       23     "policies.count": 1       24     ]

FIGURE 9 – EDIT SUBNET FILTER RULE

#### **Example 4 – Delete Subnet Filter Rule**

The endpoint includes the filter ID.

DELETE	~	https://{{host}}/api/subnets_filters 102 api_key={{API_KEY}}									
Params •	Autho	prization H	leaders (5)	Body	Pre-r	eques	t Script •	Tests	Settings		
Body Cod	okies H	eaders (12)	Test Results							æ	Status: 204 No Content
Pretty	Raw	Preview	Visualize	Text	$\sim$	<del>-</del> 2					
1											

FIGURE 10 – DELETE SUBNET FILTER RULE

## RUCKUS solutions are part of CommScope's comprehensive portfolio for Enterprise environments (indoor and outdoor).

We encourage you to visit commscope.com to learn more about:

- RUCKUS Wi-Fi Access Points
- RUCKUS ICX switches
- SYSTIMAX and NETCONNECT: Structured cabling solutions (copper and fiber)
- imVision: Automated Infrastructure Management
- Era and OneCell in-building cellular solutions
- Our extensive experience about supporting PoE and IoT

#### www.ruckusnetworks.com

Visit our website or contact your local RUCKUS representative for more information.

© 2023 CommScope, Inc. All rights reserved.

All trademarks identified by  $^{\rm TM}$  or  $^{\otimes}$  are trademarks or registered trademarks in the US and may be registered in other countries. All product names, trademarks and registered trademarks are property of their respective owners.

