Barracuda Load Balancer ADC
REST API Version 2

Barracuda Networks Inc.
3175 S. Winchester Blvd
Campbell, CA 95008
http://www.barracuda.com

Contents

Barracuda Load Balancer ADC REST API 1
Availability 1
Version 1
Revision History 1
Change Log 2
Accessing the API 4
Authentication 4
Logging Out 5
Request Format 6
Response Format 6
Resources 7
Resource Descriptions 70
- Virtual Service Groups 70
  - Create a Service Group 70
  - Rename a Service Group 71
  - Retrieve Service Groups 72
  - Delete a Service Group 73
- Virtual Services 73
  - Create a Service 73
  - Update a Service 75
  - Retrieve Services 76
  - Delete a Service 78
  - Virtual Service Parameters 79
- Certificates 101
  - Generate a Self-Signed Certificate 101
  - Upload a Signed Certificate 103
  - Upload a Trusted Certificate 105
  - Download a Certificate 106
- Servers 107
  - Create a Server 107

Update a Server 108
Retrieve Servers 109
Delete a Server 111
Server Parameters 111
Content Rules 129
Create a Content Rule 129
Update a Content Rule 130
Retrieve Content Rules 131
Delete a Content Rule 133
Content Rule Parameters 133
Rule Group Servers 139
Create a Rule Group Server 139
Update a Rule Group Server 140
Retrieve Rule Group Servers 141
Delete a Rule Group Server 143
Rule Group Server Parameters 144
Security Policies 156
Create a Custom Security Policy 156
Update a Security Policy 157
Retrieve Security Policies 158
Delete a Security Policy 161
Security Policy Parameters 161
Global ACLs 167
Create a Global ACL Rule 167
Update a Global ACL Rule 169
Retrieve Global ACL Rules 170
Delete a Global ACL Rule 172
Action Policy 173
Retrieve the Attack Actions for an Attack Group 173
Update an Attack Action 174
Attack Action Parameters 175
Data Theft Protections 177
Create a Custom Data Theft Element 177

Update a Data Theft Element 178
Retrieve Data Theft Elements 178
Delete a Data Theft Element 180
Data Theft Element Parameters 180
Monitor Groups 182
Create a Monitor Group 182
Retrieve Monitor Groups 183
Delete a Monitor Group 184
Monitors 184
Create a Monitor 184
Update a Monitor 185
Retrieve Monitors 186
Delete a Monitor 187
Monitor Parameters 188
Perl Implementation 196
Barracuda::Rest:API Dependencies 196
Install the Barracuda::Rest:API Module 196
Perl Module Methods 197
new 197
login 198
logout 199
create 199
list 200
get 200
update 201
remove 201
Object Descriptions 202
Virtual Service Groups 202
Virtual Services 203
Certificates 205
Servers 209
Content Rules 211
Rule Group Servers 213

Security Policies 215
Global ACLs 216
Action Policy 219
Data Theft Protections 220
Monitor Groups 223
Monitors 224
Sample Scripts 233
HTTP Scripts 233
HTTPS Scripts 233
Example Perl Code 234

Barracuda Load Balancer ADC REST API

The Barracuda Load Balancer ADC Representational State Transfer (REST) Application Programming Interface (API) lets you remotely administer and configure your Barracuda Load Balancer ADC. In general, the API is useful for performing large configurations by automating the manual configuration available on the web interface. For example, you can use the REST API to create services.

You identify resources by their URIs and use HTTP methods to send requests to the Barracuda Load Balancer ADC. Then your application parses the response, which is always returned using JavaScript Object Notation (JSON). You can use any programming language to interact with the API. Code examples in this reference guide are written in Curl. If you are using Perl, see the Perl Implementation section of this guide.

If you have any questions after reading this API guide, please contact Barracuda Networks Technical Support at +1-408-342-5400 or email support@barracuda.com.

Availability

The REST API is available for all Barracuda Load Balancer ADC models running firmware version 5.1 or version 5.2.

Version

The most recent version of the Barracuda Load Balancer ADC REST API is version 2. The REST API version is separate from the firmware version for the appliance. For a list of the changes that are included in the current API version, see the Change Log section.

Revision History

Date	Revision	Description
January 6, 2015	1	Published guide for the Barracuda Load Balancer ADC REST API version 2 to the Barracuda TechLibrary.
May 29, 2015	2	Resolved errors and added SNAT rules.

Change Log

The following changes are included in Barracuda Load Balancer ADC REST API version 2:

Virtual Services

- Added virtual services group object.
- Introduced POST, GET, PUT, and DELETE for Virtual Service Groups.
- Added **`protocol`** parameter.
- Added parameters for Instant SSL services.
- Added caching parameters.
- Added compression parameters.
- Added **`enabled`** parameter and **`passive`** mode to **`security`** object.
- Added **`load_threshold`** and **`polling_interval`** parameters.
- Replaced **`load_url`** and **`snmp_community_string`** with **`adaptive_param`**.
- For **`service_monitor`**, changed **`test_method`** to **`test`**.
- For **`service_monitor`**, changed **`status_code`** to **`code`**.
- Added **`post_body`** field to **`service_monitor`**.
- Added **`sni`**, **`sni_strict`**, **`sni_domain`**, **`sni_certs`**, **`ciphers`**, **`cipher_list`**, **`enable_client_authentication`**,**`enforce_client_certificate`**,and **`trusted_certificates`** to **`ssl_offloading`**. Also changed **`enable_tsl*`** to **`ssl_enable_tls*`**.
- Added **`client_ip_addr_header`**, **`ignore_case`**, and **`keepalive_request`** fields.
- Added **`ftp_pasv_config`** options.
- Changed **`security.passive`**to **`security.mode`**.
- Removed **`round_robin`** option from **`lb_policy`**.
- Added **`tcp_keepalive_timeout`**to enable keepalive probes.
- Changed **`weighted_least_requests`** to **`weighted_least_connection`**.
- Added **`last_resort_action`** under Load Balancing for Virtual Services.
- Changed **`ssl_offloading.sni_domain`** to **`ssl_offloading.sni_domains`**.
- Added **`sni_ecdsa_certs`**, **`ecdsa_status`**, and **`ecdsa_certificate`**.
- Added **`ignore_expect_header`**.

SNAT Rules

- Introduced GET, POST, PUT, and DELETE for SNAT rules.

Content Rules

- Changed server URI to have **`virtual_service_group`** before **`virtual_service`** for content rules.

Security Policies

- Introduced POST, GET, PUT, and DELETE for Security Policies.

Global ACLs

- Introduced POST, GET, PUT, and DELETE for Global ACLs.

Action Policy

- Introduced GET and PUT for Action Policy.

Data Theft Protection

- Introduced POST, GET, PUT, and DELETE for Data Theft Protections.

Accessing the API

You can invoke REST API calls on the Barracuda Load Balancer ADC in either HTTP or HTTPS. For simplicity, this guide mainly provides examples of HTTP requests. In HTTP and HTTPS requests, specify the port number through which the Barracuda Load Balancer ADC web interface is accessed. The Barracuda Load Balancer ADC REST API endpoints for making calls are:

http://<IP address of ADC>:<port of ADC>/restapi/v2/
https://<IP address of ADC>:<port of ADC>/restapi/v2/

where:

- **http|https** – The protocol used to invoke a REST API call.
- **IP address of ADC** – The IP address of the Barracuda Load Balancer ADC.
- **port of ADC –** The web interface port number on the **BASIC > Administration** page.
- **restapi** – The name of the API.
- **v2** – The current REST API version.

In HTTPS requests, the Barracuda Load Balancer ADC uses the same certificate as the web interface, as specified in the **ADVANCED > Secure Administration > SSL Certificate Configuration** section. The common name in the certificate must match the URL to where you are sending the request. For example, if the URL is https://ADC1.com:443, then the common name must be ADC1.com. Download the certificate from the **ADVANCED > Secure Administration > Private** section, and copy it to the client machine where you are executing the REST API calls.

Combine the endpoint with the appropriate resource URI to make a call. For example, to create a virtual service group for a Barracuda Load Balancer ADC at http://10.11.19.104:8000, the REST API call is:

```
http://10.11.19.104:8000/restapi/v2/virtual_service_groups
```

For a quick reference of the URIs and supported methods for all Barracuda Load Balancer ADC REST API resources, see the Resources section. For detailed descriptions, list of parameters, and request examples for each resource, see the Resource Descriptions section.

Authentication

To access the Barracuda Load Balancer ADC through the REST API, send a login request with your username and password to authenticate yourself. In response, you receive an access token that you must include in every subsequent request.

HTTPS Login Request Example

The following example displays a login request and response for HTTPS, where:

- **cacert** – The parameter to verify the associated certificate.
- **/tmp/ssl_private_cert.pem** – The path and name of the certificate.

Request:

```
$ curl -X POST \
   -H "Content-Type:application/json" \
   -d '{"username": "admin", "password": "admin"}' --cacert/tmp\
   /ssl_private_cert.pem \
   https://10.11.19.104:443/restapi/v2/login
```

Response:

```
{"token":"eyJldCI6IjEzODAyMzE3NTciLCJwYXNzd29yZCI6ImY3NzY2ZTFmNTgwMzgyNmE1YTAzZ
WZlMzcy\nYzgzOTMyIiwidXNlciI6ImFkbWluIn0=\n"}
```

HTTP Login Request Example

The following example displays an example of a login request and response for HTTP:

Request:

```
$ curl -X POST \
   -H "Content-Type:application/json" \
   -d '{"username": "admin", "password": "admin"}' \
   http://10.11.19.104:8000/restapi/v2/login
```

Response:

```
{"token":"eyJldCI6IjEzODAyMzE3NTciLCJwYXNzd29yZCI6ImY3NzY2ZTFmNTgwMzgyNmE1YTAzZ
WZlMzcy\nYzgzOTMyIiwidXNlciI6ImFkbWluIn0=\n"}
```

Logging Out

Send a logout request to delete the token that was generated for you.

Example Logout Request

```
Request:

$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/logout

Response:
{
    "msg": "Success"
}
```

Request Format

For authentication, every request that you make must include the token that you receive after sending a **Error! Reference source not found.**, followed by a colon (:). You do not need to specify your username and password because they are embedded in the token.

Unless you are generating or uploading a certificate, the body of your POST and PUT requests must be JSON with the Content-Type header set to **application/json**.

For example, see the following request to create a service named **demo_service**:

```
$ curl -X POST -H "Content-Type:application/json"
   -d '
   {"name": "demo_test", "netmask": "255.255.255.0", "interface":"ge-1-1",
"ip_address": "199.9.9.9", "port": "80", "type": "HTTP","address_version":
"ipv4"}

   '
   -u
'eyJldCI6IjE0MTA4ODc3MDgiLCJwYXNzd29yZCI6IjdmNDMzNjk4ZTRjZDM0YTIxZTQ4ZDkxOTE3\n
YjllNmY1IiwidXNlciI6ImFkbWluIn0=\n:'
http://10.66.44.13:8000/restapi/v2/virtual_service_groups/default/virtual_servi
ces
```

Response Format

All responses are returned as JSON. GET requests return resource data as JSON.

For POST, PUT, and DELETE requests, the message in the response indicates success or failure.

- If a request succeeds, the response includes a message with a short description of the operation. For example, see the following response to a request to delete a resource:

```
{
    "msg": "Successfully deleted",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

- If a request fails, an error message is returned with a description of the failure and an HTTP status code. For example, see the following response to a request that was sent with an incorrectly spelled resource URI:

```
{
    "error": {
        "msg": "Invalid URI",
        "type": 404
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Resources

A quick reference for the URIs and supported methods for all Barracuda Load Balancer ADC REST API resources are provided in the following tables. For detailed descriptions, list of parameters, and request examples for each resource, see the Resource Descriptions section.

Virtual Service Groups

URI	Method	Functionality
/virtual_service_groups	POST	Create a Service Group.
	GET	Retrieve Service Groups.
/virtual_service_groups/*<group name>*	PUT	Rename a Service Group.
	GET	Retrieve Service Groups.
	DELETE	Delete a Service Group.

Virtual Services

URI	Method	Functionality
/virtual_service_groups/*<group name>*\ /virtual_services	POST	Create a Service.
	GET	Retrieve Services.
/virtual_service_groups/*<group name>*\ /virtual_services/*<service name>*	PUT	Update a Service.
	GET	Retrieve Services.
	DELETE	Delete a Service.

Virtual Service Parameters

The tables in this section list the parameters that you can configure for services.

Service Configuration

Parameter	Data Type	Description
name	String	The name of the service.
address_version	Enum	The Internet protocol version of the service. Possible values: • **ipv4** • **ipv6**

Parameter	Data Type	Description
ip_address	String	The virtual IP address. The IP address format depends on the specified **address_version**.
port	Integer	The port number for the service.
type	Enum	The type of the service. Possible values include: • **HTTP** • **HTTPS** • **INSTANTSSL** • **FTP** • **FTPSSL** • **UDP** • **L4** • **L7UDP** • **L7Tcp RDP**
netmask	String	The netmask depends on the **address_version** specified.
interface	Enum	The interface for the service. The value depends on the appliance. For example, ge-1-1, ge-1-2, and ge-2-1.
service_hostname	String	The domain name to identify and rewrite HTTP requests to HTTPS. **Conditional:** Required only for Instant SSL services.
certificate	String	The certificate that is presented by the service when it authenticates itself to a browser or other client. **Conditional:** Required only for HTTPS, Instant SSL, FTP SSL, and SSL services.
redirect_port	Integer	The HTTP redirect port for an Instant SSL service.
enable	Boolean	Enable the service. Possible values: • **true** • **false**
auto_recover	Boolean	Automatically re-enables the real servers after they are detected as unavailable. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enable_access_log	Boolean	Logs every request made to this service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Parameter	Data Type	Description
session_timeout	Integer	The time-out period in seconds for persistent connections with clients. Zero (**0**) indicates that the session never times out (session lives forever).
comments	String	Description about the service.

Load Balancing

Parameter	Data Type	Description
lb_policy	Enum	How traffic is distributed among the real servers associated with the service. Possible values: • **weighted_round_robin** • **weighted_least_connection**
lb_adaptive_scheduling	Enum	Whether the weight of each real server is adjusted according to its CPU usage, the number of active Terminal sessions it has, or from information polled at the LOAD URL. Possible values: • **None** – Uses the static preconfigured real server weights. Past connections / UDP datagrams are not considered when directing clients to servers. • **SNMP_CPU** – Polls the SNMP OID for CPU load and manipulates the real server weights accordingly. To use this option, real servers must allow the Barracuda Load Balancer ADC SNMP access to the public community. **Dependent variables:** ○ adaptive_param ○ snmp_cpu_load_oid • **LOAD_URL** – Polls a specified URL, expecting the output to look like LOAD=23 (showing the load as an integer between 0 and 100). For this method to work, each real server must be running a web server that responds to the poll at the real server's IP address and port 80. **Dependent variable:** ○ adaptive_param • **SNMP_TS_SESSIONS** – Redistributes connections between Windows Terminal Servers based on the number of sessions per server. **Dependent variable:** ○ adaptive_param

Parameter	Data Type	Description
adaptive_param	String	**Conditional:** The value of this parameter differs for each adaptive scheduling type: • If the **lb_adaptive_scheduling** is **SNMP_CPU** or **SNMP_TS_SESSIONS**, enter the community string. • If the **lb_adaptive_scheduling** is **LOAD_URL**, enter the URL to poll the server for information.
load_threshold	Integer	The maximum acceptable real server load. If this load limit is exceeded, the weight of the real server is adjusted according to the adaptive scheduling policy.
last_resort_action	Enum	The action to take if all Real Servers configured within a Load Balancer ADC service become unavailable. The following are the possible values: • **default** – Varies depending on the service type: ○ For Layer 4 TCP and UDP services, the client will fail to connect. ○ For Layer 7 services, a 503 default error page is sent to the client. • **reset_conn** – Forces a reset of the connection to the Real Server. Select this option if you do not have a backup server configured for the service. When the Real Servers become unavailable, either the connection is closed (TCP) or an ICMP port unreachable error is returned to the client (UDP).
polling_interval	Integer	The interval between test start times, and also the maximum time that tests are given to complete, in seconds, for adaptive scheduling.
snmp_cpu_load_oid	String	The CPU Load OID to use for SNMP CPU adaptive scheduling.

Instant SSL

To configure an Instant SSL service, you must configure the following parameters for the service.

Parameter	Data Type	Description
instantssl.status	Boolean	Rewrites `http links` in responses to https. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Parameter	Data Type	Description
instantssl.host	String	The domains that are rewritten from `http to https in` responses, if instantssl`.status` is **`true`**. To include all domains, enter an asterisk (*). For example, if you enter www**`.example.com`**, any instances of **`http://www.example.com in`** outgoing responses are rewritten to **`https://www.example.com`**. Ensure that the certificate that you upload for this service is valid for the secure site domain.
instantssl.sharepoint_support	Boolean	Enables SharePoint rewrite support. Normally, an Instant SSL service rewrites http links in responses to https using HTML tags, like href. But SharePoint applications also insert hyperlinks outside of the basic HTML tags. To rewrite those links to https, set value to **`true`**. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Caching

Caching is available for only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
cache.enabled	Boolean	Enables caching to local memory for quick retrieval so some requests do not have to be sent to the web server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.file_extension	Array	A list of the extensions of the file types to be cached.
cache.expiry_age	Integer	The default expiration age in minutes, to be used when the response does not have an Expires header. A value of 0 (expiry-age = 0) sets the expiry age to 60 minutes.
cache.max_objsize	Integer	The maximum size of files that can be cached.
cache.min_objsize	Integer	The minimum size of files that can be cached.
cache.req_cachehdrs_ignore	Boolean	Ignores HTTP request cache-control headers to instruct upstream caching servers. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true

Parameter	Data Type	Description
		• **0** – false
cache.resp_cachehdrs_ ignore	Boolean	Ignores HTTP response cache-control headers to instruct upstream caching servers. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.negative_responses	Boolean	Caches HTTP negative responses with status codes 204, 305, 404, 405, 414, 501, 502, and 504. Only negative responses with the Expires or Last **`Modified headers`** are cached. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Compression

Compression is available for only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
compress.enabled	Boolean	Enables compression of the specified content types. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
compress.content_types	Array	A list of content types to be compressed. For example: **`text/css`** **`text/html`** **`text/js`** **`text/plain`**
compress.min_obj_size	Integer	The minimum size of objects that can be compressed, from 1 to2147483646 in bytes. The default value is 8192.

Service Monitoring

Parameter	Data Type	Description
service_monitor.test	Enum	The test that is used by the server monitor to determine the availability of the servers that are associated with the service. You can use either a monitor group or a test type.

Parameter	Data Type	Description

		To specify a Error! Not a valid result for table., use the following syntax:
		`GROUP_<group name>`
		For example, to use a monitor group named `Group 1`:
		`"service_monitor":{"test":"GROUP_Group1 "}`
		
		To select a test type, select one of the following values:
		○ **ICMP_PING** – Performs a PING test.
		○ **TCP_PORT_CHECK** – Validates that the configured service port is open.
		Dependent variable:
		○ service_monitor.port
		• **UDP_PORT_CHECK** – Sends a 0 byte datagram to the IP address and port of the real server to verify that the UDP port is open. Waits to receive an ICMP Port Unreachable message to determine the result. If there is a firewall prevents outbound ICMP messages, the port is assumed to be open.
		Dependent variable:
		○ service_monitor.port
		• **HTTP_TEST** – Sends an HTTP request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled.
		Dependent variables:
		○ service_monitor.port
		○ service_monitor.target
		○ service_monitor.match
		○ service_monitor.method
		○ service_monitor.hosts
		○ service_monitor.code
		• **SIMPLE_HTTP** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code.
		Dependent variables:
		○ service_monitor.port
		○ service_monitor.target
		○ service_monitor.match
		○ service_monitor.method
		○ service_monitor.hosts
		○ service_monitor.code
		• **HTTP_SLOW** – Sends an HTTP request to the

Parameter	Data Type	Description
		specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_TEST** – Sends an HTTPS request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_SLOW** – Sends an HTTPS request to the specified relative URL to verify that the response contains that the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTPS** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target

Parameter	Data Type	Description
		○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **DNS_TEST** – Sends a DNS query and compares the returned IP address to the entered IP address. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **IMAP_TEST** – Simple test for IMAP service. If no username and password are provided, this test verifies only the availability of the service on the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **POP_TEST** – Simple test for POP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SMTP_TEST** – Simple test for SMTP servers. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SNMP_TEST** – Sends an SNMP GET to the specified OID to verify that the response contains an expected pattern. If an OID is not specified, this test verifies only the availability of the service on the server. **Dependent variable:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SIP_TEST** – Simple test for SIP service. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service. **Dependent variable:** ○ service_monitor.port

Parameter	Data Type	Description
		• **SIP_TLS_TEST** – Simple Test for SIP service over TLS. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service over TLS. **Dependent variable:** ○ service_monitor.port • **LDAP_AD_TEST** – Bind test for LDAP/AD service. If no username and password are provided, the LDAP/AD test verifies availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **LDAP_AD_SSL_TEST** – Bind test for LDAPS/AD service. If no username and password are provided, the LDAP/AD test verifies availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SPAMFIREWALL_TEST** – Test for use with Barracuda Spam Firewalls. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FORCED_PORT_HTTP** – (Specific HTTP Port test) Sends an HTTP GET request using the specified port to a relative URL on the real server, to verify that the retrieved HTML contains an expected pattern. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_TEST** – Verifies the availability of a RADIUS server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match

Parameter	Data Type	Description
		• **RADIUS_ACCT** – Tests the availability of a RADIUS server by making an accounting request. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RDP_TEST** – Attempts an RDP connection to each real server to verify the availability of the Terminal Service. **Dependent variable:** ○ service_monitor.port • **FTP_TEST** – Attempts a TCP connection to each real server to check FTP availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FTPS_TEST**– Attempts a TCP connection to each real server to verify FTPS availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SFTP_TEST** – Test for FTP over SSH. **Dependent variables:** ○ service_monitor.port ○ service_monitor.username ○ service_monitor.password • **NTLM_TEST** – Simple test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.username ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **NTLMS_TEST** – Secure test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.username ○ service_monitor.password ○ service_monitor.target

Parameter	Data Type	Description
		○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **ALWAYS_PASS** Used to troubleshooting or for services with management access to real servers. This test always passes regardless of the condition of the real server.
service_monitor.port	Integer	By default, tests use the configured real server port for the service unless the real server port is set to ALL. In that case, tests use the default port for the test type (e.g., SMTP = 25, HTTP = 80, DNS = 53, HTTPS = 443, IMAP = 143, POP = 110, and SNMP = 161). Enter a port number to override the default port.
service_monitor.username	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The username for the SSH account. • MS SharePoint – The username for the SharePoint service.
service_monitor.password	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The password for the SSH account. • MS SharePoint – The username for the SharePoint service.
service_monitor.target	String	**Conditional:** The value for this parameter differs for each test type: • Barracuda Spam Firewall Test – The domain for the mail server. • DNS – The fully qualified domain name. • FTP – The username to log into server. The default username is **`anonymous`**. • FTPS – The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service on the server. • HTTP and HTTPS – The complete URL (starting with **`http`** or **`https`**). • HTTP Slow and HTTPS Slow – The complete URL (starting with **`http`** or **`https`**). • IMAP and POP3 – (Optional) The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service on the server. • LDAP and LDAPS – (Optional) The username with full LDAP schema.

Parameter	Data Type	Description
		Note: If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The root relative URL. For example: **`/cgi-bin/index.cgi`** • RADIUS Auth and RADIUS Acct – The secret to use with the RADIUS server. • Simple HTTP and Simple HTTPS – The root relative URL. For example: **`/cgi-bin/index.cgi`** The actual URL used is **`https://<real_server_ip>:<port><URL>`** • SNMP – (Optional) Enter a valid SNMP OID. **Note:** If an OID is not specified, this test verifies only the availability of the service in the server. • Specific HTTP Port – The TCP port followed by colon (:) and the root relative URL. For example: **`8080:/cgi-bin/index.cgi`**.
service_monitor.match	String	**Conditional:** The value for this parameters differs for each test type: • Barracuda Spam Firewall – (Optional) A pattern that is expected in the banner of the SMTP server. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. • DNS – The IP address of the hostname. • FTP – The password to log into the server. The default password is **`anonymous`**. • FTPS – The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service on the server. • HTTP and HTTPS – The expected pattern in the retrieved HTML. • HTTP Slow and HTTPS Slow – The expected pattern in the retrieved HTML. • IMAP and POP3 – (Optional) The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The password to log into the server. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user.

Parameter	Data Type	Description
		• MS SharePoint and MS SharePoint Secure – The pattern that is expected in the retrieved HTML. • RADIUS Auth – The username and password, separated by a backslash (\). For example: **`username\password`** • Simple HTTP and Simple HTTPS – The pattern expected in the retrieved HTML. • SMTP – (Optional) The pattern that is expected in the banner for the SMTP server. • SNMP – (Required if an OID is specified in the test target) A pattern to match in the response. • Specific HTTP Port – The pattern that is expected in the retrieved HTML.
service_monitor.hosts	String	Additional headers. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
service_monitor.code	Integer	The expected status code. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
service_monitor.test_delay	Integer	The interval between test start times, in seconds.
service_monitor.method	Enum	The method for HTTP and HTTPS tests. Possible values: • **GET** • **HEAD** • **POST** **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, and HTTPS Slow tests.
server_monitor.post_body	String	The data that is being sent in the POST request. **Conditional:** Required when the **`server_monitor.method`** is **`POST`**.

Persistence

Parameter	Data Type	Description
persistence.type	Enum	How clients are redirected to the real server that they were initially routed to, after a period of inactivity. Possible values: • **NONE** – Past connections / UDP datagrams are not considered when directing clients to servers. • **SCRIPNETMASK** – Directs requests based on

Parameter	Data Type	Description
		the client IP address. Requests from clients that are grouped by the persistence netmask are routed to the same real server as the first request received from the group. **Dependent variable:** ○ persistence.netmask • **COOKIEINSERT** – Directs requests based on a cookie (BNI_cookie_name) that is inserted by the Barracuda Load Balancer ADC after it routes the first request from the client. **Dependent variables:** ○ persistence.cookie_name ○ persistence.cookie_domain ○ persistence.cookie_path ○ persistence.cookie_age • **COOKIEPASSIVE** – Directs requests based on a cookie (BNI_cookie_name) that is inserted in the response by the Barracuda Load Balancer ADC only if the real server inserts a cookie. **Dependent variables:** ○ persistence.cookie_name ○ persistence.cookie_domain ○ persistence.cookie_path ○ persistence.cookie_age • **HEADERFIELD** – Directs all incoming HTTP requests based on the HTTP header. **Dependent variable:** ○ persistence.header_name • **URLPARAM** – Directs all incoming HTTP requests based on the specified parameter name. **Dependent variable:** ○ persistence.param_name
persistence.timeout	String	For HTTP/S services, the expiration age for the cookie is set to the current time plus this value whenever the browser sends a request. If the browser does not honor the expiry age and sends the cookie even after its expiration, the same real server is used. For all other service types, this is the maximum length of time that a client can remain idle during a persistent session and still be redirected to the same real server.
persistence.failover_method	Enum	How to handle a request in a persistent session when the server that must respond to the request is unavailable. Possible values: • **load_balance** – The requests are load balanced

Parameter	Data Type	Description
		among the remaining servers in the pool. • **error** – A "503 service unavailable" error message is sent.
persistence.netmask	String	The netmask for persistence using source IP addresses (including Layer 4-UDP). A more specific netmask (such as 255.255.255.255) tracks each client independently and can cause a higher memory load on the Barracuda Load Balancer ADC. A less specific netmask (such as 255.255.0.0) organizes multiple clients under the same network identifier and connects them all to the same server.
persistence.cookie_name	String	The name of the cookie used for persistence.
persistence.cookie_domain	String	The domain property of the cookie. If blank, it is the domain name in the request. **Conditional:** Required only when the **persistence.type** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence.cookie_path	String	The path where the cookie is valid.
persistence.cookie_age	String	The maximum age for the session cookie, in seconds. **Conditional:** Required only when the **persistence.type** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence.header_name	String	The name of the header value to verify in the HTTP requests. **Conditional:** Required only when the **persistence.type** is **HEADERFIELD**.
persistence.param_name	String	The name of the parameter value to verify in the URL. **Conditional:** Required only when the **persistence.type** is **URLPARAM**.
persistence.cookie_security	Boolean	Transmits the cookie over only HTTPS connections. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
persistence.cookie_httponly	Boolean	Makes the cookie inaccessible by client-side scripts, if supported by the browser. Helps mitigate most common cross-site scripting (XSS) attacks. Possible values: • **true** • **false**

Parameter	Data Type	Description
		For Perl, possible values are: • **1** – true • **0** – false

Notifications

Parameter	Data Type	Description
notification_enable	Boolean	Generates an alert whenever a real server goes up or down. The alert is emailed to the system alerts email address entered on the **BASIC > Administration** page. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false **Dependent variable:** minimum_notificate_real_server
minimum_notificate_real_server	Integer	The minimum number of operating real servers for this service. If this number is not met, an alert is generated. If you enter **0,** no alerts are generated.

Headers and URLs

The following parameters are available for only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
client_ip_addr_header	String	The HTTP header name (e.g., X-Forwarded-For or X-Client-IP) to be used in logs as a substitute for any client IP addresses in the request headers.
ignore_case	Boolean	Ignores the letter case in URLs when processing rules (e.g., content rules, HTTP request and response rewrite rules and response body rewrite rules, and the URL and Header allow/deny rules). Only the letter case of the URLs is ignored; parameter names are unaffected. **Recommended settings:** • For Windows-based servers: **true** • For Linux- and Unix-based servers:**false** For Perl, possible values are: • **1** – true • **0** – false

Parameter	Data Type	Description
ignore_expect_header	Boolean	Ignores the **'Expect'** entry in http headers while buffering HTTP requests in HTTP and HTTPS services. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Keepalive and Timeouts

The following parameters apply to only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
keepalive_request	Integer	The maximum number of requests allowed on a persistent HTTP connection. A value of **0** does not enforce any limit, allowing the client to control the number of requests on the connection.
tcp_keepalive_timeout	Integer	The maximum time which connections with clients can remain idle, in seconds, before timing out. Default setting: 64 **Note:** A value of 0 indicates that the session never times out. Only enter 0 if the URL takes so long to return that it is difficult to set a max timeout value.

SSL Offloading

The following parameters apply to only secure service types (i.e., HTTPS, Instant SSL, FTP SSL, and Secure TCP Proxy).

Parameter	Data Type	Description
ssl_offloading.status	Boolean	Enables SSL offloading. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.certificate	String	The certificate to be presented to any browser accessing the service. **Conditional:** Requiredwhen **`ssl_offloading.status`** is **`true`**.
ssl_offloading.enable_ssl_3	Boolean	Allows clients to use SSL 3.0 to connect to

Parameter	Data Type	Description
		the service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.ssl_enable_tls	Boolean	Allows clients to use SSL TLS 1.0 to connect to the service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.ssl_enable_tls_1_1	Boolean	Allows clients to use SSL TLS 1.1 to connect to the service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.ssl_enable_tls_1_2	Boolean	Allows clients to use SSL TLS 1.2 to connect to the service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.sni	Boolean	Enables Server Name Indication (SNI), an extension of SSL and TLS protocols. Allows a client to request a certificate for a specific domain from a server hosting more than one domain. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.sni_strict	Boolean	Blocks non-SNI clients. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true

Parameter	Data Type	Description
		• **0** – false
ssl_offloading.sni_domains	Array	The domain name for which the SNI check is enforced. You can specify multiple domain names with a comma (**,**) as a delimiter without any spaces.
ssl_offloading.sni_certs	Array	The certificates associated with the specified domain names. **Conditional:** Required when SNI is enabled.
sni_ecdsa_certs	String	The ECDSA certificate associated with the specified domain names.
ssl_offloading.ciphers	Enum	The ciphers that are used for the service. Possible values: • **default –** Includes all available ciphers. • **custom –** Includes a select list of available ciphers. **Dependent variable:** ○ ssl_offloading.cipher_list
ssl_offloading.cipher_list	Array	The list of SSL ciphers to use. Possible values: • **ECDHE-RSA-AES256-GCM-SHA384** • **ECDHE-ECDSA-AES256-GCM-SHA384** • **ECDHE-RSA-AES256-SHA384** • **ECDHE-ECDSA-AES256-SHA384** • **ECDHE-RSA-AES256-SHA** • **ECDHE-ECDSA-AES256-SHA** • **AES256-GCM-SHA384** • **AES256-SHA256** • **AES256-SHA** • **CAMELLIA256-SHA** • **ECDHE-RSA-AES128-GCM-SHA256** • **ECDHE-ECDSA-AES128-GCM-SHA256** • **ECDHE-RSA-AES128-SHA256** • **ECDHE-ECDSA-AES128-SHA256** • **ECDHE-RSA-AES128-SHA** • **ECDHE-ECDSA-AES128-SHA** • **AES128-GCM-SHA256** • **AES128-SHA256** • **AES128-SHA** • **CAMELLIA128-SHA** • **SEED-SHA**

Parameter	Data Type	Description
		• **IDEA-CBC-SHA** • **ECDHE-RSA-RC4-SHA** • **ECDHE-ECDSA-RC4-SHA** • **RC4-SHA** • **ECDHE-RSA-DES-CBC3-SHA** • **ECDHE-ECDSA-DES-CBC3-SHA** • **DES-CBC3-SHA** **Conditional:** Requiredwhen **`ssl_offloading.ciphers`** is **`custom`**.
ssl_offloading.enforce_client_certificate	Boolean	Requires clients to present a certificate or the SSL handshake is terminated. Possible values: • **true** • **false**
ssl_offloading.enable_client_authentication	Boolean	Requires users connecting to this site to present a trusted certificate for validation. Possible values: • **true** • **false** **Conditional:** Required when **`ssl_offloading.enforce_client_certificate`** is **`true`**.
ssl_offloading.trusted_certificates	Array	A list of trusted certificates.

Security Policy

The following parameters are apply to only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
security.enabled	Boolean	Enforces the conifgured security policy. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
security.mode	Enum	Determines handling of anomalies and intrusions. Possible values: • **ACTIVE** – Blocks requests when an anomaly or intrusion is detected. • **PASSIVE** – Logs all detected anomalies and intrusions but allows traffic to pass through the

Parameter	Data Type	Description
		Barracuda Load Balancer ADC. Use this mode in the initial stages of deployment, to prevent false positives from paralyzing the service.
security.web_firewall_policy	Enum	For HTTP, HTTPS, and Instant SSL services, you can assign a security policy. Either create a custom policy or use one of the following predefined security policies: • **default** • **sharepoint** • **sharepoint2013** • **owa** • **owa2010** • **owa2013** • **oracle**
security.trusted_host_group	String	The trusted hosts group for this service.
security.web_firewall_log_level	Enum	The threshold for logging error messages for the service. Possible values: • **0** – Emergency: system is unusable (highest priority) • **1** – Alert: response needed immediately • **2** – Critical: critical conditions • **3** – Error: error conditions • **4** – Warning: warning conditions • **5** – Notice: normal but significant condition • **6** – Information: informational messages (on ACL configuration changes) • **7** –Debug: debug level messages (lowest priority)

FTP Passive

The following parameters apply to only FTP and FTP SSL services.

Parameter	Data Type	Description
ftp_pasv_config.ftp_aps_status	Boolean	Enables a security policy to only allow specific FTP verbs. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false You can edit the list of allowed FTP verbs for this service on the **SECURITY > FTP Security** page.

Parameter	Data Type	Description
ftp_pasv_config.ftp_pasv_ip_address	String	The IP address that the FTP application uses to open an FTP data connection after receiving a PASV request from the client. This IP address is used only for PASV requests from FTP clients; it is ignored for PORT requests. • If you leave this field empty, the FTP application uses the virtual IP address of the service. • If a NAT'ing firewall sits in front of the Barracuda Load Balancer ADC, enter the public IP address that translates to the virtual IP address of this service.
ftp_pasv_config.ftp_pasv_port_ranges	String	Port numbers that the FTP application uses to respond to a PASV request from the FTP client. You can enter a single port number or a range of ports where the start port and end port are separated by a hyphen (-). If you enter a port range, it includes the start and end port. • If a firewall sits in front of the Barracuda Load Balancer ADC, typically you would enter the ports that are allowed by the firewall. • If you leave this field blank, any available port is used.

Certificates

URI	Method	Functionality
/certificates	POST	Generate a Self-Signed Certificate.
/certificates?upload=signed	POST	Upload a Signed Certificate.
/certificates?upload=trusted	POST	Upload a Trusted Certificate.
/certificates/*<certificate name>*	GET	Download aCertificate.

.

Servers

URI	Method	Functionality
/virtual_service_groups\ /*<group name>*/virtual_services\ /*<service name>*/servers	POST	You can add and configure multiple back-end servers to load balance incoming traffic for a service. Create a Server.
	GET	Retrieve Servers.
/virtual_service_groups\	PUT	Update a Server.

/*<group name>*/virtual_services\ /*<service name>*/servers/*<server name>*	GET DELETE	Retrieve Servers. Delete a Server.

Server Parameters

You can configure the following parameters for each real server.

Server Configuration

Parameter	Data Type	Description
name	String	A name to identify the server.
address_version	Enum	The internet protocol version to be used. Possible values: • **ipv4** • **ipv6**
identifier	Enum	How the Barracuda Load Balancer ADC identifies the server. Possible values: • **hostname** • **ip_address**
ip_address	String	The IP address of the server. **Conditional:** Required when the **`identifier`** is **`ip_address`**.
port	String	The port number of the server.
hostname	String	The hostname of the server. **Conditional:** Required when the **`identifier`** is **`hostname`**.
status	Enum	Indicates when requests are forwarded to the server. Possible values: • **Enable** – Requests can be forwarded to the server. • **Disable** – Requests cannot be forwarded to the server. All existing connections to this server are immediately terminated. • **Maintenance** – Requests cannot be forwarded to the server. Existing connections are terminated only after in-progress requests are completed. • **Sticky** – (Available for only Layer 7 services) Only requests from existing persistent connections for the service can be forwarded to the server. Existing connections are maintained until the Persistence **Time on** the BASIC **> Services page** is exceeded.

Parameter	Data Type	Description
backup_server	Boolean	Configures the server as a backup server that is used only when all other servers fail or are out of service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
weight	Integer	The static load balancing weight of the server. The server with the highest weight receives the most requests. **Range:** 1 to 65535 **Note:** The server weight is ignored if adaptive scheduling is enabled.

SSL

The following parameters apply only to servers being added to secure services (i.e., HTTPS, Instant SSL, FTP SSL, and Secure TCP Proxy).

Parameter	Data Type	Description
enable_https	Boolean	Encrypts all traffic between the server and service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enabled_ssl_3	Boolean	Enables the service to use SSL 3.0 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enable_tls_1	Boolean	Enables the service to use SSL TLS 1.0 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

enable_tls_1_1	Boolean	Enables the service to use SSL TLS 1.1 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enabled_tls_1_2	Boolean	Enables use of SSL TLS 1.2 by the service to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
validate_certificate	Boolean	Requires validation of the server certificate using certificates from well-known Certificate Authorities. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
client_certificate	String	The certificate that the server provides when the service requires client authentication.

Server Monitor

Parameter	Data Type	Description

server_monitor.test	Enum	The tests that are used by the server monitor to determine the availability of the servers that are associated with the service. You can either use a monitor group or select a test type.
		
		To specify a **Error! Not a valid result for table.**, use the following syntax:
		`GROUP_<group name>`
		For example, to use a monitor group named `Group 1`:
		`"service_monitor":{"test":"GROUP_Group1 "}`
		
		To select a test type, use one of the following values:
		○ **ICMP_PING** – Performs a PING test.
		○ **TCP_PORT_CHECK** – Validates that the configured service port is open.
		Dependent variable:
		○ service_monitor.port

Parameter	Data Type	Description
		• **UDP_PORT_CHECK** – Verifies that the UDP port is open by sending a 0 byte datagram to the IP address and port of the real server. This test depends on receiving an ICMP Port Unreachable message to determine the result. If a firewallprevents outbound ICMP messages, the port is assumed to be open. **Dependent variable:** ○ service_monitor.port • **HTTP_TEST** – Sends an HTTP request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTP** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTP_SLOW** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts

Parameter	Data Type	Description
		○ service_monitor.code • **HTTPS_TEST** – Sends an HTTPS request to the specified URL to verify that the respose contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_SLOW** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTPS** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **DNS_TEST** – Sends a DNS query and compares the returned IP address to the entered IP address. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **IMAP_TEST** – Simple test for an IMAP service. If no

Parameter	Data Type	Description
		username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **POP_TEST** – Simple test for a POP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SMTP_TEST** – Simple test for SMTP servers. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SNMP_TEST** – Sends an SNMP GET to the specified OID to verify that the response contains an expected pattern. If no OID is specified, this test verifies only the availability of the service in the server. **Dependent variable:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SIP_TEST** – Simple test for a SIP service. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service. **Dependent variable:** ○ service_monitor.port • **SIP_TLS_TEST** – Simple test for a SIP service over TLS. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service over TLS. **Dependent variable:** ○ service_monitor.port • **LDAP_AD_TEST** – Bind test for an LDAP/AD service. If no username and password are provided, the LDAP/AD test verifies the availability of the anonymous user. **Dependent variables:** ○ service_monitor.port

Parameter	Data Type	Description
		○ service_monitor.target ○ service_monitor.match • **LDAP_AD_SSL_TEST** – Bind test for an LDAPS/AD service. If no username and password are provided, the LDAP/AD test verifies the availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SPAMFIREWALL_TEST** – Test for use with Barracuda Spam Firewalls. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FORCED_PORT_HTTP** – (Specific HTTP Port test) Sends an HTTP GET request using a specified port to a relative URL on the real server, to verify that the retrieved HTML contains an expected pattern. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_TEST** – Tests the availability of a RADIUS server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_ACCT** – Tests the availability of a RADIUS server by making an accounting request. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RDP_TEST** – Attempts an RDP connection to each real server to check the availability of the Terminal service. **Dependent variable:** ○ service_monitor.port • **FTP_TEST** – Attempts a TCP connection to each real

Parameter	Data Type	Description
		server to check FTP availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FTPS_TEST**– Attempts a TCP connection to each real server to check FTPS availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SFTP_TEST** – Test for FTP over SSH. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password • **NTLM_TEST** – Simple test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **NTLMS_TEST** – Secure test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **ALWAYS_PASS** – This test is used for troubleshooting or for services with management access to real servers. This test always passes regardless of the condition of the real server.

Parameter	Data Type	Description
server_monitor.port	Integer	By default, tests use the configured real server port for the service unless the real server port is set to ALL. In that case, tests use the default port for the test type (e.g., SMTP = 25, HTTP = 80, DNS = 53, HTTPS = 443, IMAP = 143, POP = 110, and SNMP = 161). Enter a port number to override the default port.
server_monitor.user_name	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The username for the SSH account. • MS SharePoint – The username for the SharePoint service.
server_monitor.password	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The password for the SSH account. • MS SharePoint – The password for the SharePoint service.
server_monitor.target	String	**Conditional:** The value for this parameter differs for each test type: • Barracuda Spam Firewall Test – The domain for the mail server. • DNS – The fully qualified domain name. • FTP – The username to log into the server. The default username is **`anonymous`**. • FTPS – The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service on the server. • HTTP and HTTPS – The complete URL (starting with **`http`** or **`https`**). • HTTP Slow and HTTPS Slow – The complete URL (starting with **`http`** or **`https`**). • IMAP and POP3 – (Optional) The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The username with full LDAP schema. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The root relative URL. For example: **`/cgi-bin/index.cgi`** • RADIUS Auth and RADIUS Acct – The secret to use with the RADIUS server.

Parameter	Data Type	Description
		• Simple HTTP and Simple HTTPS – The root relative URL. For example: **`/cgi-bin/index.cgi`** The actual URL used is **`https://<real_server_ip>:<port><URL>`** • SNMP – (Optional) Enter a valid SNMP OID. **Note:** If no OID is specified, this test verifies only the availability of the service in the server. • Specific HTTP Port – The TCP port followed by colon (:) and the root relative URL. For example: **`8080:/cgi-bin/index.cgi`**
server_monitor.match	String	**Conditional:** This parameter value differs for each test type: • Barracuda Spam Firewall – (Optional) A pattern that is expected in the banner of the SMTP server. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. • DNS – The IP address of the hostname. • FTP – The password to log into the server. The default password is **`anonymous`**. • FTPS – The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The expected pattern in the retrieved HTML. • HTTP Slow and HTTPS Slow – The expected pattern in the retrieved HTML. • IMAP and POP3 – (Optional) The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The password to log into the server. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The pattern that is expected in the retrieved HTML. • RADIUS Auth – The username and password, separated by a backslash (****). For example: **`username\password`** • Simple HTTP and Simple HTTPS – The pattern expected in the retrieved HTML.

Parameter	Data Type	Description
		• SMTP – (Optional) The pattern that is expected in the banner for the SMTP server. • SNMP – (Required if an OID is specified in the test target) A pattern to match in the response. • Specific HTTP Port – The pattern that is expected in the retrieved HTML.
server_monitor. hosts	String	Additional headers. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
server_monitor.code	Integer	The expected status code. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
server_monitor.test_delay	Integer	The interval between test start times, in seconds.
server_monitor.method	Enum	The method for HTTP and HTTPS tests. Possible values: • **GET** • **HEAD** • **POST** **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, and HTTPS Slow tests.
server_monitor.post_body	String	Data that is sent in the POST request. **Conditional:** Required when the **`server_monitor.method`** is **`POST`**.

Advanced Options

Parameter	Data Type	Description
enable_client_impersonation	Boolean	Enables the Barracuda Load Balancer ADC to use the client IP address as the source IP address to communicate to the server. If this option is disabled, the Barracuda Load Balancer ADC uses its own IP address. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false **Note:** Available for only TCP Proxy, Secure TCP Proxy, HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
route	Enum	Indicates how outgoing traffic is delivered from the real server to the client. Direct server return is ideal for high-bandwidth requirements such as content delivery networks and lets you keep the existing IP addresses of your real servers. Possible values: • **standard** – Uses standard route for outgoing traffic. • **direct** – Uses direct server return.
enable_connection_pooling	Boolean	Allows a server connection to be used for multiple client requests. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false **Note:** Available for only HTTP, HTTPS, and Instant SSL services. **Dependent Variables:** • keepalive_timeout • max_connections • max_request • max_keepalive_requests • max_establishing_connections • max_spare_connections • timeout
keepalive_timeout	Integer	The maximum length of time, in milliseconds, that a persistent connection to the real server can be idle before timing out. (Default: 900000) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_connections	Integer	The maximum number of simultaneous TCP connections between the Barracuda Load Balancer ADC and this server at any time. (Default: 10000) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_request	Integer	The maximum number of HTTP requests that can be queued on the Barracuda Load Balancer ADC for this server. (Default: 1000) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.

Parameter	Data Type	Description
max_keepalive_requests	Integer	The maximum number of requests allowed on a persistent connection. If set to 0, the number of requests allowed is unlimited. (Default: 0) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_establishing_connections	Integer	The maximum number of connections that can be initiated with the server at one time. When this limit is reached, additional connection requests are ignored. (Default: 100 connections) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_spare_connections	Integer	The maximum number of pre-allocated connections that can be established to this server. Spare connections stay in the pool to be used for future requests. (Default: 0 connections) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
timeout	Integer	The maximum length of time, in milliseconds, that the connection can remain idle before being terminated. (Default: 300000) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.

Content Rules

URI	Method	Functionality
/virtual_service_groups\ /*<group name>*/virtual_services\ /*<service name>*/content_rules	POST	Create a Content Rule.
	GET	Retrieve Content Rules.
/virtual_service_groups\ /*<group name>*/virtual_services\ /*<service name>*/content_rules\ /*<rule name>*	PUT	Update a Content Rule.
	GET	Retrieve Content Rules.
	DELETE	Delete a Content Rule.

Content Rule Parameters

You can configure the following parameters for content rules.

Basic Configuration

Parameter	Data Type	Description
name	String	A name for the content rule.
status	Boolean	The status of the content rule. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
host_match	String	A host name to be matched against the host in the request header.
url_match	String	A URL to be matched to the URL in the request header.
extended_match	String	An expression that consists of a combination of HTTP headers and/or query String parameters.
extended_match_sequence	Integer	A number to specify the order of this rule when multiple content rules are configured for this service. Content rules are evaluated sequentially according their extended match sequence number.
comments	String	Description about the content rule.

Caching

Parameter	Data Type	Description
cache.enabled	Boolean	Enables caching of the specified file types in local memory. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.file_extensions	Array	The extensions of the file types to be cached.
cache.expiry_age	Integer	The default expiration age in minutes, to be used when the response does not have an Expires header. A value of 0 (expiry-age = 0) sets the expiry age to 60 minutes.
cache.max_objsize	Integer	The maximum size of files that can be cached.
cache.min_objsize	Integer	The minimum size of files that can be cached.

Parameter	Data Type	Description
cache.negative_responses	Boolean	Caches HTTP negative responses with status codes like 204, 305, 404, 405, 414, 501, 502, and 504. Only negative responses with the Expires or Last `Modified headers` are cached. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.req_cachehdrs_ignore	Boolean	Ignores HTTP request cache-control headers to instruct upstream caching servers. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.resp_cachehdrs_ignore	Boolean	Ignores HTTP response cache-control headers to instruct upstream caching servers. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Compression

Parameter	Data Type	Description
compress.enabled	Boolean	Enables compression of the specified content types. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
compress.content_types	Array	The content types to be compressed. For example: **`text/css`** **`text/html`** **`text/js`** **`text/plain`**
compress.min_obj_size	Integer	The minimum size of objects that can be compressed, from 1 to **2147483646 in** bytes. The default value is 8192.

Load Balancing

Parameter	Data Type	Description
lb_algorithm	Enum	How traffic is distributed among the real servers associated with this content rule. If no servers are configured, the requests are distributed to the servers of the service. Possible values: • **weighted_round_robin** • **weighted_least_connection**

Persistence

Parameter	Data Type	Description
persistence_method	Enum	How clients are redirected after a period of inactivity to the real server that they were initially routed to. Possible values: • **NONE** – Past connections / UDP datagrams are not considered when directing clients to servers. • **SCRIPNETMASK** – Directs requests based on the client IP address. Requests from clients that are grouped by the persistence netmask are routed to the same real server as the first request received from the group. **Dependent variable:** ○ source_ip_netmask • **COOKIEINSERT** – Directs requests based on a cookie (BNI_cookie_name) that is inserted by the Barracuda Load Balancer ADC after it routes the first request from the client. **Dependent variables:** ○ persistence_cookie_name ○ persistence_cookie_domain ○ persistence_cookie_path ○ cookie_age ○ persistence_cookie_httponly ○ persistence_cookie_security • **COOKIEPASSIVE** – Directs requests based on a cookie (BNI_cookie_name) that is inserted by the Barracuda Load Balancer ADC in the response only if the real server inserts a cookie. **Dependent variables:** ○ persistence_cookie_name ○ persistence_cookie_domain ○ persistence_cookie_path ○ cookie_age

Parameter	Data Type	Description
		○ persistence_cookie_httponly ○ persistence_cookie_security • **HEADERFIELD** – Directs all incoming HTTP requests based on the HTTP header. **Dependent variable:** ○ header_name • **URLPARAM** – Directs all incoming HTTP requests based on the specified parameter name. **Dependent variable:** ○ parameter_name
persistence_time	String	The maximum idle time (in seconds) for a persistent connection. A client is directed to the same real server unless the connection is inactive for more than the specified number of seconds.
failover_method	Enum	How to handle a request in a persistent session when the server that must respond to the request is unavailable. Possible values: • **LB** – The requests are load balanced among the remaining servers in the pool. • **ERROR** – A "503 service unavailable" error message is sent. This method is not supported when the **persistence_method** is **SRCIPNETMASK**.
source_ip_netmask	String	The netmask for persistence using source IP addresses (including Layer 4-UDP). A more specific netmask (such as 255.255.255.255) tracks each client independently and can cause a higher memory load on the Barracuda Load Balancer ADC. A less specific netmask (such as 255.255.0.0) organizes multiple clients under the same network identifier and connects them all to the same server. **Conditional:** Required only when the **persistence_method** is **SRCIPNETMASK**.
persistence_cookie_name	String	The name of the cookie used for persistence. **Conditional:** Required only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence.cookie_domain	String	The domain name of server of the cookie. **Conditional:** Applicable only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence_cookie_path	String	The path where the cookie is valid. **Conditional:**

Parameter	Data Type	Description
		Applicable only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence_cookie_httponly	Boolean	Makes the cookie inaccessible by client-side scripts, if supported by the browser. Helps mitigate most common cross-site scripting (XSS) attacks. Possible values: • **0** – false • **1** – true **Conditional:** Applicable only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence_cookie_security	Boolean	Only transmits the cookie over HTTPS connections. Possible values: • **0** – false • **1** – true **Conditional:** Applicable only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
cookie_age	Integer	The maximum age for the session cookie, in minutes. **Conditional:** Applicable when the when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
header_name	String	The name of the header for which the value needs to be checked in the HTTP requests. **Conditional:** Required only when the **persistence_method** is **HEADERFIELD**.
parameter_name	String	The name of the parameter for which the value needs to be checked in the URL. **Conditional:** Required only when the **persistence_method** is **URLPARAM**.

Rule Group Servers

URI	Method	Functionality
/virtual_service_groups\ /*<group name>*/virtual_services\ /*<service name>*/content_rules\ /*<rule name>*/rg_servers	POST	Create a Rule Group Server.
	GET	Retrieve Rule Group Servers.
/virtual_service_groups\ /*<group name>*/virtual_services\ /*<service name>*/content_rules\ /*<rule name>*/rg_servers\ /*<rule group server name>*	PUT	Update a Rule Group Server.
	GET	Retrieve Rule Group Servers.
	DELETE	Delete a Rule Group Server.

Rule Group Server Parameters

You can configure the following parameters for rule group servers.

Server Configuration

Parameter	Data Type	Description
name	String	A name to identify the server.
address_version	Enum	The Internet protocol version to be used. Possible values: • **ipv4** • **ipv6**
identifier	Enum	Indicates whether the Barracuda Load Balancer ADC identifies the server by its IP address or hostname. Possible values: • **hostname** • **ipaddr**
ip_address	String	The IP address of the server. **Conditional:** Required when the **`identifier`** is **`ipaddr`**.
hostname	String	The hostname of the server. **Conditional:** Required when the **`identifier`** is **`hostname`**.
port	String	The port number of the server.

status	Enum	Indicates if requests are forwarded to the server. Possible values: • **Enable** – Requests can be forwarded to the server. • **Disable** – Requests cannot be forwarded to the server. All existing connections to this server are immediately terminated. • **Maintenance** – Requests cannot be forwarded to the server. Existing connections are terminated only after in-progress requests are completed.

		• **Sticky** – Only requests from existing persistent connections for the service can be forwarded to the server. Existing connections are maintained until the Persistence **Time on** the BASIC **> Services page** is exceeded.
backup_server	Boolean	Configures the server as a backup server that is only used when all other servers fail or are out of service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
weight	Integer	The static load balancing weight of the server. The server with the highest weight receives the most requests. **Range:** 1 to 65535

SSL

Parameter	Data Type	Description
enable_https	Boolean	Encrypts all traffic between the server and service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enabled_ssl_3	Boolean	Allows the service to use SSL 3.0 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enable_tls_1	Boolean	Allows the service to use SSL TLS 1.0 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Parameter	Data Type	Description
enable_tls_1_1	Boolean	Allows the service to use SSL TLS 1.1 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enabled_tls_1_2	Boolean	Allows the service to use SSL TLS 1.2 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
validate_certificate	Boolean	Requires validation of the server certificate using certificates from well-known Certificate Authorities. Possible values: • **1** – false • **0** – true
client_certificate	String	The name of the certificate that the server provides when the service requires client authentication.

Server Monitor

Parameter	Data Type	Description

server_monitor.test	Enum	The tests that the server monitor uses to determine the availability of the servers that are associated with the service. You can use either a monitor group or a test type.
		
		To specify a **Error! Not a valid result for table.**, use the following syntax:
		`GROUP_<group name>`
		For example, to use a monitor group named `Group 1`:
		`"service_monitor":{"test":"GROUP_Group1 "}`
		
		To select a test type, use one of the following values:
		○ **ICMP_PING** – Performs a PING test.
		○ **TCP_PORT_CHECK** – Verifies that the configured service port is open.
		Dependent variable:
		○ service_monitor.port
		• **UDP_PORT_CHECK** – Verifies that the UDP port is open by sending a 0 byte datagram to the IP address and port of the real server. Expects to

Parameter	Data Type	Description
		receive an ICMP Port Unreachable message. If a firewall prevents outbound ICMP messages, the test assumes that the port is open. **Dependent variable:** ○ service_monitor.port • **HTTP_TEST** – Sends an HTTP request to the specified URL and verifies that the response containes an expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTP** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTP_SLOW** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_TEST** – Sends an HTTPS request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code.

Parameter	Data Type	Description
		The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_SLOW** – Sends an HTTPS request to the specified relative URL to verify that the response contains that the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTPS** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **DNS_TEST** – Sends a DNS query and compares the returned IP address to the entered IP address. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **IMAP_TEST** – Simple test for an IMAP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:**

Parameter	Data Type	Description
		○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **POP_TEST** – Simple test for a POP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SMTP_TEST** – Simple test for SMTP servers. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SNMP_TEST** – Performs an SNMP GET to the specified OID and verifies that the response contains an expected pattern. If no OID is specified, this test verifies only the availability of the service in the server. **Dependent variable:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SIP_TEST** – Simple test for a SIP service. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service. **Dependent variable:** ○ service_monitor.port • **SIP_TLS_TEST** – Simple test for a SIP service over TLS. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service over TLS. **Dependent variable:** ○ service_monitor.port • **LDAP_AD_TEST** – Bind test for an LDAP/AD service. If no username and password are provided, the LDAP/AD test verifies the availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **LDAP_AD_SSL_TEST** – Bind test for an LDAPS/AD

Parameter	Data Type	Description
		service. If no username and password are provided, the LDAP/AD test verifies the availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SPAMFIREWALL_TEST** – Test for use with Barracuda Spam Firewalls. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FORCED_PORT_HTTP** – (Specific HTTP Port test) Sends an HTTP GET request using a specified port to a relative URL on the real server, and verifies that the retrieved HTML contains an expected pattern. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_TEST** – Tests the availability of a RADIUS server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_ACCT** – Tests the availability of a RADIUS server by making an accounting request. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RDP_TEST** – Attempts an RDP connection to each real server to check the availability of the Terminal service. **Dependent variable:** ○ service_monitor.port • **FTP_TEST** – Attempts a TCP connection to each real server to check FTP availability. **Dependent variables:**

Parameter	Data Type	Description
		○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FTPS_TEST**– Attempts a TCP connection to each real server to check FTPS availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SFTP_TEST** – Test for FTP over SSH. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password • **NTLM_TEST** – Simple test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **NTLMS_TEST** – Secure test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **ALWAYS_PASS** – This test is used for troubleshooting or for services with management access to real servers. This test always passes regardless of the condition of the real server.
server_monitor.port	Integer	By default, tests use the configured real server port for the service unless the real server port is set to ALL. In that case, the tests use the default port for the test type (e.g., SMTP = 25, HTTP = 80, DNS = 53, HTTPS = 443, IMAP = 143, POP = 110, and SNMP = 161). Enter a port number to override the default port.

Parameter	Data Type	Description
server_monitor.user_name	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The username for the SSH account. • MS SharePoint – The username for the SharePoint service.
server_monitor.password	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The password for the SSH account. • MS SharePoint – The password for the SharePoint service.
server_monitor.target	String	**Conditional:** The value for this parameter differs for each test type: • Barracuda Spam Firewall Test – The domain for the mail server. • DNS – The fully qualified domain name. • FTP – The username to log into server. The default username is **`anonymous`**. • FTPS – The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The complete URL (starting with **`http`** or **`https`**). • HTTP Slow and HTTPS Slow – The complete URL (starting with **`http`** or **`https`**). • IMAP and POP3 – (Optional) The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The username with full LDAP schema. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The root relative URL. For example: **`/cgi-bin/index.cgi`** • RADIUS Auth and RADIUS Acct – The secret to use with the RADIUS server. • Simple HTTP and Simple HTTPS – The root relative URL. For example: **`/cgi-bin/index.cgi`** The actual URL used is **`https://<real_server_ip>:<port><URL>`** • SNMP – (Optional) Enter a valid SNMP OID.

Parameter	Data Type	Description
		Note: If an OID is not specified, this test verifies only the availability of the service in the server. • Specific HTTP Port – The TCP port followed by colon (:) and the root relative URL. For example: **`8080:/cgi-bin/index.cgi`**
server_monitor.match	String	**Conditional:** The value for this parameters differs for each test type: • Barracuda Spam Firewall – (Optional) A pattern that is expected in the banner of the SMTP server. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. • DNS – The IP address of the hostname. • FTP – The password to log into the server. The default password is **`anonymous`**. • FTPS – The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The expected pattern in the retrieved HTML. • HTTP Slow and HTTPS Slow – The expected pattern in the retrieved HTML. • IMAP and POP3 – (Optional) The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The password to log into the server. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The pattern that is expected in the retrieved HTML. • RADIUS Auth – The username and password, separated by a backslash (****). For example: **`username\password`** • Simple HTTP and Simple HTTPS – The pattern expected in the retrieved HTML. • SMTP – (Optional) The pattern that is expected in the banner for the SMTP server. • SNMP – (Required if an OID is specified in the test target) A pattern to match in the response. • Specific HTTP Port – The pattern that is expected in the retrieved HTML.

Parameter	Data Type	Description
server_monitor.hosts	String	Additional headers. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
server_monitor.code	nteger	The expected status code. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
server_monitor.test_delay	nteger	The interval between test start times, in seconds.
server_monitor.method	Enum	The method for HTTP and HTTPS tests. Possible values: • **GET** • **HEAD** • **POST** **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, and HTTPS Slow tests.
server_monitor.post_body	String	The data that is being sent in the POST request. **Conditional:** Required when the **`server_monitor.method`** is **`POST`**.

Advanced Options

Parameter	Data Type	Description
enable_client_impersonation	Boolean	Enables the Barracuda Load Balancer ADC to use the client IP address as the source IP address to communicate to the server. If this option is disabled, the Barracuda Load Balancer ADC uses its own IP address. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
connection_pooling.enable_connection_pooling	Boolean	Allows the server connection to be used for multiple client requests. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false **Dependent Variables:** • connection_pooling.keepalive_timeout

Parameter	Data Type	Description
		• max_connections • max_request • max_keepalive_requests • max_establishing_connections • max_spare_connections • timeout
max_connections	Integer	The maximum number of simultaneous TCP connections that the Barracuda Load Balancer ADC can have with this server at any time. (Default: 10000) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
max_request	Integer	The maximum number of HTTP requests that can be queued on the Barracuda Load Balancer ADC for this server. (Default: 1000) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
max_keepalive_requests	Integer	The maximum number of requests allowed on a persistent connection. If set to 0, the number of requests allowed is unlimited. (Default: 0) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
max_establishing_connections	Integer	The maximum number of connections that can be initiated with the server at one time. After this limit is reached, additional connection requests are ignored. (Default: 100 connections) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
max_spare_connections	Integer	The maximum number of pre-allocated connections that can be established to this server. Spare connections stay in the pool to be used for future requests. (Default: 0 connections) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
timeout	Integer	The maximum length of time, in milliseconds, that the connection can remain idle before being terminated. (Default: 300000)

Parameter	Data Type	Description
connection_pooling. keepalive_timeout	Integer	The time in milliseconds to time out a connection that is used at least once. This is the maximum amount of time a connection is kept alive. This value is applicable per 1024 connections, where a timeout error had occurred. **Conditional:** Applicable only when **`connection_pooling.enable_connection_pooling`** is **`true`**.

Security Policies

URI	Method	Functionality
/security_policies	POST	Create a Custom Security Policy.
	GET	Retrieve Security Policies.
/security_policies/<*policy name*>	PUT	Update a Security Policy.
	GET	Retrieve Security Policies.
	DELETE	Delete a Security Policy.

Security Policy Parameters

You can configure the following parameters for each security policy.

Name

Parameter	Data Type	Description
name	String	The name of the security policy.

Request Limits

Parameter	Data Type	Description
request_limits.enable	Enum	Enforces size limit checks on request headers. Possible values: • **yes** • **no**
request_limits.max_request_length	Integer	The maximum allowable request length. This includes the Request Line and all HTTP request headers (for example, User Agent, Cookies, Referer, etc.).
request_limits.max_request_line_length	Integer	The maximum allowable length for the request line. The request line consists of the method, the URL (including any query strings) and the HTTP version.

request_limits.max_url_length	Integer	The maximum allowable URL length including the query string portion of the URL.
request_limits.max_query_length	Integer	The maximum allowable length for the query string portion of the URL.
request_limits.max_number_of_cookies	Integer	The maximum number of cookies to be allowed.
request_limits.max_cookie_name_length	Integer	The maximum allowable length for a cookie name.
request_limits.max_cookie_value_length	Integer	The maximum allowable length for a cookie value.
request_limits.max_number_of_headers	Integer	The maximum number of headers to be allowed in a request.
request_limits.max_header_name_length	Integer	The maximum allowable length for a header name.
request_limits.max_header_value_length	Integer	The maximum allowable length for header value in a request.

Cookie Security

Parameter	Data Type	Description
cookie_security.tamper_proof_mode	Enum	Indicates whether tamper proofing method for cookies. Possible values: • **signed** • **encrypted** • **none**
cookie_security.cookie_max_age	Integer	The maximum age for session cookies, in minutes.
cookie_security.cookie_replay_protection_type	Enum	Indicates whether header values and/or client IP addresses are encoded in the cookies that are sent to the client. These are validated in the incoming cookies. If the IP address or header value does not match the IP address or header value of the sent cookie, the incoming cookie is considered a possible replay attack. Possible values: • **none** • **IP** • **IP_and_custom_headers** • **custom_headers**

Parameter	Data Type	Description
cookie_security.custom_headers	Array	The custom headers to be used in the cookie. **Conditional:** Required only if the **cookie_security.cookie_replay_protection_type** is **IP_and_custom_headers** or **custom_headers**.
cookie_security.secure_cookie	Enum	Allows cookies only if the client makes secure HTTPS connection. Possible values: • **yes** • **no**
cookie_security.http_only	Enum	Makes the cookie inaccessible by client-side scripts, if supported by the browser. Helps mitigate most common cross-site scripting (XSS) attacks. Possible values: • **yes** • **no**
cookie_security.allow_unrecognized_cookies	Enum	Whether unrecognized cookies are allowed. Possible values: • **custom** • **always** • **never**
cookie_security.days_allowed	Integer	The maximum number of days that unrecognized cookies are allowed. **Conditional:** Required only when **cookie_security.allow_unrecognized_cookies** is **custom**.
cookie_security.cookies_exempted	Array	The names of the cookies that are exempted from the cookie security policy.

URL Protection

Parameter	Data Type	Description
url_protection.enable	Enum	Enforces URL protection. Possible values: • **yes** • **no**
url_protection.allowed_methods	Array	The list of allowable methods in a request.
url_protection.allowed_content_types	Array	The list of content types to be allowed in the POST body of a request.
url_protection.max_content_length	Integer	The maximum content length to be allowed for POST request body.

Parameter	Data Type	Description
url_protection.max_parameters	Integer	The maximum number of parameters to be allowed in a request.
url_protection.maximum_upload_files	Integer	The maximum number of files that can be of file-upload type in a request.
url_protection.csrf_prevention	Enum	The Cross-Site Request Forgery (CSRF) prevention for forms and URLs. Possible values: • **forms_and_urls** • **none** • **forms**
url_protection.maximum_parameter_name_length	Integer	The maximum length of a parameter name in a request.
url_protection.blocked_attack_types	Array	The attack types to be matched in a request. Possible values: • **cross_site_scripting** • **remote_file_inclusion** • **sql_injection_strict** • **sql_injection** • **os_command_injection** • **remote_file_inclusion_strict** • **os_command_injection_strict** • **cross_site_scripting_strict**
url_protection.custom_blocked_attack_types	Array	The custom attack types defined on the **ADVANCED > Libraries** page (if any).
url_protection.exception_patterns	Array	Patterns that are allowed despite matching a malicious pattern group. **Note:** Configure the exact pattern name that is displayed on the **ADVANCED > View Internal Patterns** page, or as defined when creating a new group on the **ADVANCED > Libraries** page.

Parameter Protection

Parameter	Data Type	Description
parameter_protection.enable	Enum	Enforces parameter protection. Possible values: • **yes** • **no**

Parameter	Data Type	Description
parameter_protection.denied_ metacharacters	String	The meta-characters that are not allowed in a parameter value. Meta-characters must be URL encoded. Non-printable characters such as "backspace" and web interface reserved characters like "?" must be URL encoded.
parameter_protection.maximum_ parameter_value_length	Integer	The maximum allowed length of any parameter value, including no-name parameters.
parameter_protection.maximum_instances	Integer	The maximum number of times a parameter is allowed in a request.
parameter_protection.file_upload_ extensions	Array	The extensions of the files types that can be uploaded.
parameter_protection.maximum_upload_ file_size	Integer	The maximum size (in KB) for an individual file that can be uploaded in a request.
parameter_protection.blocked_attack_types	Array	The attack types to be matched in a request. Possible values: • **directory_traversal** • **directory_traversal_strict** • **cross_site_scripting** • **remote_file_inclusion** • **sql_injection_strict** • **sql_injection_medium** • **os_command_injection** • **remote_file_inclusion_strict** • **os_command_injection_strict** • **cross_site_scripting_strict**
parameter_protection.custom_blocked_ attack_types	Array	The custom attack types defined on the **ADVANCED > Libraries** page (if any).
parameter_protection.exception_patterns	Array	The patterns that are allowed despite matching a malicious pattern group. **Note:** Configure the exact pattern name displayed on the **ADVANCED > View Internal Patterns** page, or as defined when creating a new group on the **ADVANCED > Libraries** page.
parameter_protection.ignore_parameters	String	The parameters that are exempt from *all* validations.

Cloaking

Parameter	Data Type	Description
cloaking.suppress_return_code	Enum	Suppresses an HTTP Status code in the response header and inserts a default or custom response page in case of any error responses from the server. Possible values: • **yes** • **no**
cloaking.return_codes_to_exempt	String	The HTTP response codes that are exempt from cloaking.
cloaking.filter_response_header	Enum	Removes the HTTP headers in responses. Possible values: • **yes** • **no**
cloaking.headers_to_filter	String	The list of headers that are to be removed from a response before serving it to a client.

URL Normalization

Parameter	Data Type	Description
url_normalization.detect_response_charset	Enum	Detects the character set decoding from the response. Possible values: • **yes** • **no**
url_normalization.parameter_separators	Enum	The URL-decoded parameter separator to be used. Possible values: • **ampersand** • **ampersand_and_semicolon** • **semicolon**
url_normalization.double_decoding	Enum	Enables double-decoding of the character set. Possible values: • **yes** • **no**

url_normalization.default_charset	Enum	The character set decoding type to be used for incoming requests. Possible values: • **GBK** • **ASCII** • **UTF-8** • **Shift-JIS** • **JOHAB** • **EUC-KR** • **ISO-8859-1** • **ISO-2022-KR** • **ISO-2022-CN** • **ISO-2022-JP** • **HZ** • **BIG5** • **GB2312** • **EUC-TW** • **EUC-JP**

Global ACLs

URI	Method	Functionality
/security_policies\ /*<policy name>*/global_acls	POST	Create a Global ACL Rule.
	GET	Retrieve Global ACL Rules.
/security_policies\ /*<policy name>*/global_acls\ /*<acl rule name>*	PUT	Update a Global ACL Rule.
	GET	Retrieve Global ACL Rules.
	DELETE	Delete a Global ACL Rule.

Action Policy

URI	Method	Functionality
/security_policies\ /*<policy name>*/attack_groups\ /*<attack group name>*/actions/	GET	Retrieve the Attack Actions for an Attack Group
/security_policies\ /*<policy name>*/attack_groups\ /*<attack group name>*/actions\ /*<attack action name>*	PUT	Update an Attack Action.
	GET	Retrieve the Attack Actions for an Attack Group.

Attack Action Parameters

You can configure the following parameters for attack actions.

Parameter	Data Type	Description
action	Enum	The action to be taken for an invalid request. Possible values: • **none** • **protect_and_log** • **allow_and_log** • **protect_with_no_log**
attack_action_deny_response	Enum	How to respond to the client if the request is denied. Possible values: • **close_connection** • **send_response** **Dependent variable:** ○ response_page • **temporary_redirect** **Dependent variable:** ○ redirect_url • **permanent_redirect** **Dependent variable:** ○ redirect_url
follow_up_action	Enum	The follow up action to be taken if the request is denied. Possible values: • **none** • **block_client_ip** **Dependent variable:** ○ follow_up_action_time
follow_up_action_time	Integer	The time in seconds to block the client IP. **Conditional:** Required only when the **`follow_up_action`** is **`block_client_ip`**.
redirect_url	String	The URL used to redirect the request.

		Conditional: Required only when the **attack_action_deny_response** is **temporary_redirect** or **permanent_redirect**.
response_page	Enum	The response page to be sent to the client. Possible values: • **default** • **default-error-resp** • **default-virus** **Conditional:** Required only when the **attack_action_deny_response** is **send_response**.

Data Theft Protection

URI	Method	Functionality
/security_policies\ /*<policy name>*/data_theft_protections	POST	Create a Custom Data Theft Element.
	GET	Retrieve Data Theft Elements.
/security_policies\ /*<policy name>*/data_theft_protections\ /*<element name>*	PUT	Update a Data Theft Element.
	GET	Retrieve Data Theft Elements.
	DELETE	Delete a Data Theft Element.

Data Theft Element Parameters

You can configure the following parameters for each data theft element.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for this data theft element. **Note:** You cannot change the name of the predefined data theft elements.
enabled	Enum	Yes	Enables this data theft element. Possible values: • **yes** • **no**
identity_theft_type	Enum	Yes	The identity theft pattern of the data theft element. Possible values: • **directory_indexing** • **credit_cards** • **social_security_numbers** • **custom**

Parameter	Data Type	Mandatory	Description
custom_identity_theft_type	Enum	Conditional	The identity theft pattern defined on the **SECURITY > Libraries** page (if any). **Note:** Required only when **`identity_theft_type`** is **`custom`**.
action	Enum	Yes	How to handle any page containing this data type. Possible values: • **cloak** – Overwrites the matching data type with Xs. **Dependent variables:** ○ initial_characters_to_keep ○ trailing_characters_to_keep • **block** – Blocks the page.
initial_characters_to_keep	Integer	Conditional	The number of initial characters to be displayed to the user. **Note:** Required only when **`action`**is **`cloak`**.
trailing_characters_to_keep	Integer	Conditional	The number of trailing characters to be displayed to the user. **Note:** Required only when **`action`**is **`cloak`**.

Monitor Groups

URI	Method	Functionality
/monitor_groups	POST	Create a Monitor Group.
	GET	Retrieve Monitor Groups.
/monitor_groups/*<group name>*	GET	Retrieve Monitor Groups.
	DELETE	Delete a Monitor Group.

Monitors

URI	Method	Functionality
/monitor_groups\ /*<group name>*/monitors	POST	Create a Monitor.
	GET	Retrieve Monitors.
/monitor_groups\ /*<group name>*/monitors\ /*<monitor name>*	PUT	Update a Monitor.
	GET	Retrieve Monitors.
	DELETE	Delete a Monitor.

Resource Descriptions

The following sections provide descriptions, URIs, and example requests for all Barracuda Load Balancer ADC REST API resources. If you are using Perl, see the Perl Implementation section of this guide.

Virtual Service Groups

A virtual service group associates multiple services that you add to it. For example, you can create a service group to associate services by type (e.g., Instant SSL) or application (e.g., Lync). You can also use the predefined virtual service group named **default**.

Create a Service Group

To create a service group, make a POST request to the following URI:

`/virtual_service_groups`

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass a name for the service group:

Required Parameter	Data Type	Description
name	String	A name for the service group.

Request and Response Examples

The following examples display a request and response for creating a service group named **`group1`**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"group1"}' \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups
```

Example Response

```
{
    "id": "group1",
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Rename a Service Group

To rename a service group, make a PUT request to the following URI:

`/virtual_service_groups/<group name>`

where *<group name>* is the current name of the service group. The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the new name for the service group:

Required Parameter	Data Type	Description
name	String	A new name for the service group.

Request and Response Examples

The following examples display a request and response for changing the name of a service group from **`group1`** to **`demo`**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X PUT \
   -H "Content-Type:application/json" \
   -d '{"name":"demo"}' \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/group1
```

Example Response

```
{
    "id": "demo",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Service Groups

To retrieve data for all configured service groups, make a GET request to the following URI:

/virtual_service_groups

To retrieve data for only a specific service group, include the name of the group in the URI:

/virtual_service_groups/*<group name>*

Request and Response Examples

The following examples display a request and response for retrieving the default service group.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X GET \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default
```

Example Response

The following example displays an excerpt of the information retrieved for the services organized in the default service group.

```
{
    "id": "default",
    "name": "default",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n:",
    "virtual_services": [
        {
            "adaptive_param": "",
            "address_version": "ipv4",
            "auto_recover": true,
            "cache": {
                "enabled": false,
                "expiry_age": "60",
                "file_extensions": [
                    "gif",
                    "tif",
                    "jpg",
                    "jpeg",
                    "png",
                    "bmp",
                    "ico",
                    "js",
                    "jsp",
                    "css",
                    "jar",
                    "swf",
                    "pdf"
                ],
                "max_objsize": "256",
                "min_objsize": "256",
```

```
                "negative_responses": false,
                "req_cachehdrs_ignore": false,
                "resp_cachehdrs_ignore": false
……..……
            },
              "tcp_keepalive_timeout": false,
              "type": "FTPSSL"
          }
    ]
}
```

Delete a Service Group

You can delete only empty service groups. After you verify that a service group does not contain any services, delete it by making a DELETE request to the following URI:

/virtual_service_groups/*<group name>*

Request and Response Examples

The following examples display a request and response for deleting a service group named **group1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/group1
```

Example Response

```
{
    "msg": "Successfully deleted",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n:"
}
```

Virtual Services

A virtual service is a combination of a virtual IP (VIP) address and a TCP port, which listens and directs the traffic to a real server associated with the service.

Create a Service

To create a service, make a POST request to the following URI:

/virtual_service_groups/*<group name>*/virtual_services

where *<group name>* specifies the name of the service group that you are associating the service with. The body of your request must be JSON, with the Content-Type header set to application/json. In the body of

your request, pass the parameters listed in the following table:

Parameter	Data Type	Mandatory	Description
name	String	Yes	The name of the service.
address_version	Enum	Yes	The Internet protocol version of the service. Possible values include: • **ipv4** • **ipv6**
ip_address	String	Yes	The virtual IP address. The IP address format depends on the specified **`address_version`**.
port	Integer	Yes	The port number for the service.
type	Enum	Yes	The type of the service. Possible values: • **HTTP** • **HTTPS** • **INSTANTSSL** • **FTP** • **FTPSSL** • **UDP** • **L4** • **L7UDP** • **L7Tcp RDP**
netmask	String	Yes	The netmask depends on the **`address_version`** specified.
interface	Enum	Yes	The interface for the service. The value depends on the appliance. For example, ge-1-1, ge-1-2, and ge-2-1.
service_hostname	String	Conditional	The domain name to identify and rewrite HTTP requests to HTTPS. **Note:** Required only for Instant SSL services.
certificate	String	Conditional	The certificate that is presented by the service when it authenticates itself to a browser or other client. **Note:** Required only for HTTPS, Instant SSL, FTP SSL, and SSL services.

For a complete list of parameters that you can pass, see Virtual Service Parameters.

Request and Response Examples

The following examples display a request and response for creating a service named **service1** in the default service group.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"service1", "ip_address":"12.17.1.14", "port":"4", \
      "address_version":"ipv4", "type":"http", "netmask":"255.255.255.255", \
      "interface":"ge-1-1"}'
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services
```

Example Response

```
{
  "info": {
      "msg": [
        "Configuration updated"
      ]
  },
  "id": "service1",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Update a Service

To update a service, make a PUT request to the following URI:

/virtual_service_groups/*<group name>*/virtual_services/*<service name>*

where:

- *<group name>* is the service group that the service is associated with.
- *<service name>* is the service to update.

The body of your request must be JSON, with the Content-Type header set to application/json. Only the values of the parameters that are passed in the body of your request are changed or added. For a complete list of parameters that you can pass, see Virtual Service Parameters.

Request and Response Examples

The following examples display a request and response for updating the netmask for a service named **service1** in the default service group.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X PUT \
   -H "Content-Type:application/json" \
   -d '{"netmask":"255.255.255.0"}' \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
  /virtual_services/service1
```

Example Response

```
{
  "info": {
      "msg": [
        "Configuration updated"
      ]
  },
  "id": "service1",
"token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Services

To retrieve data for all the services that are included in a service group, make a GET request to the following URI:

/virtual_service_groups/*<group name>*/virtual_services

where *<group name>* is the name of the service group.

To retrieve data for only a specific service in the service group, include the name of the service in the URI:

/virtual_service_groups/*<group name>*/virtual_services/*<service name>*

Request and Response Examples

The following examples display a request and response for retrieving a service named **service1** in the default service group.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X GET \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1 \
```

Example Response


```
{
    "adaptive_param": "",
    "address_version": "ipv4",
    "auto_recover": true,
    "cache": {
        "enabled": false,
        "expiry_age": "60",
        "file_extensions": [
            "gif",
            "tif",
            "jpg",
            "jpeg",
            "png",
            "bmp",
            "ico",
            "js",
            "jsp",
            "css",
            "jar",
            "swf",
            "pdf"
        ],
        "max_objsize": "256",
        "min_objsize": "256",
        "negative_responses": false,
        "req_cachehdrs_ignore": false,
        "resp_cachehdrs_ignore": false
……
    },
    "interface": "ge-1-1",
    "ip_address": "172.33.33.33",
    "keepalive_request": "64",
    "lb_adaptive_scheduling": "NONE",
    "lb_policy": "weighted_round_robin",
    "load_threshold": "35",
    "minimum_notificate_real_server": "0",
    "name": "servicel",
    "netmask": "255.255.255.0",
    "notification_enable": false,
    "persistence": {
        "cookie_age": null,
        "cookie_domain": null,
        "cookie_httponly": "1",
        "cookie_name": "persistence",
        "cookie_path": null,
        "cookie_security": "0",
        "failover_method": "LB",
        "header_name": "",
        "netmask": null,
        "param_name": null,
        "timeout": "600",
        "type": "NONE"
    },
    "polling_interval": "300",
    "port": "443",
    "protocol": "TCP",
    "redirect_port": "80",
```

```
    "security": {
        "enabled": "0",
        "mode": "PASSIVE",
        "trusted_host_action": "DEFAULT",
        "trusted_host_group": null,
        "web_firewall_log_level": "5",
        "web_firewall_policy": "default"
.....
    },
    "tcp_keepalive_timeout": false,
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n",
    "type": "INSTANTSSL"
}
```

Delete a Service

To delete a service, make a DELETE request to the following URI:

`/virtual_service_groups/<group name>/virtual_services/<service name>`

where:

- *<group name>* is the service group that the service is associated with.
- *<service name>* is the service to delete.

Request and Response Examples

The following examples display a request and response for deleting a service named **`service1`** in the default service group.

Example Request

```
$ curl http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
  /virtual_services/service1 \
   -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X DELETE
```

Example Response

```
{
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "msg": "Successfully deleted",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Virtual Service Parameters

The tables in this section list the parameters that you can configure for services.

Service Configuration

Parameter	Data Type	Description
name	String	The name of the service.
address_version	Enum	The Internet protocol version of the service. Possible values: • **ipv4** • **ipv6**
ip_address	String	The virtual IP address. The IP address format depends on the specified **`address_version`**.
port	Integer	The port number for the service.
type	Enum	The type of the service. Possible values include: • **HTTP** • **HTTPS** • **INSTANTSSL** • **FTP** • **FTPSSL** • **UDP** • **L4** • **L7UDP** • **L7Tcp RDP**
netmask	String	The netmask depends on the **`address_version`** specified.
interface	Enum	The interface for the service. The value depends on the appliance. For example, ge-1-1, ge-1-2, and ge-2-1.
service_hostname	String	The domain name to identify and rewrite HTTP requests to HTTPS. **Conditional:** Required only for Instant SSL services.
certificate	String	The certificate that is presented by the service when it authenticates itself to a browser or other client. **Conditional:** Required only for HTTPS, Instant SSL, FTP SSL, and SSL services.
redirect_port	Integer	The HTTP redirect port for an Instant SSL service.
enable	Boolean	Enable the service. Possible values: • **true** • **false**

Parameter	Data Type	Description
auto_recover	Boolean	Automatically re-enables the real servers after they are detected as unavailable. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enable_access_log	Boolean	Logs every request made to this service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
session_timeout	Integer	The time-out period in seconds for persistent connections with clients. Zero (**0**) indicates that the session never times out (session lives forever).
comments	String	Description about the service.

Load Balancing

Parameter	Data Type	Description
lb_policy	Enum	How traffic is distributed among the real servers associated with the service. Possible values: • **weighted_round_robin** • **weighted_least_connection**
lb_adaptive_scheduling	Enum	Whether the weight of each real server is adjusted according to its CPU usage, the number of active Terminal sessions it has, or from information polled at the LOAD URL. Possible values: • **None** – Uses the static preconfigured real server weights. Past connections / UDP datagrams are not considered when directing clients to servers. • **SNMP_CPU** – Polls the SNMP OID for CPU load and manipulates the real server weights accordingly. To use this option, real servers must allow the Barracuda Load Balancer ADC SNMP access to the public community. **Dependent variables:** ○ adaptive_param ○ snmp_cpu_load_oid • **LOAD_URL** – Polls a specified URL, expecting the output to look like LOAD=23 (showing the load as an integer between 0 and 100). For this method to work, each real server must be running a web

Parameter	Data Type	Description
		server that responds to the poll at the real server's IP address and port 80. **Dependent variable:** ○ adaptive_param • **SNMP_TS_SESSIONS** – Redistributes connections between Windows Terminal Servers based on the number of sessions per server. **Dependent variable:** ○ adaptive_param
adaptive_param	String	**Conditional:** The value of this parameter differs for each adaptive scheduling type: • If the **lb_adaptive_scheduling** is **SNMP_CPU** or **SNMP_TS_SESSIONS**, enter the community string. • If the **lb_adaptive_scheduling** is **LOAD_URL**, enter the URL to poll the server for information.
last_resort_action	Enum	The action to take if all Real Servers configured within a Load Balancer ADC service become unavailable. The following are the possible values: • **default** – Varies depending on the service type: ○ For Layer 4 TCP and UDP services, the client will fail to connect. ○ For Layer 7 services, a 503 default error page is sent to the client. • **reset_conn** – Forces a reset of the connection to the Real Server. Select this option if you do not have a backup server configured for the service. When the Real Servers become unavailable, either the connection is closed (TCP) or an ICMP port unreachable error is returned to the client (UDP).
load_threshold	Integer	The maximum acceptable real server load. If this load limit is exceeded, the weight of the real server is adjusted according to the adaptive scheduling policy.
polling_interval	Integer	The interval between test start times, and also the maximum time that tests are given to complete, in seconds, for adaptive scheduling.
snmp_cpu_load_oid	String	The CPU Load OID to use for SNMP CPU adaptive scheduling.

Instant SSL

To configure an Instant SSL service, you must configure the following parameters for the service.

Parameter	Data Type	Description
instantssl.status	Boolean	Rewrites `http` links in responses to `https`. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
instantssl.host	String	The domains that are rewritten from `http` to `https` in responses, if **`instantssl.status`** is **`true`**. To include all domains, enter an asterisk (*). For example, if you enter **`www.example.com`**, any instances of **`http://www.example.com`** in outgoing responses are rewritten to **`https://www.example.com`**. Ensure that the certificate that you upload for this service is valid for the secure site domain.
instantssl.sharepoint_support	Boolean	Enables SharePoint rewrite support. Normally, an Instant SSL service rewrites http links in responses to https using HTML tags, like href. But SharePoint applications also insert hyperlinks outside of the basic HTML tags. To rewrite those links to https, set value to **`true`**. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Caching

Caching is available for only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
cache.enabled	Boolean	Enables caching to local memory for quick retrieval so some requests do not have to be sent to the web server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.file_extension	Array	A list of the extensions of the file types to be cached.
cache.expiry_age	Integer	The default expiration age in minutes, to be used when the response does not have an `Expires` header. A value of **0** (expiry-age = 0) sets the expiry age to 60 minutes.
cache.max_objsize	Integer	The maximum size of files that can be cached.

Parameter	Data Type	Description
cache.min_objsize	Integer	The minimum size of files that can be cached.
cache.req_cachehdrs_ignore	Boolean	Ignores HTTP request cache-control headers to instruct upstream caching servers. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.resp_cachehdrs_ ignore	Boolean	Ignores HTTP response cache-control headers to instruct upstream caching servers. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.negative_responses	Boolean	Caches HTTP negative responses with status codes 204, 305, 404, 405, 414, 501, 502, and 504. Only negative responses with the **`Expires`** or **`Last Modified`** headers are cached. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Compression

Compression is available for only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
compress.enabled	Boolean	Enables compression of the specified content types. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
compress.content_types	Array	A list of content types to be compressed. For example: **`text/css`** **`text/html`** **`text/js`** **`text/plain`**
compress.min_obj_size	Integer	The minimum size of objects that can be compressed, from **`1 to2147483646`** in bytes. The default value is **`8192.`**

Service Monitoring

Parameter	Data Type	Description

service_monitor.test	Enum	The test that is used by the server monitor to determine the availability of the servers that are associated with the service. You can use either a monitor group or a test type.
		
		To specify a Error! Not a valid result for table., use the following syntax:
		`GROUP_<group name>`
		For example, to use a monitor group named `Group 1`:
		`"service_monitor":{"test":"GROUP_Group1 "}`
		
		To select a test type, select one of the following values:
		○ **ICMP_PING** – Performs a PING test.
		○ **TCP_PORT_CHECK** – Validates that the configured service port is open.
		Dependent variable:
		○ service_monitor.port
		• **UDP_PORT_CHECK** – Sends a 0 byte datagram to the IP address and port of the real server to verify that the UDP port is open. Waits to receive an ICMP Port Unreachable message to determine the result. If there is a firewall prevents outbound ICMP messages, the port is assumed to be open.
		Dependent variable:
		○ service_monitor.port
		• **HTTP_TEST** – Sends an HTTP request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled.
		Dependent variables:
		○ service_monitor.port
		○ service_monitor.target
		○ service_monitor.match
		○ service_monitor.method
		○ service_monitor.hosts
		○ service_monitor.code
		• **SIMPLE_HTTP** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code.
		Dependent variables:
		○ service_monitor.port

Parameter	Data Type	Description
		○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTP_SLOW** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_TEST** – Sends an HTTPS request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_SLOW** – Sends an HTTPS request to the specified relative URL to verify that the response contains that the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTPS** – Sends an HTTPS request to the

Parameter	Data Type	Description
		specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **DNS_TEST** – Sends a DNS query and compares the returned IP address to the entered IP address. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **IMAP_TEST** – Simple test for IMAP service. If no username and password are provided, this test verifies only the availability of the service on the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **POP_TEST** – Simple test for POP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SMTP_TEST** – Simple test for SMTP servers. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SNMP_TEST** – Sends an SNMP GET to the specified OID to verify that the response contains an expected pattern. If an OID is not specified, this test verifies only the availability of the service on the server. **Dependent variable:** ○ service_monitor.port ○ service_monitor.target

Parameter	Data Type	Description
		○ service_monitor.match • **SIP_TEST** – Simple test for SIP service. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service. **Dependent variable:** ○ service_monitor.port • **SIP_TLS_TEST** – Simple Test for SIP service over TLS. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service over TLS. **Dependent variable:** ○ service_monitor.port • **LDAP_AD_TEST** – Bind test for LDAP/AD service. If no username and password are provided, the LDAP/AD test verifies availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **LDAP_AD_SSL_TEST** – Bind test for LDAPS/AD service. If no username and password are provided, the LDAP/AD test verifies availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SPAMFIREWALL_TEST** – Test for use with Barracuda Spam Firewalls. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FORCED_PORT_HTTP** – (Specific HTTP Port test) Sends an HTTP GET request using the specified port to a relative URL on the real server, to verify that the retrieved HTML contains an expected pattern. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match

Parameter	Data Type	Description
		• **RADIUS_TEST** – Verifies the availability of a RADIUS server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_ACCT** – Tests the availability of a RADIUS server by making an accounting request. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RDP_TEST** – Attempts an RDP connection to each real server to verify the availability of the Terminal Service. **Dependent variable:** ○ service_monitor.port • **FTP_TEST** – Attempts a TCP connection to each real server to check FTP availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FTPS_TEST**– Attempts a TCP connection to each real server to verify FTPS availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SFTP_TEST** – Test for FTP over SSH. **Dependent variables:** ○ service_monitor.port ○ service_monitor.username ○ service_monitor.password • **NTLM_TEST** – Simple test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.username ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **NTLMS_TEST** – Secure test for Microsoft

Parameter	Data Type	Description
		SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.username ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **ALWAYS_PASS** Used to troubleshooting or for services with management access to real servers. This test always passes regardless of the condition of the real server.
service_monitor.port	Integer	By default, tests use the configured real server port for the service unless the real server port is set to ALL. In that case, tests use the default port for the test type (e.g., SMTP = 25, HTTP = 80, DNS = 53, HTTPS = 443, IMAP = 143, POP = 110, and SNMP = 161). Enter a port number to override the default port.
service_monitor.username	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The username for the SSH account. • MS SharePoint – The username for the SharePoint service.
service_monitor.password	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The password for the SSH account. • MS SharePoint – The username for the SharePoint service.
service_monitor.target	String	**Conditional:** The value for this parameter differs for each test type: • Barracuda Spam Firewall Test – The domain for the mail server. • DNS – The fully qualified domain name. • FTP – The username to log into server. The default username is **`anonymous`**. • FTPS – The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service on the server. • HTTP and HTTPS – The complete URL (starting with **`http`** or **`https`**). • HTTP Slow and HTTPS Slow – The complete URL (starting with **`http`** or **`https`**). • IMAP and POP3 – (Optional) The username to log

Parameter	Data Type	Description
		into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service on the server. • LDAP and LDAPS – (Optional) The username with full LDAP schema. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The root relative URL. For example: **`/cgi-bin/index.cgi`** • RADIUS Auth and RADIUS Acct – The secret to use with the RADIUS server. • Simple HTTP and Simple HTTPS – The root relative URL. For example: **`/cgi-bin/index.cgi`** The actual URL used is **`https://<real_server_ip>:<port><URL>`** • SNMP – (Optional) Enter a valid SNMP OID. **Note:** If an OID is not specified, this test verifies only the availability of the service in the server. • Specific HTTP Port – The TCP port followed by colon (:) and the root relative URL. For example: **`8080:/cgi-bin/index.cgi`**.
service_monitor.match	String	**Conditional:** The value for this parameters differs for each test type: • Barracuda Spam Firewall – (Optional) A pattern that is expected in the banner of the SMTP server. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. • DNS – The IP address of the hostname. • FTP – The password to log into the server. The default password is **`anonymous`**. • FTPS – The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service on the server. • HTTP and HTTPS – The expected pattern in the retrieved HTML. • HTTP Slow and HTTPS Slow – The expected pattern in the retrieved HTML. • IMAP and POP3 – (Optional) The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in

Parameter	Data Type	Description
		the server. • LDAP and LDAPS – (Optional) The password to log into the server. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The pattern that is expected in the retrieved HTML. • RADIUS Auth – The username and password, separated by a backslash (****). For example: **`username\password`** • Simple HTTP and Simple HTTPS – The pattern expected in the retrieved HTML. • SMTP – (Optional) The pattern that is expected in the banner for the SMTP server. • SNMP – (Required if an OID is specified in the test target) A pattern to match in the response. • Specific HTTP Port – The pattern that is expected in the retrieved HTML.
service_monitor.hosts	String	Additional headers. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
service_monitor.code	Integer	The expected status code. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
service_monitor.test_delay	Integer	The interval between test start times, in seconds.
service_monitor.method	Enum	The method for HTTP and HTTPS tests. Possible values: • **GET** • **HEAD** • **POST** **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, and HTTPS Slow tests.
server_monitor.post_body	String	The data that is being sent in the POST request. **Conditional:** Required when the **`server_monitor.method`** is **`POST`**.

Persistence

Parameter	Data Type	Description

Parameter	Data Type	Description
persistence.type	Enum	How clients are redirected to the real server that they were initially routed to, after a period of inactivity. Possible values: • **NONE** – Past connections / UDP datagrams are not considered when directing clients to servers. • **SCRIPNETMASK** – Directs requests based on the client IP address. Requests from clients that are grouped by the persistence netmask are routed to the same real server as the first request received from the group. **Dependent variable:** ○ persistence.netmask • **COOKIEINSERT** – Directs requests based on a cookie (BNI_cookie_name) that is inserted by the Barracuda Load Balancer ADC after it routes the first request from the client. **Dependent variables:** ○ persistence.cookie_name ○ persistence.cookie_domain ○ persistence.cookie_path ○ persistence.cookie_age • **COOKIEPASSIVE** – Directs requests based on a cookie (BNI_cookie_name) that is inserted in the response by the Barracuda Load Balancer ADC only if the real server inserts a cookie. **Dependent variables:** ○ persistence.cookie_name ○ persistence.cookie_domain ○ persistence.cookie_path ○ persistence.cookie_age • **HEADERFIELD** – Directs all incoming HTTP requests based on the HTTP header. **Dependent variable:** ○ persistence.header_name • **URLPARAM** – Directs all incoming HTTP requests based on the specified parameter name. **Dependent variable:** ○ persistence.param_name
persistence.timeout	String	For HTTP/S services, the expiration age for the cookie is set to the current time plus this value whenever the browser sends a request. If the browser does not honor the expiry age and sends the cookie even after its expiration, the same real server is used. For all other service types, this is the maximum length

Parameter	Data Type	Description
		of time that a client can remain idle during a persistent session and still be redirected to the same real server.
persistence.failover_method	Enum	How to handle a request in a persistent session when the server that must respond to the request is unavailable. Possible values: • **load_balance** – The requests are load balanced among the remaining servers in the pool. • **error** – A "503 service unavailable" error message is sent.
persistence.netmask	String	The netmask for persistence using source IP addresses (including Layer 4-UDP). A more specific netmask (such as 255.255.255.255) tracks each client independently and can cause a higher memory load on the Barracuda Load Balancer ADC. A less specific netmask (such as 255.255.0.0) organizes multiple clients under the same network identifier and connects them all to the same server.
persistence.cookie_name	String	The name of the cookie used for persistence.
persistence.cookie_domain	String	The domain property of the cookie. If blank, it is the domain name in the request. **Conditional:** Required only when the **persistence.type** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence.cookie_path	String	The path where the cookie is valid.
persistence.cookie_age	String	The maximum age for the session cookie, in seconds. **Conditional:** Required only when the **persistence.type** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence.header_name	String	The name of the header value to verify in the HTTP requests. **Conditional:** Required only when the **persistence.type** is **HEADERFIELD**.
persistence.param_name	String	The name of the parameter value to verify in the URL. **Conditional:** Required only when the **persistence.type** is **URLPARAM**.
persistence.cookie_security	Boolean	Transmits the cookie over only HTTPS connections. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Parameter	Data Type	Description
persistence.cookie_httponly	Boolean	Makes the cookie inaccessible by client-side scripts, if supported by the browser. Helps mitigate most common cross-site scripting (XSS) attacks. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Notifications

Parameter	Data Type	Description
notification_enable	Boolean	Generates an alert whenever a real server goes up or down. The alert is emailed to the system alerts email address entered on the **BASIC > Administration** page. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false **Dependent variable:** minimum_notificate_real_server
minimum_notificate_real_server	Integer	The minimum number of operating real servers for this service. If this number is not met, an alert is generated. If you enter **0**, no alerts are generated.

Headers and URLs

The following parameters are available for only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
client_ip_addr_header	String	The HTTP header name (e.g., X-Forwarded-For or X-Client-IP) to be used in logs as a substitute for any client IP addresses in the request headers.
ignore_case	Boolean	Ignores the letter case in URLs when processing rules (e.g., content rules, HTTP request and response rewrite rules and response body rewrite rules, and the URL and Header allow/deny rules). Only the letter case of the URLs is ignored; parameter names are unaffected. **Recommended settings:** • For Windows-based servers: **true** • For Linux- and Unix-based servers: **false**

Parameter	Data Type	Description
		For Perl, possible values are: • **1** – true • **0** – false

Keepalive and Timeouts

The following parameters apply to only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
keepalive_request	Integer	The maximum number of requests allowed on a persistent HTTP connection. A value of **0** does not enforce any limit, allowing the client to control the number of requests on the connection.
tcp_keepalive_timeout	Integer	The maximum time that connections with clients can remain idle, in seconds, before timing out. Default setting: **64** **Note:** A value of 0 indicates that the session never times out. Only enter 0 if the URL takes so long to return that it is difficult to set a max timeout value.

SSL Offloading

The following parameters apply to only secure service types (i.e., HTTPS, Instant SSL, FTP SSL, and Secure TCP Proxy).

Parameter	Data Type	Description
ssl_offloading.status	Boolean	Enables SSL offloading. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.certificate	String	The certificate to be presented to any browser accessing the service. **Conditional:** Requiredwhen **`ssl_offloading.status`** is **`true`**.
ssl_offloading.enable_ssl_3	Boolean	Allows clients to use SSL 3.0 to connect to the service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.ssl_enable_tls	Boolean	Allows clients to use SSL TLS 1.0 to connect to the service. Possible values: • **true** • **false**

Parameter	Data Type	Description
		For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.ssl_enable_tls_1_1	Boolean	Allows clients to use SSL TLS 1.1 to connect to the service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.ssl_enable_tls_1_2	Boolean	Allows clients to use SSL TLS 1.2 to connect to the service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.sni	Boolean	Enables Server Name Indication (SNI), an extension of SSL and TLS protocols. Allows a client to request a certificate for a specific domain from a server hosting more than one domain. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.sni_strict	Boolean	Blocks non-SNI clients. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
ssl_offloading.sni_domains	Array	The domain name for which the SNI check is enforced. You can specify multiple domain names with a comma (**,**) as a delimiter without any spaces.
ssl_offloading.sni_certs	Array	The certificates associated with the specified domain names. **Conditional:** Required when SNI is enabled.

Parameter	Data Type	Description
ssl_offloading.ciphers	Enum	The ciphers that are used for the service. Possible values: • **default –** Includes all available ciphers. • **custom –** Includes a select list of available ciphers. **Dependent variable:** ○ ssl_offloading.cipher_list
ssl_offloading.cipher_list	Array	The list of SSL ciphers to use. Possible values: • **ECDHE-RSA-AES256-GCM-SHA384** • **ECDHE-ECDSA-AES256-GCM-SHA384** • **ECDHE-RSA-AES256-SHA384** • **ECDHE-ECDSA-AES256-SHA384** • **ECDHE-RSA-AES256-SHA** • **ECDHE-ECDSA-AES256-SHA** • **AES256-GCM-SHA384** • **AES256-SHA256** • **AES256-SHA** • **CAMELLIA256-SHA** • **ECDHE-RSA-AES128-GCM-SHA256** • **ECDHE-ECDSA-AES128-GCM-SHA256** • **ECDHE-RSA-AES128-SHA256** • **ECDHE-ECDSA-AES128-SHA256** • **ECDHE-RSA-AES128-SHA** • **ECDHE-ECDSA-AES128-SHA** • **AES128-GCM-SHA256** • **AES128-SHA256** • **AES128-SHA** • **CAMELLIA128-SHA** • **SEED-SHA** • **IDEA-CBC-SHA** • **ECDHE-RSA-RC4-SHA** • **ECDHE-ECDSA-RC4-SHA** • **RC4-SHA** • **ECDHE-RSA-DES-CBC3-SHA** • **ECDHE-ECDSA-DES-CBC3-SHA** • **DES-CBC3-SHA** **Conditional:** Requiredwhen `ssl_offloading.ciphers` is `custom`.

Parameter	Data Type	Description
ssl_offloading.enforce_client_certificate	Boolean	Requires clients to present a certificate or the SSL handshake is terminated. Possible values: • **true** • **false**
ssl_offloading.enable_client_authentication	Boolean	Requires users connecting to this site to present a trusted certificate for validation. Possible values: • **true** • **false** **Conditional:** Required when `ssl_offloading.enforce_client_certificate` is `true`.
ssl_offloading.trusted_certificates	Array	A list of trusted certificates.

Security Policy

The following parameters are apply to only HTTP, HTTPS, and Instant SSL services.

Parameter	Data Type	Description
security.enabled	Boolean	Enforces the conifgured security policy. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
security.mode	Enum	Determines handling of anomalies and intrusions. Possible values: • **ACTIVE** – Blocks requests when an anomaly or intrusion is detected. • **PASSIVE** – Logs all detected anomalies and intrusions but allows traffic to pass through the Barracuda Load Balancer ADC. Use this mode in the initial stages of deployment, to prevent false positives from paralyzing the service.
security.web_firewall_policy	Enum	For HTTP, HTTPS, and Instant SSL services, you can assign a security policy. Either create a custom policy or use one of the following predefined security policies: • **default** • **sharepoint** • **sharepoint2013** • **owa** • **owa2010**

Parameter	Data Type	Description
		• **owa2013** • **oracle**
security.trusted_host_group	String	The trusted hosts group for this service.
security.web_firewall_log_level	Enum	The threshold for logging error messages for the service. Possible values: • **0** – Emergency: system is unusable (highest priority) • **1** – Alert: response needed immediately • **2** – Critical: critical conditions • **3** – Error: error conditions • **4** – Warning: warning conditions • **5** – Notice: normal but significant condition • **6** – Information: informational messages (on ACL configuration changes) • **7** –Debug: debug level messages (lowest priority)

FTP Passive

The following parameters apply to only FTP and FTP SSL services.

Parameter	Data Type	Description
ftp_pasv_config.ftp_aps_status	Boolean	Enables a security policy to only allow specific FTP verbs. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false You can edit the list of allowed FTP verbs for this service on the **SECURITY > FTP Security** page.
ftp_pasv_config.ftp_pasv_ip_address	String	The IP address that the FTP application uses to open an FTP data connection after receiving a PASV request from the client. This IP address is used only for PASV requests from FTP clients; it is ignored for PORT requests. • If you leave this field empty, the FTP application uses the virtual IP address of the service. • If a NAT'ing firewall sits in front of the Barracuda Load Balancer ADC, enter the public IP address that translates to the virtual IP address of this service.

Parameter	Data Type	Description
ftp_pasv_config.ftp_pasv_port_ranges	String	Port numbers that the FTP application uses to respond to a PASV request from the FTP client. You can enter a single port number or a range of ports where the start port and end port are separated by a hyphen (-). If you enter a port range, it includes the start and end port. • If a firewall sits in front of the Barracuda Load Balancer ADC, typically you would enter the ports that are allowed by the firewall. • If you leave this field blank, any available port is used.

Certificates

A signed certificate is a digital identity document that enables both server and client to authenticate each other. Certificates are used with the HTTPS protocol to encrypt secure information transmitted over the Internet. A certificate contains information such as username, expiration date, a unique serial number assigned to the certificate by a trusted CA, the public key, and the name of the CA that issued the certificate.

You can generate a self-signed certificate, upload a self-signed certificate, upload a trusted self-signed certificate, and download a certificate.

Generate a Self-Signed Certificate

To generate a self-signed certificate, make a POST request to the following URI:

```
/certificates
```

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the parameters listed in the following table:

Parameter	Data Type	Mandatory	Description
name	String	Yes	The name of the certificate.
common_name	String	Yes	The domain name (DN) of the web server for the certificate.
country_code	String	Yes	The two-letter country code of the location of the organization.
state	String	Optional	The full name of the state or province of the location of the organization.
city	String	Optional	The full name of the locality (city) where the organization is located.
organization_name	String	Optional	The legally registered name of the organization or company.

Parameter	Data Type	Mandatory	Description
organization_unit	String	Optional	The department or unit within the organization.
key_size	Enum	Yes	The private key size for the certificate in bits. Possible values: • **1024** • **2048** • **4096**
allow_private_key_export	Boolean	Yes	Locks the Private Key corresponding to this certificate. Possible values: • **true –** Export the private key corresponding to the certificate. • **false –** Lock the private key. In this case, the certificate can be downloaded only in PEM format and a backup of the system configuration cannot be taken. For Perl, possible values are: • **1** – true • **0** – false Normally, certificates are downloaded in PKCS #12 format which includes the Private Key and Certificate. When a key is locked, you can only download the certificate in PEM format. Also, you cannot take a backup when the Private Key is locked. **Note:** This option is valid only for created and uploaded (generated and signed by a trusted CA) certificates.

Request and Response Examples

The following examples display a request and response for generating a signed certificate named **sign_cert**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"sign_cert", "common_name":"barracuda.yourdomain.com", \
      "country_code":"US", "state":"CA", "city":"Campbell", \
      "organization_name":"Barracuda", "organization_unit":"Engineering",\
      "key_size":"1024", "allow_private_key_export":"true"}' \
   http://10.11.19.104:8000/restapi/v2/certificates
```

Example Response

```
{
    "id": "sign_cert",
    "info": {
        "msg": [
            "Successfully created certificate."
        ]
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Upload a Signed Certificate

To upload a signed certificate in PEM or PKCS #12 format, make a POST request to the following URI:

`/certificates?upload=signed`

When specifying the parameters for the certificate, use the `-F` option for `curl` to POST data as multipart/form-data. The `upload` field is passed in the URI. In the body of your request, pass the parameters listed in the following table:

Parameter	Data Type	Mandatory	Description
name	String	Yes	The name of the certificate.
type	Enum	Yes	The certificate type. Possible values: • **pkcs** • **pem** **Dependent variables:** • intermediary_certificate • password
signed_certificate	Cert	Yes	The path and name of the signed certificate file that must be uploaded.
password	String	Conditional	The password used to generate the PKCS #12 token for the signed certificate being uploaded. **Note:** Required only for uploading certificates in PKCS #12 Token format.

Parameter	Data Type	Mandatory	Description
assigned_associated_key	Boolean	Conditional	Possible values: • **true** –The CSR corresponding to this certificate was generated on the Barracuda Load Balancer ADC. • **false** – Upload the private key corresponding to this certificate specified by the **`key`** parameter. For Perl, possible values are: • **1** – true • **0** – false **Note:** Required only for uploading certificates in PEM format.
key	Public Key	Conditional	The path and name of the corresponding private key for the signed certificate being uploaded. **Note:** Required only for uploading certificates in PEM format.
intermediary_certificate	Array	Conditional	The path and name of the intermediary CA certificate file to upload. If your certificate is signed by a trusted CA, upload the certificate in the following order: 1. Leaf certificate 2. Intermediate certificate(s) 3. Root CA certificate **Note:** Required only for uploading certificates in PEM format.
allow_private_key_export	Boolean	Yes	Exports the private key corresponding to the certificate. Possible values: • **true** – Export the private key corresponding to the certificate. • **false** – Lock the private key. In this case, the certificate can be downloaded only in PEM format and a backup of the system configuration cannot be taken. For Perl, possible values are: • **1** – true • **0** – false

Request and Response Examples

The following examples display a request and response for uploading a signed certificate named **`sign_cert`**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X POST \
   -F name="sign_cert" \
   -F type="pem" \
   -F signed_certificate=@server.crt \
   -F assign_associated_key=false \
   -F key=@server.key \
   -F allow_private_key_export=true \
   http://10.11.19.104:8000/restapi/v2/certificates?upload=signed
```

Example Response

```
{
   "info": {
       "message": [
             "Configuration updated"
       ],
       "status": 201, "type": "Success",
       "params": {},
       "content_type": "application/json"
   },
   "id": "sign_cert",
   "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Upload a Trusted Certificate

To upload a trusted certificate in PEM format, make a POST request to the following URI:

`/certificates?upload=trusted`

To specify the parameters for the certificate, use the **`-F`** option for **`curl`** to POST data as multipart/form-data. The **`upload`** field is passed in the URI. In the body of your request, pass the following parameters:

Parameter	Data Type	Mandatory	Description
name	String	Yes	The name of the certificate.
trusted_certificate	String	Yes	The path and name of the trusted server certificate that must be uploaded.

Request and Response Examples

The following examples display a request and response for uploading a trusted certificate named **`trust_cert`**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X POST \
   -F name="trust_cert" \
   -F trusted_certificate=@mycert.crt \
   http://10.11.19.104:8000/restapi/v2/certificates?upload=trusted
```

Example Response

```
{
  "info": {
      "message": [
            "Configuration
  updated"
      ],
      "status": 201,
      "type": "Success",
      "params": {},
      "content_type": "application/json"
  },
  "id": "trust_cert",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

DownloadaCertificate

To download a certificate, make a GET request to the following URI:

/certificates/<certificate name>

In the body of your request, pass the following parameters:

Parameter	Data Type	Mandatory	Description
download	Enum	Yes	The type of certificate. Possible values: • **trusted** • **signed** • **private**
encrypt_password	String	Yes	The password to save the certificate

Request Example

The following example displays a request for downloading a certificate named **cert**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X GET \
   -G \
   -d download=signed \
   -d encrypt_password=test \
   -O \
   http://10.11.19.104:8000/restapi/v2/certificates/cert
```

After you issue the request, the certificate downloads to your current directory, unless you specify another folder.

Servers

You can add and configure multiple back-end servers to load balance incoming traffic for a service.

Create a Server

To create a server for a service, make a POST request to the following URI:

/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*/servers

where:

- *<group name>* is the service group that the service was added to.
- *<service name>* is the name of the service.

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the following parameters:

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name to identify the server.
identifier	Enum	Yes	How the Barracuda Load Balancer ADC identifies the server. Possible values: • **hostname** • **ip_address**
ip_address	String	Conditional	The IP address of the server. **Note:** Required when the **identifier** is **ip_address**.
hostname	String	Conditional	The hostname of the server. **Note:** Required when the **identifier** is **hostname**.
port	String	Yes	The port number of the server.

For a complete list of parameters that you can pass, see Server Parameters.

Request and Response Examples

The following examples display a request and response for adding a server named **new_server** for a service named **service1** in the default service group.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"new_server", "identifier":"ipaddr", \
       "ip_address":"10.66.44.104", "port":"80"}' \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/servers
```

Example Response

```
{
  "info": {
    "msg": [
      "Configuration updated"
    ]
  },
  "id": "new_server",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Update a Server

To update a server for a service, make a PUT request to the following URI:

/virtual_service_groups/<group name>/virtual_services/<service name>
/servers/<server name>

where:

- *<group name>* is the service group that the service was added to.
- *<service name>* is the service of the server.
- *<server name>* is the server to update.

The body of your request must be JSON, with the Content-Type header set to application/json. Only the values of the parameters that are passed in the body of your request are updated. For a complete list of parameters that you can pass, see Server Parameters.

Request and Response Examples

The following examples display a request and response for enabling direct server return and changing the weight for a server named **test_server**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -H "Content-Type:application/json" \
   -X PUT \
   -d '{"route":"direct","weight":"100"}' \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/servers/test_server
```

Example Response

```
{
  "info": {
                         "msg": [
      "Configuration updated"
    ]
  },
  "id": "test_server",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Servers

If you want to retrieve data for all the servers that are associated with a service, make a GET request to the following URI:

/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*/servers

where:

- *<group name>* is the name of the service group that the service was added to.
- *<service name>* is the service.

If you want data for only a specific server, add the server name to the URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*\
/servers/*<server name>***

Request and Response Examples

The following examples display a request and response for retrieving a server named **test_server**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X GET \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/servers/test_server
```

Example Response


```
{
    "address_version": "ipv4",
    "backup_server": false,
    "client_certificate": "test_gen_cert",
    "comments": null,
    "enable_client_impersonation": false,
    "enable_connection_pooling": true,
    "enable_https": false,
    "enable_ssl_3": true,
    "enable_tls_1": true,
    "enable_tls_1_1": true,
    "enable_tls_1_2": true,
    "hostname": null,
    "id": "testl4server",
    "identifier": "ipaddr",
    "index": "md5zDwbE71gMws3RHrGB41UFQ",
    "ip_address": "10.13.13.3",
    "keepalive_timeout": "900000",
    "max_connections": "10000",
    "max_establishing_connections": "100",
    "max_keepalive_requests": "0",
    "max_request": "1000",
    "max_spare_connections": "0",
    "name": "test_server",
    "operational_status": "out-of-service-health",
    "port": "21",
    "route": "direct",
    "scope_data": "testft:10.13.13.3_21",
    "server_monitor": {
        "method": "GET",
        "post_body": "",
        "retries": "1",
        "test": "",
        "test_delay": "10"
    },
    "status": "enable",
    "timeout": "300000",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n",
    "traffic": 0,
    "validate_certificate": "0",
    "weight": "1"
}
```

Delete a Server

To delete a server from a service, make a DELETE request to the following URI:

```
/virtual_service_groups/<group name>/virtual_services/<servicename>\
/servers/<server name>
```

where:

- *<group name>* is the service group that the service was added to.
- *<service name>* is the service of the server.
- *<server name>* is the server to delete.

Request and Response Examples

The following examples display a request and response for deleting a server named **test_server**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/servers/test_server
```

Example Response

```
{
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "msg": "Successfully deleted",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Server Parameters

You can configure the following parameters for each real server.

Server Configuration

Parameter	Data Type	Description
name	String	A name to identify the server.
address_version	Enum	The internet protocol version to be used. Possible values: • **ipv4** • **ipv6**

Parameter	Data Type	Description
identifier	Enum	How the Barracuda Load Balancer ADC identifies the server. Possible values: • **hostname** • **ip_address**
ip_address	String	The IP address of the server. **Conditional:** Required when the **identifier** is **ip_address**.
port	String	The port number of the server.
hostname	String	The hostname of the server. **Conditional:** Required when the **identifier** is **hostname**.
status	Enum	Indicates when requests are forwarded to the server. Possible values: • **Enable** – Requests can be forwarded to the server. • **Disable** – Requests cannot be forwarded to the server. All existing connections to this server are immediately terminated. • **Maintenance** – Requests cannot be forwarded to the server. Existing connections are terminated only after in-progress requests are completed. • **Sticky** – (Available for only Layer 7 services) Only requests from existing persistent connections for the service can be forwarded to the server. Existing connections are maintained until the **Persistence Time** on the **BASIC > Services** page is exceeded.
backup_server	Boolean	Configures the server as a backup server that is used only when all other servers fail or are out of service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
weight	Integer	The static load balancing weight of the server. The server with the highest weight receives the most requests. **Range:** 1 to 65535 **Note:** The server weight is ignored if adaptive scheduling is enabled.

SSL

The following parameters apply only to servers being added to secure services (i.e., HTTPS, Instant SSL, FTP SSL, and Secure TCP Proxy).

Parameter	Data Type	Description
enable_https	Boolean	Encrypts all traffic between the server and service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enabled_ssl_3	Boolean	Enables the service to use SSL 3.0 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enable_tls_1	Boolean	Enables the service to use SSL TLS 1.0 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enable_tls_1_1	Boolean	Enables the service to use SSL TLS 1.1 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enabled_tls_1_2	Boolean	Enables use of SSL TLS 1.2 by the service to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
validate_certificate	Boolean	Requires validation of the server certificate using certificates from well-known Certificate Authorities. Possible values:

		• **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
client_certificate	String	The certificate that the server provides when the service requires client authentication.

Server Monitor

Parameter	Data Type	Description

server_monitor.test	Enum	The tests that are used by the server monitor to determine the availability of the servers that are associated with the service. You can either use a monitor group or select a test type.
		
		To specify a **Error! Not a valid result for table.**, use the following syntax:
		`GROUP_<group name>`
		For example, to use a monitor group named `Group 1`:
		`"service_monitor":{"test":"GROUP_Group1 "}`
		
		To select a test type, use one of the following values:
		○ **ICMP_PING** – Performs a PING test.
		○ **TCP_PORT_CHECK** – Validates that the configured service port is open.
		Dependent variable:
		○ service_monitor.port
		• **UDP_PORT_CHECK** – Verifies that the UDP port is open by sending a 0 byte datagram to the IP address and port of the real server. This test depends on receiving an ICMP Port Unreachable message to determine the result. If a firewallprevents outbound ICMP messages, the port is assumed to be open.
		Dependent variable:
		○ service_monitor.port
		• **HTTP_TEST** – Sends an HTTP request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled.
		Dependent variables:
		○ service_monitor.port
		○ service_monitor.target
		○ service_monitor.match

Parameter	Data Type	Description
		○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTP** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTP_SLOW** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_TEST** – Sends an HTTPS request to the specified URL to verify that the respose contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_SLOW** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting

Parameter	Data Type	Description
		on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTPS** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **DNS_TEST** – Sends a DNS query and compares the returned IP address to the entered IP address. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **IMAP_TEST** – Simple test for an IMAP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **POP_TEST** – Simple test for a POP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SMTP_TEST** – Simple test for SMTP servers. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target

Parameter	Data Type	Description
		○ service_monitor.match • **SNMP_TEST** – Sends an SNMP GET to the specified OID to verify that the response contains an expected pattern. If no OID is specified, this test verifies only the availability of the service in the server. **Dependent variable:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SIP_TEST** – Simple test for a SIP service. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service. **Dependent variable:** ○ service_monitor.port • **SIP_TLS_TEST** – Simple test for a SIP service over TLS. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service over TLS. **Dependent variable:** ○ service_monitor.port • **LDAP_AD_TEST** – Bind test for an LDAP/AD service. If no username and password are provided, the LDAP/AD test verifies the availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **LDAP_AD_SSL_TEST** – Bind test for an LDAPS/AD service. If no username and password are provided, the LDAP/AD test verifies the availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SPAMFIREWALL_TEST** – Test for use with Barracuda Spam Firewalls. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match

Parameter	Data Type	Description
		• **FORCED_PORT_HTTP** – (Specific HTTP Port test) Sends an HTTP GET request using a specified port to a relative URL on the real server, to verify that the retrieved HTML contains an expected pattern. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_TEST** – Tests the availability of a RADIUS server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_ACCT** – Tests the availability of a RADIUS server by making an accounting request. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RDP_TEST** – Attempts an RDP connection to each real server to check the availability of the Terminal service. **Dependent variable:** ○ service_monitor.port • **FTP_TEST** – Attempts a TCP connection to each real server to check FTP availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FTPS_TEST**– Attempts a TCP connection to each real server to check FTPS availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SFTP_TEST** – Test for FTP over SSH. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password • **NTLM_TEST** – Simple test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port

Parameter	Data Type	Description
		○ service_monitor.user_name ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **NTLMS_TEST** – Secure test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **ALWAYS_PASS** – This test is used for troubleshooting or for services with management access to real servers. This test always passes regardless of the condition of the real server.
server_monitor.port	Integer	By default, tests use the configured real server port for the service unless the real server port is set to ALL. In that case, tests use the default port for the test type (e.g., SMTP = 25, HTTP = 80, DNS = 53, HTTPS = 443, IMAP = 143, POP = 110, and SNMP = 161). Enter a port number to override the default port.
server_monitor.user_name	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The username for the SSH account. • MS SharePoint – The username for the SharePoint service.
server_monitor.password	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The password for the SSH account. • MS SharePoint – The password for the SharePoint service.
server_monitor.target	String	**Conditional:** The value for this parameter differs for each test type: • Barracuda Spam Firewall Test – The domain for the mail server. • DNS – The fully qualified domain name. • FTP – The username to log into the server. The default username is `anonymous`. • FTPS – The username to log into the server. **Note:** If no username and password are provided,

Parameter	Data Type	Description
		this test verifies only the availability of the service on the server. • HTTP and HTTPS – The complete URL (starting with **`http`** or **`https`**). • HTTP Slow and HTTPS Slow – The complete URL (starting with **`http`** or **`https`**). • IMAP and POP3 – (Optional) The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The username with full LDAP schema. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The root relative URL. For example: **`/cgi-bin/index.cgi`** • RADIUS Auth and RADIUS Acct – The secret to use with the RADIUS server. • Simple HTTP and Simple HTTPS – The root relative URL. For example: **`/cgi-bin/index.cgi`** The actual URL used is **`https://<real_server_ip>:<port><URL>`** • SNMP – (Optional) Enter a valid SNMP OID. **Note:** If no OID is specified, this test verifies only the availability of the service in the server. • Specific HTTP Port – The TCP port followed by colon (:) and the root relative URL. For example: **`8080:/cgi-bin/index.cgi`**
server_monitor.match	String	**Conditional:** This parameter value differs for each test type: • Barracuda Spam Firewall – (Optional) A pattern that is expected in the banner of the SMTP server. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. • DNS – The IP address of the hostname. • FTP – The password to log into the server. The default password is **`anonymous`**. • FTPS – The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The expected pattern in the

Parameter	Data Type	Description
		retrieved HTML. • HTTP Slow and HTTPS Slow – The expected pattern in the retrieved HTML. • IMAP and POP3 – (Optional) The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The password to log into the server. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The pattern that is expected in the retrieved HTML. • RADIUS Auth – The username and password, separated by a backslash (****). For example: **`username\password`** • Simple HTTP and Simple HTTPS – The pattern expected in the retrieved HTML. • SMTP – (Optional) The pattern that is expected in the banner for the SMTP server. • SNMP – (Required if an OID is specified in the test target) A pattern to match in the response. • Specific HTTP Port – The pattern that is expected in the retrieved HTML.
server_monitor. hosts	String	Additional headers. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
server_monitor.code	Integer	The expected status code. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
server_monitor.test_delay	Integer	The interval between test start times, in seconds.
server_monitor.method	Enum	The method for HTTP and HTTPS tests. Possible values: • **GET** • **HEAD** • **POST** **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, and HTTPS Slow tests.

Parameter	Data Type	Description
server_monitor.post_body	String	Data that is sent in the POST request. **Conditional:** Required when the **server_monitor.method** is **POST**.

Advanced Options

Parameter	Data Type	Description
enable_client_impersonation	Boolean	Enables the Barracuda Load Balancer ADC to use the client IP address as the source IP address to communicate to the server. If this option is disabled, the Barracuda Load Balancer ADC uses its own IP address. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false **Note:** Available for only TCP Proxy, Secure TCP Proxy, HTTP, HTTPS, and Instant SSL services.
route	Enum	Indicates how outgoing traffic is delivered from the real server to the client. Direct server return is ideal for high-bandwidth requirements such as content delivery networks and lets you keep the existing IP addresses of your real servers. Possible values: • **standard** – Uses standard route for outgoing traffic. • **direct** – Uses direct server return.
enable_connection_pooling	Boolean	Allows a server connection to be used for multiple client requests. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false **Note:** Available for only HTTP, HTTPS, and Instant SSL services. **Dependent Variables:** • keepalive_timeout • max_connections • max_request • max_keepalive_requests • max_establishing_connections • max_spare_connections • timeout

Parameter	Data Type	Description
keepalive_timeout	Integer	The maximum length of time, in milliseconds, that a persistent connection to the real server can be idle before timing out. (Default: 900000) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_connections	Integer	The maximum number of simultaneous TCP connections between the Barracuda Load Balancer ADC and this server at any time. (Default: 10000) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_request	Integer	The maximum number of HTTP requests that can be queued on the Barracuda Load Balancer ADC for this server. (Default: 1000) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_keepalive_requests	Integer	The maximum number of requests allowed on a persistent connection. If set to 0, the number of requests allowed is unlimited. (Default: 0) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_establishing_connections	Integer	The maximum number of connections that can be initiated with the server at one time. When this limit is reached, additional connection requests are ignored. (Default: 100 connections) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
max_spare_connections	Integer	The maximum number of pre-allocated connections that can be established to this server. Spare connections stay in the pool to be used for future requests. (Default: 0 connections) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.
timeout	Integer	The maximum length of time, in milliseconds, that the connection can remain idle before being terminated. (Default: 300000) **Conditional:** Applicable only when **`enable_connection_pooling`** is **`true`**.

SNAT Rules

You can add a source NAT (SNAT) rule to map IP addresses from one IP address range to another.

Get SNAT Rules

You can get all SNAT rules that have been configured on the Barracuda Load Balancer ADC.
To get the SNAT rules, make a GET request to the following URI:

/interfaces/<*interface name*>/NAT/<NAT identifier>

where:

- <*interface name*> is the name of the interface on which the SNAT rules have been configured.
- <NAT identifier> is the identifier for a specific SNAT rule configured on the specified interface (Optional).

Example Response


```
{
    "data": [
        {
            "protocol": "tcp",
            "comment": "",
            "interface": "ge-1-1",
            "interface_address": "199.1.1.1",
            "netmask": "255.255.255.255",
            "id": "1.1.1.1",
            "address": "1.1.1.1",
            "port": "1-65535"
        }
    ],
    "token":
"eyJldCI6IjE0MzEwMjMxNzciLCJwYXNzd29yZCI6ImVjNGZhMzQ4MzI1ZWJiNDQyM2MwYTFhOGUy\n
NDAwNWEzIiwidXNlciI6ImFkbWluIn0=\n"
}
```


Example Response (for specific SNAT)


```
{
    "protocol": "tcp",
    "interface_address": "199.1.1.1",
    "port": "1-65535",
    "comment": "",
    "interface": "ge-1-1",
    "id": null,
    "address": "1.1.1.1",
    "netmask": "255.255.255.255",
    "token":
"eyJldCI6IjE0MzEwMjMyMzIiLCJwYXNzd29yZCI6ImU0OTBjYmQ4OTAwZWZjYmYwNDFkZGQ2Yjkx\n
YTM4MzIxIiwidXNlciI6ImFkbWluIn0=\n"
}
```

Create a new SNAT Rule

To create a new SNAT rule, make a POST request to the following URI:

`/interfaces/<interface name>/NAT/IP:PORT:PROTOCOL`

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the following parameters:

Parameter	Data Type	Mandatory	Description
protocol	String	Yes	Specify the protocol used by the connected virtual service. Possible values: • tcp • udp
comment	String	No	Text describing SNAT rule.
interface	String	Yes	Name of the interface on the Barracuda Load Balancer ADC (for example, ge-1-1).
interface_address	String	Yes	IP address specified for the interface.
netmask	String	Yes	Subnetwork mask specified for the interface address.
id	String	Yes	Identifier for the SNAT rule.
address	String	Yes	IP address specified for the SNAT rule.
Port	String	Yes	Port specified for the SNAT rule.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -H "Content-Type:application/json" \
   -X POST \
   -d '{"protocol": "tcp","interface_address": "199.1.1.1","port": "1-
65535","comment": "","interface": "ge-1-1","id": null, "address": "2.2.2.2",
"netmask": "255.255.255.255"}' \
   http://10.11.19.104:8000/restapi/v2/interfaces/ge-1-1/NAT/2.2.2.2:1-65535:TCP
```

Example Response


```
{
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "data": [
        {
            "protocol": "tcp",
            "comment": "",
            "interface": "ge-1-1",
            "interface_address": "199.1.1.1",
            "netmask": "255.255.255.255",
            "id": "1.1.1.1",
            "address": "1.1.1.1",
            "port": "1-65535"
        },
        {
            "protocol": "tcp",
            "comment": "",
            "interface": "ge-1-1",
            "interface_address": "199.1.1.1",
            "netmask": "255.255.255.255",
            "id": "null",
            "address": "2.2.2.2",
            "port": "1-65535"
        }
    ],
    "token":
"eyJldCI6IjE0MzEwMjMzNzkiLCJwYXNzd29yZCI6IjZjMGMzYWU5ZjI5MTRhOTUzZWU5OGExMjNm\n
YzYxOGI5IiwidXNlciI6ImFkbWluIn0=\n"
}
```


Edit an SNAT Rule

You can edit an SNAT rule using the PUT command. You must specify the SNAT ID in the path.

To edit an SNAT rule, make a PUT request to the following URI:

/interfaces/<*interface name*>/NAT/<NAT identifier>/IP:PORT:PROTOCOL

where:

- <*interface name*> is the name of the interface on which the NAT rules have been configured.
- <NAT identifier> is the identifier for a specific NAT rule you need to delete on the specified interface.

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the following parameters:

Parameter	Data Type	Mandatory	Description
protocol	String	Yes	Specify the protocol used by the connected virtual service. Possible values: • tcp • udp
comment	String	No	Text describing SNAT rule.
interface	String	Yes	Name of the interface on the Barracuda Load Balancer ADC (for example, ge-1-1).
interface_address	String	Yes	IP address specified for the interface.
netmask	String	Yes	Subnetwork mask specified for the interface address.
id	String	Yes	Identifier for the SNAT rule.
address	String	Yes	IP address specified for the SNAT rule.
Port	String	Yes	Port specified for the SNAT rule.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -H "Content-Type:application/json" \
   -X PUT \
   -d '{"protocol": "tcp","interface_address": "199.1.1.1","port": "1-
65535","comment": "","interface": "ge-1-1","id": null, "address": "1.0.0.0",
"netmask": "255.0.0.0"}' \
   http://10.11.19.104:8000/restapi/v2/interfaces/ge-1-1/NAT/1.0.0.0/2.2.2.2:1-
   65535:TCP
```

Example Response

```
{
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "data": [
        {
            "protocol": "tcp",
            "comment": "",
            "interface": "ge-1-1",
            "interface_address": "199.1.1.1",
            "netmask": "255.0.0.0",
            "id": "1.0.0.0",
            "address": "1.0.0.0",
            "port": "1-65535"
        },
    ],
    "token":
"eyJldCI6IjE0MzEwMjM5NzUiLCJwYXNzd29yZCI6IjdiMzMwOTJmOGFjZWNkMDJjYWZhOTcyOWYx\n
ZGNiYWU0IiwidXNlciI6ImFkbWluIn0=\n"
}
```

Deleting an SNAT Rule

You can delete an existing SNAT rule with the DELETE command. You must specify the SNAT ID.

To delete an SNAT rule, make a DELETE request to the following URI:
/interfaces/<*interface name*>/NAT/<NAT identifier>

where:

- <*interface name*> is the name of the interface on which the SNAT rules have been configured.
- <NAT identifier> is the identifier for a specific SNAT rule you need to delete on the specified interface.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -H "Content-Type:application/json" \
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/interfaces/ge-1-
   1/NAT/1.0.0.0/IP:PORT:PROTOCOL
```

Example Response

```
{
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "msg": "Successfully deleted",
    "token":
"eyJldCI6IjE0MzEwMjM2MDQiLCJwYXNzd29yZCI6ImNmMTgyMDNjYTlhNzA1NmVkMGNkMmI5NTBm\n
YWZjZjcwIiwidXNlciI6ImFkbWluIn0=\n"
}
```

Content Rules

For HTTP, HTTPS, and Instant SSL services, you can create content rules to apply caching, compression, load balancing, and persistence settings to incoming web traffic matching the URL, host, and extended match patterns specified in the rules.

The extended match pattern is one or more expressions that consist of a combination of HTTP headers and/or query string parameters. If there are multiple rules for a service, they are evaluated in the order established by their extended match sequence numbers. The rule with the most specific host and URL match is executed.

Create a Content Rule

To create a content rule for a service, make a POST request to the following URI:

`/virtual_service_groups/<group name>/virtual_services/<service name>/content_rules`

where:

- *<group name>* is the service group that the service was added to.
- *<service name>* is the service.

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the following parameters:

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for the content rule.
host_match	String	Yes	A host name to be matched against the host in the request header.
url_match	String	Yes	A URL to be matched to the URL in the request header.
extended_match	String	Yes	An expression that consists of a combination of HTTP headers and/or query string parameters.
extended_match_sequence	Integer	Yes	A number to specify the order of this rule when multiple content rules are configured for this service. Content rules are evaluated sequentially according their extended match sequence number.

For a complete list of parameters that you can pass, see
.

Request and Response Examples

The following examples display a request and response for creating a content rule named **rule1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"rule1", "host_match":"www.barracuda.com","url_match":\
      "/index.html", "extended_match":"*", "extended_match_sequence":1}'\
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/content_rules
```

Example Response

```
{
    "id": "rule1",
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Update a Content Rule

To update a content rule for a service, make a PUT request to the following URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*\
/content_rules/*<rule name>***

where:

- *<group name>* is the service group that the service was added to.
- *<service name>* is the service.
- *<rule name>* is the rule.

The body of your request must be JSON, with the Content-Type header set to application/json. Only the values of the parameters that are passed in the body of your request are changed or added. For a complete list of parameters that you can pass, see
.

Request and Response Examples

The following examples display a request and response for updating the URL match for a content rule named **rule1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X PUT \
   -H "Content-Type:application/json" \
   -d '{"url_match":"/index2.html"}' \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/content_rules/rule1
```

Example Response

```
{
  "info": {
     "msg": [
      "Configuration updated"
    ]
  },
  "id": "rule1",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Content Rules

To retrieve data for all the content rules of a service, make a GET request to the following URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*\
/content_rules**

where:

- *<group name>* is the service group that the service was added to.
- *<service name>* is the service.

To retrieve data for only a specific content rule, add the rule name to the URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*\
/content_rules/*<rule name>***

Request and Response Examples

The following examples display a request and response for retrieving a content rule named **rule1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X GET \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/content_rules/rule1
```

Example Response


```
{
    "cache": {
        "enabled": true,
        "expiry_age": "60",
        "file_extensions": [
            "gif",
            "tif",
            "jpg",
            "jpeg",
            "png",
            "bmp",
            "ico",
            "js",
            "jsp",
            "css",
            "jar",
            "swf",
            "pdf"
        ],
        "max_objsize": "256",
        "min_objsize": "256",
        "negative_responses": true,
        "req_cachehdrs_ignore": true,
        "resp_cachehdrs_ignore": true
    },
    "comments": null,
    "compress": {
        "content_types": [
            "text/html",
            "text/plain"
        ],
        "enabled": true,
        "min_obj_size": "8192"
    },
    "cookie_age": null,
    "extended_match": "*",
    "extended_match_sequence": "5",
    "failover_method": "LB",
    "group": "default",
    "header_name": "header1",
    "host_match": "barracuda.com",
    "id": "rule1",
    "lb_algorithm": "weighted_round_robin",
    "name": "rule1",
```

```
    "parameter_name": null,
    "persistence_cookie_domain": null,
    "persistence_cookie_httponly": "1",
    "persistence_cookie_name": "persistence",
    "persistence_cookie_path": null,
    "persistence_cookie_security": "0",
    "persistence_method": "HEADERFIELD",
    "persistence_time": "600",
    "servers": [],
    "service_name": "service1",
    "service_type": "HTTPS",
    "status": true,
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n",
    "url_match": "/index.html"
}
```


Delete a Content Rule

To delete a content rule for a service, make a DELETE request to the following URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*\
/content_rules/*<rule name>***

where:

- *<group name>* is the service group that the service was added to.
- *<service name>* is the service.
- *<rule name>* is the rule.

Request and Response Examples

The following examples display a request and response for deleting a content rule named **rule1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/content_rules/rule1
```

Example Response


```
{
  "info": {
    "msg": [
      "Configuration updated"
    ]
  },
  "msg": "Successfully deleted",
   "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Content Rule Parameters

You can configure the following parameters for content rules.

Basic Configuration

Parameter	Data Type	Description
name	String	A name for the content rule.
status	Boolean	The status of the content rule. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
host_match	String	A host name to be matched against the host in the request header.
url_match	String	A URL to be matched to the URL in the request header.
extended_match	String	An expression that consists of a combination of HTTP headers and/or query String parameters.
extended_match_sequence	Integer	A number to specify the order of this rule when multiple content rules are configured for this service. Content rules are evaluated sequentially according their extended match sequence number.
comments	String	Description about the content rule.

Caching

Parameter	Data Type	Description
cache.enabled	Boolean	Enables caching of the specified file types in local memory. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.file_extensions	Array	The extensions of the file types to be cached.
cache.expiry_age	Integer	The default expiration age in minutes, to be used when the response does not have an `Expires` header. A value of **0** (expiry-age = 0) sets the expiry age to 60 minutes.
cache.max_objsize	Integer	The maximum size of files that can be cached.
cache.min_objsize	Integer	The minimum size of files that can be cached.

Parameter	Data Type	Description
cache.negative_responses	Boolean	Caches HTTP negative responses with status codes like 204, 305, 404, 405, 414, 501, 502, and 504. Only negative responses with the `Expires` or `Last Modified` headers are cached. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.req_cachehdrs_ignore	Boolean	Ignores HTTP request cache-control headers to instruct upstream caching servers. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
cache.resp_cachehdrs_ignore	Boolean	Ignores HTTP response cache-control headers to instruct upstream caching servers. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Compression

Parameter	Data Type	Description
compress.enabled	Boolean	Enables compression of the specified content types. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
compress.content_types	Array	The content types to be compressed. For example: **`text/css`** **`text/html`** **`text/js`** **`text/plain`**
compress.min_obj_size	Integer	The minimum size of objects that can be compressed, from **1** to **2147483646** in bytes. The default value is **8192**.

Load Balancing

Parameter	Data Type	Description
lb_algorithm	Enum	How traffic is distributed among the real servers associated with this content rule. If no servers are configured, the requests are distributed to the servers of the service. Possible values: • **weighted_round_robin** • **weighted_least_connection**

Persistence

Parameter	Data Type	Description
persistence_method	Enum	How clients are redirected after a period of inactivity to the real server that they were initially routed to. Possible values: • **NONE** – Past connections / UDP datagrams are not considered when directing clients to servers. • **SCRIPNETMASK** – Directs requests based on the client IP address. Requests from clients that are grouped by the persistence netmask are routed to the same real server as the first request received from the group. **Dependent variable:** ○ source_ip_netmask • **COOKIEINSERT** – Directs requests based on a cookie (BNI_cookie_name) that is inserted by the Barracuda Load Balancer ADC after it routes the first request from the client. **Dependent variables:** ○ persistence_cookie_name ○ persistence_cookie_domain ○ persistence_cookie_path ○ cookie_age ○ persistence_cookie_httponly ○ persistence_cookie_security • **COOKIEPASSIVE** – Directs requests based on a cookie (BNI_cookie_name) that is inserted by the Barracuda Load Balancer ADC in the response only if the real server inserts a cookie. **Dependent variables:** ○ persistence_cookie_name ○ persistence_cookie_domain ○ persistence_cookie_path ○ cookie_age

Parameter	Data Type	Description
		○ persistence_cookie_httponly ○ persistence_cookie_security • **HEADERFIELD** – Directs all incoming HTTP requests based on the HTTP header. **Dependent variable:** ○ header_name • **URLPARAM** – Directs all incoming HTTP requests based on the specified parameter name. **Dependent variable:** ○ parameter_name
persistence_time	String	The maximum idle time (in seconds) for a persistent connection. A client is directed to the same real server unless the connection is inactive for more than the specified number of seconds.
failover_method	Enum	How to handle a request in a persistent session when the server that must respond to the request is unavailable. Possible values: • **LB** – The requests are load balanced among the remaining servers in the pool. • **ERROR** – A "503 service unavailable" error message is sent. This method is not supported when the **persistence_method** is **SRCIPNETMASK**.
source_ip_netmask	String	The netmask for persistence using source IP addresses (including Layer 4-UDP). A more specific netmask (such as 255.255.255.255) tracks each client independently and can cause a higher memory load on the Barracuda Load Balancer ADC. A less specific netmask (such as 255.255.0.0) organizes multiple clients under the same network identifier and connects them all to the same server. **Conditional:** Required only when the **persistence_method** is **SRCIPNETMASK**.
persistence_cookie_name	String	The name of the cookie used for persistence. **Conditional:** Required only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence.cookie_domain	String	The domain name of server of the cookie. **Conditional:** Applicable only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence_cookie_path	String	The path where the cookie is valid. **Conditional:**

Parameter	Data Type	Description
		Applicable only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence_cookie_httponly	Boolean	Makes the cookie inaccessible by client-side scripts, if supported by the browser. Helps mitigate most common cross-site scripting (XSS) attacks. Possible values: • **0** – false • **1** – true **Conditional:** Applicable only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
persistence_cookie_security	Boolean	Only transmits the cookie over HTTPS connections. Possible values: • **0** – false • **1** – true **Conditional:** Applicable only when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
cookie_age	Integer	The maximum age for the session cookie, in minutes. **Conditional:** Applicable when the when the **persistence_method** is **COOKIEINSERT** or **COOKIEPASSIVE**.
header_name	String	The name of the header for which the value needs to be checked in the HTTP requests. **Conditional:** Required only when the **persistence_method** is **HEADERFIELD**.
parameter_name	String	The name of the parameter for which the value needs to be checked in the URL. **Conditional:** Required only when the **persistence_method** is **URLPARAM**.

Rule Group Servers

For each content rule, you can add rule group servers to handle matching requests.

Create a Rule Group Server

To create a rule group server for a content rule, make a POST request to the following URI:

```
/virtual_service_groups/<group name>/virtual_services/<servicename>\
/content_rules/<rule name>/rg_servers
```

where:

- *<group name>* is the service group containing the service of the content rule.
- *<service name>* is the service that the content rule is configured under.
- *<rule name>* is the content rule.

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the parameters listed in the following table:

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name to identify the server.
identifier	Enum	Yes	Indicates whether the Barracuda Load Balancer ADC identifies the server by its IP address or hostname. Possible values: • **hostname** • **ipaddr**
ip_address	String	Conditional	The IP address of the server. **Conditional:** Required when the **identifier** is **ipaddr**.
hostname	String	Conditional	The hostname of the server. **Conditional:** Required when the **identifier** is **hostname**.
port	Integer	Yes	The port number of the server.

For a complete list of parameters that you can pass, see Rule Group Server Parameters.

Request and Response Examples

The following examples display a request and response for adding a server named **rule_server** for a content rule named **rule 1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"rule_server", "identifier":"ipaddr", \
      "ip_address":"10.66.44.104", "port":"80"}' \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/content_rules/rule1/rg_servers
```

Example Response

```
{
  "info": {
    "msg": [
      "Configuration updated"
    ]
  },
  "id": "rule_server",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Update a Rule Group Server

To update a rule group server for a content rule, make a PUT request to the following URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<service name>*\
/content_rules/*<rule name>*/rg_servers/*<server name>***

where:

- *<group name> is the name of the service group containing the service of the content rule.*
- *<service name> is the name of the content rule's service.*
- *<rule name> is the name of the content rule.*
- *<server name>* is the name of the server.

The body of your request must be JSON, with the Content-Type header set to application/json. Only the values of the parameters that are passed in the body of your request are changed or added. For a complete list of parameters that you can pass, see Rule Group Server Parameters.

Request and Response Examples

The following examples display a request and response for updating the weight and status for a rule group server named **rule_server**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X PUT \
   -H "Content-Type:application/json" \
   -d '{"weight":"5", "status":"sticky"}' \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/content_rules/rule1/rg_servers/rule_server
```

Example Response

```
{
  "info": {
    "msg": [
      "Configuration updated"
    ]
  },
  "id": "rule _server",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Rule Group Servers

To retrieve data for all the rule group servers for a content rule, make a GET request to the following URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*\
/content_rules/*<rule name>*/rg_servers**

where:

- *<group name>* is the name of the service group containing the service of the contentrule.
- *<service name>* is the name of the content rule'sservice.
- *<rule name>* is the name of the contentrule.

To retrieve data for only a specific server, add the server name to the URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<servicename>*\
/content_rules/*<rule name>*/rg_servers/*<server name>***

Request and Response Examples

The following examples display a request and response for retrieving a rule group server named **rule_server**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X GET \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/content_rules/rule1/rg_servers/rule_server
```

Example Response


```
{
    "address_version": "ipv4",
    "backup_server": true,
    "client_certificate": "test_gen_cert",
    "comments": null,
    "connection_pooling": {
        "enable_connection_pooling": true,
        "keepalive_timeout": "900000"
    },
    "enable_client_impersonation": false,
    "enable_https": true,
    "enable_ssl_3": true,
    "enable_tls_1": true,
    "enable_tls_1_1": true,
    "enable_tls_1_2": true,
    "hostname": "barracuda",
    "id": "rule_server1",
    "identifier": "hostname",
    "ip_address": "216.129.105.116",
    "max_connections": "10000",
    "max_establishing_connections": "100",
    "max_keepalive_requests": "0",
    "max_request": "1000",
    "max_spare_connections": "13",
    "name": "rule_server",
    "operational_status": "disable",
    "port": "80",
    "scope_data": "testssl:rule1:216.129.105.116_80",
    "server_monitor": {
        "code": "420",
        "hosts": "header1:testheader",
        "match": "testmatchpattern",
        "method": "GET",
        "password": "password",
        "port": "",
        "post_body": "",
        "retries": "1",
        "target": "/index",
        "test": "NTLM_TEST",
        "test_delay": "10",
        "user_name": "username"
    },
```

```
    "status": "disable",
    "timeout": "300000",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n",
    "traffic": 0,
    "validate_certificate": "0",
    "weight": "0"
}
```

Delete a Rule Group Server

To delete the rule group server for a content rule, make a DELETE request to the following URI:

**/virtual_service_groups/*<group name>*/virtual_services/*<service name>*\
/content_rules/*<rule name>*/rg_servers/*<server name>***

where:

- *<group name>* is the service group containing the service of the content rule.
- *<service name>* is the service that the content rule is configured under.
- *<rule name>* is the content rule.
- *<server name>* is the server.

Request and Response Examples

The following examples display a request and response for deleting a rule group server named **rule_server**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/virtual_service_groups/default\
   /virtual_services/service1/content_rules/rule1/rg_servers/rule_server
```

Example Response

```
{
  "info": {
    "msg": [
      "Configuration updated"
    ]
  },
  "msg": "Successfully deleted",
   "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Rule Group Server Parameters

You can configure the following parameters for rule group servers.

Server Configuration

Parameter	Data Type	Description
name	String	A name to identify the server.
address_version	Enum	The Internet protocol version to be used. Possible values: • **ipv4** • **ipv6**
identifier	Enum	Indicates whether the Barracuda Load Balancer ADC identifies the server by its IP address or hostname. Possible values: • **hostname** • **ipaddr**
ip_address	String	The IP address of the server. **Conditional:** Required when the **`identifier`** is **`ipaddr`**.
hostname	String	The hostname of the server. **Conditional:** Required when the **`identifier`** is **`hostname`**.
port	String	The port number of the server.
status	Enum	Indicates if requests are forwarded to the server. Possible values: • **Enable** – Requests can be forwarded to the server. • **Disable** – Requests cannot be forwarded to the server. All existing connections to this server are immediately terminated. • **Maintenance** – Requests cannot be forwarded to the server. Existing connections are terminated only after in-progress requests are completed. • **Sticky** – Only requests from existing persistent connections for the service can be forwarded to the server. Existing connections are maintained until the **Persistence Time** on the **BASIC > Services** page is exceeded.
backup_server	Boolean	Configures the server as a backup server that is only used when all other servers fail or are out of service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

weight	Integer	The static load balancing weight of the server. The server with the highest weight receives the most requests. **Range:** 1 to 65535

SSL

Parameter	Data Type	Description
enable_https	Boolean	Encrypts all traffic between the server and service. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enabled_ssl_3	Boolean	Allows the service to use SSL 3.0 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enable_tls_1	Boolean	Allows the service to use SSL TLS 1.0 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enable_tls_1_1	Boolean	Allows the service to use SSL TLS 1.1 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
enabled_tls_1_2	Boolean	Allows the service to use SSL TLS 1.2 to connect with the server. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false

Parameter	Data Type	Description
validate_certificate	Boolean	Requires validation of the server certificate using certificates from well-known Certificate Authorities. Possible values: • **1** – false • **0** – true
client_certificate	String	The name of the certificate that the server provides when the service requires client authentication.

Server Monitor

Parameter	Data Type	Description

server_monitor.test	Enum	The tests that the server monitor uses to determine the availability of the servers that are associated with the service. You can use either a monitor group or a test type.
		
		To specify a **Error! Not a valid result for table.**, use the following syntax:
		`GROUP_<group name>`
		For example, to use a monitor group named `Group 1`:
		`"service_monitor":{"test":"GROUP_Group1 "}`
		
		To select a test type, use one of the following values:
		○ **ICMP_PING** – Performs a PING test.
		○ **TCP_PORT_CHECK** – Verifies that the configured service port is open.
		Dependent variable:
		○ service_monitor.port
		• **UDP_PORT_CHECK** – Verifies that the UDP port is open by sending a 0 byte datagram to the IP address and port of the real server. Expects to receive an ICMP Port Unreachable message. If a firewall prevents outbound ICMP messages, the test assumes that the port is open.
		Dependent variable:
		○ service_monitor.port
		• **HTTP_TEST** – Sends an HTTP request to the specified URL and verifies that the response containes an expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled.
		Dependent variables:
		○ service_monitor.port
		○ service_monitor.target
		○ service_monitor.match

Parameter	Data Type	Description
		○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTP** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTP_SLOW** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_TEST** – Sends an HTTPS request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **HTTPS_SLOW** – Sends an HTTPS request to the specified relative URL to verify that the response contains that the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting

Parameter	Data Type	Description
		on the real server must be enabled. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **SIMPLE_HTTPS** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match ○ service_monitor.method ○ service_monitor.hosts ○ service_monitor.code • **DNS_TEST** – Sends a DNS query and compares the returned IP address to the entered IP address. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **IMAP_TEST** – Simple test for an IMAP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **POP_TEST** – Simple test for a POP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SMTP_TEST** – Simple test for SMTP servers. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target

Parameter	Data Type	Description
		○ service_monitor.match • **SNMP_TEST** – Performs an SNMP GET to the specified OID and verifies that the response contains an expected pattern. If no OID is specified, this test verifies only the availability of the service in the server. **Dependent variable:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SIP_TEST** – Simple test for a SIP service. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service. **Dependent variable:** ○ service_monitor.port • **SIP_TLS_TEST** – Simple test for a SIP service over TLS. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service over TLS. **Dependent variable:** ○ service_monitor.port • **LDAP_AD_TEST** – Bind test for an LDAP/AD service. If no username and password are provided, the LDAP/AD test verifies the availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **LDAP_AD_SSL_TEST** – Bind test for an LDAPS/AD service. If no username and password are provided, the LDAP/AD test verifies the availability of the anonymous user. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SPAMFIREWALL_TEST** – Test for use with Barracuda Spam Firewalls. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match

Parameter	Data Type	Description
		• **FORCED_PORT_HTTP** – (Specific HTTP Port test) Sends an HTTP GET request using a specified port to a relative URL on the real server, and verifies that the retrieved HTML contains an expected pattern. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_TEST** – Tests the availability of a RADIUS server. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RADIUS_ACCT** – Tests the availability of a RADIUS server by making an accounting request. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **RDP_TEST** – Attempts an RDP connection to each real server to check the availability of the Terminal service. **Dependent variable:** ○ service_monitor.port • **FTP_TEST** – Attempts a TCP connection to each real server to check FTP availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **FTPS_TEST**– Attempts a TCP connection to each real server to check FTPS availability. **Dependent variables:** ○ service_monitor.port ○ service_monitor.target ○ service_monitor.match • **SFTP_TEST** – Test for FTP over SSH. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password • **NTLM_TEST** – Simple test for Microsoft SharePoint. **Dependent variables:**

Parameter	Data Type	Description
		○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **NTLMS_TEST** – Secure test for Microsoft SharePoint. **Dependent variables:** ○ service_monitor.port ○ service_monitor.user_name ○ service_monitor.password ○ service_monitor.target ○ service_monitor.match ○ service_monitor.hosts ○ service_monitor.code • **ALWAYS_PASS** – This test is used for troubleshooting or for services with management access to real servers. This test always passes regardless of the condition of the real server.
server_monitor.port	Integer	By default, tests use the configured real server port for the service unless the real server port is set to ALL. In that case, the tests use the default port for the test type (e.g., SMTP = 25, HTTP = 80, DNS = 53, HTTPS = 443, IMAP = 143, POP = 110, and SNMP = 161). Enter a port number to override the default port.
server_monitor.user_name	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The username for the SSH account. • MS SharePoint – The username for the SharePoint service.
server_monitor.password	String	**Conditional:** The value for this parameter differs for each test type: • SFTP – The password for the SSH account. • MS SharePoint – The password for the SharePoint service.
server_monitor.target	String	**Conditional:** The value for this parameter differs for each test type: • Barracuda Spam Firewall Test – The domain for the mail server. • DNS – The fully qualified domain name. • FTP – The username to log into server. The default username is **anonymous**.

Parameter	Data Type	Description
		• FTPS – The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The complete URL (starting with **`http`** or **`https`**). • HTTP Slow and HTTPS Slow – The complete URL (starting with **`http`** or **`https`**). • IMAP and POP3 – (Optional) The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The username with full LDAP schema. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The root relative URL. For example: **`/cgi-bin/index.cgi`** • RADIUS Auth and RADIUS Acct – The secret to use with the RADIUS server. • Simple HTTP and Simple HTTPS – The root relative URL. For example: **`/cgi-bin/index.cgi`** The actual URL used is **`https://<real_server_ip>:<port><URL>`** • SNMP – (Optional) Enter a valid SNMP OID. **Note:** If an OID is not specified, this test verifies only the availability of the service in the server. • Specific HTTP Port – The TCP port followed by colon (:) and the root relative URL. For example: **`8080:/cgi-bin/index.cgi`**
server_monitor.match	String	**Conditional:** The value for this parameters differs for each test type: • Barracuda Spam Firewall – (Optional) A pattern that is expected in the banner of the SMTP server. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. • DNS – The IP address of the hostname. • FTP – The password to log into the server. The default password is **`anonymous`**. • FTPS – The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in

Parameter	Data Type	Description
		the server. • HTTP and HTTPS – The expected pattern in the retrieved HTML. • HTTP Slow and HTTPS Slow – The expected pattern in the retrieved HTML. • IMAP and POP3 – (Optional) The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The password to log into the server. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The pattern that is expected in the retrieved HTML. • RADIUS Auth – The username and password, separated by a backslash (****). For example: **`username\password`** • Simple HTTP and Simple HTTPS – The pattern expected in the retrieved HTML. • SMTP – (Optional) The pattern that is expected in the banner for the SMTP server. • SNMP – (Required if an OID is specified in the test target) A pattern to match in the response. • Specific HTTP Port – The pattern that is expected in the retrieved HTML.
server_monitor.hosts	String	Additional headers. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
server_monitor.code	Integer	The expected status code. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
server_monitor.test_delay	Integer	The interval between test start times, in seconds.

Parameter	Data Type	Description
server_monitor.method	Enum	The method for HTTP and HTTPS tests. Possible values: • **GET** • **HEAD** • **POST** **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, and HTTPS Slow tests.
server_monitor.post_body	String	The data that is being sent in the POST request. **Conditional:** Required when the **`server_monitor.method`** is **`POST`**.

Advanced Options

Parameter	Data Type	Description
enable_client_impersonation	Boolean	Enables the Barracuda Load Balancer ADC to use the client IP address as the source IP address to communicate to the server. If this option is disabled, the Barracuda Load Balancer ADC uses its own IP address. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false
connection_pooling.enable_connection_pooling	Boolean	Allows the server connection to be used for multiple client requests. Possible values: • **true** • **false** For Perl, possible values are: • **1** – true • **0** – false **Dependent Variables:** • connection_pooling.keepalive_timeout • max_connections • max_request • max_keepalive_requests • max_establishing_connections • max_spare_connections • timeout

Parameter	Data Type	Description
max_connections	Integer	The maximum number of simultaneous TCP connections that the Barracuda Load Balancer ADC can have with this server at any time. (Default: 10000) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
max_request	Integer	The maximum number of HTTP requests that can be queued on the Barracuda Load Balancer ADC for this server. (Default: 1000) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
max_keepalive_requests	Integer	The maximum number of requests allowed on a persistent connection. If set to 0, the number of requests allowed is unlimited. (Default: 0) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
max_establishing_connections	Integer	The maximum number of connections that can be initiated with the server at one time. After this limit is reached, additional connection requests are ignored. (Default: 100 connections) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
max_spare_connections	Integer	The maximum number of pre-allocated connections that can be established to this server. Spare connections stay in the pool to be used for future requests. (Default: 0 connections) **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.
timeout	Integer	The maximum length of time, in milliseconds, that the connection can remain idle before being terminated. (Default: 300000)
connection_pooling.keepalive_timeout	Integer	The time in milliseconds to time out a connection that is used at least once. This is the maximum amount of time a connection is kept alive. This value is applicable per 1024 connections, where a timeout error had occurred. **Conditional:** Applicable only when **connection_pooling.enable_connection_pooling** is **true**.

Security Policies

For HTTP, HTTPS, and Instant SSL services, you can configure a security policy to protect your application from vulnerabilities and malicious attacks. Each security policy is comprised of the following subpolicies to encrypt, cloak, and restrict the data that is included in HTTP requests and responses:

Request Limits
- Cookie Security
- URL Protection
- Parameter Protection

Cloaking
- URL Normalization

All security policies are global and can be applied to multiple services configured on the Barracuda Load Balancer ADC. You can either create custom security policies or use the predefined security policies. A default security policy is available, as well as predefined security policies for SharePoint, OWA, and Oracle.

Create a Custom Security Policy

To create a security policy, make a POST request to the following URI:

`/security_policies`

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the name of the security policy:

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name to identify the security policy.

For a complete list of parameters that you can pass, see Security Policy Parameters.

Request and Response Examples

The following examples display a request and response for creating a security policy named **`security_policy1`**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"security_policy1"}' \
   http://10.11.19.104:8000/restapi/v2/security_policies
```

Update a Security Policy

To update a security policy, make a PUT request to the following URI:

`/security_policies/<policy name>`

You can update custom security policies or the following predefined security policies:

- default
- sharepoint
- sharepoint2013
- owa
- owa2010
- owa2013
- oracle

The body of your request must be JSON, with the Content-Type header set to application/json. Only the parameters that are passed in the body of your request are updated. For a complete list of parameters that you can pass, see Security Policy Parameters.

Request and Response Examples

The following examples display a request and response for updating the cookie security settings for a security policy named **`security_policy1`**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X PUT \
   -H "Content-Type:application/json" \
   -d '{"cookie_security": {"cookie_replay_protection_type":"none", \
      "allow_unrecognized_cookies":"never", "tamper_proof_mode":"encrypted"}}' \
      http://10.11.19.104:8000/restapi/v2/security_policies/security_policy1
```

Example Response

```
{
  "info": {
    "msg": [
      "Configuration updated"
    ]
  },
  "id": "security_policy1",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Security Policies

To retrieve data for all configured security policies, make a GET request to the following URI:

```
/security_policies
```

To retrieve data for only a specific security policy, include the name of the policy in the URI:

```
/security_policies/<policy name>
```

You can retrieve custom security policies or the following predefined security policies:

- default
- sharepoint
- sharepoint2013
- owa
- owa2010
- owa2013
- oracle

Request and Response Examples

The following examples display a request and response for retrieving the default security policy.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X GET \
   http://10.11.19.104:8000/restapi/v2/security_policies/default
```

Example Response

```
{
    "cloaking": {
        "filter_response_header": "yes",
        "headers_to_filter": [
            "Server",
            "X-Powered-By",
```

```
            "X-AspNet-Version"
        ],
        "return_codes_to_exempt": [],
        "suppress_return_code": "yes"
    },
    "cookie_security": {
        "allow_unrecognized_cookies": "always",
        "cookie_max_age": "1440",
        "cookie_replay_protection_type": "IP",
        "cookies_exempted": [
            "__utma",
            "__utmc",
            "__utmz",
            "__utmb",
            "AuthSuccessURL",
            "CTSESSION",
            "SMSESSION",
            "SMCHALLENGE"
        ],
        "custom_headers": [],
        "days_allowed": "Always",
        "http_only": "no",
        "secure_cookie": "no",
        "tamper_proof_mode": "signed"
    },
    "id": "default",
    "name": "default",
    "parameter_protection": {
        "blocked_attack_types": [
            "cross-site-scripting",
            "sql-injection-medium",
            "os-command-injection"
        ],
        "custom_blocked_attack_types": [],
        "denied_metacharacters": "%00%04%1b%08%7f",
        "enable": "yes",
        "exception_patterns": [],
        "file_upload_extensions": [
            "JPG",
            "GIF",
            "PDF"
        ],
        "ignore_parameters": [
            "  VIEWSTATE"
        ],
        "maximum_instances": null,
        "maximum_parameter_value_length": "1000",
        "maximum_upload_file_size": "1024"
    },
    "request_limits": {
        "enable": "yes",
        "max_cookie_name_length": "64",
        "max_cookie_value_length": "4096",
```

```
        "max_header_name_length": "32",
        "max_header_value_length": "512",
        "max_number_of_cookies": "40",
        "max_number_of_headers": "20",
        "max_query_length": "4096",
        "max_request_length": "32768",
        "max_request_line_length": "4096",
        "max_url_length": "4096"
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n",
    "url_normalization": {
        "default_charset": "UTF-8",
        "detect_response_charset": "no",
        "double_decoding": "no",
        "parameter_separators": "ampersand"
    },
    "url_protection": {
        "allowed_content_types": [
            "application/x-www-form-urlencoded",
            "multipart/form-data",
            "text/xml"
        ],
        "allowed_methods": [
            "GET",
            "POST",
            "HEAD"
        ],
        "blocked_attack_types": [
            "cross-site-scripting",
            "sql-injection-medium",
            "os-command-injection"
        ],
        "csrf_prevention": "none",
        "custom_blocked_attack_types": [],
        "enable": "yes",
        "exception_patterns": [],
        "max_content_length": "32768",
        "max_parameters": "40",
        "maximum_parameter_name_length": "64",
        "maximum_upload_files": "5"
    }
```

Delete a Security Policy

To delete a security policy, make a DELETE request to the following URI:

`/security_policies/<policy name>`

Request and Response Examples

The following examples display a request and response for deleting a security policy named **`security_policy1`**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/security_policies/security_policy1
```

Example Response

```
{
  "info": {
    "msg": [
      "Configuration updated"
    ]
  },
  "msg": "Successfully deleted",
   "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Security Policy Parameters

You can configure the following parameters for each security policy.

Name

Parameter	Data Type	Description
name	String	The name of the security policy.

Request Limits

Parameter	Data Type	Description
request_limits.enable	Enum	Enforces size limit checks on request headers. Possible values: • **yes** • **no**

request_limits.max_request_length	Integer	The maximum allowable request length. This includes the Request Line and all HTTP request headers (for example, User Agent, Cookies, Referer, etc.).
request_limits.max_request_line_length	Integer	The maximum allowable length for the request line. The request line consists of the method, the URL (including any query strings) and the HTTP version.
request_limits.max_url_length	Integer	The maximum allowable URL length including the query string portion of the URL.
request_limits.max_query_length	Integer	The maximum allowable length for the query string portion of the URL.
request_limits.max_number_of_cookies	Integer	The maximum number of cookies to be allowed.
request_limits.max_cookie_name_length	Integer	The maximum allowable length for a cookie name.
request_limits.max_cookie_value_length	Integer	The maximum allowable length for a cookie value.
request_limits.max_number_of_headers	Integer	The maximum number of headers to be allowed in a request.
request_limits.max_header_name_length	Integer	The maximum allowable length for a header name.
request_limits.max_header_value_length	Integer	The maximum allowable length for header value in a request.

Cookie Security

Parameter	Data Type	Description
cookie_security.tamper_proof_mode	Enum	Indicates whether tamper proofing method for cookies. Possible values: • **signed** • **encrypted** • **none**
cookie_security.cookie_max_age	Integer	The maximum age for session cookies, in minutes.

Parameter	Data Type	Description
cookie_security.cookie_replay_protection_type	Enum	Indicates whether header values and/or client IP addresses are encoded in the cookies that are sent to the client. These are validated in the incoming cookies. If the IP address or header value does not match the IP address or header value of the sent cookie, the incoming cookie is considered a possible replay attack. Possible values: • **none** • **IP** • **IP_and_custom_headers** • **custom_headers**
cookie_security.custom_headers	Array	The custom headers to be used in the cookie. **Conditional:** Required only if the **`cookie_security.cookie_replay_protection_type`** is **`IP_and_custom_headers`** or **`custom_headers`**.
cookie_security.secure_cookie	Enum	Allows cookies only if the client makes secure HTTPS connection. Possible values: • **yes** • **no**
cookie_security.http_only	Enum	Makes the cookie inaccessible by client-side scripts, if supported by the browser. Helps mitigate most common cross-site scripting (XSS) attacks. Possible values: • **yes** • **no**
cookie_security.allow_unrecognized_cookies	Enum	Whether unrecognized cookies are allowed. Possible values: • **custom** • **always** • **never**
cookie_security.days_allowed	Integer	The maximum number of days that unrecognized cookies are allowed. **Conditional:** Required only when **`cookie_security.allow_unrecognized_cookies`** is **`custom`**.
cookie_security.cookies_exempted	Array	The names of the cookies that are exempted from the cookie security policy.

URL Protection

Parameter	Data Type	Description
url_protection.enable	Enum	Enforces URL protection. Possible values: • **yes** • **no**
url_protection.allowed_methods	Array	The list of allowable methods in a request.
url_protection.allowed_content_types	Array	The list of content types to be allowed in the POST body of a request.
url_protection.max_content_length	Integer	The maximum content length to be allowed for POST request body.
url_protection.max_parameters	Integer	The maximum number of parameters to be allowed in a request.
url_protection.maximum_upload_files	Integer	The maximum number of files that can be of file-upload type in a request.
url_protection.csrf_prevention	Enum	The Cross-Site Request Forgery (CSRF) prevention for forms and URLs. Possible values: • **forms_and_urls** • **none** • **forms**
url_protection.maximum_parameter_name_length	Integer	The maximum length of a parameter name in a request.
url_protection.blocked_attack_types	Array	The attack types to be matched in a request. Possible values: • **cross_site_scripting** • **remote_file_inclusion** • **sql_injection_strict** • **sql_injection** • **os_command_injection** • **remote_file_inclusion_strict** • **os_command_injection_strict** • **cross_site_scripting_strict**
url_protection.custom_blocked_attack_types	Array	The custom attack types defined on the **ADVANCED > Libraries** page (if any).
url_protection.exception_patterns	Array	Patterns that are allowed despite matching a malicious pattern group. **Note:** Configure the exact pattern name that is displayed on the **ADVANCED > View Internal Patterns** page, or as defined when creating a new group on the **ADVANCED > Libraries** page.

Parameter Protection

Parameter	Data Type	Description
parameter_protection.enable	Enum	Enforces parameter protection. Possible values: • **yes** • **no**
parameter_protection.denied_ metacharacters	String	The meta-characters that are not allowed in a parameter value. Meta-characters must be URL encoded. Non-printable characters such as "backspace" and web interface reserved characters like "?" must be URL encoded.
parameter_protection.maximum_ parameter_value_length	Integer	The maximum allowed length of any parameter value, including no-name parameters.
parameter_protection.maximum_instances	Integer	The maximum number of times a parameter is allowed in a request.
parameter_protection.file_upload_ extensions	Array	The extensions of the files types that can be uploaded.
parameter_protection.maximum_upload_ file_size	Integer	The maximum size (in KB) for an individual file that can be uploaded in a request.
parameter_protection.blocked_attack_types	Array	The attack types to be matched in a request. Possible values: • **directory_traversal** • **directory_traversal_strict** • **cross_site_scripting** • **remote_file_inclusion** • **sql_injection_strict** • **sql_injection_medium** • **os_command_injection** • **remote_file_inclusion_strict** • **os_command_injection_strict** • **cross_site_scripting_strict**
parameter_protection.custom_blocked_ attack_types	Array	The custom attack types defined on the **ADVANCED > Libraries** page (if any).
parameter_protection.exception_patterns	Array	The patterns that are allowed despite matching a malicious pattern group. **Note:** Configure the exact pattern name displayed on the **ADVANCED > View Internal Patterns** page, or as defined when creating a new group on the **ADVANCED > Libraries** page.
parameter_protection.ignore_parameters	String	The parameters exempt from *all* validations.

Cloaking

Parameter	Data Type	Description
cloaking.suppress_return_code	Enum	Suppresses an HTTP Status code in the response header and inserts a default or custom response page in case of any error responses from the server. Possible values: • **yes** • **no**
cloaking.return_codes_to_exempt	String	The HTTP response codes exempt from cloaking.
cloaking.filter_response_header	Enum	Removes the HTTP headers in responses. Possible values: • **yes** • **no**
cloaking.headers_to_filter	String	The list of headers that are to be removed from a response before serving it to a client.

URL Normalization

Parameter	Data Type	Description
url_normalization.detect_response_charset	Enum	Detects the character set decoding from the response. Possible values: • **yes** • **no**
url_normalization.parameter_separators	Enum	The URL-decoded parameter separator to be used. Possible values: • **ampersand** • **ampersand_and_semicolon** • **semicolon**
url_normalization.double_decoding	Enum	Enables double-decoding of the character set. Possible values: • **yes** • **no**

url_normalization.default_charset	Enum	The character set decoding type to be used for incoming requests. Possible values: • **GBK** • **ASCII** • **UTF-8** • **Shift-JIS** • **JOHAB** • **EUC-KR** • **ISO-8859-1** • **ISO-2022-KR** • **ISO-2022-CN** • **ISO-2022-JP** • **HZ** • **BIG5** • **GB2312** • **EUC-TW** • **EUC-JP**

Global ACLs

Global ACLs (URL ACLs) define strict allow and deny rules that are matched to requests by URL and extended match expressions. As part of a security policy, global ACLs can be applied to multiple services configured on the Barracuda Load Balancer ADC.

Create a Global ACL Rule

To create a global ACL rule for a security policy, make a POST request to the following URI:

`/security_policies/<policy name>/global_acls`

where *<policy name>* is the name of the policy. The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass the parameters listed in the following table:

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for the URL ACL rule.
enabled	Enum	Optional	Enables this rule. Possible values: • **yes** • **no**

Parameter	Data Type	Mandatory	Description
url_match	String	Yes	The URL to be matched to the URL in the request. The URL must start with a forward slash (**/**) and can have a maximum of one asterisk (*****) anywhere in the URL. To apply the ACL for all URLs in a domain, use: **/***
extended_match	String	Yes	An expression that consists of a combination of HTTP headers and/or query string parameters. For more information on how to write extended match expressions, see https://techlib.barracuda.com/ADC/ExtendedMatchSyntax.
extended_match_sequence	Integer	Yes	A number indicating the sequential priority of this match, compared to other URL/Host matches.
action	Enum	Yes	The action to execute for requests matching the URL for this rule. Possible values: • **process** • **allow** • **deny_and_log** • **redirect** • **deny_with_no_log**
redirect_url	String	Conditional	A URL to which a user is redirected. **Note:** Required only when the **`action`** is **`redirect`**.
comments	String	Optional	Description about the global ACL rule.

Request and Response Examples

The following examples display a request and response for adding a global ACL rule named **`acl_1`** to the default security policy.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"acl_1", "extended_match_sequence":"2", "extended_match":"*",\
      "url_match":"/*/new_acl", "action":"process", "enabled":"yes"}' \
   http://10.11.19.104:8000/restapi/v2/security_policies/default\
   /global_acls
```

Example Response

```
{
  "info": {
    "msg": [
      "Configuration updated"
    ]
  },
  "id": "acl_1",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Update a Global ACL Rule

To change the parameter values of a global ACL rule, make a PUT request to the following URI:

`/security_policies/<policy name>/global_acls/<acl rule name>`

where:

- *<policy name>* is the security policy that the ACL rule is part of.
- *<acl rule name>* is the ACL rule.

The body of your request must be JSON, with the Content-Type header set to application/json. Only the parameters values that are passed in the body of your request are updated. In the body of your request, you can pass the parameters listed in the following table:

Parameter	Data Type	Description
name	String	A name for the URL ACL rule.
enabled	Enum	Enables this rule. Possible values: • **yes** • **no**
url_match	String	The URL to be matched to the URL in the request. The URL must start with a forward slash (**/**) and can have a maximum of one asterisk (*****) in it. To apply the ACL for all URLs in a domain, use: **/***
extended_match	String	An expression that consists of a combination of HTTP headers and/or query string parameters. For more information on how to write extended match expressions, see https://techlib.barracuda.com/ADC/ExtendedMatchSyntax.
extended_match_sequence	Integer	A number indicating the sequential priority of this match, compared to other URL/Host matches.

Parameter	Data Type	Description
action	Enum	The action to execute for requests matching this rule. Possible values: • **process** • **allow** • **deny_and_log** • **redirect** • **deny_with_no_log**
redirect_url	String	A URL to which a user should be redirected. **Conditional:** Required only when the **action** is **redirect**.
comments	String	Description about the global ACL rule.

Request and Response Examples

The following examples display a request and response for updating a global ACL rule named **acl_1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -H "Content-Type:application/json" \
   -X PUT \
   -d '{"extended_match_sequence":"1", "comments":"first rule in sequence"}' \
   http://10.11.19.104:8000/restapi/v2/security_policies/new_policy\
   /global_acls/acl_1
```

Example Response

```
{
    "id": "acl_1",
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Global ACL Rules

To retrieve data for all the global ACL rules that are part of a security policy, make a GET request to the following URI:

/security_policies/*<policy name>*/global_acls

To retrieve data for only a specific global ACL rule, add the rule name to the URI:

/security_policies/*<policy name>*/global_acls/*<acl rule name>*

Request and Response Examples

The following examples display a request and response for retrieving all global ACL rules for a security policy named **corp_policy**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -H "Content-Type:application/json" \
   -X GET \
   http://10.11.19.104:8000/restapi/v2/security_policies/corp_policy\
   /global_acls
```

Example Response


```
{
    "data": [
        {
            "action": "process",
            "comments": "",
            "enabled": "yes",
            "extended_match": "*",
            "extended_match_sequence": "1",
            "id": "robots.txt",
            "name": "robots.txt",
            "redirect_url": "",
            "url_match": "/*/robots.txt"
        },
        {
            "action": "allow",
            "comments": "",
            "enabled": "yes",
            "extended_match": "*",
            "extended_match_sequence": "1",
            "id": "favicon.ico",
            "name": "favicon.ico",
            "redirect_url": "",
            "url_match": "/*/favicon.ico"
        },
        {
            "action": "deny_and_log",
            "comments": "",
            "enabled": "yes",
            "extended_match": "(Header Range rco
\"bytes=([[:blank:]]*[[:digit:]                    ]*-
[[:blank:]]*[[:digit:]]*,){5}\")",
            "extended_match_sequence": "3",
            "id": "apache_range_header_vulnerability",
            "name": "apache_range_header_vulnerability",
            "redirect_url": "",
            "url_match": "/*"
        }
    ],
```

```
    "fields": null,
    "limit": null,
    "object": "GlobalAcls",
    "offset": null,
    "policy_id": "corp_policy",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Delete a Global ACL Rule

To delete a global ACL rule, make a DELETE request to the following URI:

/security_policies/*<policy name>*/global_acls/*<acl rule name>*

where:

- *<policy name>* is the security policy that the ACL rule is part of.
- *<acl rule name>* is the ACL rule.

Request and Response Examples

The following examples display a request and response for deleting a global ACL rule named **acl_1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/security_policies/new_policy\
   /global_acls/acl_1
```

Example Response

```
{
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "msg": "Successfully deleted",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Action Policy

The action policy specifies the actions to take for detected violations. The Barracuda Load Balancer ADC provides a predefined set of attack groups containing attack actions. Each attack action specifies how to handle a particular type of web attack. You can update predefined attack actions but you cannot delete or create them.

As part of a security policy, action policies are shareable among multiple services configured on the Barracuda Load Balancer ADC.

Retrieve the Attack Actions for an Attack Group

If you want to retrieve data for all the attack actions that are part of an attack group, make a GET request to the following URI:

`/security_policies/<policy name>/attack_groups/<attack group name>/actions`

where:

- *<policy name>* is the security policy that the attack group is part of.
- *<attack group name>* is the name of the attackgroup.

If you want data for only a specific attack action, add the action name to the URI:

```
/security_policies/<policy name>/attack_groups/<attack group name>\
/actions/<attack action name>
```

The following attack groups are available:

- advanced-policy-violations
- application-profile-violations
- param-profile-violations
- protocol-violations
- request-policy-violations
- response-violations
- url-profile-violations
- header-violations

Request and Response Examples

The following examples display a request and an excerpt of the response for retrieving all the attack actions for the **`protocol-violations`** attack group that is part of the default security policy.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X GET \
   http://10.11.19.104:8000/restapi/v2/security_policies/default\
   /attack_groups/protocol-violations/actions
```

Example Response


```
{
    "data": [
        {
            "action": "protect_and_log",
            "attack_action_deny_response":
"send_response",
            "attack_group": "protocol-violations",
            "follow_up_action": "none",
            "follow_up_action_time": "60",
            "id": "directory-traversal-beyond-root",
            "name": "directory-traversal-beyond-root",
            "numeric_id": "16",
            "redirect_url": "",
            "response_page": "default"
        },
        {
            "action": "protect_and_log",
            "attack_action_deny_response":
"send_response",
            "attack_group": "protocol-violations",
            "follow_up_action": "none",
            "follow_up_action_time": "60",
            "id": "post-request-without-content-length",
            "name": "post-request-without-content-length",
            "numeric_id": "25",
            "redirect_url": "",
            "response_page": "default"
        },
.....
    ],
    "fields": null,
    "limit": null,
    "object": "ActionPolicy",
    "offset": null,
    "policy_id": "default",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```


Update an Attack Action

To update an attack action, make a PUT request to the following URI:

```
/security_policies/<policy name>/attack_groups/<attack group name>\
/actions/<attack action name>
```

where:

- *<policy name>* is the security policy that the attack group is part of.
- *<attack group name>* is the attack group.
- *<attack action name>* is the attack action.

The body of your request must be JSON, with the Content-Type header set to application/json. Only the

parameter values that are passed in the body of your request are updated. For a complete list of the attack action parameters that you can update, see Attack Action Parameters.

Request and Response Examples

The following examples display a request and response for updating the response page for the **directory-traversal-beyond-root** attack action in the **protocol-violations** attack group that is part of the default security policy.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:'\
   -H "Content-Type:application/json" \
   -X PUT \
   -d '{"response_page":"default"}' \
   http://10.11.19.104:8000/restapi/v2/security_policies/default\
   /attack_groups/protocol-violations/actions/directory-traversal-beyond-root
```

Example Response

```
{
    "id": "protocol-violations",
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Attack Action Parameters

You can configure the following parameters for attack actions.

Parameter	Data Type	Description
action	Enum	The action to be taken for an invalid request. Possible values: • **none** • **protect_and_log** • **allow_and_log** • **protect_with_no_log**

attack_action_deny_response	Enum	How to respond to the client if the request is denied. Possible values: • **close_connection** • **send_response** **Dependent variable:** ○ response_page • **temporary_redirect** **Dependent variable:** ○ redirect_url • **permanent_redirect** **Dependent variable:** ○ redirect_url
follow_up_action	Enum	The follow up action to be taken if the request is denied. Possible values: • **none** • **block_client_ip** **Dependent variable:** ○ follow_up_action_time
follow_up_action_time	Integer	The time in seconds to block the client IP. **Conditional:** Required only when the **follow_up_action** is **block_client_ip**.
redirect_url	String	The URL used to redirect the request. **Conditional:** Required only when the **attack_action_deny_response** is **temporary_redirect** or **permanent_redirect**.
response_page	Enum	The response page to be sent to the client. Possible values: • **default** • **default-error-resp** • **default-virus** **Conditional:** Required only when the **attack_action_deny_response** is **send_response**.

Data Theft Protections

Data theft protection prevents unauthorized disclosure of confidential information. The Barracuda Load Balancer ADC intercepts responses from the server and compares them to the patterns associated with data theft elements, which specify how to handle matching responses. The following predefined data theft elements are provided to prevent the disclosure of passwords, credit card information, and U.S. Social Security numbers (SSNs):

- credit-cards
- directory-indexing
- ssn

As part of a security policy, data theft protection elements are shareable among multiple services configured on the Barracuda Load Balancer ADC. You can update the predefined data theft elements but you cannot delete them. You can also create and update custom data theft elements.

Create a Custom Data Theft Element

To create a data theft element for a security policy, make a POST request to the following URI:

`/security_policies/<policy name>/data_theft_protections`

The body of your request must be JSON, with the Content-Type header set to application/json. For a list of the parameters that you must pass, see Data Theft Element Parameters.

Request and Response Examples

The following examples display a request and response for adding a custom data theft element named **`element_1`** to the default security policy.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -H "Content-Type:application/json" \
   -X POST \
   -d '{"name":"element_1", "action":"block", \
      "identity_theft_type":"social_security_numbers", \
      "custom_identity_theft_type":"CUSTOM"}' \
   http://10.11.19.104:8000/restapi/v2/security_policies/default\
   /data_theft_protections
```

Example Response

```
{
  "id" : "element_1",
  "info" : {
     "msg" : [
        "Configuration updated"
     ]
  },
  "token" : "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Update a Data Theft Element

To change the parameters for a data theft element, make a PUT request to the following URI:

`/security_policies/<policy name>/data_theft_protections/<element name>`

The body of your request must be JSON, with the Content-Type header set to application/json. Only the parameter values that are passed in the body of your request are changed or added. For a complete list of the data theft element parameters that you can update, see Data Theft Element Parameters.

Request and Response Examples

The following examples display a request and response for updating a custom data theft element named **`element_1`** that is part of the default security policy.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -H "Content-Type:application/json" \
   -X PUT \
   -d '{"response_page":"default"}' \
   http://10.11.19.104:8000/restapi/v2/security_policies/default\
   /data_theft_protections/element_1
```

Example Response

```
{
    "id": "element_1",
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Data Theft Elements

To retrieve data for all the data theft elements that are part of a security policy, make a GET request to the following URI:

`/security_policies/<policy name>/data_theft_protections`

To retrieve data for only a specific data theft element, add the element name to the URI:

`/security_policies/<policy name>/data_theft_protections/<element name>`

Request and Response Examples

The following examples display a request and response for retrieving all the data theft elements for the default security policy.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X GET \
   http://10.11.19.104:8000/restapi/v2/security_policies/default\
   /data_theft_protections
```

Example Response


```
{
    "data": [
        {
            "action": "cloak",
            "custom_identity_theft_type": "",
            "enabled": "yes",
            "id": "credit-cards",
            "identity_theft_type": "credit_cards",
            "initial_characters_to_keep": "0",
            "name": "credit-cards",
            "trailing_characters_to_keep": "4"
        },
        {
            "action": "cloak",
            "custom_identity_theft_type": "",
            "enabled": "yes",
            "id": "ssn",
            "identity_theft_type": "social_security_numbers",
            "initial_characters_to_keep": "0",
            "name": "ssn",
            "trailing_characters_to_keep": "4"
        },
        {
            "action": "block",
            "custom_identity_theft_type": "",
            "enabled": "no",
            "id": "directory-indexing",
            "identity_theft_type": "directory_indexing",
            "initial_characters_to_keep": "0",
            "name": "directory-indexing",
            "trailing_characters_to_keep": "4"
        },
        {
            "action": "block",
            "custom_identity_theft_type": "",
            "enabled": "no",
            "id": "test",
            "identity_theft_type": "custom",
            "initial_characters_to_keep": "20",
            "name": "test",
            "trailing_characters_to_keep": "4"
        }
    ],
```

```
    "fields": null,
    "limit": null,
    "object": "Data Theft Protection",
    "offset": null,
    "policy_id": "default",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Delete a Data Theft Element

To delete a data theft element from a security policy, make a DELETE request to the following URI:

/security_policies/*<policy name>*/data_theft_protections/*<element name>*

Request and Response Examples

The following examples display a request and response for deleting a custom data theft element named **element_1** that is part of the default security policy.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/security_policies/default\
   /data_theft_protections/element_1
```

Example Response

```
{
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "msg": "Successfully deleted",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Data Theft Element Parameters

You can configure the following parameters for each data theft element.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for this data theft element. **Note:** You cannot change the name of the predefined data theft elements.

Parameter	Data Type	Mandatory	Description
enabled	Enum	Yes	Enables this data theft element. Possible values: • **yes** • **no**
identity_theft_type	Enum	Yes	The identity theft pattern of the data theft element. Possible values: • **directory_indexing** • **credit_cards** • **social_security_numbers** • **custom**
custom_identity_theft_type	Enum	Conditional	The identity theft pattern defined on the **SECURITY > Libraries** page (if any). **Note:** Required only when **`identity_theft_type`** is **`custom`**.
action	Enum	Yes	How to handle any page containing this data type. Possible values: • **cloak** – Overwrites the matching data type with Xs. **Dependent variables:** ○ initial_characters_to_keep ○ trailing_characters_to_keep • **block** – Blocks the page.
initial_characters_to_keep	Integer	Conditional	The number of initial characters to be displayed to the user. **Note:** Required only when **`action`** is **`cloak`**.
trailing_characters_to_keep	Integer	Conditional	The number of trailing characters to be displayed to the user. **Note:** Required only when **`action`** is **`cloak`**.

Monitor Groups

Monitor groups are sets of monitors, or tests, that are performed by the Service Monitor on the real servers associated with a service. Use monitor groups to get a more complete picture of the health of your real servers. If a real server fails any of the tests, it is considered unavailable and is removed from the load-balancing pool.

After you create a monitor group, you can add two or more monitors to it. For information on configuring monitors, see Monitors.

Create a Monitor Group

To create a monitor group, make a POST request to the following URI:

`/monitor_groups`

The body of your request must be JSON, with the Content-Type header set to application/json. In the body of your request, pass a name for the monitor group:

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for the monitor group. **Note:** You cannot change the name of a monitor group after it is created.

Request and Response Examples

The following examples display a request and response for creating a monitor group named **`MS-IIS App Pool Monitor`**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"MS-IIS App Pool Monitor"}' \
   http://10.11.19.104:8000/restapi/v2/monitor_groups
```

Example Response

```
{
  "id": "Group1",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Monitor Groups

To retrieve data for all configured monitor groups, make a GET request to the following URI:

/monitor_groups

To retrieve data for only a specific monitor group, add the group name to the URI:

/monitor_groups/*<group name>*

Request and Response Examples

The following examples display a request and response for retrieving all monitor groups.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X GET \
   http://10.11.19.104:8000/restapi/v2/monitor_groups
```

Example Response

```
{
    "data": [
        {
            "id": "Group",
            "monitor": [
                {
                    "delay": "10",
                    "headers": "header",
                    "id": "test1",
                    "ip_address": "",
                    "match": "/testmatch",
                    "name": "test2",
                    "port": "",
                    "status_code": "200",
                    "target": "http://barracuda.com",
                    "type": "HTTP_TEST"
                },
                {
                    "delay": "10",
                    "id": "test2",
                    "ip_address": "",
                    "name": "test1",
                    "type": "ICMP_PING"
                }
            ],
            "name": "Group1"
        }
    ],
    "fields": null,
    "limit": null,
```

```
    "object": "MonitorGroups",
    "offset": null,
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Delete a Monitor Group

To delete a monitor group, make a DELETE request to the following URI:

/monitor_groups/*<group name>*

Request and Response Examples

The following examples display a request and response for deleting a monitor group named **Group1**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/monitor_groups/Group1
```

Example Response

```
{
    "msg": "Successfully deleted",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Monitors

Monitors test the real servers associated with a service. If a real server fails any tests, it is considered unavailable and is removed from the load-balancing pool.

Create a Monitor

To create a monitor, make a POST request to the following URI:

/monitor_groups/*<group name>*/monitors

where *<group name>* is the monitor group that you are adding the monitor to. The body of your request must be JSON, with the Content-Type header set to application/json. For a list of the parameters that you must pass, see Monitor Parameters.

Request and Response Examples

The following examples display a request and response for adding an MS SharePoint test named **Exchange-Owa-Monitor** to the **MSGroup** monitor group.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X POST \
   -H "Content-Type:application/json" \
   -d '{"name":"Exchange-Owa-Monitor", "type":"ms_sharepoint_secure", \
      "ip_address":"192.23.2.2", "address_version":"ipv4", "delay":"30",\
      "port":"80","username":"user1", "password":"msft","target":\
      "http://barracuda.com", "match":"/*", "additional_header":\
      "Header1:Value1", "status_code":"200"}' \
   http://10.11.19.104:8000/restapi/v2/monitor_groups/MSGroup/monitors
```

Example Response

```
{
  "id": "Exchange-Owa-Monitor",
  "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Update a Monitor

To update a monitor, make a PUT request to the following URI:

/monitor_groups/<group name>/monitors/<monitor name>

The body of your request must be JSON, with the Content-Type header set to application/json. Only the parameters values that are passed in the body of your request are updated. For a complete list of parameters that you can pass, see _____

Monitor Parameters.

Request and Response Examples

The following examples display a request and response for updating a monitor named **Exchange-Owa-Monitor**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -H "Content-Type:application/json" \
   -X PUT \
   -d '{"match": "/index"}' \
   http://10.11.19.104:8000/restapi/v2/monitor_groups/MSGroup/monitors\
   /Exchange-Owa-Monitor
```

Example Response

```
{
    "id": "Exchange-Owa-Monitor",
    "info": {
        "msg": [
            "Configuration updated"
        ]
    },
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Retrieve Monitors

To retrieve data for all monitors in a monitor group, make a GET request to the following URI:

/monitor_groups/*<group name>*/monitors

To retrieve data for only a specific monitor, add the monitor name to the URI:

/monitor_groups/*<group name>*/monitors/*<monitor name>*

Request and Response Examples

The following examples display a request and response for retrieving all the monitors that are part of the **MSGroup** monitor group.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X GET \
   http://10.11.19.104:8000/restapi/v2/monitor_groups/MSGroup/monitors
```

Example Response

```
{
    "data": [
        {
            "delay": "10",
            "headers": "header1",
            "id": "Exchange-Owa-Monitor",
            "ip_address": "",
            "match": "testmatch",
            "name": "Exchange-Owa-Monitor",
            "port": "",
            "status_code": "200",
            "target": "http://barracuda.com",
            "type": "HTTP_TEST"
        }
    ],
    "fields": null,
    "limit": null,
    "monitor_group_id": "MSGroup",
    "object": "Monitors",
    "offset": null,
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```


Delete a Monitor

To delete a monitor, make a DELETE request to the following URI:

/monitor_groups/*<group name>*/monitors/*<monitor name>*

Request and Response Examples

The following examples display a request and response for deleting a monitor named **Exchange-Owa-Monitor**.

Example Request

```
$ curl -u 'J3sidXNlcm5hbWUiOiJ5Jw=\n:' \
   -X DELETE \
   http://10.11.19.104:8000/restapi/v2/monitor_groups/MSGroup/monitors\
   /Exchange-Owa-Monitor
```

Example Response


```
{
    "msg": "Successfully deleted",
    "token": "J3sidXNlcm5hbWUiOiJ5Jw=\n"
}
```

Monitor Parameters

You can configure the following parameters for each monitor.

Parameter	Data Type	Description
name	String	The name of the monitor.
ip_address	Integer	The IP address of the real server to be tested. To test all of the servers associated with the service, leave this value blank.
address_version	Enum	The Internet protocol version. Possible values: • **ipv4** • **ipv6**
type	Enum	The tests that are used by the server monitor to determine the availability of the servers that are associated with the service. Possible values: ○ **ICMP_PING** – Performs a PING test. ○ **TCP_PORT_CHECK** – Validates that the configured service port is open. **Dependent variable:** ○ port • **UDP_PORT_CHECK** – Verifies that the UDP port is open by sending a 0 byte datagram to the real server IP address and port. This test expects to receive an ICMP Port Unreachable message to determine the result. If a firewall prevents outbound ICMP messages, the test assumes that the port is open. **Dependent variable:** ○ port • **HTTP_TEST** – Sends an HTTP request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **SIMPLE_HTTP** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ port

Parameter	Data Type	Description
		○ target ○ match ○ headers ○ status_code • **HTTP_SLOW** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **HTTPS_TEST** – Sends an HTTPS request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **HTTPS_SLOW** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **SIMPLE_HTTPS –** Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:**

Parameter	Data Type	Description
		○ port ○ target ○ match ○ headers ○ status_code • **DNS_TEST** – Sends a DNS query and compares the returned IP address to the entered IP address. **Dependent variables:** ○ port ○ target ○ match • **IMAP_TEST –** Simple test for an IMAP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ port ○ target ○ match • **POP_TEST** – Simple test for a POP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ port ○ target ○ match • **SMTP_TEST** – Simple test for SMTP servers. **Dependent variables:** ○ port ○ target ○ match • **SNMP_TEST –** Performs an SNMP GET to the specified OID and verifies that the response contains an expected pattern. If an OID is not specified, this test verifies only the availability of the service in the server. **Dependent variables:** ○ port ○ target ○ match • **SIP_TEST** – Simple test for SIP service. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service. **Dependent variable:**

Parameter	Data Type	Description
		○ port • **SIP_TLS_TEST** – Simple Test for SIP service over TLS. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service over TLS. **Dependent variable:** ○ port • **LDAP_AD_TEST** – Bind test for an LDAP/AD service. If no username and password are provided, the LDAP/AD test verifies availability of the anonymous user. **Dependent variables:** ○ port ○ target ○ match • **LDAP_AD_SSL_TEST** – Bind test for an LDAPS/AD service. If no username and password are provided, the LDAP/AD test verifies availability of the anonymous user. **Dependent variables:** ○ port ○ target ○ match • **SPAMFIREWALL_TEST** – Test for use with Barracuda Spam Firewalls. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. **Dependent variables:** ○ port ○ target ○ match • **FORCED_PORT_HTTP** – (Specific HTTP Port test) Sends an HTTP GET request using a specified port to a relative URL on the real server, and verifies that the retrieved HTML contains an expected pattern. **Dependent variables:** ○ port ○ target ○ match • **RADIUS_TEST** – Tests the availability of a RADIUS server. **Dependent variables:** ○ port ○ target

Parameter	Data Type	Description
		○ match • **RADIUS_ACCT** – Tests the availability of a RADIUS server by making an accounting request. **Dependent variables:** ○ port ○ target ○ match • **RDP_TEST** – Attempts an RDP connection to each real server to check the availability of the Terminal service. **Dependent variable:** ○ port • **FTP_TEST** – Attempts a TCP connection to each real server to check FTP availability. **Dependent variables:** ○ port ○ target ○ match • **FTPS_TEST**– Attempts a TCP connection to each real server to check FTPS availability. **Dependent variables:** ○ port ○ target ○ match • **SFTP_TEST** – Test for FTP over SSH. **Dependent variables:** ○ port ○ username ○ password • **NTLM_TEST** – Simple test for Microsoft SharePoint. **Dependent variables:** ○ port ○ username ○ password ○ target ○ match ○ headers ○ status_code • **NTLMS_TEST** – Secure test for Microsoft SharePoint. **Dependent variables:** ○ port ○ username ○ password

Parameter	Data Type	Description
		○ target ○ match ○ headers ○ status_code • **ALWAYS_PASS** – This test is used for troubleshooting or for services with management access to real servers. This test always passes regardless of the condition of the real server.
port	Integer	By default, tests use the configured real server port for the service unless the real server port is set to ALL. In that case, the tests use the default port for the test type (e.g., SMTP = 25, HTTP = 80, DNS = 53, HTTPS = 443, IMAP = 143, POP = 110, and SNMP = 161). Enter a port number to override the default port.
username	String	**Conditional:** The value for this parameter depends on the test type: • SFTP – The username for the SSH account. • MS SharePoint – The username for the SharePoint service.
password	String	**Conditional:** The value for this parameter depends on the test type: • SFTP – The password for the SSH account. • MS SharePoint – The password for the SharePoint service.
target	String	**Conditional:** The value for this parameter depends on the test type: • Barracuda Spam Firewall Test – The domain for the mail server. • DNS – The fully qualified domain name. • FTP – The username to log into server. The default username is **`anonymous`**. • FTPS – The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The complete URL (starting with **`http`** or **`https`**). • HTTP Slow and HTTPS Slow – The complete URL (starting with **`http`** or **`https`**). • IMAP and POP3 – (Optional) The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The username with

Parameter	Data Type	Description
		full LDAP schema. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The root relative URL. For example: **`/cgi-bin/index.cgi`** • RADIUS Auth and RADIUS Acct – The secret to use with the RADIUS server. • Simple HTTP and Simple HTTPS – The root relative URL. For example: **`/cgi-bin/index.cgi`** The actual URL used is **`https://<real_server_ip>:<port><URL>`** • SNMP – (Optional) Enter a valid SNMP OID. **Note:** If an OID is not specified, this test verifies only the availability of the service in the server. • Specific HTTP Port – The TCP port followed by colon (:) and the root relative URL. For example: **`8080:/cgi-bin/index.cgi`**
match	String	**Conditional:** The value for this parameters depends on the test type: • Barracuda Spam Firewall – (Optional) A pattern that is expected in the banner of the SMTP server. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. • DNS – The IP address of the hostname. • FTP – The password to log into the server. The default password is **`anonymous`**. • FTPS – The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The expected pattern in the retrieved HTML. • HTTP Slow and HTTPS Slow – The expected pattern in the retrieved HTML. • IMAP and POP3 – (Optional) The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The password to log into the server. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the

Parameter	Data Type	Description
		availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The pattern that is expected in the retrieved HTML. • RADIUS Auth – The username and password, separated by a backslash (\). For example: **`username\password`** • Simple HTTP and Simple HTTPS – The pattern expected in the retrieved HTML. • SMTP – (Optional) The pattern that is expected in the banner for the SMTP server. • SNMP – (Required if an OID is specified in the test target) A pattern to match in the response. • Specific HTTP Port – The pattern that is expected in the retrieved HTML.
headers	String	Additional headers. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
status_code	Integer	The expected status code. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
delay	Integer	The interval between test start times, in seconds.

Perl Implementation

You can download and use the Barracuda::Rest:API module to implement the Barracuda Load Balancer ADC REST API using Perl. The following sections provide methods, object descriptions, sample scripts, and example code to help you get started.

Barracuda::Rest:API Dependencies

The Barracuda::Rest:API module requires these other modules and libraries:

- Common-Sense
- JSON-XS
- Types-Serializer
- WWW-CURL
- WWW-CURL-EASY

Install the Barracuda::Rest:API Module

To install the Barracuda::Rest:API module and verify that you can use it to connect with the Barracuda Load Balancer ADC:

1. Download and extract the API.zip file. In the Barracuda TechLibrary, see the *Barracuda Load Balancer ADC - REST API* article.
2. Change to the `/t`directory in the extracted **API** folder.
3. Edit the **Barracuda-Rest-API.t** file to include the IP address of your Barracuda Load Balancer ADC.
4. Run the following commands to install and test the module:
   ```
   perl Makefile.PL
   make
   make test
   make install
   ```
   If the test successfully completes, the following message displays:
   "Obtained token. Test passed"

Perl Module Methods

The Barracuda::Rest:API module contains the following methods.

Method	Functionality
new	Initialize your environment.
login	Log into the Barracuda Load Balancer ADC to make requests.
logout	Log out of the Barracuda Load Balancer ADC.
create	Create an object.
list	Retrieve a list of objects.
get	Get data for an object.
update	Update an object or upload a certificate.
remove	Delete an object.

new

Use the **new** method to initialize your environment.

Format

```
new(ip_address, port, version, type, cacert_verify, cacert);
```

Arguments

The **new** method takes the arguments listed in the following table to specify the IP address and port number of the Barracuda Load Balancer ADC and whether you are making HTTP or HTTPS requests. For HTTPS requests, you must either specify the location of the certificate or specify insecure request.

Argument	Data Type	Mandatory	Description
ip_address	String	Yes	The IP address of the Barracuda Load Balancer ADC.
port	String	Yes	The port over which the web interface of the Barracuda Load Balancer ADC is reached.
version	String	Yes	The version of the Barracuda Load Balancer ADC REST API, which is v2.
type	String	Optional	Whether you are making HTTP or HTTPS requests. Possible values: • **http** • **https**
cacert_verify	Boolean	Conditional	Verifies that client certificates are signed by a trusted Certificate Authority. Possible values: • **true** • **false** **Note:** Only required for HTTPS requests.

Argument	Data Type	Mandatory	Description
cacert	String	Conditional	The location of the certificate. **Note:** Only required when **cacert_verify** is **true**.

Request Examples

Example HTTP Setup

```
my $API = Barracuda::ADC::API->new('10.5.6.160', '8000', 'v2','http');
```

Example CA Verified HTTPS Setup

```
my $API = Barracuda::ADC::API->new('10.5.6.160', '443', 'v2', 'https',\
'yes', 'server.crt');
```

Example HTTPS Setup without CA Verification

```
my $API = Barracuda::ADC::API->new('10.5.6.160', '443', 'v2','https', 'no');
```

login

Use the **login** method to log into the Barracuda Load Balancer ADC.

Format

```
login(username, password);
```

Arguments

Argument	Data Type	Mandatory	Description
username	String	Yes	Your username for logging into the Barracuda Load Balancer ADC.
password	String	Yes	Your password for logging into the Barracuda Load Balancer ADC.

Request Example

```
my $result = $API->login("admin", "admin");
```

logout

Use the **logout** method to log out of the Barracuda Load Balancer ADC.

Format

```
logout();
```

Request Example

```
my $result - $API->logout();
```

create

Use the **create** method to create a new instance of an object or generate a self-signed certificate.

Format

```
create(object_name, required_ids, parameters);
```

Arguments

Argument	Data Type	Mandatory	Description
object_name	String	Yes	The name of the object that you are creating.
required_ids	Hash	Yes	The ID of the object. Some objects also require the IDs of their parent objects.
parameters	Hash	Yes	The parameters of the object that you are creating.

For a list of objects with their IDs and parameters, see the Object Descriptions section.

Request Example

```
my $result = $API->create('virtual_services',
      {virtual_service_group_id => 'default'},
      {
       name => "service1",
       ip_address => "192.168.17.125",
       port => "80",
       type => "HTTP"
       address_version => "ipv4",
       netmask => "255.255.255.0",
       interface => "ge-1-1"
      }
);
```

list

Use the the **list** method to retrieve a list of objects under a parent object .

Format

```
list(object_name, required_ids);
```

Arguments

Argument	Data Type	Mandatory	Description
object_name	String	Yes	The objects that you want to retrieve.
required_ids	Hash	Yes	The ID of the parent object. If you want to get data for only a specific object, include its ID. If you do not include the ID of a specific object, then all objects under the specified parent object id are retrieved.

For a list of objects with their IDs and parameters, see the Object Descriptions section.

Request Example

```
my $result = $API->list('virtual_services',
      {virtual_service_group_id => 'default'});
```

get

Use the **get** method to get data for a specific object.

Format

```
get(object_name, required_ids, fields);
```

Arguments

Argument	Data Type	Mandatory	Description
object_name	String	Yes	The object that you want to get.
required_ids	Hash	Yes	The ID of the object. Some objects also require the IDs of their parent objects.
fields	Hash	Optional	The parameters of the object that you want to get. Only these parameters are returned for the object.

For a list of objects with their IDs and parameters, see the Object Descriptions section.

Request Example

```
my $result = $API->get('virtual_services',
       {
        virtual_service_group_id => 'default',
        id => "service1"
       },
       {fields => "name,ip_address"}
);
```

update

Use the **update** method to update an object or upload a certificate. Only the parameter values that you pass are updated.

Format

```
update(object_name, required_ids, parameters);
```

Arguments

Argument	Data Type	Mandatory	Description
object_name	String	Yes	The object to update.
required_ids	Hash	Yes	The ID of the object. Some objects also require the IDs of their parent objects.
parameters	Hash	Yes	The parameters to update or add for the object.

For a list of objects with their IDs and parameters, see the Object Descriptions section.

Request Example

```
my $result = $API->update('virtual_services',
       {virtual_service_group_id => 'default',
        id => "service1"},
       {"enable": 1});
```

remove

Use the **remove** method to delete an object.

Format

```
remove(object_name, required_ids);
```

Arguments

Argument	Data Type	Mandatory	Description
object_name	String	Yes	The object that you want to update.
required_ids	Hash	Yes	The ID of the object. Some objects also require the IDs of their

Argument	Data Type	Mandatory	Description
			parent objects.

For a list of objects with their IDs and parameters, see the Object Descriptions section.

Request Example

```
my $result = $API->remove('virtual_services',
       {
        virtual_service_group_id => 'default',
        id => "service1"
       }
);
```

Object Descriptions

The following sections list each Perl object with their names, IDs, and parameters.

Virtual Service Groups

A virtual service group is a container that you can add multiple services to. For example, you can create a virtual service group to organize services by type (e.g., Instant SSL) or application (e.g., Lync). You can also used the predefined virtual service group named **default**.

Object Name

`virtual_service_groups`

ID

ID	Data Type	Description
id	String	The name of the virtual service group.

Parameter

When you create a virtual service group, you only need to pass a name for it.

Parameter	Data Type	Description
name	String	A name for the service group.

Request Examples

The following table lists example requests for retrieving virtual service groups and configuring a virtual service group named **group1**.

Method	Example Usage	Example Request
create	Create a virtual service group named group 1.	`my $result = $API->create('virtual_service_groups',undef, {name => "group1"});`
list	List all virtual service groups.	`my $result = $API->list('virtual_service_groups');`
get	Retrieve a service named Exchange that is part of group 1.	`my $result = $API->show('virtual_service_groups', {id => "Exchange"});`
update	Rename group 1 as Exchange Services.	`my $result = $API->update('virtual_service_groups', {id => "group1"}, {name => "Exchange Services"});`
delete	Delete group 1.	`my $result = $API->delete('virtual_service_groups', {id => "group1"});`

Virtual Services

A virtual service is a combination of a virtual IP (VIP) address and a TCP port, which listens and directs the traffic to a real server associated with the service. You must create the virtual service as part of a virtual service group.

Object Name

`virtual_services`

IDs

ID	Data Type	Description
virtual_service_groups_id	String	The name of the virtual service group that that virtual service is part of.
id	String	The name of the virtual service group.

Parameters

The following table lists only the parameters that you must pass when creating a virtual service. For a complete list of service parameters, see Virtual Service Parameters.

Parameter	Data Type	Mandatory	Description
name	String	Yes	The name of the service.
address_version	Enum	Yes	The Internet protocol version of the service. Possible values include: • **ipv4** • **ipv6**
ip_address	String	Yes	The virtual IP address. The IP address format depends on the specified **address_version**.
port	Integer	Yes	The port number for the service.
type	Enum	Yes	The type of the service. Possible values: • **HTTP** • **HTTPS** • **INSTANTSSL** • **FTP** • **FTPSSL** • **UDP** • **L4** • **L7UDP** • **L7Tcp RDP**
netmask	String	Yes	The netmask depends on the **address_version** specified.
interface	Enum	Yes	The interface for the service. The value depends on the appliance. For example, ge-1-1, ge-1-2, and ge-2-1.
service_hostname	String	Conditional	The domain name to identify and rewrite HTTP requests to HTTPS. **Note:** Required only for Instant SSL services.
certificate	String	Conditional	The certificate that is presented by the service when authenticating itself to a browser or some other client. **Note:** Required only for HTTPS, Instant SSL, FTP SSL, and SSL services.

Request Examples

The following table lists example requests for retrieving virtual services and configuring a virtual service named service1 in the default virtual service group.

Method	Example Usage	Example Request

create	Create a virtual service named service1 in the default virtual service group.	`my $result = $API->create('virtual_services',`
		`{virtual_service_group_id => 'default'},`
		`{`
		`name => "service1",`
		`ip_address => "192.168.17.125",`
		`port => 80,`
		`type => "HTTP",`
		`address_version => "ipv4",`
		`netmask => "255.255.255.0",`
		`interface => "ge-1-1"`
		`}`
		`);`

list	List all virtual services that are part of the default service group.	`my $result = $API->list('virtual_services',` `{virtual_service_group_id => 'default'});`

get	Get the name and IP address of service1.	`my $result = $API->get('virtual_services',`
		`{virtual_service_group_id => 'default',`
		`id => "service1"},`
		`{fields => "name, ip_address"}`
		`);`

update	Enable service1.	`my $result = $API->update('virtual_services',`
		`{virtual_service_group_id => 'default',`
		`id => "service1"},`
		`{enable => true}`
		`);`

remove	Remove service1.	`my $result = $API->remove('virtual_services',`
		`{virtual_service_group_id => 'default',`
		`id => "service1" });`

Certificates

A signed certificate is a digital identity document that enables both server and client to authenticate each other. Certificates are used with the HTTPS protocol to encrypt secure information transmitted over the Internet. A certificate contains information such as user name, expiration date, a unique serial number assigned to the certificate by a trusted CA, the public key, and the name of the CA that issued the certificate.

You can:

- Generate a signed certificate.
- Upload a signed certificate.
- Upload a trusted certificate.
- Download a certificate.

Object Name

`certificates`

ID

ID	Data Type	Description
id	String	The name of the certificate.
ecdsa_status	Boolean	Enables the ECDSA ciper suite if you have an ECDSA certificate key pair. It is presented to any browser attempting to access the Service. Possible values: • **true** • **false**
ecdsa_certificate	String	The name of the ECDSA certificate.

Self-Signed Certificate Parameters

The following table lists all of the parameters that you must pass for generating a self-signed certificate.

Parameter	Data Type	Mandatory	Description
name	String	Yes	The name of the certificate.
common_name	String	Yes	The domain name (DN) of the web server for which you want to generate the certificate.
country_code	String	Yes	The two-letter country code of the location of the organization.
state	String	Optional	The full name of the state or province of the location of the organization.
city	String	Optional	The full name of the locality (city) where the organization is located.
organization_name	String	Optional	The legally registered name of the organization or company.
organization_unit	String	Optional	The department or unit within the organization.
key_size	Enum	Yes	The private key size for the certificate in bits. Possible values: • **1024** • **2048** • **4096**
allow_private_key_export	Boolean	Yes	Locks the private key corresponding to this certificate. Possible values: • **1** – true • **0** – false Normally, certificates are downloaded in

Parameter	Data Type	Mandatory	Description
			PKCS #12 format which includes the private key and certificate. When a key is locked, you can only download the certificate in PEM format. Also, you cannot take a backup when the private key is locked. **Note:** This option is valid only for created and uploaded (generated and signed by a trusted CA) certificates.

Signed Certificate Parameters

You can upload a signed certificate in PEM or PKCS12 format. Use the **update** method and pass the parameters listed in the following table:

Parameter	Data Type	Mandatory	Description
name	String	Yes	The name of the certificate.
type	Enum	Yes	The certificate type. Possible values: • **pkcs** • **pem** **Dependent variables:** • intermediary_certificate • password
signed_certificate	Cert	Yes	The path and name of the signed certificate file that must be uploaded.
password	String	Conditional	The password used to generate the PKCS #12 token for the signed certificate being uploaded. **Note:** Required only for uploading certificates in PKCS #12 Token format.
assigned_associated_key	Boolean	Conditional	Possible values: • **1** – If the CSR corresponding to this certificate was generated on the Barracuda Load Balancer ADC. • **0** – Upload the private key corresponding to this certificate specified by the **key** parameter. **Note:** Required only for uploading certificates in PEM format.
key	Public Key	Conditional	The path and name of the corresponding private key for the signed certificate being uploaded. **Note:** Required only for uploading certificates in PEM format.
intermediary_certificate	Array	Conditional	The path and name of the intermediary CA certificate file that must be uploaded.

Parameter	Data Type	Mandatory	Description
			If your certificate is signed by a trusted CA, upload the certificate in the following order: 1. Leaf certificate 2. Intermediate certificate(s) 3. Root CA certificate **Note:** Required only for uploading certificates in PEM format.
allow_private_key_export	Boolean	Yes	Exports the private key corresponding to the certificate. Possible values: • **1** – Export the private key corresponding to the certificate. • **0** – Lock the private key. In this case, the certificate can be downloaded only in PEM format and a backup of the system configuration cannot be taken.
upload	String	Yes	Possible value: **signed**

Trusted Certificate Parameters

You can upload a trusted certificate in PEM format. Use the **update** method and pass the following parameters:

Parameter	Data Type	Mandatory	Description
name	String	Yes	The name of the certificate.
trusted_certificate	String	Yes	The path and name of the trusted server certificate that must be uploaded.
upload	String	Yes	Possible value: **trusted**

Request Examples

The following table provides example requests for generating, uploading, and downloading certificates.

Method	Example Usage	Example Request

create	Generate a self-signed certificate named sign_cert.	`my $result = $API->create('certificates',undef,`
		`{`
		`name => "sign_cert",`
		`common_name =>`
		`"barracuda.yourdomain.com",`
		`country_code => "US",`
		`state => "CA",`
		`city => "Campbell",`
		`organization_name => "Barracuda",`
		`organization_unit => "Engineering",`
		`key_size => "1024",`
		`allow_private_key_export => "true"`
		`}`

Method	Example Usage	Example Request
		);
create	Upload a signed certificate named sign_cert.	my $result = $API->create('certificates',undef, { name => "sign_cert", type => "pem", signed_certificate => "@server.crt", assign_associated_key => "false", key => "@server.key", allow_private_key_export => "true" }, {upload => "signed"});
create	Upload a a trusted certificate named trust_cert.	my $result = $API->create('certificates',undef, { name => "trust_cert", trusted_certificate => "@mycert.crt" }, {upload => "trusted"});

get	Download a certificate named cert.	my $result = $API->get('certificates',undef,
		{
		name => "cert",
		encrypt_password => "test"
		}
		);

Servers

A server object represents a real back-end server. You can add and configure multiple real servers to load balance incoming traffic for a service.

Object Name

servers

IDs

ID	Data Type	Description
virtual_service_groups_id	String	The name of the virtual service group of the server's service.
virtual_services_id	String	The name of the virtual service that you are adding the server to.

Parameters

The following table lists only the parameters that you must pass when creating a server. For a complete list of server parameters, see Server Parameters.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name to identify the server.

Parameter	Data Type	Mandatory	Description
identifier	Enum	Yes	How the Barracuda Load Balancer ADC identifies the server. Possible values: • **hostname** • **ip_address**
ip_address	String	Conditional	The IP address of the server. **Note:** Required when the **identifier** is **ip_address** .
hostname	String	Conditional	The hostname of the server. **Note:** Required when the **identifier** is **hostname**.
port	String	Yes	The port number of the server.

Request Examples

The following table provides example requests for retrieving servers and configuring a server named Server_12.0.0.147 for a virtual service named service1 that is part of the default virtual service group.

Method	Example Usage	Example Request

create	Create a server named Server_12.0.0.147 for the service1 service that is part of the default virtual service group.	`my $result = $API->create('servers',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1"`
		`},`
		`{`
		`name => "Server_12.0.0.147",`
		`address_version => "ipv4",`
		`ip_address => "12.0.0.147",`
		`port => 80,`
		`identifier => "ipaddr"`
		`}`
		`);`

list	List all of the servers that are associated with service1.	`my $result = $API->list('servers',`
		`{virtual_service_group_id => 'default',`
		`virtual_service_id => "service1"});`

get	Retrieve data for Server_12.0.0.147.	`my $result = $API->get('servers',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1",`
		`id => "Server_12.0.0.147"`
		`}`
		`);`

Method	Example Usage	Example Request
update	Update the port for Server_12.0.0.147.	`my $result = $API->update('servers', { virtual_service_group_id => 'default', virtual_service_id => "service1", id => "Server_12.0.0.147" }, {port => 22} );`
remove	Remove Server_12.0.0.147	`my $result = $API->remove('servers', { virtual_service_group_id => 'default', virtual_service_id => "service1", id => "Server_12.0.0.147" } );`

Content Rules

For HTTP, HTTPS, and Instant SSL services, you can create content rules to apply caching, compression, load balancing, and persistence settings to incoming web traffic matching the URL, host, and extended match patterns specified in the rules.

The extended match pattern is one or more expressions that consist of a combination of HTTP headers and/or query string parameters. Multiple rules for a service are evaluated in the order established by their extended match sequence numbers. The rule with the most specific host and URL match is executed.

Object Name

`content_rules`

IDs

ID	Data Type	Description
virtual_service_groups_id	String	The name of the virtual service group of the content rule's service.
virtual_services_id	String	The name of the virtual service that you are adding the content rule to.

Parameters

The following table lists only the parameters that are required when you create a content rule. For a complete list of content rule parameters, see
Content Rule Parameters.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for the content rule.
host_match	String	Yes	A host name to be matched against the host in the request header.

Parameter	Data Type	Mandatory	Description
url_match	String	Yes	A URL to be matched to the URL in the request header.
extended_match	String	Yes	An expression that consists of a combination of HTTP headers and/or query string parameters.
extended_match_sequence	Integer	Yes	A number to specify the rank of this rule when multiple content rules are configured for this service. Content rules are evaluated sequentially according their extended match sequence number.

Request Examples

The following table provides example requests for retrieving content rules and configuring a content rule named rule1 for the service1 virtual service that is part of the default virtual service group.

Method	Example Usage	Example Request

create	Create a content rule named rule1 for the service1 virtual service that is part of the default virtual service group.	`my $result = $API->create('content_rules',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1"`
		`},`
		`{`
		`name => "rule1",`
		`host_match => "*.barracuda.com",`
		`url_match => "/*",`
		`extended_match => "*",`
		`extended_match_sequence => 5`
		`}`
		`);`

list	List all of the content rules for service1.	`my $result = $API->list('content_rules',`
		`{virtual_service_group_id => 'default',`
		`virtual_service_id => "service1"});`

get	Retrieve data for rule1.	`my $result = $API->get('content_rules',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1",`
		`id => "rule1"`
		`}`
		`);`

update	Update the URL match for rule1.	`my $result = $API->update('content_rules',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1",`
		`id => "rule1"`
		`},`
		`{url_match => "/barracuda"}`
		`);`

Method	Example Usage	Example Request

remove	Remove rule1.	`my $result = $API->remove('content_rules',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1",`
		`id => "rule1"`
		`}`
		`);`

Rule Group Servers

For each content rule, you can add rule group servers to handle matching requests.

Object Name

`rg_servers`

IDs

ID	Data Type	Description
virtual_service_groups_id	String	The name of the virtual service group of the content rule's service.
virtual_services_id	String	The name of the content rule's service.
content_rules_id	String	The name of the content rule that you are adding the server to.

Parameters

The following table lists only the parameters that are required when you create a rule group server. For a complete list of rule group server parameters, see Rule Group Server Parameters.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name to identify the server.
identifier	Enum	Yes	Indicates whether the Barracuda Load Balancer ADC identifies the server by its IP address or hostname. Possible values: • **hostname** • **ipaddr**
ip_address	String	Conditional	The IP address of the server. **Conditional:** Required when the **identifier** is **ipaddr**.
hostname	String	Conditional	The hostname of the server. **Conditional:** Required when the **identifier** is **hostname**.
port	Integer	Yes	The port number of the server.

Request Examples

The following table provides example requests for retrieving rule group servers and configuring a rule group server named rule_server.

Method	Example Usage	Example Request

create	Create a rule group server named rule_server for a content rule named rule1, which is part of the service1 virtual service in the default virtual service group.	`my $result = $API->create('rg_servers',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1"`
		`content_rules_id => "rule1"`
		`},`
		`{`
		`name => "rule_server",`
		`address_version => "ipv4",`
		`ip_address => "10.66.44.104",`
		`port => 80,`
		`identifier => "ipaddr"`
		`}`
		`);`

list	List all of the rule group servers for rule1.	`my $result = $API->list('rg_servers',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1"`
		`content_rules_id => "rule1"`
		`}`
		`);`

get	Retrieve data for rule_server.	`my $result = $API->get('rg_servers',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1"`
		`content_rules_id => "rule1"`
		`id => "rule_server"`
		`}`
		`);`

update	Update the weight and status for rule_server.	`my $result = $API->update('rg_servers',`
		`{`
		`virtual_service_group_id => 'default',`
		`virtual_service_id => "service1"`
		`content_rules_id => "rule1"`
		`id => "rule_server"`
		`}`
		`{`
		`weight => 5,`
		`status => "sticky"`
		`}`
		`);`

Method	Example Usage	Example Request
remove	Remove rule_server.	`my $result = $API->remove('rg_servers', { virtual_service_group_id => 'default', virtual_service_id => "service1" content_rules_id => "rule1" id => "rule_server" } );`

Security Policies

For HTTP, HTTPS, and Instant SSL services, you can configure a security policy to protect your application against vulnerabilities and malicious attacks. Each security policy is comprised of the following subpolicies to encrypt, cloak, and restrict the data that is included in HTTP requests and responses:

- Request Limits
- Cookie Security
- URL Protection
- Parameter Protection
- Cloaking
- URL Normalization

All security policies are global and can be shared among multiple services configured on the Barracuda Load Balancer ADC. You can either create custom security policies or use the predefined security policies. A default security policy is available, as well as predefined security policies for SharePoint, OWA, and Oracle.

Object Name

`security_policies`

ID

ID	Data Type	Description
id	String	The name of the security policy.

Parameter

When you create a security policy, you are only required to pass a name for it. For a complete list of security policy parameters, see Security Policy Parameters.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name to identify the security policy.

Request Examples

The following table provides example requests for retrieving security policies and configuring a security policy named security_policy1.

Method	Example Usage	Example Request
create	Create a security policy named security_policy1.	`my $result = $API->create('security_policies',undef {name => "security_policy1",});`
list	List all security policies.	`my $result = $API->list('security_policies');`
get	Retrieve data for security_policy1.	`my $result = $API->get('security_policies', {id => "security_policy1"});`
update	Update the cookie security settings for security_policy1.	`my $result = $API->update('security_policies', {id => "security_policy1"} { cookie_security => { cookie_replay_protection_type => "none" allow_unrecognized_cookies => "never", "tamper_proof_mode => "encrypted" } } );`
remove	Remove security_policy1.	`my $result = $API->remove('security_policies', {id => "security_policy1"});`

Global ACLs

Global ACLs (URL ACLs) define strict allow and deny rules that are matched to requests by URL and extended match expressions. As part of a security policy, global ACLs are shareable among multiple services configured on the Barracuda Load Balancer ADC.

Object Name

`global_acls`

IDs

ID	Data Type	Description
security_policies_id	String	The name of the security policy that the global ACL rule is part of.
id	String	The name of the global ACL rule.

Parameters

The following table lists all of the parameters that are required when you create a global ACL rule.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for the URL ACL rule.
enabled	Enum	Optional	Enables this rule. Possible values: • **yes** • **no**
url_match	String	Yes	The URL to be matched to the URL in the request. The URL must start with a forward slash (**/**) and can have a maximum of one asterisk (*****) anywhere in the URL. To apply the ACL for all URLs in a domain, use: **/***
extended_match	String	Yes	An expression that consists of a combination of HTTP headers and/or query string parameters. Updating extended match parameters value is shown in the example below. For more information on how to write extended match expressions, see Extended Match Syntax.
extended_match_sequence	Integer	Yes	The order in which to evaluate this rule's Extended Match expression when a request matches multiple rules with the same URL match and Host match.
action	Enum	Yes	The action to be taken on the request matching the URL for this rule. Possible values: • **process** • **allow** • **deny_and_log** • **redirect** • **deny_with_no_log**
redirect_url	String	Conditional	A URL to which a user is redirected. **Note:** Required only when the **`action`** is **`redirect`**.
comments	String	Optional	Description about the global ACL rule.

Request Examples

The following table provides example requests for retrieving global ACL rules and configuring a rule named acl1.

Method	Example Usage	Example Request

create	Create a global ACL rule named acl1 for the default security policy.	`my $result = $API->create('global_acls',`
		`{security_policies_id => "default"},`
		`{`
		`name => "acl1",`
		`extended_match_sequence => "2",`
		`extended_match => "*",`
		`url_match => "/*/new_acl",`
		`action => process",`
		`enabled => "yes"`
		`}`
		`);`

list	List all global ACL rules for the default security policy.	`my $result = $API->list('global_acls',` `{security_policies_id => "default"});`

get	Retrieve data for acl1.	`my $result = $API->get('global_acls',`
		`{`
		`security_policies_id => "default"},`
		`id => "acl1"`
		`}`
		`);`

update	Update the sequence number and add a comment for acl1.	`my $result = $API->update('global_acls',`
		`{`
		`security_policies_id => "default"},`
		`id => "acl1"`
		`}`
		`{`
		`extended_match_sequence => "1",`
		`comments => "first rule in sequence",`
		`}`
		`);`

remove	Remove acl1.	`my $result = $API->remove('global_acls',`
		`{`
		`security_policies_id => "default"},`
		`id => "acl1"`
		`}`
		`);`

Action Policy

The action policy specifies the actions to take for detected violations. The Barracuda Load Balancer ADC provides a predefined set of attack groups containing attack actions. Each attack action specifies how to handle a particular type of web attack. You can update the predefined attack actions but you cannot delete them or create new ones.

As part of a security policy, action policies are shareable among multiple services configured on the Barracuda Load Balancer ADC.

Object Name

`actions`

IDs

ID	Data Type	Description
security_policies_id	String	The name of security policy that the attack group is part of.
attack_groups_id	String	The name of the attack group. The following attack groups are available: • advanced-policy-violations • application-profile-violations • param-profile-violations • protocol-violations • request-policy-violations • response-violations • url-profile-violations • header-violations
id	String	The name of the attack action. To get the names of attack actions, use the **list** method.

Parameters

For a full list of the parameters that you can pass for attack actions, see the Attack Action Parameters section

Request Examples

The following table provides example requests for retrieving attack actions and configuring the directory-traversal-beyond-root attack action in the protocol-violations attack group that is part of the default security policy.

Method	Example Usage	Example Request

list	List all attack actions for the protocol-violations attack group	`my $result = $API->list('actions',`
		`{`
		`security_policies_id => "default",`
		`attack_groups_id => "protocol-violations",`
		`}`
		`);`

get	Retrieve data for the directory-traversal-beyond-root attack action in the protocol-violations attack group that is part of the default security policy.	`my $result = $API->get('actions',`
		`{`
		`security_policies_id => "default",`
		`attack_groups_id => "protocol-violations",`
		`id => "directory-traversal-beyond-root"`
		`}`
		`);`

update	Update the response page for the directory-traversal-beyond-root attack action.	`my $result = $API->update('actions',`
		`{`
		`security_policies_id => "default",`
		`attack_groups_id => "protocol-violations",`
		`id => "directory-traversal-beyond-root"`
		`}`
		`{response_page => "default"}`
		`);`

Data Theft Protections

Data theft protection prevents unauthorized disclosure of confidential information. The Barracuda Load Balancer ADC intercepts responses from the server and compares them to the patterns associated with data theft elements, which specify how to handle matching responses. The following predefined data theft elements are provided to prevent the disclosure of passwords, credit card information, and U.S. Social Security numbers:

- credit-cards
- directory-indexing
- ssn

As part of a security policy, data theft protection elements are shareable among multiple services configured on the Barracuda Load Balancer ADC. You can update the predefined data theft elements but you cannot delete them. You can also create and update custom data theft elements.

Object Name

`data_theft_protections`

IDs

ID	Data Type	Description
security_policies_id	String	The name of the security policy that the data theft element is part of.
id	String	The name of the data theft element.

Parameters

The following table lists all of the parameters that you can pass for each data theft element.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for this data theft element. **Note:** You cannot change the name of the predefined data theft elements.
enabled	Enum	Yes	Enables this data theft element. Possible values: • **yes** • **no**
identity_theft_type	Enum	Yes	The identity theft pattern of the data theft element. Possible values: • **directory_indexing** • **credit_cards** • **social_security_numbers** • **custom**
custom_identity_theft_type	Enum	Conditional	The identity theft pattern defined on the **SECURITY > Libraries** page (if any). **Note:** Required only when **`identity_theft_type`** is **`custom`**.
action	Enum	Yes	How to handle any page containing this data type. Possible values: • **cloak** – Overwrites the matching data type with Xs. **Dependent Variables:** ○ initial_characters_to_keep ○ trailing_characters_to_keep • **block** – Blocks the page.
initial_characters_to_keep	Integer	Conditional	The number of initial characters to be displayed to the user. **Note:** Required only when **`action`** is **`cloak`**.

Parameter	Data Type	Mandatory	Description
trailing_characters_to_keep	Integer	Conditional	The number of trailing characters to be displayed to the user. **Note:** Required only when **action** is **cloak**.

Request Examples

The following table provides example requests for retrieving data theft elements and configuring a data theft element named element_1.

Method	Example Usage	Example Request

create	Create a data theft element named element_1 as part of the default security group.	`my $result = $API->create('data_theft_protections',`
		`{security_policies_id => "default"},`
		`{`
		`name => "element_1",`
		`action => "block",`
		`identity_theft_type => "social_security_numbers",`
		`custom_identity_theft_type => "CUSTOM"`
		`}`
		`);`

list	List all data theft elements for the default security policy.	`my $result =$API->list('data_theft_protections',` `{security_policies_id => "default"});`

get	Retrieve data for element_1.	`my $result = $API->get('data_theft_protections',`
		`{`
		`security_policies_id => "default"},`
		`id => "element_1"`
		`}`
		`);`

update	Disable element_1.	`my $result = $API->update('data_theft_protections',`
		`{`
		`security_policies_id => "default"},`
		`id => "element_1"`
		`}`
		`{`
		`enabled => "no"`
		`}`
		`);`

remove	Remove element_1.	`my $result = $API->remove('data_theft_protections',`
		`{`
		`security_policies_id => "default"},`
		`id => "element_1"`
		`}`
		`);`

Monitor Groups

Monitor groups are sets of monitors, or tests, that are run by the Service Monitor on the real servers associated with a service. Use monitor groups to get a more complete picture of the health of your real servers. If a real server fails any tests, it is considered unavailable and is removed from the load-balancing pool.

After you create a monitoring group, you can add two or more monitors to it. For information on configuring monitors, see the Monitors section.

Object Name

```
monitor_groups
```

ID

ID	Data Type	Description
id	String	The name of the monitor group.

Parameter

When you create a monitor group, you only need to pass a name for it.

Parameter	Data Type	Mandatory	Description
name	String	Yes	A name for the monitor group. **Note:** You cannot change the name of a monitor group after it is created.

Request Examples

The following table provides example requests for retrieving monitor groups and configuring a monitor group named MS-IIS App Pool Monitor.

Method	Example Usage	Example Request
create	Create a monitor group named MS-IIS App Pool Monitor.	`my $result = $API->create('monitor_groups',` `{name => "MS-IIS App Pool Monitor",});`
list	List all monitor groups.	`my $result = $API->list('monitor_groups');`
get	Retrieve data for MS-IIS App Pool Monitor.	`my $result = $API->get('monitor_groups',` `{id => "MS-IIS App Pool Monitor"});`

update	Rename MS-IIS App Pool Monitor as MS-IIS App Pool Monitor 2.	`my $result = $API->update('monitor_groups',`
		`{id => "MS-IIS App Pool Monitor"}`
		`{name => "MS-IIS App Pool Monitor 2"}`
		`);`

remove	Remove MS-IIS App Pool Monitor.	`my $result = $API->remove('monitor_groups',` `{id => "MS-IIS App Pool Monitor"});`

Monitors

Monitors test the real servers associated with a service. If a real server fails tests, it is considered unavailable and removed from the load-balancing pool.

Object Name

`monitors`

IDs

ID	Data Type	Description
monitor_group_id	String	The name of the monitor group that you want to add the monitor to.
id	String	The name of the monitor.

Parameters

The following table lists all of the parameters that you can pass for a monitor.

Parameter	Data Type	Description
name	String	The name of the monitor.
ip_address	Integer	The IP address of the real server to test. To test all servers that are associated with the service, leave this value blank.
address_version	Enum	The Internet protocol version. Possible values: • **ipv4** • **ipv6**
type	Enum	The tests that are used by the server monitor to determine the availability of the servers that are associated with the service. Possible values: • **ICMP_PING** – Performs a PING test. • **TCP_PORT_CHECK** – Validates that the configured service port is open. **Dependent variable:** ○ port • **UDP_PORT_CHECK** – Verifies that the UDP port is open by sending a 0 byte datagram to the real server IP address and port. This test expects to receive an ICMP Port Unreachable message to determine the result. If a firewall prevents outbound ICMP messages, the test assumes that the port is open. **Dependent variable:** ○ port • **HTTP_TEST** – Sends an HTTP request to the specified URL to verify that the response contains the expected pattern, headers,

Parameter	Data Type	Description
		and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **SIMPLE_HTTP** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **HTTP_SLOW** – Sends an HTTP request to the specified relative URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **HTTPS_TEST** – Sends an HTTPS request to the specified URL to verify that the response contains the expected pattern, headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **HTTPS_SLOW** – Sends an HTTPS request to the specified relative URL to verify that the response contains the expected pattern,

Parameter	Data Type	Description
		headers, and/or status code. The real server is used as a proxy server to retrieve the page, so the forward proxy setting on the real server must be enabled. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **SIMPLE_HTTPS** – Sends an HTTPS request to the specified relative URL to verify that the response contains theexpected pattern, headers, and/or status code. **Dependent variables:** ○ port ○ target ○ match ○ headers ○ status_code • **DNS_TEST** – Sends a DNS query and compares the returned IP address to the entered IP address. **Dependent variables:** ○ port ○ target ○ match • **IMAP_TEST** – Simple test for IMAP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ port ○ target ○ match • **POP_TEST –** Simple test for POP service. If no username and password are provided, this test verifies only the availability of the service in the server. **Dependent variables:** ○ port ○ target ○ match • **SMTP_TEST** – Simple test for SMTP servers. **Dependent variables:** ○ port ○ target ○ match

Parameter	Data Type	Description
		• **SNMP_TEST –** Sends an SNMP GET to the specified OID and verifies that the response containes an expected pattern. If no OID is specified, this test verifies only the availability of the service in the server. **Dependent variables:** ○ port ○ target ○ match • **SIP_TEST** – Simple test for SIP service. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service. **Dependent variable:** ○ port • **SIP_TLS_TEST** – Simple Test for SIP service over TLS. This test sends an OPTIONS packet to the SIP server to check the availability of the SIP service over TLS. **Dependent variable:** ○ port • **LDAP_AD_TEST** – Bind test for LDAP/AD service. If no username and password are provided, the LDAP/AD test verifies availability of the anonymous user. **Dependent variables:** ○ port ○ target ○ match • **LDAP_AD_SSL_TEST** – Bind test for LDAPS/AD service. If no username and password are provided, the LDAP/AD test verifies availability of the anonymous user. **Dependent variables:** ○ port ○ target ○ match • **SPAMFIREWALL_TEST** – Test for use with Barracuda Spam Firewalls. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. **Dependent variables:** ○ port ○ target ○ match • **FORCED_PORT_HTTP** – (Specific HTTP Port

Parameter	Data Type	Description
		test) Sends an HTTP GET request using a specified port to a relative URL on the real server, and verifies that the retrieved HTML contains an expected pattern. **Dependent variables:** ○ port ○ target ○ match • **RADIUS_TEST** – Tests the availability of a RADIUS server. **Dependent variables:** ○ port ○ target ○ match • **RADIUS_ACCT** – Tests the availability of a RADIUS server by making an accounting request. **Dependent variables:** ○ port ○ target ○ match • **RDP_TEST** – Attempts an RDP connection to each Real Server to check the availability of the Terminal Service. **Dependent variable:** ○ port • **FTP_TEST** – Attempts a TCP connection to each real server to check FTP availability. **Dependent variables:** ○ port ○ target ○ match • **FTPS_TEST**– Attempts a TCP connection to each real server to check FTPS availability. **Dependent variables:** ○ port ○ target ○ match • **SFTP_TEST –** Test for FTP over SSH. **Dependent variables:** ○ port ○ username ○ password • **NTLM_TEST** – Simple test for Microsoft SharePoint. **Dependent variables:** ○ port ○ username ○ password

Parameter	Data Type	Description
		○ target ○ match ○ headers ○ status_code • **NTLMS_TEST** – Secure test for Microsoft SharePoint. **Dependent variables:** ○ port ○ username ○ password ○ target ○ match ○ headers ○ status_code • **ALWAYS_PASS** – This test is used for troubleshooting or for services used for management access to real servers. This test always passes regardless of the condition of the real server.
port	Integer	By default, tests try to use the configured real server port for the service unless the real server port is set to ALL. In that case, the tests use the default port for the test type (e.g., SMTP = 25, HTTP = 80, DNS = 53, HTTPS = 443, IMAP = 143, POP = 110, and SNMP = 161). Enter a port number to override the default port.
username	String	**Conditional:** The value for this parameter depends on each test type: • SFTP – The username for the SSH account. • MS SharePoint – The username for the SharePoint service.
password	String	**Conditional:** The value for this parameter depends on each test type: • SFTP – The password for the SSH account. • MS SharePoint – The password for the SharePoint service.
target	String	**Conditional:** The value for this parameter depends on each test type: • Barracuda Spam Firewall Test – The domain for the mail server. • DNS – The fully qualified domain name. • FTP – The username to log into server. The default username is anonymous. • FTPS – The username to log into the server. **Note:** If no username and password are

Parameter	Data Type	Description
		provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The complete URL (starting with http or https). • HTTP Slow and HTTPS Slow – The complete URL (starting with http or https). • IMAP and POP3 – (Optional) The username to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The username with full LDAP schema. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The root relative URL. For example: **`/cgi-bin/index.cgi`** • RADIUS Auth and RADIUS Acct – The secret to use with the RADIUS server. • Simple HTTP and Simple HTTPS – The root relative URL. For example: **`/cgi-bin/index.cgi`** The actual URL used is: **`https://<real_server_ip>:<port><URL>`** • SNMP – (Optional) Enter a valid SNMP OID. **Note:** If no OID is specified, this test verifies only the availability of the service in the server. • Specific HTTP Port – The TCP port followed by colon (:) and the root relative URL. For example: **`8080:/cgi-bin/index.cgi`**
match	String	**Conditional:** The value for this parameters depends on each test type: • Barracuda Spam Firewall – (Optional) A pattern that is expected in the banner of the SMTP server. **Note:** The Barracuda Load Balancer ADC's IP address must be exempted from any Rate Control settings on the Barracuda Spam Firewall. • DNS – The IP address of the hostname. • FTP – The password to log into the server. The default password is anonymous. • FTPS – The password to log into the server.

Parameter	Data Type	Description
		Note: If no username and password are provided, this test verifies only the availability of the service in the server. • HTTP and HTTPS – The expected pattern in the retrieved HTML. • HTTP Slow and HTTPS Slow – The expected pattern in the retrieved HTML. • IMAP and POP3 – (Optional) The password to log into the server. **Note:** If no username and password are provided, this test verifies only the availability of the service in the server. • LDAP and LDAPS – (Optional) The password to log into the server. **Note:** If no username and password are provided, the LDAP/AD or SSL LDAP/AD test verifies the availability of the anonymous user. • MS SharePoint and MS SharePoint Secure – The pattern that is expected in the retrieved HTML. • RADIUS Auth – The username and password, separated by a backslash (\). For example: username\password • Simple HTTP and Simple HTTPS – The pattern expected in the retrieved HTML. • SMTP – (Optional) The pattern that is expected in the banner for the SMTP server. • SNMP – (Required if an OID is specified in the test target) A pattern to match in the response. • Specific HTTP Port – The pattern that is expected in the retrieved HTML.
headers	String	Additional headers. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
status_code	Integer	The expected status code. **Conditional:** Applicable for the HTTP, HTTPS, Simple HTTP, Simple HTTPS, HTTP Slow, HTTPS Slow, MS SharePoint, and MS SharePoint Secure tests.
delay	Integer	The interval between test start times, in seconds.

Request Examples

The following table provides example requests for retrieving monitors and configuring a monitor named Exchange-Owa-Monitor that is part of the MS-IIS App Pool Monitor group.

Method	Example Usage	Example Request

create	Create a monitor named Exchange-Owa-Monitor in the MS-IIS App Pool Monitor group.	`my $result = $API->create('monitors',`
		`{monitor_group_id => "MS-IIS App Pool Monitor"}`
		`{`
		`name =>"Exchange-Owa-Monitor",`
		`testing_method =>"ms_sharepoint_secure",`
		`ip_address =>"192.23.2.2",`
		`address_version => "ipv4",`
		`test_delay => 30,`
		`port => "80",`
		`username => "owa",`
		`password => "msft",`
		`target => "http://barracuda.com",`
		`match => "/barracuda",`
		`headers => "Header1:Value1",`
		`status_code => "200"`
		`}`
		`);`

list	List all monitors that are part of the M S-IIS App Pool Monitor group.	`my $result = $API->list('monitors',` `{monitor_group_id => "MS-IIS App Pool Monitor"});`

get	Retrieve data for Exchange-Owa-Monitor.	`my $result = $API->get('monitors',`
		`{`
		`monitor_group_id => "MS-IIS App Pool Monitor",`
		`id => "Exchange-Owa-Monitor"`
		`}`
		`);`

update	Update the test delay for Exchange-Owa-Monitor.	`my $result = $API->update('monitors',`
		`{`
		`monitor_group_id => "MS-IIS App Pool Monitor",`
		`id => "Exchange-Owa-Monitor"`
		`}`
		`{test_delay => 15}`
		`);`

remove	Remove Exchange-Owa-Monitor.	`my $result = $API->remove('monitors',`
		`{`
		`monitor_group_id => "MS-IIS App Pool Monitor",`
		`id => "Exchange-Owa-Monitor"`
		`}`
		`);`

Sample Scripts

In the API.zip file that you downloaded, change to the **Sample Scripts** folder. This folder contains scripts to access your Barracuda Load Balancer ADC over HTTP and HTTPS. Use these scripts to configure your services and servers.

HTTP Scripts

The following scripts are available in the **Sample Scripts/HTTP Scripts** folder:

Script	Function
Add_Service_and_Server_HTTP.pl	Create a virtual service and add a server to it.
Add_Content_Rule_HTTP.pl	Create a virtual service, add a content rule to the service, and then add a server to the content rule.
Add_Monitor_Group_HTTP.pl	Add a monitor group.
List_All_ADC_Config_HTTP.pl	Print out your Barracuda Load Balancer ADC configuration.
Update_Virtual_Service_HTTP.pl	Update a virtual service.
Update_Content_Rule_HTTP.pl	Update a content rule.

HTTPS Scripts

The following scripts are available in the **Sample Scripts/HTTPS Scripts** folder:

Script	Function
Add_Service_and_Server_HTTPS.pl	Create a virtual service and add a server to it.
Add_Content_Rule_HTTPS.pl	Create a virtual service, add a content rule to the service, and then add a server to the content rule.
Add_Monitor_Group_HTTPS.pl	Add a monitor group.
List_ADC_Config_HTTPS.pl	Print out your Barracuda Load Balancer ADC configuration.
Update_Virtual_Service_HTTPS.pl	Update a virtual service.
Update_Content_Rule_HTTPS.pl	Update a content rule.

Example Perl Code

```
use Barracuda::Rest::API;
use Data::Dumper;

# Initialize your environment
my $API = Barracuda::Rest::API->new('10.5.7.146', '8000', 'v2', 'http','no');

# Require user to login at initial step
$API->login("admin", "admin");
my $result;

# Creating a Resource
print "Creating Resources\n";

# Create a Virtual Service
print "Creating Virtual Service\n";
$result = $API->create('virtual_services',
      {virtual_service_group_id => 'default'},
      {
       name => "service1",
       ip_address => "192.168.17.125",
       port => "80",
       type => "http",
       address_version => "ipv4",
       netmask => "255.255.255.0",
       interface => "ge-1-1"
      }
);
print Dumper($result);

# Create a Server
print "Creating Server\n";
$result = $API->create('servers',
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1"
      },
      {
       name => "Server_12.0.0.128",
       address_version => "ipv4",
       ip_address => "12.0.0.128",
       port => 80,
       identifier => "ipaddr"
      }
);
print Dumper($result);


# Create a content rule
print "Creating Content Rule\n";
$result = $API->create('content_rules',
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1"
      },
      {
       name => "rule9000",
       host_match => "*.barracuda.com",
```

```
        url_match => "/*",
        extended_match => "*",
        extended_match_sequence => 5
       }
);
print Dumper($result);

# Create a monitor group
print "Creating Monitor Group\n";
$result = $API->create('monitor_groups', undef,
      {"name" => "MS_IIS App Pool Monitor"});
print Dumper($result);

# Add a monitor to the group print "Creating monitor\n";
$result = $API->create('monitors',
      {monitor_group_id => "MS_IIS App Pool Monitor"},
      {
       name => "Exchange-Owa-Monitor",
       type => "NTLMS_TEST",
       address_version => "ipv4",
       ip_address => "192.23.2.2",
       delay => 30,
       username => "owq",
       password => "msft",
       target => "http://barracuda.com",
       match => "/barracuda",
       headers => "",
       status_code => 200,
       port => "80"
      }
);
print Dumper($result);

# Listing a Resource
print "Listing Resources\n";

# Listing Virtual Services
print "Listing Virtual Services\n";
$result = $API->list('virtual_services',
      {virtual_service_group_id => 'default'});
print Dumper($result);

# Listing Servers
print "Listing Servers\n";
$result = $API->list('servers',
      {virtual_service_group_id => 'default', virtual_service_id =>
"service1"});
print Dumper($result);

# Listing Content Rules
print "Listing Content Rules\n";
$result = $API->list('content_rules',
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1"
      }
); print Dumper($result);

# Listing Monitor Groups
print "Listing Monitor Groups\n";
```

```
$result = $API->list('monitor_groups'); print Dumper($result);

# Listing Monitors
print "Listing Monitors\n";
$result = $API->list('monitors', { monitor_group_id => "MS_IIS App Pool
Monitor"});
print Dumper($result);

# Retrieving a Resource
print "Retrieving Resources\n";

# Retrieve a Virtual Service
print "Retrieving Virtual Service\n";
$result = $API->get('virtual_services',
      {
       virtual_service_group_id => 'default',
       id => "service1"}
);
print Dumper($result);

# Retrieve a Server
print "Retrieving Server\n";
$result = $API->get('servers',
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1",
       id => "Server_12.0.0.128"
      }
);
print Dumper($result);

# Retrieve a Content Rule
print "Retrieving Content Rule\n";
$result = $API->get('content_rules',
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1",
       id => "rule9000"
       }
);
print Dumper($result);

# Retrieve a Monitor Group
print "Retrieve Monitor Group\n";
$result = $API->get('monitor_groups'); print Dumper($result);

# Retrieve a Monitor
print "Retrieve Monitor\n";
$result = $API->get('monitors',
      {
       id => "Exchange-Owa-Monitor",
       monitor_group_id => "MS_IIS App Pool Monitor"
      }
);
print Dumper($result);

# Update Resource
print "Updating Resources\n";

# Update a Virtual Service
```

```
print "Updating Virtual Service\n";
$result = $API->update('virtual_services',
      {virtual_service_group_id => 'default', id => "service1"}, {enable =>
"true" } );
print Dumper($result);

# Update a Server
print "Updating Servers\n";
$result = $API->update('servers',
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1",
       id => "Server_12.0.0.128"
      },
      {status => "maintenance"}
);
print Dumper($result);

# Update a Content Rule
print "Updating Content Rule\n"
$result = $API->update('content_rules',
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1",
       id => "rule9000"
      },
      {host_match => "*.changeit.com"}
);
print Dumper($result);

# Update a Monitor
print "Updating Monitors\n";
$result = $API->update('monitors',
      {
       id => "Exchange-Owa- Monitor",
       monitor_group_id => "MS_IIS App Pool Monitor"
      },
      {ip_address => "192.1.1.1"}
);
print Dumper($result);

# Deleting Resources
print "Deleting Resources\n";

# Remove a Monitor
print "Removing Monitors\n";
$result = $API->remove('monitors',
      {monitor_group_id => "MS_IIS App Pool Monitor", id => "Exchange-Owa-
Monitor"}
);
print Dumper($result);

# Remove a Monitor Group
print "Removing Monitor Groups\n";
$result = $API->remove('monitor_groups', {id => "MS_IIS App Pool Monitor"});
print Dumper($result);

# Remove a Content Rule
print "Removing Content Rules\n";
$result = $API->remove('content_rules',
```

```
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1",
       id => "rule9000"}
);
print Dumper($result);

# Remove a Server
print "Removing Servers\n";
$result = $API->remove('servers',
      {
       virtual_service_group_id => 'default',
       virtual_service_id => "service1",
       id => "Server_12.0.0.128"
      }
);
print Dumper($result);

# Remove a Virtual Service
print "Removing Virtual Services";
$result = $API->remove('virtual_services',
      {virtual_service_group_id => 'default', id => "service1"}
);
print Dumper($result);

$API->logout();
```