## CONTENTS

### Preface

### Chapter 1  Welcome

Get Started .......................................................................................................................... 7  
Get API Notifications ........................................................................................................ 7  
WAF API Framework ....................................................................................................... 8  
  - Base URL to the Qualys API Server ........................................................................... 8  
  - XML Output and Schemas ......................................................................................... 9  
Introduction to WAF API Paradigm ................................................................................ 10  
Tracking API usage by user ............................................................................................. 19  
Know your Portal Version .............................................................................................. 20  

### Chapter 2  Web Applications API

Current web application count ...................................................................................... 23  
Get details on a web application ................................................................................... 24  
Search web applications ................................................................................................. 28  
Create web application ................................................................................................... 34  
Update web application .................................................................................................. 38  
Update web applications (bulk) .................................................................................... 43  
Delete web application ................................................................................................... 50  
Delete web applications (bulk) ...................................................................................... 51  
Reference: Web applications .......................................................................................... 53  

### Chapter 3  Web Servers API

Current web server count ............................................................................................... 58  
Get details on a web server ............................................................................................ 59  
Search web servers .......................................................................................................... 61  
Create web server ............................................................................................................ 65  
Update web server .......................................................................................................... 68  
Update web servers (bulk) ............................................................................................. 71  
Delete web server ............................................................................................................ 74  
Delete web servers (bulk) ............................................................................................... 75  
Reference: web servers .................................................................................................... 77  

### Chapter 4  Healthchecks API

Current healthcheck count ............................................................................................. 80  
Get details on a healthcheck ........................................................................................... 81  
Search healthchecks ........................................................................................................ 83  
Create healthcheck ......................................................................................................... 86  
Update healthcheck ......................................................................................................... 89  
Update healthchecks (bulk) ............................................................................................ 92
Contents

Delete healthcheck ................................................................. 96
Delete healthchecks (bulk) ................................................... 97
Reference: healthchecks ....................................................... 99

Chapter 5  SSL Certificates API
Current certificate count ...................................................... 102
Get details on a certificate .................................................. 103
Search certificates ............................................................. 106
Create certificate ............................................................... 110
Update certificate .............................................................. 116
Update certificates (bulk) .................................................. 119
Delete certificate .............................................................. 122
Delete certificates (bulk) .................................................. 123
Reference: certificates ....................................................... 125

Chapter 6  Custom Response Pages API
Current custom response page count .................................. 128
Get details on a custom response page ................................ 129
Search custom response pages .......................................... 132
Create custom response page ............................................. 135
Update custom response page .......................................... 138
Update custom response pages (bulk) .............................. 141
Delete custom response page ............................................ 144
Delete custom response pages (bulk) ............................... 145
Reference: custom response pages ................................. 147

Chapter 7  Security Policies API
Current security policy count ............................................. 150
Get details on a security policy ......................................... 151
Search security policies ................................................... 154
Create security policy ..................................................... 159
Update security policy ..................................................... 164
Update security policies (bulk) ........................................ 169
Delete security policy ..................................................... 175
Delete security policies (bulk) ........................................ 176
Reference: Security Policies ............................................. 178

Chapter 8  HTTP Profiles API
Current HTTP Profile count .............................................. 181
Get details on an HTTP Profile ........................................ 182
Search HTTP Profiles ..................................................... 186
Create HTTP Profile ....................................................... 190
Update HTTP Profile ..................................................... 196
Update HTTP Profiles (bulk) .......................................... 201
Delete HTTP Profile ..................................................... 206
Delete HTTP Profiles (bulk) .......................................................................................... 207
Reference: HTTP Profile .............................................................................................. 209

Chapter 9 Custom Rules API
Current Custom Rule count .......................................................................................... 214
Get details on a Custom Rule ...................................................................................... 215
Search Custom Rules .................................................................................................... 218
Create Custom Rule ..................................................................................................... 222
Update Custom Rule ..................................................................................................... 225
Update Custom Rules (bulk) ....................................................................................... 228
Delete Custom Rules .................................................................................................... 233
Delete Custom Rules (bulk) ......................................................................................... 234
Reference: Custom Rules .............................................................................................. 236

Chapter 10 Clusters API
Current cluster count .................................................................................................... 239
Get details on a cluster ............................................................................................... 240
Search clusters ............................................................................................................. 243
Create cluster ............................................................................................................... 248
Update cluster .............................................................................................................. 252
Update clusters (bulk) ................................................................................................. 256
Delete cluster ................................................................................................................ 260
Delete clusters (bulk) ................................................................................................... 261
Reference: Clusters ....................................................................................................... 263

Chapter 11 Appliance API
Current appliance count ............................................................................................... 268
Get details on appliance ............................................................................................. 269
Search appliances ......................................................................................................... 271
Delete appliance .......................................................................................................... 274
Reference: Appliances .................................................................................................. 275
Preface

Using the Qualys Web Application Firewall (WAF) API, third parties can integrate the Qualys Web Application Firewall solution into their own applications using an extensible XML interface. This user guide is intended for application developers who will use the Qualys WAF API.

About Qualys

Qualys, Inc. (NASDAQ: QLYS) is a pioneer and leading provider of cloud-based security and compliance solutions. The Qualys Cloud Platform and its integrated apps help businesses simplify security operations and lower the cost of compliance by delivering critical security intelligence on demand and automating the full spectrum of auditing, compliance and protection for IT systems and web applications.

Founded in 1999, Qualys has established strategic partnerships with leading managed service providers and consulting organizations including Accenture, BT, Cognizant Technology Solutions, Deutsche Telekom, Fujitsu, HCL, HP Enterprise, IBM, Infosys, NTT, Optiv, SecureWorks, Tata Communications, Verizon and Wipro. The company is also a founding member of the Cloud Security Alliance (CSA). For more information, please visit www.qualys.com.

Contact Qualys Support

Qualys is committed to providing you with the most thorough support. Through online documentation, telephone help, and direct email support, Qualys ensures that your questions will be answered in the fastest time possible. We support you 7 days a week, 24 hours a day. Access support information at www.qualys.com/support/.
Welcome

Welcome to Qualys Web Application Firewall API.

Get Started

*WAF API Framework* - Learn the basics about making API requests. The base URL depends on the platform where your Qualys account is located.

*XML Output and Schemas* - XML output uses schemas defined on your platform.

*Introduction to WAF API Paradigm* - We’ll tell you about making requests with authentication, making requests with payloads, using Curl, and truncation/pagination logic. API requests must authenticate using Qualys credentials.

*Authentication* - We’ll tell you about the method used for authentication. API requests must authenticate using Qualys credentials.

Get API Notifications

We recommend you join our Community and subscribe to our API notifications so you’ll get email notifications telling you about important upcoming API enhancements and changes.

From our Community

Join our Community

Subscribe to API Notifications (select Receive email notifications)
Chapter 1 — Welcome

WAF API Framework

The new Qualys Web Application Firewall (WAF) API framework introduces numerous innovations and new functionality compared to the other Qualys API frameworks.

Request URL

The URL for making API requests respects the following structure:

https://<baseurl>/qps/rest/2.0/<operation>/<module>/<object>/<object_id>

where the components are described below.

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;baseurl&gt;</td>
<td>The Qualys API server URL that you should use for API requests depends on the platform where your account is located. The base URL for Qualys US Platform 1 is: <a href="https://qualysapi.qualys.com">https://qualysapi.qualys.com</a></td>
</tr>
<tr>
<td>&lt;operation&gt;</td>
<td>The request operation, such as get a list and search.</td>
</tr>
<tr>
<td>&lt;module&gt;</td>
<td>The API module. For the WAF API, the module is: &quot;waf&quot;.</td>
</tr>
<tr>
<td>&lt;object&gt;</td>
<td>The module specific object.</td>
</tr>
<tr>
<td>&lt;object_id&gt;</td>
<td>(Optional) The module specific object ID, if appropriate.</td>
</tr>
</tbody>
</table>

Base URL to the Qualys API Server

The Qualys API documentation and sample code within it use the API server URL for Qualys US Platform 1: qualysapi.qualys.com.

The Qualys API server URL that you should use for API requests depends on the platform where your account is located.

<table>
<thead>
<tr>
<th>Account Location</th>
<th>API Server URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Qualys US Platform 1</td>
<td><a href="https://qualysapi.qualys.com">https://qualysapi.qualys.com</a></td>
</tr>
<tr>
<td>Qualys US Platform 2</td>
<td><a href="https://qualysapi.qg2.apps.qualys.com">https://qualysapi.qg2.apps.qualys.com</a></td>
</tr>
<tr>
<td>Qualys US Platform 3</td>
<td><a href="https://qualysapi.qg3.apps.qualys.com">https://qualysapi.qg3.apps.qualys.com</a></td>
</tr>
<tr>
<td>Qualys EU Platform 1</td>
<td><a href="https://qualysapi.qualys.eu">https://qualysapi.qualys.eu</a></td>
</tr>
<tr>
<td>Qualys EU Platform 2</td>
<td><a href="https://qualysapi.qg2.apps.qualys.eu">https://qualysapi.qg2.apps.qualys.eu</a></td>
</tr>
<tr>
<td>Qualys India Platform 1</td>
<td><a href="https://qualysapi.qg1.apps.qualys.in">https://qualysapi.qg1.apps.qualys.in</a></td>
</tr>
<tr>
<td>Qualys Private Cloud Platform</td>
<td><a href="https://qualysapi">https://qualysapi</a>.&lt;customer_base_url&gt;</td>
</tr>
</tbody>
</table>
XML Output and Schemas

<table>
<thead>
<tr>
<th>Category</th>
<th>XSD URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web Applications XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/webapp.xsd</td>
</tr>
<tr>
<td>Web Servers XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/webserver.xsd</td>
</tr>
<tr>
<td>Healthchecks XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/healthcheck.xsd</td>
</tr>
<tr>
<td>SSL Certificates XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/certificate.xsd</td>
</tr>
<tr>
<td>Custom Response Pages XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/custompage.xsd</td>
</tr>
<tr>
<td>Security Policies XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/securitypolicy.xsd</td>
</tr>
<tr>
<td>HTTP Profiles XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/httpprofile.xsd</td>
</tr>
<tr>
<td>Custom Rules XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/customrule.xsd</td>
</tr>
<tr>
<td>WAF Clusters XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/cluster.xsd</td>
</tr>
<tr>
<td>WAF Appliances XSD</td>
<td>https://&lt;baseurl&gt;/qps/xsd/2.0/waf/appliance.xsd</td>
</tr>
</tbody>
</table>

<baseurl> is the Qualys API server platform URL where your account is located. See Base URL to the Qualys API Server.
Introduction to WAF API Paradigm

Authentication

The application must authenticate using Qualys account credentials (user name and password) as part of the HTTP request. The credentials are transmitted using the "Basic Authentication Scheme" over HTTPS.

For more information, see the "Basic Authentication Scheme" section of RFC #2617:

http://www.faqs.org/rfcs/rfc2617.html

The exact method of implementing authentication will vary according to which programming language is used.

The allowed methods, POST and/or GET, for each API request are documented with each API call in this user guide.

Example

Basic authentication - recommended option:

curl -u "USERNAME:PASSWORD"
https://qualysapi.qualys.com/qps/rest/2.0/search/waf/webapp/

where qualysapi.qualys.com is the base URL to the Qualys API server where your account is located.

Making Requests with an XML Payload

While it is still possible to create simple API requests using the GET method, you can create API requests using the POST method with an XML payload to make an advanced request.

The XML payloads can be compared to a scripting language that allows user to make multiple actions within one single API request, like adding a parameter to an object and updating another parameter.

The XML structure of the payload is described in the XSD files.

Using Curl

Curl is a multi-platform command-line tool used to transfer data using multiple protocols. This tool is supported on many systems, including Windows, Unix, Linux and Mac. In this document Curl is used in the examples to build WAF API requests using the HTTP over SSL (https) protocol, which is required by the Qualys WAF API framework.

Want to learn more? Visit http://curl/haxx/se
The following Curl options are used according to different situations:

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-u &quot;LOGIN:PASSWORD&quot;</td>
<td>This option is used for basic authentication.</td>
</tr>
<tr>
<td>-X &quot;POST&quot;</td>
<td>This option is used to provide a method other than the default method, GET.</td>
</tr>
<tr>
<td>-H &quot;content-type&quot;</td>
<td>This option is used to provide a custom HTTP request header parameter for content type, to specify the MIME type of the curl's payload.</td>
</tr>
<tr>
<td>--data-binary</td>
<td>This option is used to specify the POST data. See the examples below.</td>
</tr>
</tbody>
</table>

The sample below shows a typical Curl request using options mentioned above and how they interact with each other. The option -X "POST" tells Curl to execute the request using the HTTP POST method. The option --data-binary @- tells Curl to read the POST data from its standard input (stdin). The string "< file.xml" is interpreted by the shell to redirect the content of the file to the stdin of the command. The option -H "content-type: text/xml" tells Curl the POST data in "file.xml" is XML in text format.

```bash
curl -H "content-type: text/xml" -X "POST" --data-binary @- "https://example.com" < file.xml
```

This documentation uses Curl examples showing the POST data in the "file.xml" file. This is referred to as Request POST Data. This can also be referred to as the Payload.

**XML Output Pagination / Truncation Logic**

The XML output of a search API request is paginated and the default page size is 100 object records. The page size can be customized to a value between 1 and 1,000. If the number of records is greater than the page size then the <ServiceResponse> element shows the response code SUCCESS with the element <hasMoreRecords>true</hasMoreRecords> as shown below.

Follow the process below to obtain the first two the XML pages for an API request. Please apply the same logic to get all the next (n+1) pages until all records are returned. This is indicated when <hasMoreRecords>false</hasMoreRecords>.

**Request 1:**

The service requests in the data file defines the search criteria for clusters and restricts the results to 3.

```bash
curl -u "USERNAME:PASSWORD" -X "POST" -H "Content-Type: text/xml" https://qualysapi.qualys.com/qps/rest/2.0/search/waf/cluster --data "<?xml version="1.0" encoding="UTF-8"?>
```
Chapter 1 — Welcome
Introduction to WAF API Paradigm

<ServiceRequest>
  <preferences>
    <startFromOffset>1</startFromOffset>
    <limitResults>3</limitResults>
  </preferences>
  <filters>
    <Criteria field="name" operator="NOT EQUALS">Demo cluster</Criteria>
  </filters>
</ServiceRequest>

Response:
The number of records is greater than the default pagination value (100) so the <ServiceResponse> element identifies the last ID of the object in the current page output.

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/cluster.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>3</count>
  <hasMoreRecords>true</hasMoreRecords>
  <lastId>25402</lastId>
  <data>
    <Cluster>
      <id>24401</id>
      <uuid>e6638fed-a2dd-43fb-87a0-b1fd5bbed00b</uuid>
      <name><![CDATA[Cluster]]></name>
      <errorResponse>
        <customPage>
          <id>1001</id>
          <uuid>9c7c6bfc-07e4-43db-2b5d124e25e7</uuid>
          <name><![CDATA[No site!!]]></name>
          <customPage>
        </errorResponse>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </owner>
  </data>
Chapter 1 — Welcome
Introduction to WAF API Paradigm

<created>2017-07-04T14:12:21Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-07-19T09:50:25Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<token><![CDATA[027AEC4A-5F83-4473-9D38-84A4D136F2F0]]></token>
<syncDate>2017-07-31T15:00:58Z</syncDate>
<status>ACTIVE</status>
<webApps>
  <WebApp>
    <id>63090275</id>
    <uuid>e662b1fc-bf1c-465e-b1c1-102ba3cfa71d</uuid>
    <name><![CDATA[http://dip01.p29.eng.sjc01.qualys.com]]></name>
  </WebApp>
</webApps>
<appliances>
  <Appliance>
    <id>17801</id>
    <uuid>c9281bfb-eabc-4162-b187-51afbf61a15</uuid>
    <name><![CDATA[C9281BFB-EABC-4162-B187-51AFBF61A15]]></name>
  </Appliance>
</appliances>
</Cluster>
<Cluster>
  <id>25401</id>
  <uuid>471fd090-429f-46b4-95b5-9b5fe24c7e24</uuid>
  <name><![CDATA[Europe]]></name>
  <errorResponse>
    <block/>
  </errorResponse>
</Cluster>

Qualys Web Application Firewall API 13
<owner>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
</owner>
<created>2017-08-01T11:03:23Z</created>
<createdBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
</createdBy>
<updated>2017-08-01T11:03:23Z</updated>
<updatedBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
</updatedBy>
<token><![CDATA[a154dd59-a889-4c0a-a0cd-79f503cf34c1]]></token>
<status>NO_SENSORS</status>
</Cluster>
<Cluster>
    <id>25402</id>
    <uuid>b7305e1d-9eaa-4e5f-a47b-07b4a30a8770</uuid>
    <name><![CDATA[South America]]></name>
    <errorResponse>
        <block/>
    </errorResponse>
    <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
    </owner>
    <created>2017-08-01T11:04:07Z</created>
    <createdBy>
        <id>3988443</id>
        <username>john_doe</username>
    </createdBy>
</Cluster>
Request 2:
To get the next page of results, you need to edit your service request in the data section that will be passed to API request as a POST payload. The next page of results are according to the <lastID> element returned in the first page.

curl -u "USERNAME:PASSWORD" -X "POST" -H "Content-Type: text/xml" https://qualysapi.qualys.com/qps/rest/2.0/search/waf/cluster --data "<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
<preferences>
  <startFromOffset>4</startFromOffset>
  <limitResults>10</limitResults>
</preferences>
<filters>
  <Criteria field="name" operator="NOT EQUALS">Demo cluster</Criteria>
</filters>
</ServiceRequest>"

Response:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd
Chapter 1 — Welcome
Introduction to WAF API Paradigm

```
<responseCode>SUCCESS</responseCode>
<count>3</count>
<hasMoreRecords>false</hasMoreRecords>
<data>
  <Cluster>
    <id>25403</id>
    <uuid>5e8f9a49-8b57-4f53-b7f0-fbfaf72c5dc6</uuid>
    <name><![CDATA[North America]]></name>
    <errorResponse>
      <block/>
    </errorResponse>
    <owner>
      <id>3988443</id>
      <username>john_doe</username>
      <firstname>John</firstname>
      <lastname>Doe</lastname>
    </owner>
    <created>2017-08-01T11:09:42Z</created>
    <createdBy>
      <id>3988443</id>
      <username>john_doe</username>
      <firstname>John</firstname>
      <lastname>Doe</lastname>
    </createdBy>
    <updated>2017-08-01T11:09:42Z</updated>
    <updatedBy>
      <id>3988443</id>
      <username>john_doe</username>
      <firstname>John</firstname>
      <lastname>Doe</lastname>
    </updatedBy>
    <token><![CDATA[200d69ea-04d5-4d7f-8d8d-727f7f6081bc]]></token>
    <status>NO_SENSORS</status>
  </Cluster>
  <Cluster>
    <id>25404</id>
    <uuid>9d01b87c-dc58-43cc-bf62-880749d42185</uuid>
    <name><![CDATA[Africa]]></name>
    <errorResponse>
```
Chapter 1 — Welcome
Introduction to WAF API Paradigm

<block/>
</errorResponse>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-08-01T11:05Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-08-01T11:10:05Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<token><![CDATA[de0ae6ca-6128-4c08-bc61-43e32329e1b]]></token>
<status>NO_SENSORS</status>
</Cluster>
<Cluster>
  <id>25405</id>
  <uuid>63873df-d9e3-4445-be9b-e13e23254bd0</uuid>
  <name><![CDATA[Asia]]></name>
  <errorResponse>
    <block/>
  </errorResponse>
  <owner>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </owner>
  <created>2017-08-01T11:10:17Z</created>
  <createdBy>
Setting the Custom Page Size

The service request needs to contain the <preferences> section with the <limitResults> parameter. For the <limitResults> parameter you can enter a value from 1 to 1,000.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
<preferences>
  <startFromOffset>4</startFromOffset>
  <limitResults>3</limitResults>
</preferences>
<filters>
  <Criteria field="name" operator="NOT EQUALS">Demo cluster</Criteria>
</filters>
</ServiceRequest>
```
Tracking API usage by user

You can track API usage per user without the need to provide user credentials such as the username and password. Contact Qualys Support to get the X-Powered-By HTTP header enabled. Once enabled, the X-Powered-By HTTP header is returned for each API request made by a user. The X-Powered-By value includes a unique ID generated for each subscription and a unique ID generated for each user. See sample headers below.

Click here to learn more.

Sample: Tracking API usage through the X-Powered-By HTTP header

```
229 HTTP/1.1 200 OK
X-Powered-By: Qualys:QAPOD4SJC:f972e2cc-69d6-7ebd-80e6-7b9a931475d8:06198167-43f3-7591-802a-1c400a0e81b1
Content-Type: application/xml
Transfer-Encoding: chunked
Date: Mon, 04 Dec 2017 05:36:29 GMT
Server: Apache
LBDEBUG:NS=10.44.1.12,SERVER=10.44.77.81:50205,CSW=cs-p04-qualysapi 443,VSERVER=vs-p04-papi-80,ACTIVE-SERVICES=2,HEALTH=100
```

Once X-Powered-By HTTP header is enabled, information is returned in the following format:

X-Powered-By Qualys:<POD_ID>:<SUB_UUID>:<USER_UUID>

Where,

- **POD_ID** is the shared POD or a PCP. Shared POD is USPOD1, USPOD2, etc.
- **SUB_UUID** is the unique ID generated for the subscription
- **USER_UUID** is the unique ID generated for the user

For example,

```
X-Powered-By: Qualys:QAPOD4SJC:f972e2cc-69d6-7ebd-80e6-7b9a931475d8:06198167-43f3-7591-802a-1c400a0e81b1
```

You can use the **USER_UUID** to track API usage per user.
Know your Portal Version

Using the Version API you can find out the installed version of Portal and its sub-modules that are available in your subscription.

URL: https://qualysapi.qualys.com/qps/rest/portal/version
Methods allowed: GET

Examples

Example 1: XML

API Request:

curl -u "USERNAME:PASSWORD" -X "GET" -H "Accept: application/xml"
https://qualysapi.qualys.com/qps/rest/portal/version

Response:

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="https://qualysapi.qualys.com/qps/xsd/version.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Portal-Version>
      <WAS-VERSION>6.0.0.0</WAS-VERSION>
      <FIM-VERSION>1.5.1</FIM-VERSION>
      <VM-VERSION>1.0.3</VM-VERSION>
      <CERTVIEW-VERSION>1.1.0.0</CERTVIEW-VERSION>
      <CM-VERSION>1.20.1</CM-VERSION>
      <MDS-VERSION>2.11.7.0</MDS-VERSION>
      <CA-VERSION>2.9.1.0</CA-VERSION>
      <IOC-VERSION>1.1.0</IOC-VERSION>
      <AV2-VERSION>0.1.0</AV2-VERSION>
      <QUESTIONNAIRE-VERSION>2.14.0.4</QUESTIONNAIRE-VERSION>
      <WAF-VERSION>2.7.0.0</WAF-VERSION>
    </Portal-Version>
  </data>
</ServiceResponse>
Example 2: JSON

API Request:

curl -u "USERNAME:PASSWORD" -X "GET" -H "Accept: application/json"
https://qualysapi.qualys.com/qps/rest/portal/version

Response:

```json
{
    "ServiceResponse": {
        "data": [
            {
                "Portal-Version": {
                    "WAS-VERSION": "6.0.0.0",
                    "VM-VERSION": "1.0.3",
                    "CM-VERSION": "1.20.1",
                    "MDS-VERSION": "2.11.7.0",
                    "CA-VERSION": "2.9.1.0",
                    "QUESTIONNAIRE-VERSION": "2.14.0.4",
                    "WAF-VERSION": "2.7.0.0"
                }
            }...
        ]
    },
    "responseCode": "SUCCESS",
    "count": 1
}
```
Web Applications API

Use these API functions to manage web applications in your WAF license.

- Current web application count
- Get details on a web application
- Search web applications
- Create web application
- Update web application
- Update web applications (bulk)
- Delete web application
- Delete web applications (bulk)

Reference: Web applications
Current web application count

Returns the total number of web applications licensed for WAF and in the user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/count/waf/webapp/

**Methods allowed:**
GET

**Input**
No input elements are available.

**Permissions**
User must have the WAF module enabled
User must have "API ACCESS" permission
Count includes web apps licensed for WAF and in the user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/webapp

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="https://qualysapi.qualys.com/qps/xsd/2.0/waf/webapp.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>72</count>
</ServiceResponse>
```
Get details on a web application

Returns details about a specific web application licensed for WAF, within the user’s scope. Want to find a web application ID to use as input? See Search web applications.

URL: https://<baseurl>/qps/rest/2.0/get/waf/webapp/<id>
Methods allowed: GET

Input

The element "id" (Integer) is required, where "id" identifies the web application ID of interest.

Permissions

User must have WAF module enabled
User must have "API ACCESS" permission
Output includes web app licensed for WAF and within the user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/webapp/63098273

Response:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webapp.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebApp>
      <id>63098273</id>
      <uuid>01bd1b58-2802-48dd-b5b5-ea1342aea21a</uuid>
      <name>
        <![CDATA[Site 01]]>
      </name>
      <owner>
        <id>3988443</id>
      </owner>
    </WebApp>
  </data>
</ServiceResponse>
<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</owner>
<created>2017-05-31T09:49Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-05-31T09:39Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
$url>https://site01.xfuentes-docker</url>
(urls/)
<webServer>
  <id>1001</id>
  <uuid>315cc797-3c73-4721-ba42-263e7e7b6cbb</uuid>
  <name>
    <![CDATA[First Pool]]>
  </name>
</webServer>
<webServerTimeout>60</webServerTimeout>
<persistencyEnabled>false</persistencyEnabled>
<healthcheck>
  <id>1001</id>
  <uuid>f479e6f5-57a1-4677-a8cb-272e2c69623a</uuid>
  <name>
    <![CDATA[Standard Healthcheck]]>
  </name>
</healthcheck>
<failureResponseCode>503</failureResponseCode>
<certificate>
  <id>1</id>
  <uuid>a21b4a1b-de54-45e8-9d29-204444ce5bb</uuid>
  <name>
Chapter 2 — Web Applications API
Get details on a web application

<!CDATA[Site01 Cert]>
</name>
</certificate>
<sslProtocols>
<!CDATA[SSLV3,TLS10,TLS11,TLS12]>
</sslProtocols>
<sslCiphers>
</sslCiphers>
<blockingMode>false</blockingMode>
<customPage>
'id>1001'</id>
<uuid>0dba4434-1118-40e5-8768-23c5616053d5</uuid>
</name>
<!CDATA[My Response]>
</name>
</customPage>
<securityPolicy>
'id>30682'</id>
<uuid>6c56416a-66ff-4016-b16f-da2cec2e97f3</uuid>
</name>
<!CDATA[Standard Policy]>
</name>
</securityPolicy>
</httpProfile>
'id>1001'</id>
<uuid>341bcf25-c9fa-45ff-ac63-728e38056443</uuid>
</name>
<!CDATA[Standard Protocol]>
</name>
</httpProfile>
<scanTrustEnabled>true</scanTrustEnabled>
<scanTrustToken>
<!CDATA[38770c30-7c79-4b75-a5ec-43d07493eca1]>
</scanTrustToken>
<customRules>
<CustomRule>
'id>1001'</id>
<uuid>20e220d3-1244-42ca-a473-c80469e95bc0</uuid>
</uuid>
<name>
  <![CDATA[Test custom rule]]>
</name>
</CustomRule>
<CustomRule>
  <id>2001</id>
  <uuid>c64c3008-claf-4969-8290-d0b1d8e9f27b</uuid>
  <name>
    <![CDATA[shamzor]]>
  </name>
</CustomRule>
</customRules>
<clusters>
  <Cluster>
    <id>24401</id>
    <uuid>48ae444d-e652-443f-8438-3a9182403b9f</uuid>
    <name>
      <![CDATA[Cluster 1]]>
    </name>
  </Cluster>
</clusters>
<status>DOWN</status>
<sslEnabled>true</sslEnabled>
<sslStatus>OK</sslStatus>
<deploymentStatus>FAILURE</deploymentStatus>
<deployed>2017-05-31T12:15:14Z</deployed>
</WebApp>
</data>
</ServiceResponse>
Chapter 2 — Web Applications API

Search web applications

Finds web applications in the user’s account matching the search criteria.

**URL:** https://<baseurl>/qps/rest/2.0/search/waf/webapp/

**Methods allowed:** POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: Web applications for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Long)</td>
<td>webServer.uuid (UUID)</td>
</tr>
<tr>
<td>uuid (UUID)</td>
<td>webServer.name (Text)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>webServerTimeout (Long)</td>
</tr>
<tr>
<td>url (Text)</td>
<td>persistencyEnabled (Boolean)</td>
</tr>
<tr>
<td>tags.tag.id (Long)</td>
<td>scanTrustEnabled (Boolean)</td>
</tr>
<tr>
<td>tags.tag.name (Text)</td>
<td>certificate.id (Long)</td>
</tr>
<tr>
<td>owner.id (Long)</td>
<td>certificate.uuid (UUID)</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>certificate.name (Text)</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>status</td>
</tr>
<tr>
<td>owner.lastname (Text)</td>
<td>deploymentStatus</td>
</tr>
<tr>
<td>created (Date)</td>
<td>deployed (Date)</td>
</tr>
<tr>
<td>createdBy.id (Long)</td>
<td>synced (Date)</td>
</tr>
<tr>
<td>createdBy.username (Text)</td>
<td>blockingMode (Boolean)</td>
</tr>
<tr>
<td>createdBy.firstname (Text)</td>
<td>customPage.id (Long)</td>
</tr>
<tr>
<td>createdBy.lastname (Text)</td>
<td>customPage.uuid (UUID)</td>
</tr>
<tr>
<td>updated (Date)</td>
<td>customPage.name (Text)</td>
</tr>
<tr>
<td>updatedBy.id (Long)</td>
<td>securityPolicy.id (Long)</td>
</tr>
<tr>
<td>updatedBy.username (Text)</td>
<td>securityPolicy.uuid (UUID)</td>
</tr>
<tr>
<td>updatedBy.firstname (Text)</td>
<td>securityPolicy.name (Text)</td>
</tr>
<tr>
<td>updatedBy.lastname (Text)</td>
<td>httpProfile.id (Long)</td>
</tr>
<tr>
<td>urls.value (Text)</td>
<td>httpProfile.uuid (UUID)</td>
</tr>
</tbody>
</table>
Web applications API only provides the id, uuid, and name of a cluster. Use the Clusters API to get more details of a cluster and its appliances.

### Permissions

- User must have the WAF module enabled
- User must have "API ACCESS" permission
- Output includes web apps licensed for WAF and within the user's scope

### Example

**Request:**
```
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/search/waf/webapp" < file.xml
```

Note: "file.xml" contains the request POST data.

The request POST data is optional. If you leave it empty all web applications in the user’s scope are returned.

**Request POST Data:**
```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
```
Chapter 2 — Web Applications API

Search web applications

Response:

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <hasMoreRecords>false</hasMoreRecords>
  <data>
    <WebApp>
      <id>63098273</id>
      <uuid>01bd1b58-2802-48dd-b5b5-ea1342aea21a</uuid>
      <name><![CDATA[Site 01]]></name>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </owner>
      <created>2017-05-31T09:01:49Z</created>
      <createdBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </createdBy>
      <updated>2017-05-31T09:19:39Z</updated>
      <updatedBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </updatedBy>
    </WebApp>
  </data>
</ServiceResponse>
```
<url>https://site01.xfuentes-docker</url>
<urls/>
<webServer>
  <id>1001</id>
  <uuid>315cc797-3c73-4721-ba42-263e7e7b6cbb</uuid>
  <name>
    <![CDATA[First Pool]]>
  </name>
</webServer>
<webServerTimeout>60</webServerTimeout>
<persistencyEnabled>false</persistencyEnabled>
<healthcheck>
  <id>1001</id>
  <uuid>f479e6f5-57a1-4677-a8cb-272e2c69623a</uuid>
  <name>
    <![CDATA[Standard Healthcheck]]>
  </name>
</healthcheck>
<failureResponseCode>503</failureResponseCode>
<certificate>
  <id>1</id>
  <uuid>a21b4a1b-de54-45e8-9d29-204444ce5bb</uuid>
  <name>
    <![CDATA[Site01 Cert]]>
  </name>
</certificate>
<sslProtocols>
  <![CDATA[SSLv3,TLS10,TLS11,TLS12]]>
</sslProtocols>
<sslCiphers>
</sslCiphers>
<blockingMode>false</blockingMode>
<customPage>
  <id>1001</id>
  <uuid>0dba4434-1118-40e5-8768-23c5616053d5</uuid>
  <name>
    <![CDATA[My Response]]>
  </name>
</customPage>
<securityPolicy>
  <id>30682</id>
  <uuid>6c56416a-66ff-4016-b16f-da2cec2e97f3</uuid>
  <name><![CDATA[Standard Policy]]></name>
</securityPolicy>

<httpProfile>
  <id>1001</id>
  <uuid>341bcf25-c9fa-45ff-ac63-728e38056443</uuid>
  <name><![CDATA[Standard Protocol]]></name>
</httpProfile>

<scanTrustEnabled>false</scanTrustEnabled>

<clusters>
  <customRules>
    <CustomRule>
      <id>1001</id>
      <uuid>20e220d3-1244-42ca-a473-c80469e95bc0</uuid>
      <name><![CDATA[Test custom rule]]></name>
    </CustomRule>
    <CustomRule>
      <id>2001</id>
      <uuid>c64c3008-claf-4969-8290-d0b1d8e9f27b</uuid>
      <name><![CDATA[shamzor]]></name>
    </CustomRule>
  </customRules>
</clusters>

<clusters>
  <Cluster>
    <id>24401</id>
    <uuid>48ae444d-e652-443f-8438-3a9182403b9f</uuid>
    <name><![CDATA[Cluster 1]]></name>
  </Cluster>
</clusters>
</name>
</Cluster>
</clusters>
<status>DOWN</status>
<sslEnabled>true</sslEnabled>
<sslStatus>OK</sslStatus>
<deploymentStatus>FAILURE</deploymentStatus>
<deployed>2017-05-31T12:15:14Z</deployed>
</WebApp>
</data>
</ServiceResponse>
Create web application

Create a web application asset that you want to monitor using WAF.

URL:  https://<baseurl>/qps/rest/2.0/create/waf/webapp

Methods allowed:  POST

Input

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: Web applications for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>persistencyEnabled (Boolean)</td>
</tr>
<tr>
<td>url (Text)</td>
<td>persistencyToken</td>
</tr>
<tr>
<td>webServer.id (Long)</td>
<td>healthcheck.id (Long)</td>
</tr>
<tr>
<td>securityPolicy.id (Long)</td>
<td>failureResponseCode (Long)</td>
</tr>
<tr>
<td>httpProfile.id (Long)</td>
<td>certificate.id (Long)</td>
</tr>
<tr>
<td></td>
<td>sslProtocols (Text)</td>
</tr>
<tr>
<td></td>
<td>sslCiphers (Text)</td>
</tr>
<tr>
<td></td>
<td>blockingMode (Boolean)</td>
</tr>
<tr>
<td></td>
<td>customPage.id (Long)</td>
</tr>
<tr>
<td></td>
<td>scanTrustEnabled (Boolean)</td>
</tr>
<tr>
<td></td>
<td>customRules.CustomRule.id (Long)</td>
</tr>
<tr>
<td></td>
<td>clusters.cluster.id (Long)</td>
</tr>
<tr>
<td></td>
<td>lastComment (Text)</td>
</tr>
<tr>
<td></td>
<td>urls</td>
</tr>
<tr>
<td></td>
<td>urls.string (Text)</td>
</tr>
<tr>
<td></td>
<td>tags</td>
</tr>
<tr>
<td></td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td></td>
<td>tags.tag.name (Text)</td>
</tr>
<tr>
<td></td>
<td>webServerTimeout (Long)</td>
</tr>
</tbody>
</table>
Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Create WAF Asset" permission

Example

Request:
```bash
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/create/waf/webapp" <
file.xml
```

Note: "file.xml" contains the request POST data.

Request POST Data:
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <WebApp>
      <name>Site created by API</name>
      <url>http://site01.xfuentes-docker</url>
      <webServer><id>1001</id></webServer>
      <webServerTimeout>90</webServerTimeout>
      <securityPolicy><id>30682</id></securityPolicy>
      <httpProfile><id>1001</id></httpProfile>
    </WebApp>
  </data>
</ServiceRequest>
```

Response:
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webapp.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebApp>
      <id>63098473</id>
      <uuid>6ecd55d8-5431-4114-ba11-90b020576f37</uuid>
    </WebApp>
  </data>
</ServiceResponse>
```
<name>
  <![CDATA[Site created by API]]>
</name>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-06-01T09:22:47Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-06-01T09:22:47Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<url>http://site01.xfuentes-docker</url>
<webServer>
  <id>1001</id>
  <uuid>315cc797-3c73-4721-ba42-263e7e7b6cbb</uuid>
  <name>
    <![CDATA[First Pool]]>
  </name>
</webServer>
<webServerTimeout>90</webServerTimeout>
<persistencyEnabled>false</persistencyEnabled>
<blockingMode>false</blockingMode>
<securityPolicy>
  <id>30682</id>
  <uuid>6c56416a-66ff-4016-b16f-da2cec2e97f3</uuid>
  <name>
    <![CDATA[Standard Policy]]>
  </name>
</securityPolicy>
</securityPolicy>
<httpProfile>
  <id>1001</id>
  <uuid>341bcf25-c9fa-45ff-ac63-728e38056443</uuid>
  <name>
    <![CDATA[Standard Protocol]]>
  </name>
  <scanTrustEnabled>false</scanTrustEnabled>
  <status>DOWN</status>
  <sslEnabled>false</sslEnabled>
  <deploymentStatus>PENDING_DEPLOY</deploymentStatus>
  <deployed>2017-06-01T09:22:47Z</deployed>
</WebApp>
</data>
</ServiceResponse>
Update web application

Update a web application asset in the user’s account. You can update all fields except tag ID and tag name.

**URL:** https://<baseurl>/qps/rest/2.0/update/waf/webapp/<id>

**Methods allowed:** POST

**Input**

The "id" (Long) element is required. This identifies the web application you want to update.

Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: Web applications for descriptions of these elements.

<table>
<thead>
<tr>
<th>name (Text)</th>
<th>sslProtocols (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>url (Text)</td>
<td>sslCiphers (Text)</td>
</tr>
<tr>
<td>webServer.id (Long)</td>
<td>blockingMode (Boolean)</td>
</tr>
<tr>
<td>webServerTimeout (Long)</td>
<td>customPage.id (Long)</td>
</tr>
<tr>
<td>securityPolicy.id (Long)</td>
<td>scanTrustEnabled (Boolean)</td>
</tr>
<tr>
<td>httpProfile.id (Long)</td>
<td>customRules.CustomRule.id (Long)</td>
</tr>
<tr>
<td>persistencyEnabled (Boolean)</td>
<td>clusters.cluster.id (Long)</td>
</tr>
<tr>
<td>persistencyToken</td>
<td>lastComment (Text)</td>
</tr>
<tr>
<td>healthcheck.id (Long)</td>
<td>urls</td>
</tr>
<tr>
<td>failureResponseCode (Long)</td>
<td>urls.string (Text)</td>
</tr>
<tr>
<td>certificate.id (Long)</td>
<td>tags</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Update WAF Asset" permission
Web application must be licensed in user’s subscription
Web application must be within the user’s scope
Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/webapp/63098473" < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <WebApp>
      <url>https://site02.xfuentes-docker</url>
      <certificate><id>1001</id></certificate>
      <sslProtocols>TLS12</sslProtocols>
      <blockingMode>true</blockingMode>

      <customRules><CustomRule><id>2001</id></CustomRule></customRules>
    </WebApp>
  </data>
</ServiceRequest>

Response:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webapp.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebApp>
      <id>63098473</id>
      <uuid>6ecd55d8-5431-4114-ba11-90b020576f37</uuid>
      <name><![CDATA[Site created by API]]></name>
      <owner>
        <id>3988443</id>
      </owner>
    </WebApp>
  </data>
</ServiceResponse>
<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</owner>
<created>2017-06-01T09:22:47Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-06-02T12:23:32Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
</updatedBy>
<url>https://site02.xfuentes-docker</url>
<html>
  <webServer>
    <id>1001</id>
    <uuid>315cc797-3c73-4721-ba42-263e7e7b6cbb</uuid>
    <name>
      <![CDATA[First Pool]]>
    </name>
  </webServer>
</html>
<webServerTimeout>60</webServerTimeout>
<persistencyEnabled>true</persistencyEnabled>
<persistencyToken>
  <![CDATA[ptoken]]>
</persistencyToken>
<healthcheck>
  <id>1001</id>
  <uuid>f479e6f5-57a1-4677-a8cb-272e2c69623a</uuid>
  <name>
    <![CDATA[Standard Healthcheck]]>
  </name>
</healthcheck>
<certificate>
  <id>1001</id>
  <uuid>5788c0eb-5bda-466f-bfb6-71a1f60856ff</uuid>
</certificate>
<name>
  <![CDATA[Site02 Cert]]>
</name>
</certificate>
<sslProtocols>
  <![CDATA[TLS12]]>
</sslProtocols>
<sslCiphers>
</sslCiphers>
<blockingMode>true</blockingMode>
<securityPolicy>
  <id>30682</id>
  <uuid>6c56416a-66ff-4016-b16f-da2cec2e97f3</uuid>
  <name>
    <![CDATA[Standard Policy]]>
  </name>
</securityPolicy>
<httpProfile>
  <id>1001</id>
  <uuid>341bcf25-c9fa-45ff-ac63-728e38056443</uuid>
  <name>
    <![CDATA[Standard Protocol]]>
  </name>
</httpProfile>
<scanTrustEnabled>true</scanTrustEnabled>
<scanTrustToken>
  <![CDATA[38770c30-7c79-4b75-a5ec-43d07493eca1]]>
</scanTrustToken>
<customRules>
  <CustomRule>
    <id>2001</id>
    <uuid>c64c3008-claf-4969-8290-d0b1d8e9f27b</uuid>
    <name>
      <![CDATA[shamzor]]>
    </name>
  </CustomRule>
</customRules>
<status>INACTIVE</status>
Update web applications (bulk)

Update multiple web application assets in the user’s account. You can update all fields except tag ID and tag name.

**URL:** https://<baseurl>/qps/rest/2.0/update/waf/webapp

**Methods allowed:** POST

**Input**

All elements for the search operation are supported. See [Search web applications](#).

Allowed input elements for bulk update are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. See [Reference: Web applications](#) for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>sslProtocols (Text)</td>
</tr>
<tr>
<td>url (Text)</td>
<td>sslCiphers (Text)</td>
</tr>
<tr>
<td>webServer.id (Long)</td>
<td>blockingMode (Boolean)</td>
</tr>
<tr>
<td>webServerTimeout (Long)</td>
<td>customPage.id (Long)</td>
</tr>
<tr>
<td>securityPolicy.id (Long)</td>
<td>scanTrustEnabled (Boolean)</td>
</tr>
<tr>
<td>httpProfile.id (Long)</td>
<td>customRules.CustomRule.id (Long)</td>
</tr>
<tr>
<td>persistencyEnabled (Boolean)</td>
<td>clusters.cluster.id (Long)</td>
</tr>
<tr>
<td>persistencyToken</td>
<td>lastComment (Text)</td>
</tr>
<tr>
<td>healthcheck.id (Long)</td>
<td>urls</td>
</tr>
<tr>
<td>failureResponseCode (Long)</td>
<td>urls.string (Text)</td>
</tr>
<tr>
<td>certificate.id (Long)</td>
<td>tags</td>
</tr>
</tbody>
</table>

**Allowed Operators**

<table>
<thead>
<tr>
<th>Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>UUID</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Text</td>
<td>CONTAINS, EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Date</td>
<td>EQUALS, NOT EQUALS, GREATER, LESSER</td>
</tr>
<tr>
<td>Boolean</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
</tbody>
</table>
Chapter 2 — Web Applications API
Update web applications (bulk)

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Update WAF Asset" permission
Web application must be licensed in user’s subscription
Web application must be within the user’s scope

Example

Request:

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/webapp" < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="sslEnabled" operator="EQUALS">true</Criteria>
  </filters>
  <data>
    <WebApp>
      <sslProtocols>TLS12</sslProtocols>
    </WebApp>
    <customRules><CustomRule><id>2001</id></CustomRule></customRules>
  </data>
</ServiceRequest>
```

Response:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webapp.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>2</count>
</data>
```

Qualys Web Application Firewall API
<WebApp>
  <id>63098273</id>
  <uuid>01bd1b58-2802-48dd-b5b5-ea1342aea21a</uuid>
  <name><![CDATA[Site 01]]></name>
  <owner>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </owner>
  <created>2017-05-31T09:01:49Z</created>
  <createdBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </createdBy>
  <updated>2017-06-02T13:23:43Z</updated>
  <updatedBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </updatedBy>
  <url>https://site01.xfuentes-docker</url>
  <webServer>
    <id>1001</id>
    <uuid>315cc797-3c73-4721-ba42-263e7e7b6cbb</uuid>
    <name><![CDATA[First Pool]]></name>
  </webServer>
  <webServerTimeout>60</webServerTimeout>
  <persistencyEnabled>false</persistencyEnabled>
  <healthcheck>
    <id>1001</id>
    <uuid>f479e6f5-57a1-4677-a8cb-272e2c69623a</uuid>
    <name><![CDATA[Standard Healthcheck]]></name>
  </healthcheck>
</WebApp>
<scanTrustToken>
  <![CDATA[38770c30-7c79-4b75-a5ec-43d07493eca1]]>
</scanTrustToken>
<customRules>
  <CustomRule>
    <id>2001</id>
    <uuid>c64c3008-claf-4969-8290-d0b1d8e9f27b</uuid>
    <name>
      <![CDATA[shamzor]]>
    </name>
  </CustomRule>
</customRules>
<clusters>
  <Cluster>
    <id>24401</id>
    <uuid>48ae444d-e652-443f-8438-3a9182403b9f</uuid>
    <name>
      <![CDATA[Cluster 1]]>
    </name>
  </Cluster>
</clusters>
<status>DOWN</status>
<sslEnabled>true</sslEnabled>
<sslStatus>OK</sslStatus>
<deploymentStatus>PENDING_DEPLOY</deploymentStatus>
<deployed>2017-06-02T16:10:06Z</deployed>
</WebApp>
<WebApp>
  <id>63098473</id>
  <uuid>6ecd55d8-5431-4114-ba11-90b020576f37</uuid>
  <name>
    <![CDATA[Site created by API]]>
  </name>
  <owner>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </owner>
</WebApp>
Chapter 2 — Web Applications API

Update web applications (bulk)

<created>2017-06-01T09:22:47Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-06-02T13:23:46Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
=url=https://site02.xfuentes-docker</url>
<webServer>
  <id>1001</id>
  <uuid>315cc797-3c73-4721-ba42-263e7e7b6cbb</uuid>
  <name>
    <![CDATA[First Pool]]>
  </name>
</webServer>
<persistencyEnabled>false</persistencyEnabled>
<healthcheck>
  <id>1001</id>
  <uuid>f479e6f5-57a1-4677-a8cb-272e2c69623a</uuid>
  <name>
    <![CDATA[Standard Healthcheck]]>
  </name>
</healthcheck>
<certificate>
  <id>1001</id>
  <uuid>5788c0eb-5bda-466f-bfb6-71a1f60856ff</uuid>
  <name>
    <![CDATA[Site02 Cert]]>
  </name>
</certificate>
<sslProtocols>
  <![CDATA[TLS12]]>
</sslProtocols>
<sslCiphers>
</sslCiphers>
<blockingMode>true</blockingMode>
<securityPolicy>
  <id>30682</id>
  <uuid>6c56416a-66ff-4016-b16f-da2cec2e97f3</uuid>
  <name>
    <![CDATA[Standard Policy]]>
  </name>
</securityPolicy>
<httpProfile>
  <id>1001</id>
  <uuid>341bcf25-c9fa-45ff-ac63-728e38056443</uuid>
  <name>
    <![CDATA[Standard Protocol]]>
  </name>
</httpProfile>
<scanTrustEnabled>false</scanTrustEnabled>
<customRules>
  <CustomRule>
    <id>2001</id>
    <uuid>c64c3008-c1af-4969-8290-d0b1d8e9f27b</uuid>
    <name>
      <![CDATA[shamzor]]>
    </name>
  </CustomRule>
</customRules>
<status>INACTIVE</status>
<sslEnabled>true</sslEnabled>
<sslStatus>OK</sslStatus>
<deploymentStatus>PENDING DEPLOY</deploymentStatus>
<deployed>2017-06-01T16:05:54Z</deployed>
</WebApp>
</data>
</ServiceResponse>
Delete web application

Delete a web application configuration in user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/webapp/<id>

**Methods allowed:**
POST

**Input**

The "id" (Long) element is required. This identifies the web application asset you want to delete.

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Delete WAF Asset" permission
Web application must be licensed in user’s subscription
Web application must be within the user's scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "POST" -H "Content-Type: text/xml" https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/webapp/5739473

**Response:**
<?xml version="1.0" encoding="UTF-8"?>
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebApp>
      <id>5739473</id>
    </WebApp>
  </data>
</ServiceResponse>
Delete web applications (bulk)

Delete multiple web application assets in the user’s account.

URL: https://<baseurl>/qps/rest/2.0/delete/waf/webapp

Methods allowed: POST

Input

All elements for the search operation are supported. See Search web applications.

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Delete WAF Asset" permission
Web application must be licensed in user’s subscription
Web application must be within the user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/webapp" < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" ?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">created by API</Criteria>
  </filters>
</ServiceRequest>

Response:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
Delete web applications (bulk)

```xml
<xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd
/2.0/waf/webapp.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebApp>
      <id>63098473</id>
    </WebApp>
  </data>
</ServiceResponse>
```
## Reference: Web applications

A reference of all web application elements is provided below.

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Long)</td>
<td>Web application identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid (UUID)</td>
<td>Web application identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name (Text)</td>
<td>The name of the Web application as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>url (Text)</td>
<td>The incoming URL used. Any requests received that match that URL should be routed to this web application. Valid action: Update</td>
</tr>
<tr>
<td>tags.tag.id (Long)</td>
<td>The identifier of one tag associated with this WebApp asset.</td>
</tr>
<tr>
<td>tags.tag.name (Text)</td>
<td>The name of one tag associated with this WebApp asset.</td>
</tr>
<tr>
<td>owner.id (Long)</td>
<td>The user ID of the Web application owner.</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>The user name of the Web application owner.</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>The first name of the Web application owner.</td>
</tr>
<tr>
<td>owner.lastname (Text)</td>
<td>The last name of the Web application owner.</td>
</tr>
<tr>
<td>created (Date)</td>
<td>The date/time when the Web application was created.</td>
</tr>
<tr>
<td>createdBy.id (Long)</td>
<td>The user ID who created the Web application.</td>
</tr>
<tr>
<td>createdBy.username (Text)</td>
<td>The user name who created the Web application.</td>
</tr>
<tr>
<td>createdBy.firstname (Text)</td>
<td>The first name of the user who created the Web application.</td>
</tr>
<tr>
<td>createdBy.lastname (Text)</td>
<td>The last name of the user who created the Web application.</td>
</tr>
<tr>
<td>updated (Date)</td>
<td>The date/time when the Web application was last updated.</td>
</tr>
<tr>
<td>updatedBy.id (Long)</td>
<td>The user ID who last updated the Web application.</td>
</tr>
<tr>
<td>updatedBy.username (Text)</td>
<td>The user name who last updated the Web application.</td>
</tr>
<tr>
<td>updatedBy.firstname (Text)</td>
<td>The first name of the user who updated the Web application.</td>
</tr>
<tr>
<td>updatedBy.lastname (Text)</td>
<td>The last name of the user who updated the Web application.</td>
</tr>
<tr>
<td><strong>Element (Data Type)</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>-------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>urls</td>
<td>The list of optional (aliases) URL for this web application (incoming URL)</td>
</tr>
<tr>
<td>urls.string</td>
<td>(Text) One aliases URL for this web application (incoming URL)</td>
</tr>
<tr>
<td>webServer.id</td>
<td>(Long) The web server pool where received requests should be routed to.</td>
</tr>
<tr>
<td>webServer.uuid</td>
<td>(UUID) The UUID of the web server assigned to the web application.</td>
</tr>
<tr>
<td>webServer.name</td>
<td>(Text) The name of the web server pool assigned to the web application.</td>
</tr>
<tr>
<td>webServerTimeout</td>
<td>(Long) Server Timeout is the maximum time to wait for an HTTP connection attempt to a server to succeed. If the HTTP request does not respond before the duration set, it will timeout and return an HTTP 503 error code. Specify a timeout period between 1 second to 3600 seconds. Default value is 60 seconds.</td>
</tr>
<tr>
<td>persistencyEnabled</td>
<td>(Boolean) Persistency allows the client to reconnect to the same server previously visited for the web application. This bypasses load balancing. Specify the cookie name to persist connection to the server previously visited by the client.</td>
</tr>
<tr>
<td>persistencyToken</td>
<td>(Text) The cookie name used to maintain sessions on the WebServer, if persistencyEnabled is true.</td>
</tr>
<tr>
<td>scanTrustEnabled</td>
<td>(Boolean) Enable scanTrust (Authenticated Scanning) for integration with Qualys WAS for vulnerability scanning. You must get this feature enabled in subscription before you can use it.</td>
</tr>
<tr>
<td>healthcheck.id</td>
<td>(Long) The ID of the healthcheck assigned to the web application.</td>
</tr>
<tr>
<td>healthcheck.uuid</td>
<td>(UUID) The UUID of the healthcheck assigned to the web application.</td>
</tr>
<tr>
<td>healthcheck.name</td>
<td>(Text) The name of the healthcheck assigned to the web application.</td>
</tr>
<tr>
<td>failureResponseCode</td>
<td>(Long) Specify the response code returned when all Web servers in the server pool are down. The default value is 503. For example, a 503 page is displayed when the Web servers are down or the Web site is not reachable.</td>
</tr>
<tr>
<td>certificate.id</td>
<td>(Long) The ID of the SSL certificate assigned to the web application.</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>certificate.uuid</td>
<td>(UUID) The UUID of the SSL certificate assigned to the web application.</td>
</tr>
<tr>
<td>certificate.name</td>
<td>(Text) The name of the SSL certificate assigned to the web application.</td>
</tr>
</tbody>
</table>
| status                      | (Text) Status information:  
|                             | **Up** - All servers assigned to the Web application are up and running.                                                                                                                                     |
|                             | **Down** - All servers assigned to the Web application are down.                                                                                                                                           |
|                             | **Degraded** - Atleast one server assigned to the Web application is down.                                                                                                                                   |
|                             | **Unused** - Web application is not deployed on any WAF cluster, or the Web application is deployed on a cluster having no appliances registered to it.                                                        |
| deploymentStatus            | (Text) The status of Web Application and other configurations being pushed to the cluster:  
|                             | SUCCESS (Web application successfully deployed)  
|                             | PENDING_DEPLOY (Deployment of a Web application is pending)  
|                             | FAILURE (Deployment of Web application configuration has failed on all associated clusters)  
|                             | PARTIAL (Deployment failed on at least one cluster or appliance assigned to the Web application)  
|                             | IN_PROGRESS (Deployment of Web application on cluster is in progress)  
|                             | UNUSED (Web application is not deployed on any WAF cluster or the Web application is deployed on a cluster having no appliances registered to it)                                                             |
| deployed                    | (Date) The latest date on which the Web Application was pushed to the cluster.                                                                                                                             |
| synced                      | (Date) The date on which the Web application last synchronized with the cluster.                                                                                                                            |
| customRules.CustomRule.id   | (Long) The ID of the custom rule assigned to the web application.                                                                                                                                               |
| blockingMode                | (Boolean) If enabled blocks incoming traffic and displays the default WAF error page when it violates the selected security policy. If customPage is provided, displays the custom response page when incoming traffic is blocked.  
<p>|                             | If not enabled, only monitors (logs) the incoming traffic.                                                                                                                                                     |
| customPage.id               | (Long) The ID of the custom response page assigned to the web application.                                                                                                                                    |</p>
<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>customPage.uuid</td>
<td>(UUID) The UUID of the custom response page assigned to the web application.</td>
</tr>
<tr>
<td>customPage.name</td>
<td>(Text) The name of the custom response page assigned to the web application.</td>
</tr>
<tr>
<td>securityPolicy.id</td>
<td>(Long) The ID of the security policy assigned to the Web application.</td>
</tr>
<tr>
<td>securityPolicy.uuid</td>
<td>(UUID) The UUID of the security policy assigned to the Web application.</td>
</tr>
<tr>
<td>securityPolicy.name</td>
<td>(Text) The name of the security policy created by a user, assigned to the Web application.</td>
</tr>
<tr>
<td>httpProfile.id</td>
<td>(Long) The ID of the HTTP profile assigned to the web application.</td>
</tr>
<tr>
<td>httpProfile.uuid</td>
<td>(UUID) The UUID of the HTTP profile assigned to the web application.</td>
</tr>
<tr>
<td>httpProfile.name</td>
<td>(Text) The name of the HTTP profile assigned to the web application.</td>
</tr>
<tr>
<td>sslEnabled</td>
<td>(Boolean) Is SSL enabled? TRUE if any incoming URL declared in 'url' or 'urls.string' are using HTTPS.</td>
</tr>
<tr>
<td>sslProtocols</td>
<td>(Text) A comma separated list of allowed SSL protocols (SSLv3,TLS10,TLS11,TLS12)</td>
</tr>
<tr>
<td>sslCiphers</td>
<td>(Text) A comma separated list of allowed SSL ciphers (ECDHE-RSA-AES256-GCM-SHA384,ECDHE-ECDSA-AES256-GCM-SHA384,ECDHE-RSA-AES256-SHA384 etc...)</td>
</tr>
<tr>
<td>clusters.cluster.id</td>
<td>(Text) A WAF cluster ID used to deploy the Web application.</td>
</tr>
<tr>
<td>clusters.cluster.name</td>
<td>(UUID) A UUID for a WAF cluster used to deploy the Web application.</td>
</tr>
<tr>
<td>clusters.cluster.uuid</td>
<td>(Text) The name of a WAF cluster, created by a user, used to deploy the Web application.</td>
</tr>
<tr>
<td>lastComment</td>
<td>(Text) The last user defined comment.</td>
</tr>
</tbody>
</table>
Web Servers API

Use these API functions to manage origin servers as a server pool.

- Current web server count
- Get details on a web server
- Search web servers
- Create web server
- Update web server
- Update web servers (bulk)
- Delete web server
- Delete web servers (bulk)
Current web server count

Returns the total number of web server pools for WAF in the user’s account.

URL: https://<baseurl>/qps/rest/2.0/count/waf/webserver
Methods allowed: GET

Input

No input elements are available.

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
Asset must be within user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/webserver

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webserver.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>6</count>
</ServiceResponse>
Get details on a web server

Returns details about a specific web server pool for WAF, within the user’s scope. Want to find a web server pool ID to use as input? See Search web servers.

**URL:**  
https://<baseurl>/qps/rest/2.0/get/waf/webserver/<id>

**Methods allowed:**  
GET

**Input**

The element "id" (Integer) is required, where "id" identifies the web server pool ID of interest.

**Permissions**

User must have WAF module enabled  
User must have "API ACCESS" permission  
Asset must be within user’s scope

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"  
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/webserver/2401

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webserver.xsd">  
  <responseCode>SUCCESS</responseCode>  
  <count>1</count>  
  <data>  
    <WebServer>  
      <id>2401</id>  
      <uuid>069446f8-dd5a-4c5e-8cbe-346c148582ba</uuid>  
      <name>  
        <![CDATA[WAF Web Server 6]]>  
      </name>  
      <description>  
        <![CDATA[This is a server pool]]>
    </WebServer>
  </data>
</ServiceResponse>
```
Chapter 3 — Web Servers API
Get details on a web server

</description>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-05-14T08:52:35Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-05-14T08:52:35Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<loadBalancingAlgorithm>FIRST</loadBalancingAlgorithm>
<addresses>
  <WebServerAddress>
    <url>http://172.17.0.1:9081</url>
    <weight>20</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://172.17.0.2:9081</url>
    <weight>10</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://172.17.0.3:9081</url>
    <weight>1</weight>
  </WebServerAddress>
</addresses>
</data>
</ServiceResponse>
Search web servers

Finds web server pools in the user’s account matching the search criteria.

**URL:**  
https://<baseurl>/qps/rest/2.0/search/waf/webserver

**Methods allowed:**  
POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: web servers for descriptions of these elements.

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Long)</td>
<td>createdBy.username (Text)</td>
</tr>
<tr>
<td>uuid (UUID)</td>
<td>createdBy.firstname (Text)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>createdBy.lastname (Text)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>updated (Date)</td>
</tr>
<tr>
<td>loadBalancingAlgorithm (Text)</td>
<td>updatedBy.id (Long)</td>
</tr>
<tr>
<td>addresses.url (Text)</td>
<td>updatedBy.username (Text)</td>
</tr>
<tr>
<td>addresses.weight (Integer)</td>
<td>updatedBy.firstname (Text)</td>
</tr>
<tr>
<td>owner.id (Long)</td>
<td>updatedBy.lastname (Text)</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>webApps.webApp.id (Long)</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>webApps.webApp.uuid (UUID)</td>
</tr>
<tr>
<td>owner.lastname (Text)</td>
<td>webApps.webApp.name (Text)</td>
</tr>
<tr>
<td>created (Date)</td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td>createdBy.id (Long)</td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

**Allowed Operators**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>EQUALS, NOT EQUALS, GREATER, LESSER, IN</td>
</tr>
<tr>
<td>Long</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>UUID</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Text</td>
<td>CONTAINS, EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Date</td>
<td>EQUALS, NOT EQUALS, GREATER, LESSER</td>
</tr>
</tbody>
</table>

Note: The elements createdBy.id and updatedBy.id only support EQUALS.
Permissions

User must have WAF module enabled
User must have "API ACCESS" permission
Asset must be within user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-"https://qualysapi.qualys.com/qps/rest/2.0/search/waf/webserver" <file.xml

Note: "file.xml" contains the request POST data.
The request POST data is optional. If you leave it empty all web server pools in the user’s scope are returned.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="addresses.url" operator="CONTAINS">172.17.0</Criteria>
    <Criteria field="addresses.weight" operator="GREATER">1</Criteria>
  </filters>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd
/2.0/waf/webserver.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <hasMoreRecords>false</hasMoreRecords>
  <data>
    <WebServer>
      <id>2401</id>
      <uuid>069446f8-dd5a-4c5e-8cbe-346c148582ba</uuid>
    </WebServer>
  </data>
</ServiceResponse>
<name>
  <![CDATA[WAF Web Server 6]]>
</name>
<description>
  <![CDATA[This is a server pool]]>
</description>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-05-14T08:52:35Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-05-14T08:52:35Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<loadBalancingAlgorithm>FIRST</loadBalancingAlgorithm>
<addresses>
  <WebServerAddress>
    <url>http://172.17.0.1:9081</url>
    <weight>20</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://172.17.0.2:9081</url>
    <weight>10</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://172.17.0.3:9081</url>
    <weight>1</weight>
  </WebServerAddress>
</addresses>
Chapter 3 — Web Servers API
Search web servers

</WebServer>
</data>
</ServiceResponse>
Create web server

Create a web server pool which you can assign to a web application.

URL: https://<baseurl>/qps/rest/2.0/create/waf/webserver
Methods allowed: POST

Input

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: web servers for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>loadBalancingAlgorithm (Text)</td>
<td>tags</td>
</tr>
<tr>
<td>addresses.WebServerAddress</td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td></td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

Each addresses.WebServerAddress element should contain a url and a weight.

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Create WAF Asset" permission

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/create/waf/webserver" < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <WebServer>
      <name>WAF Web Server 6</name>
  </WebServer>
</ServiceRequest>
<description>This is a server pool</description>
<loadBalancingAlgorithm>FIRST</loadBalancingAlgorithm>
<addresses>
  <WebServerAddress>
    <url>http://172.17.0.1:9081</url>
    <weight>20</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://172.17.0.2:9081</url>
    <weight>10</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://172.17.0.3:9081</url>
  </WebServerAddress>
</addresses>
</WebServer>
</data>
</ServiceRequest>

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webserver.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebServer>
      <id>2401</id>
      <uuid>069446f8-dd5a-4c5e-8cbe-346c148582ba</uuid>
      <name><![CDATA[WAF Web Server 6]]></name>
      <description><![CDATA[This is a server pool]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
      </owner>
    </WebServer>
  </data>
</ServiceResponse>
<last_name>Doe</last_name>
</owner>
<created>2017-05-14T08:52:35Z</created>
<created_by>
  <id>3988443</id>
  <username>john_doe</username>
  <first_name>John</first_name>
  <last_name>Doe</last_name>
</created_by>
<updated>2017-05-14T08:52:35Z</updated>
<updated_by>
  <id>3988443</id>
  <username>john_doe</username>
  <first_name>John</first_name>
  <last_name>Doe</last_name>
</updated_by>
<load_balancing_algorithm>FIRST</load_balancing_algorithm>
<addresses>
  <web_server_address>
    <url>http://172.17.0.1:9081</url>
    <weight>20</weight>
  </web_server_address>
  <web_server_address>
    <url>http://172.17.0.2:9081</url>
    <weight>10</weight>
  </web_server_address>
  <web_server_address>
    <url>http://172.17.0.3:9081</url>
    <weight>1</weight>
  </web_server_address>
</addresses>
</web_server>
</data>
</ServiceResponse>
Update web server

Update a web server pool in the user’s account. You can update all fields except tag ID and tag name.

**URL:**  
https://<baseurl>/qps/rest/2.0/update/waf/webserver/<id>

**Methods allowed:**  
POST

**Input**

The "id" (Long) element is required. This identifies the web server pool you want to update.

Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: web servers for descriptions of these elements.

<table>
<thead>
<tr>
<th>name (Text)</th>
<th>addresses.WebServerAddress</th>
</tr>
</thead>
<tbody>
<tr>
<td>description (Text)</td>
<td>tags</td>
</tr>
<tr>
<td>loadBalancingAlgorithm (Text)</td>
<td></td>
</tr>
</tbody>
</table>

Each addresses.WebServerAddress element should contain a url and a weight.

**Permissions**

User must have the WAF module enabled  
User must have "API ACCESS" permission  
User must have "Update WAF Asset" permission  
Asset must be within user’s scope

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"  
--data-binary @-  
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/webserver/2401" < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
<WebServer>
  <name>WAF Web Server 11</name>
  <description>New server pool</description>
  <loadBalancingAlgorithm>ROUNDROBIN</loadBalancingAlgorithm>
  <addresses>
    <WebServerAddress>
      <url>http://55.17.0.1:9081</url>
      <weight>5</weight>
    </WebServerAddress>
    <WebServerAddress>
      <url>http://56.17.0.2:9081</url>
      <weight>4</weight>
    </WebServerAddress>
    <WebServerAddress>
      <url>http://57.17.0.3:9081</url>
      <weight>3</weight>
    </WebServerAddress>
  </addresses>
</WebServer>
</data>
</ServiceRequest>

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webserver.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebServer>
      <id>2401</id>
      <uuid>069446f8-dd5a-4c5e-8cbe-346c148582ba</uuid>
      <name><![CDATA[WAF Web Server 11]]></name>
      <description><![CDATA[New server pool]]></description>
      <owner>
```
<id>3988443</id>
<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</owner>
<created>2017-05-14T08:52:35Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-05-14T09:21:16Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<loadBalancingAlgorithm>ROUNDROBIN</loadBalancingAlgorithm>
<addresses>
  <WebServerAddress>
    <url>http://56.17.0.2:9081</url>
    <weight>4</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://55.17.0.1:9081</url>
    <weight>5</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://57.17.0.3:9081</url>
    <weight>3</weight>
  </WebServerAddress>
</addresses>
</ServiceResponse>
Update web servers (bulk)

Update multiple web application assets in the user’s account. You can update all fields except tag ID and tag name.

**URL:**
https://<baseurl>/qps/rest/2.0/update/waf/webserver

**Methods allowed:**
POST

**Input**

All elements for the search operation are supported. See Search web servers.

Allowed input elements for bulk update are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. See Reference: web servers for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>loadBalancingAlgorithm (Text)</td>
<td>tags</td>
</tr>
<tr>
<td>addresses.WebServerAddress</td>
<td></td>
</tr>
</tbody>
</table>

**Allowed Operators**

- **Integer**: EQUALS, NOT EQUALS, GREATER, LESSER, IN
- **Long**: EQUALS, NOT EQUALS
- **UUID**: EQUALS, NOT EQUALS
- **Text**: CONTAINS, EQUALS, NOT EQUALS
- **Date**: EQUALS, NOT EQUALS, GREATER, LESSER

Each addresses.WebServerAddress element should contain a url and a weight.

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Update WAF Asset" permission
Asset must be within user's scope
Example

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/webserver" < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="loadBalancingAlgorithm" operator="NOT EQUALS">ROUNDROBIN</Criteria>
  </filters>
  <data>
    <WebServer>
      <loadBalancingAlgorithm>ROUNDROBIN</loadBalancingAlgorithm>
    </WebServer>
  </data>
</ServiceRequest>

**Response**
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webserver.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebServer>
      <id>2401</id>
      <uuid>069446f8-dd5a-4c5e-8cbe-346c148582ba</uuid>
      <name><![CDATA[WAF Web Server 11]]></name>
      <description><![CDATA[New server pool]]></description>
  </WebServer>
</ServiceResponse>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-05-14T08:52:35Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-05-14T09:25:31Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<loadBalancingAlgorithm>ROUNDROBIN</loadBalancingAlgorithm>
<addresses>
  <WebServerAddress>
    <url>http://56.17.0.2:9081</url>
    <weight>4</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://55.17.0.1:9081</url>
    <weight>5</weight>
  </WebServerAddress>
  <WebServerAddress>
    <url>http://57.17.0.3:9081</url>
    <weight>3</weight>
  </WebServerAddress>
</addresses>
</WebServer>
</data>
</ServiceResponse>
Delete web server

Delete a web server pool in user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/webserver/<id>

**Methods allowed:**
POST

**Input**
The "id" (Long) element is required. This identifies the web server pool you want to delete.

**Permissions**
User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Delete WAF Asset" permission
Asset must be within user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/webserver/1201

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/webserver.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <WebServer>
      <id>1201</id>
    </WebServer>
  </data>
</ServiceResponse>
```
Delete web servers (bulk)

Delete multiple web server pools in the user’s account.

**URL:** https://<baseurl>/qps/rest/2.0/delete/waf/webserver

**Methods allowed:** POST

**Input**

All elements for the search operation are supported. See Search web servers.

**Permissions**

- User must have the WAF module enabled
- User must have "API ACCESS" permission
- User must have "Delete WAF Asset" permission
- Asset must be within user’s scope

**Example**

**Request:**


Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">My Web Server</Criteria>
  </filters>
</ServiceRequest>
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
```
Chapter 3 — Web Servers API
Delete web servers (bulk)

```
<2.0/waf/webserver.xsd>
  <responseCode>SUCCESS</responseCode>
  <count>2</count>
  <data>
    <WebServer>
      <id>1401</id>
    </WebServer>
    <WebServer>
      <id>1601</id>
    </WebServer>
  </data>
</ServiceResponse>
```
## Reference: web servers

A reference of all web server pool elements is provided below.

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>(Long) Web server profile identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid</td>
<td>(UUID) Web server profile identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name</td>
<td>(Text) The name of the web server profile as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>description</td>
<td>(Text) The description of the web server profile.</td>
</tr>
<tr>
<td>loadBalancingAlgorithm</td>
<td>(Text) Choose the method to load balance traffic between the servers. Your choices are: <strong>roundrobin</strong> - Each server receives the connection in turns, according to their weights. Weights are dynamically adjusted for best performance. <strong>leastconn</strong> - The server with the lowest number of connections receives the connection. Use of this algorithm is recommended where very long sessions are expected, such as LDAP, SQL, TSE, etc. <strong>first</strong> - The first server with available connection slots receives the connection. The servers are chosen from the lowest numeric identifier to the highest, which defaults to the server’s position in the farm. <strong>source</strong> - Only one designated server receives the connection, based on the source IP address. The source IP address is hashed and divided by the total weight of the running servers to designate which server will receive the request. This ensures that the same client IP address will always reach the same server as long as no server goes down or up.</td>
</tr>
<tr>
<td>addresses.WebServerAddress</td>
<td>Server URL and Weight.</td>
</tr>
<tr>
<td>addresses.url</td>
<td>You can add one server or multiple servers to load balance traffic between several original URLs. All URLs must use the same protocol and port. For each server provide the IP address or a Fully Qualified Domain Name.</td>
</tr>
<tr>
<td>addresses.weight</td>
<td>For each server add the corresponding weight between 0 to 256. WAF uses weights to distribute the request load to various servers in the Web Server Pool. Default weight is 1.</td>
</tr>
<tr>
<td>owner</td>
<td>(Text) The user for Qualys Cloud Platform who owns this web server profile.</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>owner.id</td>
<td>(Long) The user ID of the web server profile owner.</td>
</tr>
<tr>
<td>owner.username</td>
<td>(Text) The user name of the web server profile owner.</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>(Text) The first name of the web server profile owner.</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>(Text) The last name of the web server profile owner.</td>
</tr>
<tr>
<td>created</td>
<td>(Date) The date/time when the web server profile was created.</td>
</tr>
<tr>
<td>createdBy.id</td>
<td>(Long) The user ID who created the web server profile.</td>
</tr>
<tr>
<td>createdBy.username</td>
<td>(Text) The user name who created the web server profile.</td>
</tr>
<tr>
<td>createdBy.firstname</td>
<td>(Text) The first name of the user who created the web server profile.</td>
</tr>
<tr>
<td>createdBy.lastname</td>
<td>(Text) The last name of the user who created the web server profile.</td>
</tr>
<tr>
<td>updated</td>
<td>(Date) The date/time when the web server profile was last updated.</td>
</tr>
<tr>
<td>updatedBy.id</td>
<td>(Long) The user ID who last updated the web server profile.</td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>(Text) The user name who last updated the web server profile.</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>(Text) The first name of the user who updated the web server profile.</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>(Text) The last name of the user who updated the web server profile.</td>
</tr>
<tr>
<td>tags</td>
<td>(Text) List of tags associated with the web server profile. Valid action: Update</td>
</tr>
<tr>
<td>webApps.webApp.id</td>
<td>(Long) The ID of the Web Application this web server profile is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.uuid</td>
<td>(UUID) The UUID of the Web Application this web server profile is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.name</td>
<td>(Text) The name of the Web Application this web server profile is associated with.</td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>(Long) The ID of a tag associated with the web server profile.</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>(Text) The name, defined by a user, of a tag associated with the web server profile.</td>
</tr>
</tbody>
</table>
Healthchecks API

Use these API functions to manage healthchecks.

Current healthcheck count
Get details on a healthcheck
Search healthchecks
Create healthcheck
Update healthcheck
Update healthchecks (bulk)
Delete healthcheck
Delete healthchecks (bulk)
Chapter 4 — Healthchecks API

Current healthcheck count

Returns the total number of healthchecks for WAF in the user’s account.

URL: https://<baseurl>/qps/rest/2.0/count/waf/healthcheck
Methods allowed: GET

Input

No input elements are available.

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
Asset must be within user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/healthcheck

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/healthcheck.xsd">
   <responseCode>SUCCESS</responseCode>
   <count>3</count>
</ServiceResponse>
Get details on a healthcheck

Returns details about a specific healthcheck for WAF, within the user’s scope. Want to find a healthcheck ID to use as input? See Search healthchecks.

**URL:**
https://<baseurl>/qps/rest/2.0/get/waf/healthcheck/<id>

**Methods allowed:**
GET

**Input**
The element "id" (Integer) is required, where "id" identifies the healthcheck ID of interest.

**Permissions**
User must have WAF module enabled
User must have "API ACCESS" permission
Asset must be within user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/healthcheck/1401

**Response**
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/healthcheck.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Healthcheck>
      <id>1401</id>
      <uuid>b491ed9c-d9fd-461d-81eb-e8ee251289c7</uuid>
      <name><![CDATA[My Healthcheck]]></name>
      <description><![CDATA[This is a healthcheck]]></description>
    </Healthcheck>
  </data>
</ServiceResponse>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-04-11T08:29:42Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-04-11T08:29:42Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<system>false</system>
<method>GET</method>
<path><![CDATA[/path]]></path>
<expectedResponseCode>200</expectedResponseCode>
<intervalUp>5</intervalUp>
<intervalDown>6</intervalDown>
<intervalFlapping>7</intervalFlapping>
<nbSuccessesUp>8</nbSuccessesUp>
<nbFailuresDown>9</nbFailuresDown>
<timeout>10</timeout>
</Healthcheck>
</data>
</ServiceResponse>
Search healthchecks

Finds healthchecks in the user’s account matching the search criteria.

**URL:**
https://<baseurl>/qps/rest/2.0/search/waf/healthcheck

**Methods allowed:** POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: healthchecks for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Long)</td>
<td>owner.lastname (Text)</td>
</tr>
<tr>
<td>uuid (UUID)</td>
<td>created (Date)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>createdBy.id (Long)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>createdBy.username (Text)</td>
</tr>
<tr>
<td>method</td>
<td>createdBy.firstname (Text)</td>
</tr>
<tr>
<td>path (Text)</td>
<td>createdBy.lastname (Text)</td>
</tr>
<tr>
<td>expectedResponseCode (Long)</td>
<td>updated (Date)</td>
</tr>
<tr>
<td>intervalUp (Long)</td>
<td>updatedBy.id (Long)</td>
</tr>
<tr>
<td>intervalDown (Long)</td>
<td>updatedBy.username (Text)</td>
</tr>
<tr>
<td>intervalFlapping (Long)</td>
<td>updatedBy.firstname (Text)</td>
</tr>
<tr>
<td>nbSuccessesUp (Long)</td>
<td>updatedBy.lastname (Text)</td>
</tr>
<tr>
<td>nbFailuresDown (Long)</td>
<td>webApps.webApp.id (Long)</td>
</tr>
<tr>
<td>timeout (Long)</td>
<td>webApps.webApp.uuid (UUID)</td>
</tr>
<tr>
<td>owner.id (Long)</td>
<td>webApps.webApp.name (Text)</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

**Allowed Operators**
- **Long**
  - EQUALS, NOT EQUALS
  - Note: The elements createdBy.id and updatedBy.id only support EQUALS.
- **UUID**
  - EQUALS, NOT EQUALS
Permissions

User must have WAF module enabled
User must have "API ACCESS" permission
Asset must be within user's scope

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-"https://qualysapi.qualys.com/qps/rest/2.0/search/waf/healthcheck"
< file.xml

Note: "file.xml" contains the request POST data.
The request POST data is optional. If you leave it empty all healthchecks in the user's
scope are returned.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="method" operator="EQUALS">GET</Criteria>
  </filters>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/healthcheck.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <hasMoreRecords>false</hasMoreRecords>
  <data>
    <Healthcheck>
      <id>10801</id>
      <uuid>0e77cf97-b33a-4105-b273-72d49217b565</uuid>
    </Healthcheck>
  </data>
</ServiceResponse>
<name>
  <![CDATA[Standard Healthcheck]]>
</name>
<owner>
  <id>2501190</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-03-01T22:22:28Z</created>
<createdBy>
  <id>2501190</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-03-01T22:22:28Z</updated>
<system>true</system>
<method>GET</method>
<path>
  <![CDATA[/]]></path>
<expectedResponseCode>200</expectedResponseCode>
<intervalUp>15</intervalUp>
<intervalDown>5</intervalDown>
<intervalFlapping>10</intervalFlapping>
<nbSuccessesUp>3</nbSuccessesUp>
<nbFailuresDown>3</nbFailuresDown>
<timeout>15</timeout>
<webApps/>
</Healthcheck>
</data>
</ServiceResponse>
Create healthcheck

Create a healthcheck which you can assign to a web application.

**URL:** https://<baseurl>/qps/rest/2.0/create/waf/healthcheck

**Methods allowed:** POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: healthchecks for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>method</td>
<td>tags</td>
</tr>
<tr>
<td>path (Text)</td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td>expectedResponseCode (Long)</td>
<td>tags.tag.name (Text)</td>
</tr>
<tr>
<td>intervalUp (Long)</td>
<td></td>
</tr>
<tr>
<td>intervalDown (Long)</td>
<td></td>
</tr>
<tr>
<td>intervalFlapping (Long)</td>
<td></td>
</tr>
<tr>
<td>nbSuccessesUp (Long)</td>
<td></td>
</tr>
<tr>
<td>nbFailuresDown (Long)</td>
<td></td>
</tr>
<tr>
<td>timeout (Long)</td>
<td></td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Create WAF Asset" permission

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/create/waf/healthcheck"
< file.xml

Note: "file.xml" contains the request POST data.
Request POST Data:
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <Healthcheck>
      <name>My Healthcheck</name>
      <description>desc</description>
      <method>GET</method>
      <path>/path</path>
      <expectedResponseCode>200</expectedResponseCode>
      <intervalUp>5</intervalUp>
      <intervalDown>6</intervalDown>
      <intervalFlapping>7</intervalFlapping>
      <nbSuccessesUp>8</nbSuccessesUp>
      <nbFailuresDown>9</nbFailuresDown>
      <timeout>10</timeout>
    </Healthcheck>
  </data>
</ServiceRequest>
```

Response
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/healthcheck.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Healthcheck>
      <id>1601</id>
      <uuid>1b743e15-0756-4ac9-870a-b71dc031c2d4</uuid>
      <name><![CDATA[My Healthcheck]]></name>
      <description><![CDATA[This is a healthcheck]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
      </owner>
    </Healthcheck>
  </data>
</ServiceResponse>
```
<Healthcheck>
  <owner>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </owner>
  <created>2017-04-11T13:54:37Z</created>
  <createdBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </createdBy>
  <updated>2017-04-11T13:54:37Z</updated>
  <updatedBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
  </updatedBy>
  <system>false</system>
  <method>GET</method>
  <path><![CDATA[/path]]></path>
  <expectedResponseCode>200</expectedResponseCode>
  <intervalUp>5</intervalUp>
  <intervalDown>6</intervalDown>
  <intervalFlapping>7</intervalFlapping>
  <nbSuccessesUp>8</nbSuccessesUp>
  <nbFailuresDown>9</nbFailuresDown>
  <timeout>10</timeout>
</Healthcheck>
</ServiceResponse>
Update healthcheck

Update a healthcheck in the user’s account. You can update all fields except tag ID and tag name.

**URL:** https://<baseurl>/qps/rest/2.0/update/waf/healthcheck/<id>

**Methods allowed:** POST

**Input**

The "id" (Long) element is required. This identifies the healthcheck you want to update.

Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: healthchecks for descriptions of these elements.

<table>
<thead>
<tr>
<th>name (Text)</th>
<th>intervalDown (Long)</th>
</tr>
</thead>
<tbody>
<tr>
<td>description (Text)</td>
<td>intervalFlapping (Long)</td>
</tr>
<tr>
<td>method</td>
<td>nbSuccessesUp (Long)</td>
</tr>
<tr>
<td>path (Text)</td>
<td>nbFailuresDown (Long)</td>
</tr>
<tr>
<td>expectedResponseCode (Long)</td>
<td>timeout (Long)</td>
</tr>
<tr>
<td>intervalUp (Long)</td>
<td>tags</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Update WAF Asset" permission
Asset must be within user’s scope

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/healthcheck/1602" < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
```

Qualys Web Application Firewall API
Chapter 4 — Healthchecks API
Update healthcheck

<data>
  <Healthcheck>
    <description>New healthcheck desc</description>
    <intervalFlapping>77</intervalFlapping>
  </Healthcheck>
</data>

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/healthcheck.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Healthcheck>
      <id>1602</id>
      <uuid>f032764e-1de3-49e7-9c22-a9f070a909ca</uuid>
      <name><![CDATA[My Healthcheck 2]]></name>
      <description><![CDATA[New healthcheck desc]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </owner>
      <created>2017-04-11T13:57:41Z</created>
      <createdBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </createdBy>
      <updatedBy>
      </updatedBy>
    </Healthcheck>
  </data>
</ServiceResponse>
<id>3988443</id>
<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</updatedBy>
<system>false</system>
<method>GET</method>
<path><![CDATA[/path]]></path>
<expectedResponseCode>200</expectedResponseCode>
<intervalUp>5</intervalUp>
<intervalDown>6</intervalDown>
<intervalFlapping>77</intervalFlapping>
<nbSuccessesUp>8</nbSuccessesUp>
<nbFailuresDown>9</nbFailuresDown>
<timeout>10</timeout>
</Healthcheck>
</data>
</ServiceResponse>
Chapter 4 — Healthchecks API
Update healthchecks (bulk)

Update multiple healthchecks in the user’s account. You can update all fields except tag ID and tag name.

URL: https://<baseurl>/qps/rest/2.0/update/waf/healthcheck
Methods allowed: POST

Input
All elements for the search operation are supported. See Search healthchecks.

Allowed input elements for bulk update are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. See Reference: healthchecks for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Text</td>
</tr>
<tr>
<td>description</td>
<td>Text</td>
</tr>
<tr>
<td>method</td>
<td></td>
</tr>
<tr>
<td>path</td>
<td>Text</td>
</tr>
<tr>
<td>expectedResponseCode</td>
<td>Long</td>
</tr>
<tr>
<td>intervalUp</td>
<td>Long</td>
</tr>
<tr>
<td>intervalDown</td>
<td>Long</td>
</tr>
<tr>
<td>intervalFlapping</td>
<td>Long</td>
</tr>
<tr>
<td>nbSuccessesUp</td>
<td>Long</td>
</tr>
<tr>
<td>nbFailuresDown</td>
<td>Long</td>
</tr>
<tr>
<td>timeout</td>
<td>Long</td>
</tr>
<tr>
<td>tags</td>
<td></td>
</tr>
</tbody>
</table>

Allowed Operators
- Long: EQUALS, NOT EQUALS
- UUID: EQUALS, NOT EQUALS
- Text: CONTAINS, EQUALS, NOT EQUALS
- Date: EQUALS, NOT EQUALS, GREATER, LESSER

Permissions
- User must have the WAF module enabled
- User must have “API ACCESS” permission
- User must have “Update WAF Asset” permission
- Asset must be within user’s scope
Example

**Request:**

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/healthcheck"

< file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="nbSuccessesUp" operator="GREATER">5</Criteria>
  </filters>
  <data>
    <Healthcheck>
      <nbFailuresDown>55</nbFailuresDown>
    </Healthcheck>
  </data>
</ServiceRequest>

**Response**

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/healthcheck.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>2</count>
  <data>
    <Healthcheck>
      <id>1601</id>
      <uuid>1b743e15-0756-4ac9-870a-b71dc031c2d4</uuid>
      <name>
        <![CDATA[My Healthcheck]]>
      </name>
      <description>
        <![CDATA[This is a healthcheck]]>
      </description>
    </Healthcheck>
  </data>
</ServiceResponse>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-04-11T13:54:37Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-04-11T14:09:13Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<system>false</system>
<method>GET</method>
<path><![CDATA[/path]]></path>
<expectedResponseCode>200</expectedResponseCode>
<intervalUp>5</intervalUp>
<intervalDown>6</intervalDown>
<intervalFlapping>7</intervalFlapping>
<nbSuccessesUp>99</nbSuccessesUp>
<nbFailuresDown>55</nbFailuresDown>
<timeout>10</timeout>
</Healthcheck>
<Healthcheck>
  <id>1602</id>
  <uuid>f032764e-1de3-49e7-9c22-a9f070a909ca</uuid>
  <name><![CDATA[My Healthcheck 2]]></name>
  <description><![CDATA[my desc]]></description>
</Healthcheck>
</description>

<owner>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
</owner>

<created>2017-04-11T13:57:41Z</created>

<createdBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
</createdBy>

<updated>2017-04-11T14:09:17Z</updated>

<updatedBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
</updatedBy>

<system>false</system>

<method>GET</method>

<path>
    <![CDATA[/path]]>
</path>

<expectedResponseCode>200</expectedResponseCode>

<intervalUp>5</intervalUp>

<intervalDown>6</intervalDown>

<intervalFlapping>77</intervalFlapping>

<nbSuccessesUp>99</nbSuccessesUp>

<nbFailuresDown>55</nbFailuresDown>

<timeout>10</timeout>

</Healthcheck>

</data>

</ServiceResponse>
Delete healthcheck

Delete a healthcheck in user’s account.

URL: https://<baseurl>/qps/rest/2.0/delete/waf/healthcheck/<id>

Methods allowed: POST

Input

The "id" (Long) element is required. This identifies the healthcheck you want to delete.

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Delete WAF Asset" permission
Asset must be within user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/healthcheck/1
402

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/healthcheck.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Healthcheck>
      <id>1402</id>
    </Healthcheck>
  </data>
</ServiceResponse>
Delete healthchecks (bulk)

Delete multiple healthchecks in the user’s account.

**URL:**  
https://<baseurl>/qps/rest/2.0/delete/waf/healthcheck

**Methods allowed:**  
POST

**Input**

All elements for the search operation are supported. See [Search healthchecks](#).

**Permissions**

User must have the WAF module enabled  
User must have "API ACCESS" permission  
User must have "Delete WAF Asset" permission  
Asset must be within user’s scope

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"  
--data-binary @-  
"https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/healthcheck"  
< file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0"?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">My Healthcheck</Criteria>
  </filters>
</ServiceRequest>
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd"/>
```
Chapter 4 — Healthchecks API
Delete healthchecks (bulk)

```
/2.0/waf/healthcheck.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Healthcheck>
      <id>1401</id>
    </Healthcheck>
  </data>
</ServiceResponse>
```
### Reference: healthchecks

A reference of all healthcheck elements is provided below.

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>(Long) Healthcheck profile identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid</td>
<td>(UUID) Healthcheck profile identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name</td>
<td>(Text) The name of the healthcheck profile as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>description</td>
<td>(Text) Description of the healthcheck profile.</td>
</tr>
<tr>
<td>method</td>
<td>(Text) The method to use in the healthcheck periodic request (GET or HEAD).</td>
</tr>
<tr>
<td>path</td>
<td>(Text) The path to use in the healthcheck periodic request. Can be empty but have to be specified.</td>
</tr>
<tr>
<td>expectedResponseCode</td>
<td>(Long) Expected web server response code to consider the web server as available.</td>
</tr>
<tr>
<td>intervalUp</td>
<td>(Long) Number of healthcheck requests per second when the web server status is up.</td>
</tr>
<tr>
<td>intervalDown</td>
<td>(Long) Number of healthcheck requests per second when the web server status is down.</td>
</tr>
<tr>
<td>intervalFlapping</td>
<td>(Long) Number of healthcheck requests per second when the web server status is flapping (not stable).</td>
</tr>
<tr>
<td>nbSuccessesUp</td>
<td>(Long) The amount of successes before considering the web server as up.</td>
</tr>
<tr>
<td>nbFailuresDown</td>
<td>(Long) The amount of failures before considering the web server as down.</td>
</tr>
<tr>
<td>timeout</td>
<td>(Long) The healthcheck request timeout in seconds.</td>
</tr>
<tr>
<td>owner</td>
<td>(Text) The user for Qualys Cloud Platform who owns this healthcheck profile.</td>
</tr>
<tr>
<td>owner.id</td>
<td>(Long) The user ID of the healthcheck profile owner.</td>
</tr>
<tr>
<td>owner.username</td>
<td>(Text) The user name of the healthcheck profile owner.</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>(Text) The first name of the healthcheck profile owner.</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>(Text) The last name of the healthcheck profile owner.</td>
</tr>
<tr>
<td>created</td>
<td>(Date) The date/time when the healthcheck profile was created.</td>
</tr>
<tr>
<td>createdBy.id</td>
<td>(Long) The user ID who created the healthcheck profile.</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>createdBy.username</td>
<td>(Text) The user name who created the healthcheck profile.</td>
</tr>
<tr>
<td>createdBy.firstname</td>
<td>(Text) The first name of the user who created the healthcheck profile.</td>
</tr>
<tr>
<td>createdBy.lastname</td>
<td>(Text) The last name of the user who created the healthcheck profile.</td>
</tr>
<tr>
<td>updated</td>
<td>(Date) The date/time when the healthcheck profile was last updated.</td>
</tr>
<tr>
<td>updatedBy.id</td>
<td>(Long) The user ID who last updated the healthcheck profile.</td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>(Text) The user name who last updated the healthcheck profile.</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>(Text) The first name of the user who updated the healthcheck profile.</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>(Text) The last name of the user who updated the healthcheck profile.</td>
</tr>
<tr>
<td>tags</td>
<td>(Text) List of tags associated with the healthcheck profile. Valid action: Update</td>
</tr>
<tr>
<td>webApps.webApp.id</td>
<td>(Long) The ID of the Web Application this healthcheck profile is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.uuid</td>
<td>(UUID) The UUID of the Web Application this healthcheck profile is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.name</td>
<td>(Text) The name of the Web Application this healthcheck profile is associated with.</td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>(Long) The ID of a tag associated with the healthcheck profile.</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>(Text) The name, defined by a user, of a tag associated with the healthcheck profile.</td>
</tr>
</tbody>
</table>
SSL Certificates API

Use these API functions to manage SSL Certificates.

Current certificate count
Get details on a certificate
Search certificates
Create certificate
Update certificate
Update certificates (bulk)
Delete certificate
Delete certificates (bulk)
Current certificate count

Returns the total number of SSL certificates for WAF in the user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/count/waf/certificate

**Methods allowed:**
GET

**Input**

No input elements are available.

**Permissions**

- User must have WAF module enabled
- User must have "API ACCESS" permission
- Asset must be within user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/certificate

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd"
    >
    <responseCode>SUCCESS</responseCode>
    <count>3</count>
</ServiceResponse>
```
Get details on a certificate

Returns details about a specific SSL certificate for WAF, within the user’s scope. Want to find an SSL certificate ID to use as input? See Search certificates.

**URL:**
https://<baseurl>/qps/rest/2.0/get/waf/certificate/<id>

**Methods allowed:**
GET

**Input**
The element "id" (Integer) is required, where "id" identifies the SSL certificate ID of interest.

**Permissions**
User must have WAF module enabled
User must have "API ACCESS" permission
Asset must be within user’s scope

**Example**

**Request**:
```
url -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/certificate/8
```

**Response**
```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Certificate>
      <id>8</id>
      <uuid>a89fe014-d1c9-446d-b38f-8f7726158595</uuid>
      <name><![CDATA[Site03 PKCS12 Certificate]]></name>
      <description><![CDATA[This is a certificate]]>
    </Certificate>
  </data>
</ServiceResponse>
```
Chapter 5 — SSL Certificates API
Get details on a certificate

</certificateMetadata>
</Certificate>
</data>
</ServiceResponse>
# Search certificates

Finds SSL Certificates in the user’s account matching the search criteria.

**URL:**
https://<baseurl>/qps/rest/2.0/search/waf/certificate

**Methods allowed:** POST

## Input

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: certificates for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Long</td>
</tr>
<tr>
<td>uuid</td>
<td>UUID</td>
</tr>
<tr>
<td>name</td>
<td>Text</td>
</tr>
<tr>
<td>description</td>
<td>Text</td>
</tr>
<tr>
<td>owner.id</td>
<td>Long</td>
</tr>
<tr>
<td>owner.username</td>
<td>Text</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>Text</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>Text</td>
</tr>
<tr>
<td>webApps.webApp.id</td>
<td>Long</td>
</tr>
<tr>
<td>webApps.webApp.uuid</td>
<td>UUID</td>
</tr>
<tr>
<td>webApps.webApp.name</td>
<td>Text</td>
</tr>
<tr>
<td>createdBy.id</td>
<td>Long</td>
</tr>
<tr>
<td>createdBy.username</td>
<td>Text</td>
</tr>
<tr>
<td>createdBy.firstname</td>
<td>Text</td>
</tr>
<tr>
<td>createdBy.lastname</td>
<td>Text</td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>Long</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>Text</td>
</tr>
</tbody>
</table>

**Allowed Operators**

- Long: **EQUALS, NOT EQUALS**
  - Note: The elements createdBy.id and updatedBy.id only support EQUALS.
- UUID: **EQUALS, NOT EQUALS**
- Text: **CONTAINS, EQUALS, NOT EQUALS**
- Date: **EQUALS, NOT EQUALS, GREATER, LESSER**
Permissions

User must have WAF module enabled
User must have "API ACCESS" permission
Asset must be within user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/search/waf/certificate" < file.xml

Note: "file.xml" contains the request POST data.
The request POST data is optional. If you leave it empty all certificates in the user’s scope are returned.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">SUB</Criteria>
  </filters>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <hasMoreRecords>false</hasMoreRecords>
  <data>
    <Certificate>
      <id>36201</id>
      <uuid>52d35d06-b6bf-4365-bbee-81fe47a81115</uuid>
      <name><![CDATA[SubUserSSL]]></name>
    </Certificate>
  </data>
</ServiceResponse>
Chapter 5 — SSL Certificates API

Search certificates

```xml
<owner>
  <id>361390</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>

<created>2017-02-22T00:22:32Z</created>

<createdBy>
  <id>361390</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>

<updated>2017-02-23T00:31:51Z</updated>

<updatedBy>
  <id>361390</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>

<tags>
  <Tag>
    <id>7531232</id>
    <name> <![CDATA[Unassigned Business Unit]]> </name>
  </Tag>
</tags>

<certificateMetadata>
  <![CDATA["fileName":"leaf.pfx","commonName":"*.eng.qualys.com","issuer":"EMAILADDRESS=vmatosyan@qualys.com, CN=level1, OU=R & D, O=Qualys Inc., L=Redwood City, ST=California, C=US","dateStart":1452544631000,"dateEnd":1530304631000,"subject": "EMAILADDRESS=vmatosyan@qualys.com, CN=*.eng.qualys.com, OU=R & D, O=Qualys Inc., L=Redwood City, ST=California, C=US","sigAlgo":"MD5WithRSAEncryption","sn":1,"version":3,"isExpired":false,"isYetValid":true,"isSelfSigned":false,"publicKey":"ULNB IFB1YmxpYyBLZXKICAgICAgICAgICAgbW9kdWx1czogZGQ1ZmUwYTNhNTE0NzA4 M2FjMDk0NGQzZDI4Mjk4dDogMTAwMDEK","subjectAltNames": ["eng.qualys.com"],"subjectOrganization":"Qualys Inc.","subjectEmail":"vmatosyan@qualys.com","issuerOrganization":"]]>
</certificateMetadata>
```
Qualys Inc.,” "publicKeySize”: 768, "privateKey": { "usePassphrase": true, "content": "LS0tLS1CRUdJTiBSU0EgUFJJVkJkFURSBLRkLtLS0tLQpQcm9jLVR5cGU6IDQsRU5DUl1QVEVECkRFSy1Jb20vOiBERVMy1DQkMsNmRmRmVkJygyMmMwMGM3Mw oKZnUzcuLXVNBNEsAwYjliMjMjM2ZDYWQ4MjkKICAyIE1BcmFtZQogMjAwMDEK" }}...]]>

</certificateMetadata>
</Certificate>
</data>
</ServiceResponse>
Create certificate

Create an SSL Certificate which you can assign to a web application.

**URL:** https://<baseurl>/qps/rest/2.0/create/waf/certificate

**Methods allowed:** POST

A certificate profile can be created in two ways:
- You can create a certificate profile using a pfx file. In this case you must provide a pkcs12 element containing a cdata with the base64 encoded content of the pfx file and a passphrase used to decrypt the file. See Using a pfx file
- You can create a certificate profile using a certificate encoded as PEM along with it's private key file as PEM. In this case you must provide a certificate and privateKey element containing a cdata with the base64 encoded content of the PEM files and a passphrase used to decrypt the private key. See Using a PEM file

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: certificates for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>passphrase (Text)</td>
<td>pkcs12 (Text)</td>
</tr>
<tr>
<td>token (Text)</td>
<td>certificate (Text)</td>
</tr>
<tr>
<td></td>
<td>privateKey (Text)</td>
</tr>
<tr>
<td></td>
<td>chain (Text)</td>
</tr>
<tr>
<td></td>
<td>tags</td>
</tr>
<tr>
<td></td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td></td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Create WAF Asset" permission
Example

Using a pfx file

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/create/waf/certificate"
< file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<ServiceRequest>
  <data>
    <Certificate>
      <name>Site01 Certificate</name>
      <pkcs12>
        <![CDATA[MIIMQQIBAzCCDAcGCSqGSIb3DQEHAaACC/gEggvOMIIL8DCCBqcGCSqGS
Ib3DQEHbqCCBpgwgaUAqEAMIIGjQYJKoZI+EfSAdX3p0yHbkfMDewITAJBgUrDqMC
GgUABBQYWh1qCxhrJ9oXXMso//j3aadWQQImOh0187j8kMCAggA...]]>
      </pkcs12>
      <passphrase><![CDATA[ssl]]></passphrase>
    </Certificate>
  </data>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Certificate>
Chapter 5 — SSL Certificates API
Create certificate

```xml
<id>1</id>
<uuid>15322a6c-e936-483a-8f76-ad0947be9bed</uuid>
<name>
  <![CDATA[Site01 Certificate]]>
</name>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-04-03T09:01:03Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-04-03T09:01:03Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<token>
  <![CDATA[3c53911c79d6c6b10878768128548a10bfc9c3785f00add9760625d08
ea00656d26682933cd07747f65caac3390bcc6a1ee60f2dc2221317a01a9ee26
f3c7c69290baa9dd6939a96ce9d4055aaf9f054a26a47d32...]]>
</token>
<certificateMetadata>
  <![CDATA[" fileName": " uploaded-cert.pfx ", " commonName": " site03.xfuentes-docker ", " issuer": " EMAILADDRESS=xfuentes@qualys.com ", " CN=Intermediate CA, OU=Engineering, O=Qualys, ST=France, C=FR", " dateStart": 1490021391000, " dateEnd": 1522421391000, " subject": " EMAILADDRESS=xfuentes@qualys.com, CN=site03.xfuentes-docker, OU=Engineering, O=Qualys, L=Carcassonne, ST=France, C=FR", " sigAlgo": " SHA256WithRSAEncryption ", " sn": " 4101 ", " version": 3, " isExpired": false, " isYetValid": true, " isSelfSigned": false, " publicKey": " U1NB1PB1YmpYyBLZXXKICAgICAgICAgICAgbW9kdWxlczogZTFkYWFiMmQ0Nj... ]]>
</certificateMetadata>
```
Chapter 5 — SSL Certificates API
Create certificate

Using a PEM file

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/create/waf/certificate"
< file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<ServiceRequest>
 <data>
  <Certificate>
   <name>Site01 Certificate</name>
   <description>A PEM certificate with ca-chain</description>
   <certificate><![CDATA[LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUdBRENDQSp2OF3SUJB201DRUFNdTT2xRXJpQWFFdS98bG4YmRjWnRud01OSwprM0k4SUE9PQtLS0tLUVORCBDRVJUSUZJQ0FURS0tLS0tCg==...]]></certificate>
   <privateKey><![CDATA[LS0tLS1CRUdJTiBSU0EgUFJJVkJFURSBLRvktLS0tLQpQcm9jLVR5cGU6IDQsRU5DUllQVEVE9vdVVZan10ZkxxUDRMdTdpbDVWMHlqNjIvcXVqWmVBPT0KLS0tLS1FTkQgU1NBIFBSSVZBEUgs0V2LS0tLS0K...]]></privateKey>
   <passphrase>furax</passphrase>
   <token>qualys</token>
   <chain><![CDATA[LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUYrRENDQSp2OF3SUJB201DRUFdXb0RUpLb1pJaH7TkFRRuuCUUF3Z1pPeEN6QUpcZ05WQkFZVEFrW1MTVE4d0RWWU5VFJREVJUSUZJQ0tLS0tCg==...]]></chain>
  </Certificate>
 </data>
</ServiceRequest>
Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Certificate>
      <id>637</id>
      <uuid>9a3caf8c-498b-463b-84ce-4316af7454f7</uuid>
      <name>![CDATA[Site01 Certificate]]</name>
      <description>![CDATA[A PEM certificate with ca-chain]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </owner>
      <created>2017-04-06T08:31:34Z</created>
      <createdBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </createdBy>
      <updated>2017-04-06T08:58:01Z</updated>
      <updatedBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </updatedBy>
      <certificateMetadata>
        <![CDATA["fileName":null,"commonName":"site01.xfuentes-]]>
    </Certificate>
  </data>
</ServiceResponse>
Create certificate

```xml
<certificateMetadata>
  <![[CDATA[{"fileName":null,"commonName":"Intermediate CA","issuer":"EMAILADDRESS=xfuentes@qualys.com, CN=Intermediate CA, OU=Engineering, O=Qualys, ST=France, C=FR","dateStart":1481796209000,"dateEnd":1797156209000,"subject": "EMAILADDRESS=xfuentes@qualys.com, CN=Intermediate CA, OU=Engineering, O=Qualys, ST=France, C=FR","sigAlgo":"SHA256WithRSAEncryption","sn":"4096","version":3,"isExpired":false,"isYetValid":true,"isSelfSigned":false,"publicKey":"UlNBIFB1YmxpYyBLZXkXICAqICAgICAgICAgICAgbW9kdWx1czogYmVkNmVhZWVlOTcxYzBkOWYxMzg1M""xfuentes@qualys.com","issuerOrganization":"Qualys","publicKeySize":1452,"privateKey":null}...]]>
</certificateMetadata>
</data>
</ServiceResponse>
```
Chapter 5 — SSL Certificates API

Update certificate

Update an SSL Certificate in the user’s account. You can update all fields except tag ID and tag name.

**URL:** https://<baseurl>/qps/rest/2.0/update/waf/certificate/<id>

**Methods allowed:** POST

**Input**

The "id" (Long) element is required. This identifies the SSL Certificate you want to update.

Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: certificates for descriptions of these elements.

<table>
<thead>
<tr>
<th>element</th>
<th>data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Text</td>
</tr>
<tr>
<td>passphrase</td>
<td>Text</td>
</tr>
<tr>
<td>description</td>
<td>Text</td>
</tr>
<tr>
<td>token</td>
<td>Text</td>
</tr>
<tr>
<td>pkcs12</td>
<td>Text</td>
</tr>
<tr>
<td>chain</td>
<td>Text</td>
</tr>
<tr>
<td>certificate</td>
<td>Text</td>
</tr>
<tr>
<td>tags</td>
<td></td>
</tr>
<tr>
<td>privateKey</td>
<td>Text</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Update WAF Asset" permission
Asset must be within user's scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/certificate/410" < file.xml

Note: "file.xml" contains the request POST data.
Request POST Data:
<ServiceRequest>
<data>
  <Certificate>
    <name>Site01 Certificate Updated</name>
    <description>A simple PEM certificate</description>
    <certificate><![CDATA[LS0tLS1CRUdJTiBDRVJUSUZJQ0FURS0tLS0tCk1JSUdBRENDQStpZ0F3SUJB201DRUFNd0RRWUpLb1pJaHZjTkFRRUxCUUF3Z11NeEN6QUpCZ05WQkFZVEFrW1M9OdDxeTArT1ZJQ0FURS0tLS0tCg==...]]></certificate>
    <privateKey><![CDATA[LS0tLS1CRUdJTiBSU0EgUFJVVkFURSBLRVktLS0tLQpQc4QzRFRDE2MjM5RgoeUKMT0KLS0IFBSSVZBVEUgS0tLS0K...]]></privateKey>
    <passphrase>furax</passphrase>
    <token>qualys</token>
  </Certificate>
</data>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Certificate>
      <id>410</id>
      <uuid>8f6398d1-e333-4c5d-95d7-c81201df4d94</uuid>
      <name><![CDATA[Site01 Certificate Updated]]></name>
      <description><![CDATA[A simple PEM certificate]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </owner>
    </Certificate>
  </data>
</ServiceResponse>
Chapter 5 — SSL Certificates API

Update certificate

</owner>

<created>2017-04-05T08:53:20Z</created>

<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>

<updated>2017-04-05T15:39:36Z</updated>

<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>

<certificateMetadata>
<!['CDATA[{"fileName":null,"commonName":"site01.xfuentes-docker","issuer":"EMAILADDRESS=xfuentes@qualys.com,CN=Intermediate CA,OU=Engineering,O=Qualys,ST=France,C=FR","dateStart":1486388496000,"dateEnd":1518788496000,"subject":"EMAILADDRESS=xfuentes@qualys.com,CN=site01.xfuentes-docker,OU=Engineering,O=Qualys,ST=France,C=FR","sigAlgo":"SHA256WithRSAEncryption","sn":"4099","version":3,"isExpired":false,"isYetValid":true,"isSelfSigned":false,"NzNmZWIyYjY3MYzBmOGYxNjRkMDEzYTMwNDl1NTVjNmF1NTg1YjhkOTU5Yjaly","subjectEmail":"xfuentes@qualys.com","issuerOrganization":"Qualys","publicKeySize":768,"privateKey":null...}]]>
</certificateMetadata>

</Certificate>
</data>
</ServiceResponse>
Chapter 5 — SSL Certificates API
Update certificates (bulk)

Update multiple SSL Certificates in the user’s account. You can update all fields except tag ID and tag name.

**URL:**  
https://<baseurl>/qps/rest/2.0/update/waf/certificate

**Methods allowed:**  
POST

## Input

All elements for the search operation are supported. See [Search certificates](#).

Allowed input elements for bulk update are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. See [Reference: certificates](#) for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Text</td>
</tr>
<tr>
<td>passphrase</td>
<td>Text</td>
</tr>
<tr>
<td>description</td>
<td>Text</td>
</tr>
<tr>
<td>token</td>
<td>Text</td>
</tr>
<tr>
<td>pkcs12</td>
<td>Text</td>
</tr>
<tr>
<td>chain</td>
<td>Text</td>
</tr>
<tr>
<td>certificate</td>
<td>Text</td>
</tr>
<tr>
<td>tags</td>
<td></td>
</tr>
<tr>
<td>privateKey</td>
<td>Text</td>
</tr>
</tbody>
</table>

**Allowed Operators**

<table>
<thead>
<tr>
<th>Data Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>UUID</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Text</td>
<td>CONTAINS, EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Date</td>
<td>EQUALS, NOT EQUALS, GREATER, LESSER</td>
</tr>
</tbody>
</table>

## Permissions

User must have the WAF module enabled  
User must have "API ACCESS" permission  
User must have "Update WAF Asset" permission  
Asset must be within user’s scope
Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @- 
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/certificate"
< file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="name"
     operator="CONTAINS">qualys</Criteria>
  </filters>
  <data>
    <Certificate>
      <name>QCumber Certificate</name>
      <description>A simple test certificate updated</description>
    </Certificate>
  </data>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Certificate>
      <id>637</id>
      <uuid>9a3caf8c-498b-463b-84ce-4316af7454f7</uuid>
      <name>
        <![CDATA[QCumber Certificate]]></name>
      <description>
<!['simple test certificate updated']>
</description>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-04-06T08:31:34Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-04-06T08:35:57Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<certificateMetadata>
  <!['fileName':"uploaded-cert.pfx","commonName":"site03.xfuentes-docker","issuer":"EMAILADDRESS=xfuentes@qualys.com, CN=Intermediate CA, O=Qualys, ST=France, C=FR","dateStart":1490021391000,"dateEnd":1522421391000,"subject": "EMAILADDRESS=xfuentes@qualys.com, CN=site03.xfuentes-docker, OU=Engineering, O=Qualys, L=Carcassonne, ST=France, C=FR","sigAlgo":"SHA256WithRSAEncryption","sn":"4101","version":3, "isExpired":false,"isYetValid":true,"isSelfSigned":false,"publicKey":"UlNBIFB1YmxpYyBLZXxKICAgICAgICAgbW9kdWxlczoqZTFkYWFiMmQ0NjNkOWQxZTIyNGYWZmU1ZDUyYzY2NDQ1NWE4NzBmY2RhNWQyMzRjM2U3NWYxNGNiYjUwM2Q3ODVnMzWY0Nj"subjectEmail":"xfuentes@qualys.com","issuerOrganization":"Qualys","publicKeySize":768,"privateKey":null...]>
</certificateMetadata>
</Certificate>
</data>
</ServiceResponse>
Delete certificate

Delete an SSL Certificate in user's account.

URL: https://<baseurl>/qps/rest/2.0/delete/waf/certificate/<id>

Methods allowed: POST

Input

The "id" (Long) element is required. This identifies the SSL Certificate you want to delete.

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Delete WAF Asset" permission
Asset must be within user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/certificate/637

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Certificate>
      <id>637</id>
    </Certificate>
  </data>
</ServiceResponse>
Delete certificates (bulk)

Delete multiple SSL Certificates in the user’s account.

**URL:** https://<baseurl>/qps/rest/2.0/delete/waf/certificate

**Methods allowed:** POST

**Input**

All elements for the search operation are supported. See Search certificates.

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS” permission
User must have "Delete WAF Asset” permission
Asset must be within user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @- "https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/certificate" < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**
<?xml version="1.0" ?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">DEMO</Criteria>
  </filters>
</ServiceRequest>

**Response**
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/certificate.xsd">
<responseCode>SUCCESS</responseCode>
<count>1</count>
<data>
  <Certificate>
    <id>638</id>
  </Certificate>
</data>
</ServiceResponse>
# Reference: certificates

A reference of all SSL certificate elements is provided below.

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>(Long) Certificate profile identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid</td>
<td>(UUID) Certificate profile identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name</td>
<td>(Text) The name of the certificate profile as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>description</td>
<td>(Text) The description of the certificate profile.</td>
</tr>
<tr>
<td>pkcs12</td>
<td>(Text) base64 encoded PFX file content (containing certificate and private key). The passphrase is required to decrypt the file.</td>
</tr>
<tr>
<td>certificate</td>
<td>(Text) base64 encoded PEM file content of the certificate (requires privateKey attribute).</td>
</tr>
<tr>
<td>privateKey</td>
<td>(Text) base64 encoded PEM file content of the private key. The passphrase is required if the key file is encrypted.</td>
</tr>
<tr>
<td>passphrase</td>
<td>(Text) Used to decrypt the provided PFX or PEM private key.</td>
</tr>
<tr>
<td>token</td>
<td>(Text) The privateKey of the certificate will be encrypted using this token. This token have to be specified on the WAF appliance where the certificate is to be installed to be able to decrypt it. Use the WAF appliance CLI to provided the token as the waf_ssl_passphrase.</td>
</tr>
<tr>
<td>chain</td>
<td>(Text) base64 encoded PEM file content of the certificate authority chain certificates.</td>
</tr>
<tr>
<td>owner</td>
<td>(Text) The user for Qualys Cloud Platform who owns this certificate profile.</td>
</tr>
<tr>
<td>owner.id</td>
<td>(Long) The user ID of the certificate profile owner.</td>
</tr>
<tr>
<td>owner.username</td>
<td>(Text) The user name of the certificate profile owner.</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>(Text) The first name of the certificate profile owner.</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>(Text) The last name of the certificate profile owner.</td>
</tr>
<tr>
<td>created</td>
<td>(Date) The date/time when the certificate profile was created.</td>
</tr>
<tr>
<td>createdBy.id</td>
<td>(Long) The user ID who created the certificate profile.</td>
</tr>
<tr>
<td>createdBy.username</td>
<td>(Text) The user name who created the certificate profile.</td>
</tr>
<tr>
<td>createdBy.firstname</td>
<td>(Text) The first name of the user who created the certificate profile.</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>createdBy.lastname</td>
<td>(Text) The last name of the user who created the certificate profile.</td>
</tr>
<tr>
<td>updated</td>
<td>(Date) The date/time when the certificate profile was last updated.</td>
</tr>
<tr>
<td>updatedBy.id</td>
<td>(Long) The user ID who last updated the certificate profile.</td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>(Text) The user name who last updated the certificate profile.</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>(Text) The first name of the user who updated the certificate profile.</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>(Text) The last name of the user who updated the certificate profile.</td>
</tr>
<tr>
<td>tags</td>
<td>(Text) List of tags associated with the certificate profile. Valid action: Update</td>
</tr>
<tr>
<td>webApps.webApp.id</td>
<td>(Long) The ID of the Web Application this certificate profile is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.uuid</td>
<td>(UUID) The UUID of the Web Application this certificate profile is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.name</td>
<td>(Text) The name of the Web Application this certificate profile is associated with.</td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>(Long) The ID of a tag associated with the certificate profile.</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>(Text) The name, defined by a user, of a tag associated with the certificate profile.</td>
</tr>
<tr>
<td>certificateMetadata</td>
<td>(Text) JSON encoded metadata about this certificate.</td>
</tr>
<tr>
<td>chainMetadata</td>
<td>(Text) JSON encoded metadata about the certificate authority chain.</td>
</tr>
</tbody>
</table>
Custom Response Pages API

Use these API functions to manage Custom Response Pages.

Current custom response page count
Get details on a custom response page
Search custom response pages
Create custom response page
Update custom response page
Update custom response pages (bulk)
Delete custom response page
Delete custom response pages (bulk)
Chapter 6 — Custom Response Pages API

Current custom response page count

Returns the total number of custom response pages for WAF in the user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/count/waf/custompage

**Methods allowed:** GET

**Input**
No input elements are available.

**Permissions**
User must have WAF module enabled
User must have "API ACCESS" permission
Asset must be within user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/custompage

**Response**
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/custompage.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
</ServiceResponse>
Get details on a custom response page

Returns details about a specific custom response page for WAF, within the user’s scope. Want to find a custom response page ID to use as input? See Search custom response pages.

**URL:**
https://<baseurl>/qps/rest/2.0/get/waf/custompage/<id>

**Methods allowed:**
GET

**Input**

The element "id" (Integer) is required, where "id" identifies the custom response page ID of interest.

**Permissions**

User must have WAF module enabled  
User must have "API ACCESS" permission  
Asset must be within user's scope

**Example**

**Request:**
url -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/custompage/1001

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/custompage.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomPage>
      <id>1001</id>
      <uuid>11bf0ac8-3bde-4e10-aa75-ce4399378c58</uuid>
      <name><![CDATA[my test page]]></name>
      <description>
```
<![CDATA[description]]>
</description>

<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>

<created>2017-06-08T12:29:40Z</created>

<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>

<updated>2017-06-08T12:29:40Z</updated>

<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>

<body>
  <! [CDATA[
  <!DOCTYPE html>
  <html>
    <body>
      <h1>My Custom Page</h1>
      <p>My custom content</p>
      </body>
  ] ]>
  </body>
</webApps>

<WebApp>
  <id>63098273</id>
  <uuid>01bd1b58-2802-48dd-b5b5-ea1342aa21a</uuid>
  <name><![CDATA[Site 01]]></name>
</WebApp>
</webApps>
Chapter 6 — Custom Response Pages API
Search custom response pages

Finds custom response pages in the user’s account matching the search criteria.

URL: https://<baseurl>/qps/rest/2.0/search/waf/custompage
Methods allowed: POST

Input

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: custom response pages for descriptions of these elements.

<table>
<thead>
<tr>
<th>id (Long)</th>
<th>createdBy.firstname (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>uuid (UUID)</td>
<td>createdBy.lastname (Text)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>updated (Date)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>updatedBy.id (Long)</td>
</tr>
<tr>
<td>body (Text)</td>
<td>updatedBy.username (Text)</td>
</tr>
<tr>
<td>owner.id (Long)</td>
<td>updatedBy.firstname (Text)</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>updatedBy.lastname (Text)</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>webApps.webApp.id (Long)</td>
</tr>
<tr>
<td>owner.lastname (Text)</td>
<td>webApps.webApp.uuid (UUID)</td>
</tr>
<tr>
<td>created (Date)</td>
<td>webApps.webApp.name (Text)</td>
</tr>
<tr>
<td>createdBy.id (Long)</td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td>createdBy.username (Text)</td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

Allowed Operators

Long        EQUALS, NOT EQUALS
            Note: The elements createdBy.id and updatedBy.id only support EQUALS.
UUID        EQUALS, NOT EQUALS
Text         CONTAINS, EQUALS, NOT EQUALS
Date         EQUALS, NOT EQUALS, GREATER, LESSER
Permissions

User must have WAF module enabled
User must have "API ACCESS" permission
Asset must be within user's scope

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/search/waf/custompage"
< file.xml

Note: "file.xml" contains the request POST data.
The request POST data is optional. If you leave it empty all custom response pages in the
user's scope are returned.

Request POST Data:

<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="id" operator="EQUALS">1601</Criteria>
  </filters>
</ServiceRequest>'

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/custompage.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <hasMoreRecords>false</hasMoreRecords>
  <data>
    <CustomPage>
      <id>1601</id>
      <uuid>8ebcf58c-9731-47b3-850e-8e20a2627f91</uuid>
      <name><![CDATA[Custom unroutable]]></name>
    </CustomPage>
  </data>
</ServiceResponse>
<owner>
   <id>354401</id>
   <username>john_doe</username>
   <firstname>John</firstname>
   <lastname>Doe</lastname>
</owner>
<created>2017-07-28T00:19Z</created>
<createdBy>
   <id>354401</id>
   <username>john_doe</username>
   <firstname>John</firstname>
   <lastname>Doe</lastname>
</createdBy>
<updated>2017-07-28T18:31:31Z</updated>
<updatedBy>
   <id>361390</id>
   <username>john_doe</username>
   <firstname>John</firstname>
   <lastname>Doe</lastname>
</updatedBy>
<body>
   <![CDATA[Wrong server has been hit.]]>
</body>
<webApps/>
</CustomPage>
</data>
</ServiceResponse>
Create custom response page

Create a custom response page which you can assign to a web application.

**URL:**
https://<baseurl>/qps/rest/2.0/create/waf/custompage

**Methods allowed:** POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: custom response pages for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>body (Text)</td>
<td>tags</td>
</tr>
<tr>
<td></td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td></td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Create WAF Asset" permission

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/create/waf/custompage"
< file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <CustomPage>
      <name>my API created page 1</name>
      <description>example description</description>
    </CustomPage>
  </data>
</ServiceRequest>
Create custom response page

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomPage>
      <id>1401</id>
      <uuid>494d5aaa-2519-4e58-a6d4-699cfe2154a3</uuid>
      <name><![CDATA[my API created page 1]]></name>
      <description><![CDATA[example description]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </owner>
      <created>2017-06-12T10:00:28Z</created>
    </CustomPage>
  </data>
</ServiceResponse>
```
<firstname>John</firstname>
<lastname>Doe</lastname>

</createdBy>
updated>2017-06-12T10:00:28Z</updated>
updatedBy>
    <id>3988443</id>
    <username>john_doe</username>
    <firstname>John</firstname>
    <lastname>Doe</lastname>
</updatedBy>
body>
<![CDATA[
<!DOCTYPE html>
<html>
<body>
<h1>My Custom Page</h1>
<p>My custom content</p>
</body></html>]]>
</body>
webApps />
</CustomPage>
</data>
</ServiceResponse>
Update custom response page

Update a custom response page in the user’s account. You can update all fields except tag ID and tag name.

**URL:** https://<baseurl>/qps/rest/2.0/update/waf/custompage/<id>

**Methods allowed:** POST

**Input**

The "id" (Long) element is required. This identifies the custom response page you want to update.

Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: custom response pages for descriptions of these elements.

<table>
<thead>
<tr>
<th>name (Text)</th>
<th>body (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>description (Text)</td>
<td>tags</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Update WAF Asset" permission
Asset must be within user’s scope

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary "<ServiceRequest><data><CustomPage><name>my API updated page 1</name></CustomPage></data></ServiceRequest>" < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <CustomPage>
      <name>my API updated page 1</name>
    </CustomPage>
  </data>
</ServiceRequest>
```
Chapter 6 — Custom Response Pages API
Update custom response page

<description>updated description</description>
<body>
  <![CDATA[
    <!DOCTYPE html>
    <html>
      <body>
        <h1>My Updated Custom Page</h1>
        <p>My custom content</p>
      </body>
    </html>
  ]]>
</body>
</CustomPage>
</data>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/custompage.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomPage>
      <id>1401</id>
      <uuid>494d5aaa-2519-4e58-a6d4-699cfe2154a3</uuid>
      <name>
        <![CDATA[my API updated page]]>
      </name>
      <description>
        <![CDATA[updated description]]>
      </description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </owner>
      <created>2017-06-12T10:00:28Z</created>
      <createdBy>
        <id>3988443</id>
      </createdBy>
    </CustomPage>
  </data>
</ServiceResponse>
Chapter 6 — Custom Response Pages API
Update custom response page

<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</createdBy>
<updated>2017-06-12T10:12:59Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<body>
  <![CDATA[
    <!DOCTYPE html>
    <html>
      <body>
        <h1>My Updated Custom Page</h1>
        <p>My custom content</p>
      </body>
    </html>
  ]]>]
</body>
</CustomPage>
</data>
</ServiceResponse>
Update custom response pages (bulk)

Update multiple custom response pages in the user’s account. You can update all fields except tag ID and tag name.

**URL:**  
https://<baseurl>/qps/rest/2.0/update/waf/custompage

**Methods allowed:**  
POST

**Input**

All elements for the search operation are supported. See [Search custom response pages](#).

Allowed input elements for bulk update are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. See [Reference: custom response pages](#) for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Text</td>
</tr>
<tr>
<td>description</td>
<td>Text</td>
</tr>
<tr>
<td>body</td>
<td>Text</td>
</tr>
<tr>
<td>tags</td>
<td></td>
</tr>
</tbody>
</table>

**Allowed Operators**

- Long: EQUALS, NOT EQUALS
- UUID: EQUALS, NOT EQUALS
- Text: CONTAINS, EQUALS, NOT EQUALS
- Date: EQUALS, NOT EQUALS, GREATER, LESSER

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Update WAF Asset" permission
Asset must be within user’s scope
Chapter 6 — Custom Response Pages API
Update custom response pages (bulk)

Example

Request:

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">API</Criteria>
  </filters>
  <data>
    <CustomPage>
      <description>bulk update</description>
    </CustomPage>
  </data>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomPage>
      <id>1401</id>
      <uuid>494d5aaa-2519-4e58-a6d4-699cfe2154a3</uuid>
      <name><![CDATA[my API updated page 1]]></name>
      <description><![CDATA[bulk update]]></description>
      <owner>
        ...
      </owner>
    </CustomPage>
  </data>
</ServiceResponse>
<id>3988443</id>
<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</owner>
<created>2017-06-12T10:00:28Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-06-12T10:29:25Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<body>
  <![CDATA[
    <!DOCTYPE html>
    <html>
      <body>
        <h1>My Updated Custom Page</h1>
        <p>My custom content</p>
      </body>
    </html>
  ]]>
</body>
<webApps/>
</CustomPage>
</data>
</ServiceResponse>
Delete custom response page

Delete a custom response page in user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/custompage/<id>

**Methods allowed:**
POST

**Input**

The "id" (Long) element is required. This identifies the custom response page you want to delete.

**Permissions**

- User must have the WAF module enabled
- User must have "API ACCESS" permission
- User must have "Delete WAF Asset" permission
- Asset must be within user’s scope

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/custompage/1202

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/custompage.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomPage>
      <id>1202</id>
    </CustomPage>
  </data>
</ServiceResponse>
```
Delete custom response pages (bulk)

Delete multiple custom response pages in the user’s account.

**URL:** https://<baseurl>/qps/rest/2.0/delete/waf/custompage

**Methods allowed:** POST

**Input**

All elements for the search operation are supported. See Search custom response pages.

**Permissions**

- User must have the WAF module enabled
- User must have "API ACCESS" permission
- User must have "Delete WAF Asset" permission
- Asset must be within user’s scope

**Example**

**Request**

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" -d-data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/custompage" < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" ?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">API</Criteria>
  </filters>
</ServiceRequest>
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/custompage.xsd">
<responseCode>SUCCESS</responseCode>
<count>1</count>
<data>
  <CustomPage>
    <id>1401</id>
  </CustomPage>
</data>
</ServiceResponse>
### Reference: custom response pages

A reference of all custom response page elements is provided below.

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>(Long) Custom response page identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid</td>
<td>(UUID) Custom response page identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name</td>
<td>(Text) The name of the custom response page as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>description</td>
<td>(Text) The description of the custom response page.</td>
</tr>
<tr>
<td>body</td>
<td>(Text) The body of the custom page (in HTML).</td>
</tr>
<tr>
<td>owner</td>
<td>(Text) The user for Qualys Cloud Platform who owns this custom response page.</td>
</tr>
<tr>
<td>owner.id</td>
<td>(Long) The user ID of the custom response page owner.</td>
</tr>
<tr>
<td>owner.username</td>
<td>(Text) The user name of the custom response page owner.</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>(Text) The first name of the custom response page owner.</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>(Text) The last name of the custom response page owner.</td>
</tr>
<tr>
<td>created</td>
<td>(Date) The date/time when the custom response page was created.</td>
</tr>
<tr>
<td>createdBy.id</td>
<td>(Long) The user ID who created the custom response page.</td>
</tr>
<tr>
<td>createdBy.username</td>
<td>(Text) The user name who created the custom response page.</td>
</tr>
<tr>
<td>createdBy.firstname</td>
<td>(Text) The first name of the user who created the custom response page.</td>
</tr>
<tr>
<td>createdBy.lastname</td>
<td>(Text) The last name of the user who created the custom response page.</td>
</tr>
<tr>
<td>updated</td>
<td>(Date) The date/time when the custom response page was last updated.</td>
</tr>
<tr>
<td>updatedBy.id</td>
<td>(Long) The user ID who last updated the custom response page.</td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>(Text) The user name who last updated the custom response page.</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>(Text) The first name of the user who updated the custom response page.</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>(Text) The last name of the user who updated the custom response page.</td>
</tr>
</tbody>
</table>
## Chapter 6 — Custom Response Pages API

Reference: custom response pages

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>webApps.webApp.id</td>
<td>(Long) The ID of the Web Application this custom response page is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.uuid</td>
<td>(UUID) The UUID of the Web Application this custom response page is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.name</td>
<td>(Text) The name of the Web Application this custom response page is associated with.</td>
</tr>
<tr>
<td>tags</td>
<td>(Text) List of tags associated with the custom response page. Valid action: Update</td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>(Long) The ID of a tag associated with the custom response page.</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>(Text) The name, defined by a user, of a tag associated with the custom response page.</td>
</tr>
</tbody>
</table>
Security Policies API

Use these API functions to manage Security Policies.

- Current security policy count
- Get details on a security policy
- Search security policies
- Create security policy
- Update security policy
- Update security policies (bulk)
- Delete security policy
- Delete security policies (bulk)

Reference: Security Policies
Current security policy count

Returns the total number of security policies for WAF in the user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/count/waf/securitypolicy

**Methods allowed:** GET

**Input**

No input elements are available.

**Permissions**

User must have the WAF module enabled
User must have “API Access” permission
Asset must be within user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/securitypolicy

**Response:**
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/securitypolicy.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>15</count>
</ServiceResponse>
Get details on a security policy

Returns details about a specific security policy for WAF, within the user’s scope. Want to find a security policy ID to use as input? See Search security policies.

**URL:** https://<baseurl>/qps/rest/2.0/get/waf/securitypolicy/<id>

**Methods allowed:** GET

### Input

The element "id" (Integer) is required, where "id" identifies the security policy of interest.

### Permissions

User must have WAF module enabled
User must have "API Access” and "Access WAF module" permission
Asset must be within user's scope

### Example

**Request:**

curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/securitypolicy/33481

**Response:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="http://lqualysapi.qualys.com/qps/xsd/2.0/waf/securitypolicy.xsd">
   <responseCode>SUCCESS</responseCode>
   <count>1</count>
   <data>
      <SecurityPolicy>
         <id>33481</id>
         <uuid>0fded90c-42da-4ae8-b5a5-998562a4990e</uuid>
         <name><![CDATA[Server Security Policy]]></name>
         <description><![CDATA[Security policies for servers]]></description>
      </SecurityPolicy>
   </data>
</ServiceResponse>
```
Get details on a security policy

</description>
<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</owner>
<created>2017-05-14T12:33:20Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</createdBy>
<updated>2017-05-14T12:33:22Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</updatedBy>
<system>0</system>
<applicationSecurity>
  <commandExecution>
    <confidence>HIGH</confidence>
  </commandExecution>
  <crossSiteScripting>
    <confidence>MEDIUM</confidence>
  </crossSiteScripting>
  <directoryTraversal>
    <confidence>DISABLED</confidence>
  </directoryTraversal>
  <formatStringAttacks>
    <confidence>LOW</confidence>
  </formatStringAttacks>
  <informationLeakage>
    <value>46</value>
  </informationLeakage>
  <ldapInjection>
    <value>47</value>
  </ldapInjection>
</applicationSecurity>
<value>48</value>
</lfiAttacks>
<pathTraversal>
  <value>49</value>
</pathTraversal>
<rfiAttacks>
  <value>51</value>
</rfiAttacks>
<sourceCodeDisclosure>
  <value>52</value>
</sourceCodeDisclosure>
<sqlInjection>
  <value>53</value>
</sqlInjection>
<ssiInjection>
  <value>54</value>
</ssiInjection>
<xpathInjection>
  <value>55</value>
</xpathInjection>
</applicationSecurity>
<threatLevel>
  <loggingThreshold>35</loggingThreshold>
  <blockingThreshold>65</blockingThreshold>
</threatLevel>
</SecurityPolicy>
</data>
</ServiceResponse>
Search security policies

Finds security policies in the user’s account matching the search criteria.

**URL:**
https://<baseurl>/qps/rest/2.0/search/waf/securitypolicy

**Methods allowed:**
POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: Security Policies for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Integer)</td>
<td>createdBy.firstname (Text)</td>
</tr>
<tr>
<td>uuid (Integer)</td>
<td>createdBy.lastname (Text)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>updated (date)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>updatedBy.id (Long)</td>
</tr>
<tr>
<td>system (Integer)</td>
<td>updatedBy.username (Text)</td>
</tr>
<tr>
<td>owner.id</td>
<td>updatedBy.firstname (Text)</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>updatedBy.lastname (Text)</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>webApps.webApp.id (Long)</td>
</tr>
<tr>
<td>owner.lastname (Text)</td>
<td>webApps.webApp.uuid (UUID)</td>
</tr>
<tr>
<td>created (date)</td>
<td>webApps.webApp.name (Text)</td>
</tr>
<tr>
<td>createdBy.id (Long)</td>
<td>tags.tag.id</td>
</tr>
<tr>
<td>createdBy.username (Text)</td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

**Allowed Operators**

<table>
<thead>
<tr>
<th>Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long</td>
<td>EQUALS</td>
</tr>
<tr>
<td>Integer</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Text</td>
<td>CONTAINS, EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Date</td>
<td>EQUALS, NOT EQUALS, GREATER, LESSER</td>
</tr>
<tr>
<td>Keyword</td>
<td>EQUALS, NOT EQUALS, IN</td>
</tr>
<tr>
<td>Boolean</td>
<td>(true/false) EQUALS, NOT EQUALS</td>
</tr>
</tbody>
</table>
Permissions

User must have WAF module enabled
User must have "API Access" permission
Asset must be within user’s scope

Example

Request:
```
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/search/waf/securitypolicy < file.xml
```

Note: "file.xml" contains the request POST data.
The request POST data is optional. If you leave it empty all security policies in the user’s scope are returned.

Request POST Data:
```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="system" operator="EQUALS">1</Criteria>
  </filters>
</ServiceRequest>
```

Response
```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/securitypolicy.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>2</count>
  <hasMoreRecords>false</hasMoreRecords>
  <data>
    <SecurityPolicy>
      <id>30681</id>
      <uuid>da1da5e5-7c2b-4a64-853b-651e41f2ed6d</uuid>
      <name><![CDATA[Pass-through]]></name>
    </SecurityPolicy>
  </data>
</ServiceResponse>
```
<owner>
  <id>3988442</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</owner>
<created>2017-03-27T13:18:47Z</created>
<createdBy>
  <id>3988442</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</createdBy>
<system>1</system>
<applicationSecurity>
  <commandExecution>
    <confidence>DISABLED</confidence>
  </commandExecution>
  <crossSiteScripting>
    <confidence>DISABLED</confidence>
  </crossSiteScripting>
  <directoryTraversal>
    <confidence>DISABLED</confidence>
  </directoryTraversal>
  <formatStringAttacks>
    <confidence>DISABLED</confidence>
  </formatStringAttacks>
  <informationLeakage>
    <confidence>DISABLED</confidence>
  </informationLeakage>
  <ldapInjection>
    <confidence>DISABLED</confidence>
  </ldapInjection>
  <lfiAttacks>
    <confidence>DISABLED</confidence>
  </lfiAttacks>
  <pathTraversal>
    <confidence>DISABLED</confidence>
  </pathTraversal>
  <rfiAttacks>
    <confidence>DISABLED</confidence>
  </rfiAttacks>
</applicationSecurity>
<confidence>DISABLED</confidence>
</rfiAttacks>
<sourceCodeDisclosure>
  <confidence>DISABLED</confidence>
</sourceCodeDisclosure>
<sqlInjection>
  <confidence>DISABLED</confidence>
</sqlInjection>
<ssiInjection>
  <confidence>DISABLED</confidence>
</ssiInjection>
<xpathInjection>
  <confidence>DISABLED</confidence>
</xpathInjection>
</applicationSecurity>
<threatLevel>
  <loggingThreshold>50</loggingThreshold>
  <blockingThreshold>50</blockingThreshold>
</threatLevel>
</SecurityPolicy>
<SecurityPolicy>
  <id>30682</id>
  <uuid>005e0d28-026c-49cc-9f40-87d5accac97f</uuid>
  <name>
    <![CDATA[Standard Policy]]>
  </name>
  <owner>
    <id>3988442</id>
    <username>john_doe</username>
    <firstName><John></firstName>
    <lastName><Doe></lastName>
  </owner>
  <created>2017-03-27T13:18:47Z</created>
  <createdBy>
    <id>3988442</id>
    <username>john_doe</username>
    <firstName><John></firstName>
    <lastName><Doe></lastName>
  </createdBy>
  <system>1</system>
</SecurityPolicy>
<applicationSecurity>
  <commandExecution>
    <confidence>MEDIUM</confidence>
  </commandExecution>
  <crossSiteScripting>
    <confidence>MEDIUM</confidence>
  </crossSiteScripting>
  <directoryTraversal>
    <confidence>MEDIUM</confidence>
  </directoryTraversal>
  ...
  <rfiAttacks>
    <confidence>MEDIUM</confidence>
  </rfiAttacks>
  <sourceCodeDisclosure>
    <confidence>MEDIUM</confidence>
  </sourceCodeDisclosure>
  <sqlInjection>
    <confidence>MEDIUM</confidence>
  </sqlInjection>
  <ssiInjection>
    <confidence>MEDIUM</confidence>
  </ssiInjection>
  <xpathInjection>
    <confidence>MEDIUM</confidence>
  </xpathInjection>
</applicationSecurity>

<threatLevel>
  <loggingThreshold>25</loggingThreshold>
  <blockingThreshold>75</blockingThreshold>
</threatLevel>

</SecurityPolicy>

</data>
</ServiceResponse>
Create security policy

Create a new security policy with given parameters.

**URL:**  
https://<baseurl>/qps/rest/2.0/create/waf/securitypolicy

**Methods allowed:**  
POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: Security Policies for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td></td>
<td>applicationSecurity (Keyword)</td>
</tr>
<tr>
<td></td>
<td>threatLevel.loggingThreshold (Integer)</td>
</tr>
<tr>
<td></td>
<td>threatLevel.blockingThreshold (Integer)</td>
</tr>
<tr>
<td>tags</td>
<td>tags.tag.id (Integer)</td>
</tr>
<tr>
<td></td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled  
User must have "API Access" and "Create Policy" permission.

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"  
--data-binary @-  
https://qualysapi.qualys.com/qps/rest/2.0/create/waf/securitypolicy < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
<data>
```

Qualys Web Application Firewall API
<SecurityPolicy>
  <name>Server Security Policy</name>
  <description>Security Policy for servers</description>
  <applicationSecurity>
    <commandExecution>
      <confidence>HIGH</confidence>
    </commandExecution>
    <crossSiteScripting/>
    <directoryTraversal>
      <confidence>DISABLED</confidence>
    </directoryTraversal>
    <formatStringAttacks>
      <confidence>LOW</confidence>
    </formatStringAttacks>
    <informationLeakage>
      <value>46</value>
    </informationLeakage>
    <ldapInjection>
      <value>47</value>
    </ldapInjection>
    <lfiAttacks>
      <value>48</value>
    </lfiAttacks>
    <pathTraversal>
      <value>49</value>
    </pathTraversal>
    <rfiAttacks>
      <value>51</value>
    </rfiAttacks>
    <sourceCodeDisclosure>
      <value>52</value>
    </sourceCodeDisclosure>
    <sqlInjection>
      <value>53</value>
    </sqlInjection>
    <ssiInjection>
      <value>54</value>
    </ssiInjection>
    <xpathInjection>
      <value>55</value>
    </xpathInjection>
  </applicationSecurity>
</SecurityPolicy>
Chapter 7 — Security Policies API

Create security policy

```xml
<applicationSecurity>
  <threatLevel>
    <loggingThreshold>35</loggingThreshold>
    <blockingThreshold>65</blockingThreshold>
  </threatLevel>
</SecurityPolicy>
</data>
</ServiceRequest>

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="https://qualysapi.qualys.com/qps/xsd/2.0/waf/securitypolicy.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <SecurityPolicy>
      <id>33481</id>
      <uuid>0fded90c-42da-4ae8-b5a5-998562a4990e</uuid>
      <name><![CDATA[Server Security Policy]]></name>
      <description><![CDATA[Security Policy for servers]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </owner>
      <created>2017-05-14T12:33:20Z</created>
      <createdBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </createdBy>
      <updated>2017-05-14T12:33:22Z</updated>
    </SecurityPolicy>
  </data>
</ServiceResponse>
```
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</updatedBy>
<system>0</system>
<applicationSecurity>
  <commandExecution>
    <confidence>HIGH</confidence>
  </commandExecution>
  <crossSiteScripting>
    <confidence>MEDIUM</confidence>
  </crossSiteScripting>
  <directoryTraversal>
    <confidence>DISABLED</confidence>
  </directoryTraversal>
  <formatStringAttacks>
    <confidence>LOW</confidence>
  </formatStringAttacks>
  <informationLeakage>
    <value>46</value>
  </informationLeakage>
  <ldapInjection>
    <value>47</value>
  </ldapInjection>
  <lfiAttacks>
    <value>48</value>
  </lfiAttacks>
  <pathTraversal>
    <value>49</value>
  </pathTraversal>
  <rfiAttacks>
    <value>51</value>
  </rfiAttacks>
  <sourceCodeDisclosure>
    <value>52</value>
  </sourceCodeDisclosure>
  <sqlInjection>
    <value>53</value>
  </sqlInjection>
</applicationSecurity>
<ssiInjection>
  <value>54</value>
</ssiInjection>
<xpathInjection>
  <value>55</value>
</xpathInjection>
</applicationSecurity>
<threatLevel>
  <loggingThreshold>35</loggingThreshold>
  <blockingThreshold>65</blockingThreshold>
</threatLevel>
</SecurityPolicy>
</data>
</ServiceResponse>
Update security policy

Update a security policy identified by its identifier with given parameters. You can update all fields except tag ID and tag name.

**URL:**
https://<baseurl>/qps/rest/2.0/update/waf/securitypolicy/<id>

**Methods allowed:** POST

**Input**

The "id" (Long) element is required. This identifies the security policy you want to update.

Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: Security Policies for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Integer)</td>
<td>threatLevel.loggingThreshold (Integer)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>threatLevel.blockingThreshold (Integer)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>tags</td>
</tr>
<tr>
<td>applicationSecurity</td>
<td>(Keyword)</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled  
User must have "API Access" and "Update Policy" permission  
Asset must be within user's scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"  
--data-binary @-  
https://qualysapi.qualys.com/qps/rest/2.0/update/waf/securitypolicy < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <SecurityPolicy>
```

164 Qualys Web Application Firewall API
<description>test policy updated</description>
<applicationSecurity>
  <commandExecution>
    <value>31</value>
  </commandExecution>
  <crossSiteScripting>
    <value>32</value>
  </crossSiteScripting>
  <directoryTraversal>
    <value>33</value>
  </directoryTraversal>
  <formatStringAttacks>
    <value>34</value>
  </formatStringAttacks>
  <information Leakage>
    <value>35</value>
  </information Leakage>
  <ldapInjection>
    <value>36</value>
  </ldapInjection>
  <lfiAttacks>
    <value>37</value>
  </lfiAttacks>
  <pathTraversal>
    <value>38</value>
  </pathTraversal>
  <rfiAttacks>
    <value>39</value>
  </rfiAttacks>
  <sourceCodeDisclosure>
    <value>40</value>
  </sourceCodeDisclosure>
  <sqlInjection>
    <value>41</value>
  </sqlInjection>
  <ssiInjection>
    <value>42</value>
  </ssiInjection>
  <xpathInjection>
    <value>43</value>
  </xpathInjection>
</applicationSecurity>
Chapter 7 — Security Policies API
Update security policy

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/securitypolicy.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <SecurityPolicy>
      <id>33482</id>
      <uuid>68f06d15-9763-4df7-83e1-fbc9e3cdea4d</uuid>
      <name>
        <![CDATA[my security policy]]>
      </name>
      <description>
        <![CDATA[test policy updated]]>
      </description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </owner>
      <created>2017-05-14T13:07:27Z</created>
      <createdBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </createdBy>
      <updated>2017-05-14T13:10:44Z</updated>
    </SecurityPolicy>
  </data>
</ServiceResponse>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</updatedBy>
<system>0</system>
<applicationSecurity>
  <commandExecution>
    <value>31</value>
  </commandExecution>
  <crossSiteScripting>
    <value>32</value>
  </crossSiteScripting>
  <directoryTraversal>
    <value>33</value>
  </directoryTraversal>
  <formatStringAttacks>
    <value>34</value>
  </formatStringAttacks>
  <informationLeakage>
    <value>35</value>
  </informationLeakage>
  <ldapInjection>
    <value>36</value>
  </ldapInjection>
  <lfiAttacks>
    <value>37</value>
  </lfiAttacks>
  <pathTraversal>
    <value>38</value>
  </pathTraversal>
  <rfiAttacks>
    <value>39</value>
  </rfiAttacks>
  <sourceCodeDisclosure>
    <value>40</value>
  </sourceCodeDisclosure>
  <sqlInjection>
    <value>41</value>
  </sqlInjection>
</applicationSecurity>
<ssiInjection>
  <value>42</value>
</ssiInjection>
<xpathInjection>
  <value>43</value>
</xpathInjection>
</applicationSecurity>
<threatLevel>
  <loggingThreshold>36</loggingThreshold>
  <blockingThreshold>64</blockingThreshold>
</threatLevel>
</SecurityPolicy>
</data>
</ServiceResponse>
Chapter 7 — Security Policies API

Update security policies (bulk)

Update security policies identified by a search with given parameters. You can update all fields except tag ID and tag name.

**URL:**  
https://<baseurl>/qps/rest/2.0/update/waf/securitypolicy

**Methods allowed:**  
POST

**Input**

All elements for the search operation are supported. See Search security policies.

The "id" (Integer) element is required where "id" identifies the security policy. Additional elements are optional. See Reference: Security Policies for descriptions of all elements.

<table>
<thead>
<tr>
<th>id (Integer)</th>
<th>threatLevel.loggingThreshold (Integer)</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>threatLevel.blockingThreshold (Integer)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>tags</td>
</tr>
<tr>
<td>applicationSecurity (Keyword)</td>
<td></td>
</tr>
</tbody>
</table>

**Allowed Operators**

- **Integer:** EQUALS, NOT EQUALS, GREATER, LESSER, IN
- **Text:** CONTAINS, EQUALS, NOT EQUALS
- **Date:** EQUALS, NOT EQUALS, GREATER, LESSER
- **Keyword:** EQUALS, NOT EQUALS, IN
- **Boolean:** (true/false) EQUALS, NOT EQUALS

**Permissions**

User must have the WAF module enabled
User must have "API Access" and "Update Policy" permission
Asset must be within user's scope
Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/update/waf/securitypolicy < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="system" operator="EQUALS">0</Criteria>
  </filters>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/securitypolicy.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>2</count>
  <data>
    <SecurityPolicy>
      <id>31881</id>
      <uuid>d9c7aba3-5139-4553-a105-5e4a94eedd6d</uuid>
      <name>
        <![CDATA[Security Policy One]]>
      </name>
      <description>
        <![CDATA[Updating multiple security policies]]>
      </description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </owner>
    </SecurityPolicy>
  </data>
</ServiceResponse>
<created>2017-04-15T12:41:09Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</createdBy>
<updated>2017-05-14T13:34:22Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</updatedBy>
<system>0</system>
<applicationSecurity>
  <commandExecution>
    <confidence>MEDIUM</confidence>
  </commandExecution>
  <crossSiteScripting>
    <value>71</value>
  </crossSiteScripting>
  <directoryTraversal>
    <confidence>DISABLED</confidence>
  </directoryTraversal>
  <formatStringAttacks>
    <confidence>DISABLED</confidence>
  </formatStringAttacks>
  <informationLeakage>
    <value>36</value>
  </informationLeakage>
  <ldapInjection>
    <confidence>DISABLED</confidence>
  </ldapInjection>
  <lfiAttacks>
    <confidence>MEDIUM</confidence>
  </lfiAttacks>
  <pathTraversal>
    <confidence>DISABLED</confidence>
  </pathTraversal>
  <rfiAttacks>
<confidence>MEDIUM</confidence>
</rfiAttacks>
<sourceCodeDisclosure>
  <confidence>MEDIUM</confidence>
</sourceCodeDisclosure>
<sqlInjection>
  <confidence>MEDIUM</confidence>
</sqlInjection>
<ssiInjection>
  <confidence>MEDIUM</confidence>
</ssiInjection>
<xpathInjection>
  <confidence>MEDIUM</confidence>
</xpathInjection>
</applicationSecurity>
<threatLevel>
  <loggingThreshold>23</loggingThreshold>
  <blockingThreshold>77</blockingThreshold>
</threatLevel>
</SecurityPolicy>
<SecurityPolicy>
  <id>33482</id>
  <uuid>68f06d15-9763-4df7-83e1-fbc9e3cdea4d</uuid>
  <name> <![CDATA[Security Policy Two]]>
</name>
  <description> <![CDATA[Updating multiple security policies]]>
</description>
  <owner>
   <id>3988443</id>
   <username>john_doe</username>
   <firstName>John</firstName>
   <lastName>Doe</lastName>
  </owner>
  <created>2017-05-14T13:07:27Z</created>
  <createdBy>
   <id>3988443</id>
   <username>john_doe</username>
   <firstName>John</firstName>
   <lastName>Doe</lastName>
  </createdBy>
</SecurityPolicy>
<applicationSecurity>
  <commandExecution>
    <value>31</value>
  </commandExecution>
  <crossSiteScripting>
    <value>32</value>
  </crossSiteScripting>
  <directoryTraversal>
    <value>33</value>
  </directoryTraversal>
  <formatStringAttacks>
    <value>34</value>
  </formatStringAttacks>
  <informationLeakage>
    <value>35</value>
  </informationLeakage>
  <ldapInjection>
    <value>36</value>
  </ldapInjection>
  <lfiAttacks>
    <value>37</value>
  </lfiAttacks>
  <pathTraversal>
    <value>38</value>
  </pathTraversal>
  <rfiAttacks>
    <value>39</value>
  </rfiAttacks>
  <sourceCodeDisclosure>
    <value>40</value>
  </sourceCodeDisclosure>
  <sqlInjection>
<value>41</value>
</sqlInjection>
<ssiInjection>
  <value>42</value>
</ssiInjection>
<xpathInjection>
  <value>43</value>
</xpathInjection>
</applicationSecurity>
<threatLevel>
  <loggingThreshold>23</loggingThreshold>
  <blockingThreshold>77</blockingThreshold>
</threatLevel>
</SecurityPolicy>
</data>
</ServiceResponse>
Delete security policy

Delete an existing security policy identified by its identifier.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/securitypolicy/<id>

**Methods allowed:**
POST

**Input**

The element "id" (Integer) is required, where "id" identifies the security policy of interest.

**Permissions**

User must have the WAF module enabled
User must have "API Access" and "Delete Policy" permission
Asset must be within user's scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -x "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/securitypolicy/32882

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/securitypolicy.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <SecurityPolicy>
      <id>32882</id>
    </SecurityPolicy>
  </data>
</ServiceResponse>
```
Chapter 7 — Security Policies API

Delete security policies (bulk)

Delete security policies identified by a search with given parameters.

**URL:** https://<baseurl>/qps/rest/2.0/delete/waf/securitypolicy/

**Methods allowed:** POST

**Input**

All elements for the search operation are supported. See [Search security policies](#).

**Permissions**

User must have the WAF module enabled
User must have "API Access" and "Delete Policy" permission
Asset must be within user's scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/securitypolicy < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">security policy</Criteria>
  </filters>
</ServiceRequest>
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/securitypolicy.xsd">
```
<responseCode>SUCCESS</responseCode>
<count>4</count>
<data>
  <SecurityPolicy>
    <id>32681</id>
  </SecurityPolicy>
  <SecurityPolicy>
    <id>32881</id>
  </SecurityPolicy>
  <SecurityPolicy>
    <id>33083</id>
  </SecurityPolicy>
  <SecurityPolicy>
    <id>33481</id>
  </SecurityPolicy>
</data>
</ServiceResponse>
## Reference: Security Policies

A reference of all security policy elements is provided below.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>(Long) Security policy identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid</td>
<td>(UUID) Security policy identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name</td>
<td>(Text) The name of the security policy as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>description</td>
<td>(Text) A description of the security policy.</td>
</tr>
<tr>
<td>system</td>
<td>(Integer) Profile type: 0 if custom, 1 if system or 2 if template</td>
</tr>
<tr>
<td>applicationSecurity</td>
<td>(Keyword: DISABLED, LOW, MEDIUM or HIGH) Used to specify the event confidence by name or by value (20-80) for all these event types: commandExecution, crossSiteScripting, directoryTraversal, formatStringAttacks, informationLeakage, ldapInjection, lfiAttacks, pathTraversal, rfiAttacks, sourceCodeDisclosure, sqlInjection, ssiInjection, xpathInjection</td>
</tr>
<tr>
<td>threatLevel.loggingThreshold</td>
<td>(Integer) Events with a confidence higher or equal than the specified value will be logged.</td>
</tr>
<tr>
<td>threatLevel.blockingThreshold</td>
<td>(Integer) Events with a confidence higher or equal than the specified value will be blocked.</td>
</tr>
<tr>
<td>owner</td>
<td>(Text) The user for Qualys Cloud Platform who owns this security policy.</td>
</tr>
<tr>
<td>owner.id</td>
<td>(Long) The user ID of the security policy owner.</td>
</tr>
<tr>
<td>owner.username</td>
<td>(Text) The user name of the security policy owner.</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>(Text) The first name of the security policy owner.</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>(Text) The last name of the security policy owner.</td>
</tr>
<tr>
<td>created</td>
<td>(Date) The date/time when the security policy was created.</td>
</tr>
<tr>
<td>Element</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>createdBy.id</td>
<td>(Long) The user ID who created the security policy.</td>
</tr>
<tr>
<td>createdBy.username</td>
<td>(Text) The user name who created the security policy.</td>
</tr>
<tr>
<td>createdBy.firstname</td>
<td>(Text) The first name of the user who created the security policy.</td>
</tr>
<tr>
<td>createdBy.lastname</td>
<td>(Text) The last name of the user who created the security policy.</td>
</tr>
<tr>
<td>updated</td>
<td>(Date) The date/time when the security policy was last updated.</td>
</tr>
<tr>
<td>updatedBy.id</td>
<td>(Long) The user ID who last updated the security policy.</td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>(Text) The user name who last updated the security policy.</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>(Text) The first name of the user who updated the security policy.</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>(Text) The last name of the user who updated the security policy.</td>
</tr>
<tr>
<td>webApps.webApp.id</td>
<td>(Long) The ID of the Web Application this security policy is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.uuid</td>
<td>(UUID) The UUID of the Web Application this security policy is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.name</td>
<td>(Text) The name of the Web Application this security policy is associated with.</td>
</tr>
<tr>
<td>tags</td>
<td>(Text) List of tags (identifier and name).</td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>(Long) The ID of a tag associated with the security policy.</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>(Text) The name, defined by a user, of a tag associated with the security policy.</td>
</tr>
</tbody>
</table>
HTTP Profiles API

Use these API functions to manage HTTP Profiles.

- Current HTTP Profile count
- Get details on an HTTP Profile
- Search HTTP Profiles
- Create HTTP Profile
- Update HTTP Profile
- Update HTTP Profiles (bulk)
- Delete HTTP Profile
- Delete HTTP Profiles (bulk)

Reference: HTTP Profile
Current HTTP Profile count

Returns the total number of HTTP profile count for WAF in the user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/count/waf/httpprofile

**Methods allowed:** GET

**Input**

No input elements are available.

**Permissions**

User must have the WAF module enabled
User must have "API Access" permission
Asset must be within user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/httpprofile

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/httpprofile.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>4</count>
</ServiceResponse>
```
Get details on an HTTP Profile

Returns details about a specific HTTP profile for WAF, within the user’s scope. Want to find a HTTP profile ID to use as input? See Search HTTP Profiles.

**URL:** https://<baseurl>/qps/rest/2.0/get/waf/httpprofile/<id>

**Methods allowed:** GET

**Input**

The element "id" (Integer) is required, where "id" identifies the HTTP profile of interest. The associated data type for each element appears in parentheses. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: HTTP Profile for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Integer)</td>
<td></td>
</tr>
<tr>
<td>uuid (Integer)</td>
<td></td>
</tr>
<tr>
<td>name (Text)</td>
<td></td>
</tr>
<tr>
<td>description (Text)</td>
<td></td>
</tr>
<tr>
<td>system (Boolean)</td>
<td></td>
</tr>
<tr>
<td>urls.string (Text)</td>
<td></td>
</tr>
<tr>
<td>requestMethod (Keyword)</td>
<td>owner.id</td>
</tr>
<tr>
<td>requestHeader (Keyword)</td>
<td>owner.username (Text)</td>
</tr>
<tr>
<td>requestContentType (Keyword)</td>
<td>owner.firstname (Text)</td>
</tr>
<tr>
<td>detectProtocolAnomalies (Boolean)</td>
<td>owner.lastname (Text)</td>
</tr>
<tr>
<td>webServiceProtection.xmlParsing. enabled (Boolean)</td>
<td>created (date)</td>
</tr>
<tr>
<td>webServiceProtection.xmlParsing. size (Integer)</td>
<td>createdBy.id</td>
</tr>
<tr>
<td>webServiceProtection.xmlParsing. items (Integer)</td>
<td>createdBy.username (Text)</td>
</tr>
<tr>
<td>webServiceProtection.xmlParsing. level (Integer)</td>
<td>createdBy.firstname (Text)</td>
</tr>
<tr>
<td>webServiceProtection.jsonParsing. enabled (Boolean)</td>
<td>createdBy.lastname (Text)</td>
</tr>
</tbody>
</table>
Chapter 8 — HTTP Profiles API

Get details on an HTTP Profile

Permissions

User must have WAF module enabled
User must have "API Access" permission
Asset must be within user's scope

Example

Request:

curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/httpprofile/4401

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd

webServiceProtection.jsonParsing.
size (Integer) | updated (date)
--- | ---
webServiceProtection.jsonParsing.
items (Integer) | updatedBy.id
webServiceProtection.jsonParsing.
level (Integer) | updatedBy.username (Text)
serverCloaking | updatedBy.firstname (Text)
suppressSensitiveHeaders | tags
onErrorMessages (Keyword) | tags.tag.id (Integer)
onSensitiveFileTypes (Keyword) | tags.tag.name (Text)
onSensitiveFileExtensions (Keyword) | updatedBy.lastname (Text)
cookieProtection

Allowed Operators

<table>
<thead>
<tr>
<th>Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>EQUALS, NOT_EQUALS,</td>
</tr>
<tr>
<td>Text</td>
<td>CONTAINS, EQUALS, NOT_EQUALS,</td>
</tr>
<tr>
<td>Date</td>
<td>EQUALS, NOT_EQUALS, GREATER,</td>
</tr>
<tr>
<td>Keyword</td>
<td>EQUALS, NOT_EQUALS, IN</td>
</tr>
<tr>
<td>Boolean</td>
<td>(true/false) EQUALS, NOT_EQUALS</td>
</tr>
</tbody>
</table>

Qualys Web Application Firewall API 183
Get details on an HTTP Profile

```xml
<responseCode>SUCCESS</responseCode>
<count>1</count>
<data>
  <HTTPProfile>
    <id>4401</id>
    <uuid>4ffd8f5e-529d-4a38-9cac-7fb805962a18</uuid>
    <name><![CDATA[My HTTP Profile]]></name>
    <description><![CDATA[My first HTTP profile]]></description>
    <owner>
      <id>3988443</id>
      <username>john_doe</username>
      <firstName>John</firstName>
      <lastName>Doe</lastName>
    </owner>
    <created>2017-04-26T15:25:26Z</created>
    <createdBy>
      <id>3988443</id>
      <username>john_doe</username>
      <firstName>John</(firstName>
      <lastName>Doe</lastName>
    </createdBy>
    <updated>2017-04-26T15:25:26Z</updated>
    <updatedBy>
      <id>3988443</id>
      <username>john_doe</username>
      <firstName>John</(firstName>
      <lastName>Doe</lastName>
    </updatedBy>
    <system>false</system>
    <requestMethod/>
    <requestHeader>
      <detectInvalid>false</detectInvalid>
      <detectRepeated>false</detectRepeated>
      <detectChunked>false</detectChunked>
    </requestHeader>
    <requestContentType/>
    <detectProtocolAnomalies>false</detectProtocolAnomalies>
  </HTTPProfile>
</data>
```
<webServiceProtection>
  <xmlParsing>
    <enabled>true</enabled>
    <size>500000</size>
    <items>50000</items>
    <level>64</level>
  </xmlParsing>
  <jsonParsing>
    <enabled>true</enabled>
    <size>600000</size>
    <items>10000</items>
    <level>32</level>
  </jsonParsing>
</webServiceProtection>
<serverCloaking>
  <enabled>false</enabled>
</serverCloaking>
<suppressSensitiveHeaders>true</suppressSensitiveHeaders>
  <onErrorMessages>BLOCK</onErrorMessages>
  <onSensitiveFileTypes>BLOCK</onSensitiveFileTypes>
  <onSensitiveFileExtensions>BLOCK</onSensitiveFileExtensions>
  <cookieProtection>
    <type>NONE</type>
  </cookieProtection>
  <discourageContentTypeSniffing>false</discourageContentTypeSniffing>
    <forceDefaultContentType>
      <enabled>false</enabled>
    </forceDefaultContentType>
    <forceDefaultCharacterEncoding>
      <type>NONE</type>
    </forceDefaultCharacterEncoding>
    <contentSecurityPolicyHeader>
      <enabled>false</enabled>
    </contentSecurityPolicyHeader>
    <discourageClickjacking>SAME_ORIGIN_FRAMING</discourageClickjacking>
    <browserXSSProtection>ENABLE_WITH_BLOCKING</browserXSSProtection>
</HTTPProfile>
</data>
</ServiceResponse>
Search HTTP Profiles

Finds HTTP profiles in the user’s account matching the search criteria.

URL: https://<baseurl>/qps/rest/2.0/search/waf/httpprofile
Methods allowed: POST

Input

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: HTTP Profile for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id (Integer)</td>
<td>createdBy.firstname (Text)</td>
</tr>
<tr>
<td>uuid (Integer)</td>
<td>createdBy.lastname (Text)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>updated (date)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>updatedBy.id (Long)</td>
</tr>
<tr>
<td>system (Boolean)</td>
<td>updatedBy.username (Text)</td>
</tr>
<tr>
<td>owner.id</td>
<td>updatedBy.firstname (Text)</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>updatedBy.lastname (Text)</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>webApps.webApp.id (Long)</td>
</tr>
<tr>
<td>owner.lastname (Text)</td>
<td>webApps.webApp.uuid (UUID)</td>
</tr>
<tr>
<td>created (date)</td>
<td>webApps.webApp.name (Text)</td>
</tr>
<tr>
<td>createdBy.id (Long)</td>
<td>tags.tag.id</td>
</tr>
<tr>
<td>createdBy.username (Text)</td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

Allowed Operators

<table>
<thead>
<tr>
<th>Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long</td>
<td>equals, not equals</td>
</tr>
<tr>
<td>Integer</td>
<td>equals, not equals, contains, equals, not equals</td>
</tr>
<tr>
<td>Text</td>
<td>equals, not equals, contains, equals, not equals</td>
</tr>
<tr>
<td>Date</td>
<td>equals, not equals, greater, lesser</td>
</tr>
<tr>
<td>Keyword</td>
<td>equals, not equals, in</td>
</tr>
<tr>
<td>Boolean</td>
<td>(true/false) equals, not equals</td>
</tr>
</tbody>
</table>
Permissions

User must have WAF module enabled
User must have "API Access" permission
Asset must be within user’s scope

Example

**Request**:

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/search/waf/httpprofile <
file.xml

Note: "file.xml" contains the request POST data.

The request POST data is optional. If you leave it empty all HTTP profiles in the user’s
scope are returned.

**Request POST Data**:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="owner.id" operator="EQUALS">2501190</Criteria>
  </filters>
</ServiceRequest>
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/httpprofile.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <hasMoreRecords>false</hasMoreRecords>
  <data>
    <HTTPProfile>
      <id>11801</id>
      <uuid>7e3c59a8-0dd1-4483-a136-22f2e8498a84</uuid>
      <name> <![CDATA[Standard Protocol]]>
    </HTTPProfile>
  </data>
</ServiceResponse>
```
<name/>
<owner>
  <id>2501190</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</owner>
<created>2017-03-01T22:22:28Z</created>
<createdBy>
  <id>2501190</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</createdBy>
<updated>2017-03-01T22:22:28Z</updated>
<system>true</system>
:requestMethod>
  <allowAll>
    <detectInvalid>false</detectInvalid>
    <detectTraceTrack>false</detectTraceTrack>
  </allowAll>
</requestMethod>
:requestHeader>
  <detectInvalid>false</detectInvalid>
  <detectRepeated>false</detectRepeated>
  <detectChunked>false</detectChunked>
</requestHeader>
:requestContentType>
  <allowAll>
    <detectFileUploads>false</detectFileUploads>
  </allowAll>
</requestContentType>
<detectProtocolAnomalies>false</detectProtocolAnomalies>
<webServiceProtection>
  <xmlParsing>
    <enabled>true</enabled>
    <size>500000</size>
    <items>50000</items>
    <level>64</level>
  </xmlParsing>
</webServiceProtection>
<jsonParsing>
  <enabled>true</enabled>
  <size>600000</size>
  <items>10000</items>
  <level>32</level>
</jsonParsing>
</webServiceProtection>
<serverCloaking>
  <enabled>false</enabled>
</serverCloaking>

<suppressSensitiveHeaders>false</suppressSensitiveHeaders>
<onErrorMessages>LOG</onErrorMessages>
<onSensitiveFileTypes>LOG</onSensitiveFileTypes>

<onSensitiveFileExtensions>LOG</onSensitiveFileExtensions>
<cookieProtection>
  <type>NONE</type>
</cookieProtection>

<discourageContentTypeSniffing>false</discourageContentTypeSniffing>
<forceDefaultContentType>
  <enabled>false</enabled>
</forceDefaultContentType>
<forceDefaultCharacterEncoding>
  <type>NONE</type>
</forceDefaultCharacterEncoding>
<contentSecurityPolicyHeader>
  <enabled>false</enabled>
</contentSecurityPolicyHeader>
<discourageClickjacking>NONE</discourageClickjacking>
<browserXSSProtection>DABLE</browserXSSProtection>
</HTTPProfile>
</data>
</ServiceResponse>
Create HTTP Profile

Create a new HTTP profile with given parameters.

**URL:**
https://<baseurl>/qps/rest/2.0/create/waf/httpprofile

**Methods allowed:**
POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: HTTP Profile for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>requestMethod.allowAll or requestMethod.denyAll (one of allowAll or denyAll is required)</td>
<td>requestMethod.allowAll.detectInvalid (Boolean)</td>
</tr>
<tr>
<td>requestHeader</td>
<td>requestMethod.allowAll.detectTraceTrack (Boolean)</td>
</tr>
<tr>
<td>requestContentType.allowAll or requestContentType.denyAll (one of allowAll or denyAll is required)</td>
<td>requestHeader.detectInvalid (Boolean)</td>
</tr>
<tr>
<td>webServiceProtection.xmlParsing.enabled (Boolean)</td>
<td>requestHeader.detectRepeated (Boolean)</td>
</tr>
<tr>
<td>webServiceProtection.jsonParsing.enabled (Boolean)</td>
<td>requestHeader.detectChunked (Boolean)</td>
</tr>
<tr>
<td>detectProtocolAnomalies (Boolean)</td>
<td>requestContentType.allowAll.detectFileUploads (Boolean)</td>
</tr>
<tr>
<td>serverCloaking</td>
<td>webServiceProtection.xmlParsing.size (Integer)</td>
</tr>
<tr>
<td>serverCloaking.value (Text)</td>
<td>webServiceProtection.xmlParsing.items (Integer)</td>
</tr>
<tr>
<td>suppressSensitiveHeaders (Boolean)</td>
<td>webServiceProtection.xmlParsing.level (Integer)</td>
</tr>
<tr>
<td>onErrorMessages (Keyword)</td>
<td>webServiceProtection.jsonParsing.size (Integer)</td>
</tr>
<tr>
<td>onSensitiveFileTypes (Keyword)</td>
<td>webServiceProtection.jsonParsing.items (Integer)</td>
</tr>
<tr>
<td>onSensitiveFileExtensions (Keyword)</td>
<td>webServiceProtection.jsonParsing.level (Integer)</td>
</tr>
<tr>
<td>cookieProtection</td>
<td>serverCloaking.enabled (Boolean)</td>
</tr>
<tr>
<td>discourageContentTypeSniffing (Boolean)</td>
<td>cookieProtection.type</td>
</tr>
</tbody>
</table>
Chapter 8 — HTTP Profiles API

Create HTTP Profile

Permissions

User must have the WAF module enabled
User must have "API Access" and "Create WAF Asset" permission

Example

Request:

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/create/waf/httpprofile <
file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:

<?xml version="1.0" encoding="UTF-8"?><ServiceRequest>
  <data>
    <HTTPProfile>
      <name>HTTP profile for server</name>
      <description>HTTP profile created for servers</description>
      <requestMethod>
        <allowAll>
          <detectInvalid>false</detectInvalid>
          <detectTraceTrack>false</detectTraceTrack>
        </allowAll>
        <forceDefaultContentType)
        <forceDefaultContentType.value>(Text)
        <forceDefaultContentType.enabled>(Boolean)
        <forceDefaultCharacterEncoding)
        <forceDefaultCharacterEncoding.value>(Text)
        <contentSecurityPolicyHeader)
        <contentSecurityPolicyHeader.value>(Text)
        <discourageClickjacking)
        <browserXSSProtection

<table>
<thead>
<tr>
<th>forceDefaultContentType</th>
<th>cookieProtection.value</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Keyword)</td>
<td></td>
</tr>
<tr>
<td>forceDefaultContentType.value</td>
<td>forceDefaultContentType.enabled</td>
</tr>
<tr>
<td>(Text)</td>
<td>(Boolean)</td>
</tr>
<tr>
<td>forceDefaultCharacterEncoding</td>
<td>forceDefaultCharacterEncoding.type</td>
</tr>
<tr>
<td>forceDefaultCharacterEncoding.value</td>
<td>contentSecurityPolicyHeader.enabled</td>
</tr>
<tr>
<td>contentSecurityPolicyHeader</td>
<td>tags</td>
</tr>
<tr>
<td>contentSecurityPolicyHeader.value</td>
<td>tags.tag.id</td>
</tr>
<tr>
<td>discourageClickjacking</td>
<td>tags.tag.name</td>
</tr>
<tr>
<td>browserXSSProtection</td>
<td></td>
</tr>
</tbody>
</table>
Create HTTP Profile

```xml
<HTTPProfile>
  <requestMethod/>
  <detectProtocolAnomalies>false</detectProtocolAnomalies>
  <requestHeader>
    <detectInvalid>false</detectInvalid>
    <detectRepeated>false</detectRepeated>
    <detectChunked>false</detectChunked>
  </requestHeader>
  <requestContentType>
    <allowAll>
      <detectFileUploads>false</detectFileUploads>
    </allowAll>
  </requestContentType>
  <onErrorMessages>BLOCK</onErrorMessages>
  <onSensitiveFileTypes>BLOCK</onSensitiveFileTypes>
  <onSensitiveFileExtensions>BLOCK</onSensitiveFileExtensions>
  <cookieProtection>
    <type>NONE</type>
  </cookieProtection>
  <discourageContentTypeSniffing>true</discourageContentTypeSniffing>
  <forceDefaultContentType>
    <enabled>false</enabled>
  </forceDefaultContentType>
  <forceDefaultCharacterEncoding>
    <type>NONE</type>
  </forceDefaultCharacterEncoding>
  <contentSecurityPolicyHeader>
    <enabled>false</enabled>
  </contentSecurityPolicyHeader>
  <discourageClickjacking>SAME_ORIGIN_FRAMING</discourageClickjacking>
  <browserXSSProtection>ENABLE_WITH_BLOCKING</browserXSSProtection>
  <serverCloaking>
    <enabled>false</enabled>
  </serverCloaking>
  <suppressSensitiveHeaders>true</suppressSensitiveHeaders>
</HTTPProfile>
```
Create HTTP Profile

Response

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/httpprofile.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <HTTPProfile>
      <id>4801</id>
      <uuid>d6f60c16-5146-477d-a005-a2d182cb0632</uuid>
      <name><![CDATA[HTTP profile for server]]></name>
      <description><![CDATA[HTTP profile created for servers]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName><John></firstName>
        <lastName><Doe></lastName>
      </owner>
      <created>2017-05-04T10:04:54Z</created>
      <createdBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName><John></firstName>
        <lastName><Doe></lastName>
      </createdBy>
      <updated>2017-05-04T10:04:54Z</updated>
      <updatedBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName><John></firstName>
        <lastName><Doe></lastName>
      </updatedBy>
    </HTTPProfile>
  </data>
</ServiceResponse>
```
<system>false</system>

<requestMethod>
  <allowAll>
    <detectInvalid>false</detectInvalid>
    <detectTraceTrack>false</detectTraceTrack>
  </allowAll>
</requestMethod>

$requestHeader>
  <detectInvalid>false</detectInvalid>
  <detectRepeated>false</detectRepeated>
  <detectChunked>false</detectChunked>
</requestHeader>

$requestContentType>
  <allowAll>
    <detectFileUploads>false</detectFileUploads>
  </allowAll>
</requestContentType>

<detectProtocolAnomalies>false</detectProtocolAnomalies>

<webServiceProtection>
  <xmlParsing>
    <enabled>false</enabled>
  </xmlParsing>
  <jsonParsing>
    <enabled>false</enabled>
  </jsonParsing>
</webServiceProtection>

<serverCloaking>
  <enabled>false</enabled>
</serverCloaking>

<suppressSensitiveHeaders>true</suppressSensitiveHeaders>

<onErrorMessages>BLOCK</onErrorMessages>

<onSensitiveFileTypes>BLOCK</onSensitiveFileTypes>

<onSensitiveFileExtensions>BLOCK</onSensitiveFileExtensions>

<cookieProtection>
  <type>NONE</type>
</cookieProtection>

<discourageContentTypeSniffing>true</discourageContentTypeSniffing>
Chapter 8 — HTTP Profiles API
Create HTTP Profile

<forceDefaultContentType>
  <enabled>false</enabled>
</forceDefaultContentType>
<forceDefaultCharacterEncoding>
  <type>NONE</type>
</forceDefaultCharacterEncoding>
<contentSecurityPolicyHeader>
  <enabled>false</enabled>
</contentSecurityPolicyHeader>
<discourageClickjacking>SAME_ORIGIN_FRAMING</discourageClickjacking>

<browserXSSProtection>ENABLE_WITH_BLOCKING</browserXSSProtection>
</HTTPProfile>
</data>
</ServiceResponse>
Update HTTP Profile

Update a HTTP profile identified by its identifier with given parameters. You can update all fields except tag ID and tag name.

**URL:**  
https://<baseurl>/qps/rest/2.0/update/waf/httpprofile/<id>

**Methods allowed:**  
POST

### Input

The "id" (Long) element is required. This identifies the HTTP profile you want to update. Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: HTTP Profile for descriptions of these elements.

<table>
<thead>
<tr>
<th>name (Text)</th>
<th>description (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestMethod.allowAll or requestMethod.denyAll (one of allowAll or denyAll is required)</td>
<td>requestMethod.allowAll.detectTraceTrack (Boolean)</td>
</tr>
<tr>
<td>requestMethod.allowAll.detectInvalid (Boolean)</td>
<td>requestHeader.detectInvalid (Boolean)</td>
</tr>
<tr>
<td>requestHeader</td>
<td>requestHeader.detectRepeated (Boolean)</td>
</tr>
<tr>
<td>requestContentType.allowAll or requestContentType.denyAll (one of allowAll or denyAll is required)</td>
<td>requestHeader.detectChunked (Boolean)</td>
</tr>
<tr>
<td>detectProtocolAnomalies (Boolean)</td>
<td>requestContentType.allowAll.detectFileUploads (Boolean)</td>
</tr>
<tr>
<td>serverCloaking</td>
<td>webServiceProtection.xmlParsing.enabled (Boolean)</td>
</tr>
<tr>
<td>serverCloaking.value (Text)</td>
<td>webServiceProtection.xmlParsing.size (Integer)</td>
</tr>
<tr>
<td>suppressSensitiveHeaders (Boolean)</td>
<td>webServiceProtection.xmlParsing.items (Integer)</td>
</tr>
<tr>
<td>onErrorMessages</td>
<td>webServiceProtection.xmlParsing.level (Integer)</td>
</tr>
<tr>
<td>onSensitiveFileTypes</td>
<td>webServiceProtection.jsonParsing.enabled (Boolean)</td>
</tr>
<tr>
<td>onSensitiveFileExtensions</td>
<td>webServiceProtection.jsonParsing.size (Integer)</td>
</tr>
<tr>
<td>cookieProtection</td>
<td>webServiceProtection.jsonParsing.items (Integer)</td>
</tr>
<tr>
<td>discourageContentTypeSniffing (Boolean)</td>
<td>webServiceProtection.jsonParsing.level (Integer)</td>
</tr>
<tr>
<td>forceDefaultContentType</td>
<td>serverCloaking.enabled (Boolean)</td>
</tr>
</tbody>
</table>
Update HTTP Profile

Permissions

User must have the WAF module enabled
User must have "API Access" and "Update WAF Asset" permission
Asset must be within user's scope

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/update/waf/httpprofile/4801 < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <HTTPProfile>
      <description>Update HTTP profile</description>
      <forceDefaultContentType>
        <enabled>true</enabled>
        <value><![CDATA[application/xml]]></value>
      </forceDefaultContentType>
      <discourageClickjacking>NO_FRAMING</discourageClickjacking>
    </HTTPProfile>
  </data>
</ServiceRequest>
Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/httpprofile.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <HTTPProfile>
      <id>4801</id>
      <uuid>d6f60c16-5146-477d-a005-a2d182cb0632</uuid>
      <name><![CDATA[HTTP Profile]]></name>
      <description><![CDATA[Update HTTP profile description]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName><John /></firstName>
        <lastName><Doe /></lastName>
      </owner>
      <created>2017-05-04T10:04:54Z</created>
      <createdBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName><John /></firstName>
        <lastName><Doe /></lastName>
      </createdBy>
      <updated>2017-05-04T12:40:22Z</updated>
      <updatedBy>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName><John /></firstName>
        <lastName><Doe /></lastName>
      </updatedBy>
      <system>false</system>
    </HTTPProfile>
  </data>
</ServiceResponse>
<requestMethod>
  <allowAll>
    <detectInvalid>false</detectInvalid>
    <detectTraceTrack>false</detectTraceTrack>
  </allowAll>
</requestMethod>

<requestHeader>
  <detectInvalid>false</detectInvalid>
  <detectRepeated>false</detectRepeated>
  <detectChunked>false</detectChunked>
</requestHeader>

<requestContentType>
  <allowAll>
    <detectFileUploads>false</detectFileUploads>
  </allowAll>
</requestContentType>

<detectProtocolAnomalies>false</detectProtocolAnomalies>
</WebServiceProtection>

<xmlParsing>
  <enabled>true</enabled>
  <size>500000</size>
  <items>50000</items>
  <level>64</level>
</xmlParsing>

<jsonParsing>
  <enabled>true</enabled>
  <size>600000</size>
  <items>10000</items>
  <level>32</level>
</jsonParsing>

</WebServiceProtection>

<serverCloaking>
  <enabled>false</enabled>
</serverCloaking>

<suppressSensitiveHeaders>true</suppressSensitiveHeaders>
<onErrorMessages>BLOCK</onErrorMessages>
<onSensitiveFileTypes>BLOCK</onSensitiveFileTypes>
<onSensitiveFileExtensions>BLOCK</onSensitiveFileExtensions>

<cookieProtection>
  <type>NONE</type>
</cookieProtection>
<discourageContentTypeSniffing>true</discourageContentTypeSniffing>

<forceDefaultContentType>
  <enabled>true</enabled>
  <value>
    <![CDATA[application/xml]]>
  </value>
</forceDefaultContentType>

<forceDefaultCharacterEncoding>
  <type>NONE</type>
</forceDefaultCharacterEncoding>

<contentSecurityPolicyHeader>
  <enabled>false</enabled>
</contentSecurityPolicyHeader>

<discourageClickjacking>NO_FRAMING</discourageClickjacking>

<browserXSSProtection>ENABLE_WITH_BLOCKING</browserXSSProtection>

</HTTPProfile>

</data>

</ServiceResponse>
Update HTTP Profiles (bulk)

Update HTTP profiles identified by a search with given parameters. You can update all fields except tag ID and tag name.

**URL:** https://<baseurl>/qps/rest/2.0/update/waf/httpprofile

**Methods allowed:** POST

### Input

All elements for the search operation are supported. See Search HTTP Profiles.

Allowed input elements are listed below. The associated data type for each element appears in parentheses. All elements are optional. See Reference: HTTP Profile

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>requestMethod.allowAll or</td>
<td>requestMethod.denyAll (one of</td>
</tr>
<tr>
<td>requestMethod.denyAll (one of</td>
<td>allowAll or denyAll is required)</td>
</tr>
<tr>
<td>allowAll or denyAll is required)</td>
<td>requestHeader.detectInvalid (Boolean)</td>
</tr>
<tr>
<td>requestHeader</td>
<td>requestHeaderValue (Text)</td>
</tr>
<tr>
<td>requestHeaderValue.detectRepeated</td>
<td>requestHeaderValue.detectInvalid (Boolean)</td>
</tr>
<tr>
<td>requestContentType.allowAll or</td>
<td>requestHeaderValue.detectChunked (Boolean)</td>
</tr>
<tr>
<td>requestContentType.denyAll (one of</td>
<td>requestHeaderValue.detectInvalid (Boolean)</td>
</tr>
<tr>
<td>allowAll or denyAll is required)</td>
<td>requestHeaderValue.detectChunked (Boolean)</td>
</tr>
<tr>
<td>detectProtocolAnomalies</td>
<td>requestContentType.allowAll.denyAll.detectFileUploads (Boolean)</td>
</tr>
<tr>
<td>serverCloaking.value (Text)</td>
<td>webServiceProtection.xmlParsing.enabled (Boolean)</td>
</tr>
<tr>
<td>suppressSensitiveHeaders</td>
<td>webServiceProtection.xmlParsing.size (Integer)</td>
</tr>
<tr>
<td>onSensitiveFileTypes</td>
<td>webServiceProtection.xmlParsing.items (Integer)</td>
</tr>
<tr>
<td>onSensitiveFileExtensions</td>
<td>webServiceProtection.xmlParsing.items (Integer)</td>
</tr>
<tr>
<td>cookieProtection</td>
<td>webServiceProtection.jsonParsing.enabled (Boolean)</td>
</tr>
<tr>
<td>discourageContentTypeSniffing</td>
<td>webServiceProtection.jsonParsing.items (Integer)</td>
</tr>
<tr>
<td>forceDefaultContentType</td>
<td>serverCloaking.enabled (Boolean)</td>
</tr>
</tbody>
</table>
Permissions

User must have the WAF module enabled
User must have "API Access" and "Update WAF Asset" permission
Asset must be within user's scope

Example

Request:

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/update/waf/httpprofile <
file.xml

Note: "file.xml" contains the request POST data.
Request POST Data:
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="description" operator="CONTAINS">updated</Criteria>
  </filters>
  <data>
    <HTTPProfile>
      <description>Update description</description>
      <requestHeader>
        <detectInvalid>true</detectInvalid>
        <detectRepeated>true</detectRepeated>
        <detectChunked>true</detectChunked>
      </requestHeader>
    </HTTPProfile>
  </data>
</ServiceRequest>
```

Response
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/httpprofile.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <HTTPProfile>
      <id>4801</id>
      <uuid>d6f60c16-5146-477d-a005-a2d182cb0632</uuid>
      <name><![CDATA[HTTP Profile]]></name>
      <description><![CDATA[Update description]]></description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName>John</firstName>
      </owner>
    </HTTPProfile>
  </data>
</ServiceResponse>
```
<lastName><Doe></lastName>
</owner>
<created>2017-05-04T10:04:54Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</createdBy>
<updated>2017-05-04T12:58:03Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</updatedBy>
<system>false</system>
:requestMethod>
  <allowAll>
    <detectInvalid>false</detectInvalid>
    <detectTraceTrack>false</detectTraceTrack>
  </allowAll>
</requestMethod>
:requestHeader>
  <detectInvalid>true</detectInvalid>
  <detectRepeated>true</detectRepeated>
  <detectChunked>true</detectChunked>
</requestHeader>
:requestContentType>
  <allowAll>
    <detectFileUploads>false</detectFileUploads>
  </allowAll>
</requestContentType>
<detectProtocolAnomalies>false</detectProtocolAnomalies>
<webServiceProtection>
  <xmlParsing>
    <enabled>true</enabled>
    <size>500000</size>
    <items>50000</items>
    <level>64</level>
  </xmlParsing>
</webServiceProtection>
<jsonParsing>
  <enabled>true</enabled>
  <size>600000</size>
  <items>10000</items>
  <level>32</level>
</jsonParsing>
</webServiceProtection>
<serverCloaking>
  <enabled>false</enabled>
</serverCloaking>
<suppressSensitiveHeaders>true</suppressSensitiveHeaders>
<onErrorMessages>BLOCK</onErrorMessages>
<onSensitiveFileTypes>BLOCK</onSensitiveFileTypes>
<onSensitiveFileExtensions>BLOCK</onSensitiveFileExtensions>
<cookieProtection>
  <type>NONE</type>
</cookieProtection>
<discourageContentTypeSniffing>true</discourageContentTypeSniffing>
<forceDefaultContentType>
  <enabled>true</enabled>
  <value>
    <![CDATA[application/xml]]>
  </value>
</forceDefaultContentType>
<forceDefaultCharacterEncoding>
  <type>NONE</type>
</forceDefaultCharacterEncoding>
<contentSecurityPolicyHeader>
  <enabled>false</enabled>
</contentSecurityPolicyHeader>
<discourageClickjacking>NO_FRAMING</discourageClickjacking>
<browserXSSProtection>ENABLE_WITH_BLOCKING</browserXSSProtection>
</HTTPProfile>
</data>
</ServiceResponse>
Delete HTTP Profile

Delete an existing HTTP profile identified by its identifier.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/httpprofile/<id>

**Methods allowed:**
POST

**Input**

The element "id" (Integer) is required, where "id" identifies the HTTP profile of interest.

**Permissions**

User must have the WAF module enabled
User must have "API Access" and "Delete WAF Asset" permission
Asset must be within user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/httpprofile/1401

**Response**

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/httpprofile.xsd">
    <responseCode>SUCCESS</responseCode>
    <count>1</count>
    <data>
        <HTTPProfile>
            <id>1401</id>
        </HTTPProfile>
    </data>
</ServiceResponse>
Delete HTTP Profiles (bulk)

Delete an existing HTTP profile identified by search operation.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/httpprofile/

**Methods allowed:** POST

**Input**
All elements for the search operation are supported. See Search HTTP Profiles.

**Permissions**
User must have the WAF module enabled
User must have "API Access" and "Delete WAF Asset" permission
Asset must be within user’s scope

**Example**

**Request:**

Note: "file.xml" contains the request POST data.

**Request POST Data:**
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">API</Criteria>
  </filters>
</ServiceRequest>
```

**Response**
```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/httpprofile.xsd">  
  <responseCode>SUCCESS</responseCode>
</ServiceResponse>
```
<count>3</count>
<data>
  <HTTPProfile>
    <id>4401</id>
  </HTTPProfile>
  <HTTPProfile>
    <id>4601</id>
  </HTTPProfile>
  <HTTPProfile>
    <id>4801</id>
  </HTTPProfile>
</data>
</ServiceResponse>
### Reference: HTTP Profile

A reference of all HTTP profile elements is provided below.

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>(Long) HTTP profile identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid</td>
<td>(UUID) HTTP profile identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name</td>
<td>(Text) The name of the HTTP profile as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>description</td>
<td>(Text) A description of the HTTP profile.</td>
</tr>
<tr>
<td>system</td>
<td>(Boolean) True if this is a system profile</td>
</tr>
<tr>
<td>urls.string</td>
<td>(Text) At least one backend web server address (HTTP URL).</td>
</tr>
<tr>
<td>requestMethod</td>
<td>Request Method protection settings.</td>
</tr>
<tr>
<td>requestMethod.allowAll</td>
<td>Activate AllowAll Policy for HTTP Methods</td>
</tr>
<tr>
<td>requestMethod.denyAll</td>
<td>Activate DenyAll Policy for HTTP Methods except the one listed in the Cdata content</td>
</tr>
<tr>
<td>requestMethod.allowAll.detectInvalid</td>
<td>(Boolean) If true enables invalid methods detection.</td>
</tr>
<tr>
<td>requestMethod.allowAll.detectTraceTrack</td>
<td>(Boolean) If true enables trace track detection</td>
</tr>
<tr>
<td>requestHeader</td>
<td>(Keyword) Request Header protection settings.</td>
</tr>
<tr>
<td>requestHeader.detectInvalid</td>
<td>(Boolean) If true enables invalid headers detection</td>
</tr>
<tr>
<td>requestHeader.detectRepeated</td>
<td>(Boolean) If true enables repeated headers detection.</td>
</tr>
<tr>
<td>requestHeader.detectChunked</td>
<td>(Boolean) If true enables chunked headers detection.</td>
</tr>
<tr>
<td>requestContentType</td>
<td>Request Content Type protection settings. (allowAll or denyAll sub elements)</td>
</tr>
<tr>
<td>requestContentType.allowAll</td>
<td>Activate AllowAll Policy for request Content Types</td>
</tr>
<tr>
<td>requestContentType.denyAll</td>
<td>Activate DenyAll Policy for request Content Types except for the one listed in the Cdata content</td>
</tr>
<tr>
<td>requestContentType.allowAll.detectFileUploads</td>
<td>(Boolean) If true enables file uploads detection</td>
</tr>
<tr>
<td>detectProtocolAnomalies</td>
<td>(Boolean) Enable/disable protocol anomalies detection</td>
</tr>
</tbody>
</table>
### Chapter 8 — HTTP Profiles API

Reference: HTTP Profile

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>webServiceProtection.xmlParsing.enabled</td>
<td>(Boolean) XML parser enabled or disabled. Default is disabled.</td>
</tr>
<tr>
<td>webServiceProtection.xmlParsing.size</td>
<td>(Integer) Maximum size of data parsed (in bytes). Note that the size of data can get inflated due to pattern reuse (e.g., Reuse of XML entities). Size also includes extra payloads added for preventing against attacks. For example to prevent a Billion laughs attack (a DoS attack aimed at XML parsers). Default is 100000 characters.</td>
</tr>
<tr>
<td>webServiceProtection.xmlParsing.items</td>
<td>(Integer) Maximum number of items parsed. An &quot;item&quot; can be an attribute, element tag, etc. (Depending on format: whether XML or JSON). Default is 10000.</td>
</tr>
<tr>
<td>webServiceProtection.xmlParsing.level</td>
<td>(Integer) Maximum depth reachable when parsing structured content. This enables you to avoid parsing data with huge depth, but protects servers against DDOS attacks. Default is 32.</td>
</tr>
<tr>
<td>webServiceProtection.jsonParsing.enabled</td>
<td>(Boolean) JSON parser enabled or disabled. Default is disabled.</td>
</tr>
<tr>
<td>webServiceProtection.jsonParsing.size</td>
<td>(Integer) Maximum size of data parsed (in bytes). Note that the size of data can get inflated due to pattern reuse (e.g., Reuse of XML entities). Size also includes extra payloads added for preventing against attacks. For example to prevent a Billion laughs attack (a DoS attack aimed at XML parsers). Default is 100000 characters.</td>
</tr>
<tr>
<td>webServiceProtection.jsonParsing.items</td>
<td>(Integer) Maximum number of items parsed. An &quot;item&quot; can be an attribute, element tag, etc. (Depending on format: whether XML or JSON). Default is 10000.</td>
</tr>
<tr>
<td>webServiceProtection.jsonParsing.level</td>
<td>(Integer) Maximum depth reachable when parsing structured content. This enables you to avoid parsing data with huge depth, but protects servers against DDOS attacks. Default is 32.</td>
</tr>
<tr>
<td>serverCloaking</td>
<td>Server Cloaking settings.</td>
</tr>
<tr>
<td>serverCloaking.enabled</td>
<td>(Boolean) If true enable server Cloaking</td>
</tr>
<tr>
<td>serverCloaking.value</td>
<td>(Text) Use to specify the server header value that will be enforced</td>
</tr>
<tr>
<td>suppressSensitiveHeaders</td>
<td>(Boolean) Suppress sensitive headers if true.</td>
</tr>
<tr>
<td>onErrorMessages</td>
<td>(Keyword: ALLOW, LOG, BLOCK) Action when error messages are returned.</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>onSensitiveFileTypes</td>
<td>(Keyword:ALLOW, LOG, BLOCK) Action when sensitive file types are requested.</td>
</tr>
<tr>
<td>onSensitiveFileExtensions</td>
<td>(Keyword:ALLOW, LOG, BLOCK) Action when sensitive file extensions are requested.</td>
</tr>
<tr>
<td>cookieProtection</td>
<td>Cookie protection settings.</td>
</tr>
<tr>
<td>cookieProtection.type</td>
<td>(Keyword:NONE, ALL, SELECTED) Cookie protection type</td>
</tr>
<tr>
<td>cookieProtection.value</td>
<td>(Text) Use to specify the list of selected cookies if cookie protection type is &quot;SELECTED&quot;</td>
</tr>
<tr>
<td>discourageContentTypeSniffing</td>
<td>(Boolean) If true discourage content type sniffing.</td>
</tr>
<tr>
<td>forceDefaultContentType</td>
<td>(Text) Force default content type when unknown (enabled attribute and cdata value)</td>
</tr>
<tr>
<td>forceDefaultContentType.enabled</td>
<td>(Boolean) If true forces default content type when not specified</td>
</tr>
<tr>
<td>forceDefaultContentType.value</td>
<td>(Text) Use to specify the default content type to enforce</td>
</tr>
<tr>
<td>forceDefaultCharacterEncoding</td>
<td>(Text) Force default character encoding (type attribute and cdata value)</td>
</tr>
<tr>
<td>forceDefaultCharacterEncoding.type</td>
<td>(Keyword: NONE, ALWAYS_APPLY, APPLY_WHEN_NOT_SET) Type of enforcement .</td>
</tr>
<tr>
<td>forceDefaultCharacterEncoding.value</td>
<td>(Text) Use to specify the default character encoding to enforce</td>
</tr>
<tr>
<td>contentSecurityPolicyHeader</td>
<td>Text) Content security policy header (enabled attribute and cdata value)</td>
</tr>
<tr>
<td>contentSecurityPolicyHeader.enabled</td>
<td>(Boolean) If true enables content security header</td>
</tr>
<tr>
<td>contentSecurityPolicyHeader.value</td>
<td>(Text) Use to specify the value of the content security header</td>
</tr>
<tr>
<td>discourageClickjacking</td>
<td>(Keyword:NONE, NO_FRAMING, SAME_ORIGIN_FRAMING) Discourage click jacking</td>
</tr>
<tr>
<td>browserXSSProtection</td>
<td>(Keyword: NONE, DISABLE, ENABLE_WITHOUT_BLOCKING, ENABLE_WITH_BLOCKING) Protect browser from XSS attacks.</td>
</tr>
<tr>
<td>owner</td>
<td>(Text) The user for Qualys Cloud Platform who owns this HTTP profile.</td>
</tr>
<tr>
<td>owner.id</td>
<td>(Long) The user ID of the HTTP profile owner.</td>
</tr>
<tr>
<td><strong>Element (Data Type)</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>--------------------------------------</td>
<td>-------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>owner.username</td>
<td>(Text) The user name of the HTTP profile owner.</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>(Text) The first name of the HTTP profile owner.</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>(Text) The last name of the HTTP profile owner.</td>
</tr>
<tr>
<td>created</td>
<td>(Date) The date/time when the HTTP profile was created.</td>
</tr>
<tr>
<td>createdBy.id</td>
<td>(Long) The user ID who created the HTTP profile.</td>
</tr>
<tr>
<td>createdBy.username</td>
<td>(Text) The user name who created the HTTP profile.</td>
</tr>
<tr>
<td>createdBy.firstname</td>
<td>(Text) The first name of the user who created the HTTP profile.</td>
</tr>
<tr>
<td>createdBy.lastname</td>
<td>(Text) The last name of the user who created the HTTP profile.</td>
</tr>
<tr>
<td>updated</td>
<td>(Date) The date/time when the HTTP profile was last updated.</td>
</tr>
<tr>
<td>updatedBy.id</td>
<td>(Long) The user ID who last updated the HTTP profile.</td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>(Text) The user name who last updated the HTTP profile.</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>(Text) The first name of the user who updated the HTTP profile.</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>(Text) The last name of the user who updated the HTTP profile.</td>
</tr>
<tr>
<td>webApps.webApp.id</td>
<td>(Long) The ID of the Web Application this HTTP profile is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.uuid</td>
<td>(UUID) The UUID of the Web Application this HTTP profile is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.name</td>
<td>(Text) The name of the Web Application this HTTP profile is associated with.</td>
</tr>
<tr>
<td>tags</td>
<td>(Text) List of tags (identifier and name).</td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>(Long) A tag identifier in tag list of that HTTP profile.</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>(Text) A tag name in tag list of that HTTP profile.</td>
</tr>
</tbody>
</table>
Custom Rules API

Use these API functions to manage Custom Rules.

- Current Custom Rule count
- Get details on a Custom Rule
- Search Custom Rules
- Create Custom Rule
- Update Custom Rule
- Update Custom Rules (bulk)
- Delete Custom Rules
- Delete Custom Rules (bulk)

Reference: Custom Rules
Current Custom Rule count

Returns the total number of custom rules for WAF in the user’s account.

URL: https://<baseurl>/qps/rest/2.0/count/waf/customrule
Methods allowed: GET

Input
No input elements are available.

Permissions
User must have the WAF module enabled
User must have "API Access" permission
Asset must be within user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/customrule

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/customrule.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>2</count>
</ServiceResponse>
Get details on a Custom Rule

Returns details about a specific custom rule for WAF, within the user’s scope. Want to find a custom rule ID to use as input? See Search Custom Rules.

**URL:**

https://<baseurl>/qps/rest/2.0/get/waf/customrule/<id>

**Methods allowed:**

GET

**Input**

The element "id" (Integer) is required, where "id" identifies the custom rule of interest. The associated data type for each element appears in parentheses. All dates must be entered in UTC date/time format. See Reference: Custom Rules for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>uuid</td>
<td>Integer</td>
</tr>
<tr>
<td>name</td>
<td>Text</td>
</tr>
<tr>
<td>description</td>
<td>Text</td>
</tr>
<tr>
<td>conditions</td>
<td></td>
</tr>
<tr>
<td>action</td>
<td></td>
</tr>
<tr>
<td>owner.id</td>
<td></td>
</tr>
<tr>
<td>owner.username</td>
<td>Text</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>Text</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>Text</td>
</tr>
<tr>
<td>updated (date)</td>
<td></td>
</tr>
<tr>
<td>updatedBy.id</td>
<td></td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>Text</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>Text</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>Text</td>
</tr>
<tr>
<td>tags</td>
<td></td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>Integer</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>Text</td>
</tr>
</tbody>
</table>

**Permissions**

User must have WAF module enabled
User must have "API Access" permission
Asset must be within user's scope
Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/customrule/1001

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/customrule.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomRule>
      <id>1001</id>
      <uuid>183186b8-4c91-490c-90e0-1bdf4752f3fe</uuid>
      <name>
        <![CDATA[Custom Rule Server]]>
      </name>
      <description>
        <![CDATA[Description for Custom Rule]]>
      </description>
      <owner>
        <id>3988443</id>
        <username>johndoe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </owner>
      <created>2017-04-14T13:57:32Z</created>
      <createdBy>
        <id>3988443</id>
        <username>johndoe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </createdBy>
      <updated>2017-05-16T07:32:28Z</updated>
      <updatedBy>
        <id>3988443</id>
        <username>johndoe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </updatedBy>
    </CustomRule>
  </data>
</ServiceResponse>
<lastName><Doe></lastName>
</updatedBy>
<conditions>
  <RuleCondition>
    <subject>
      <![CDATA[request.header]]>
    </subject>
    <key>
      <![CDATA[Secret]]>
    </key>
    <operator>EQUAL</operator>
    <value>
      <![CDATA[Qualys]]>
    </value>
  </RuleCondition>
  <RuleCondition>
    <subject>
      <![CDATA[client.ip.address]]>
    </subject>
    <operator>EQUAL</operator>
    <value>
      <![CDATA[truc]]>
    </value>
  </RuleCondition>
</conditions>
<action>
  <log>true</log>
  <block/>
</action>
</CustomRule>
</data>
</ServiceResponse>
Search Custom Rules

Finds custom rules in the user’s account matching the search criteria.

**URL:**
https://<baseurl>/qps/rest/2.0/search/waf/customrule

**Methods allowed:** POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: Custom Rules for descriptions of these elements.

<table>
<thead>
<tr>
<th>id (Integer)</th>
<th>createdBy.username (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>uuid (Integer)</td>
<td>createdBy.firstname (Text)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>createdBy.lastname (Text)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>updated (date)</td>
</tr>
<tr>
<td>owner.id</td>
<td>updatedBy.id (Long)</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>updatedBy.username (Text)</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>updatedBy.firstname (Text)</td>
</tr>
<tr>
<td>owner.lastname (Text)</td>
<td>updatedBy.lastname (Text)</td>
</tr>
<tr>
<td>created (date)</td>
<td>tags.tag.id</td>
</tr>
<tr>
<td>createdBy.id (Long)</td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

**Allowed Operators**

- **Long:** EQUALS
- **Integer:** EQUALS, NOT EQUALS,
- **Text:** CONTAINS, EQUALS, NOT EQUALS
- **Date:** EQUALS, NOT EQUALS, GREATER, LESSER
- **Keyword:** EQUALS, NOT EQUALS, IN
- **Boolean:** (true/false) EQUALS, NOT EQUALS
Permissions

User must have WAF module enabled
User must have "API Access" permission
Asset must be within user's scope

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/search/waf/customrule < file.xml

Note: "file.xml" contains the request POST data.
The request POST data is optional. If you leave it empty all custom rules in the user's scope are returned.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
 <filters>
   <Criteria field="name" operator="CONTAINS">cust</Criteria>
 </filters>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
 <responseCode>SUCCESS</responseCode>
 <count>1</count>
 <hasMoreRecords>false</hasMoreRecords>
 <data>
  <CustomRule>
   <id>1001</id>
   <uuid>183186b8-4c91-490c-90e0-1bdf4752f3fe</uuid>
   <name>
    <![CDATA[Custom Rule Servers]]>
   </name>
  </CustomRule>
</data>
</ServiceResponse>
<description>
  <![CDATA[My custom rule]]>
</description>

<owner>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</owner>

<created>2017-04-14T13:57:32Z</created>

<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</createdBy>

<updated>2017-05-16T07:32:28Z</updated>

<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</updatedBy>

<conditions>
  <RuleCondition>
    <subject>
      <![CDATA[request.header]]>
    </subject>
    <key><![CDATA[Secret]]></key>
    <operator>EQUAL</operator>
    <value><![CDATA[Qualys]]></value>
  </RuleCondition>
  <RuleCondition>
    <subject>
      <![CDATA[client.ip.address]]>
    </subject>
    <operator>EQUAL</operator>
    <value><![CDATA[truc]]></value>
  </RuleCondition>
</conditions>
<action>
  <log>true</log>
  <block/>
</action>
</CustomRule>
</data>
</ServiceResponse>
Create Custom Rule

Create a new custom rule with given parameters.

**URL:** https://<baseurl>/qps/rest/2.0/create/waf/customrule

**Methods allowed:** POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: Custom Rules for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>conditions</td>
<td>tags</td>
</tr>
<tr>
<td>action</td>
<td>tags.tag.id (Integer)</td>
</tr>
<tr>
<td></td>
<td>tags.tag.name (Text)</td>
</tr>
</tbody>
</table>

**Permissions**

User must have the WAF module enabled
User must have "API Access" and "Create Patch/Exception Rule" permission

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/create/waf/customrule < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <CustomRule>
      <name>My First Custom Rule</name>
      <description>checking servers</description>
      <conditions>
        
      </conditions>
    </CustomRule>
  </data>
</ServiceRequest>
```
Chapter 9 — Custom Rules API

Create Custom Rule

```
<RuleCondition>
  <subject>request.body.parameter</subject>
  <key>blague</key>
  <operator>EQUAL</operator>
  <value>toto</value>
</RuleCondition>
</conditions>
<action>
  <log>true</log>
  <block/>
</action>
</CustomRule>
</data>
</ServiceRequest>

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/customrule.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomRule>
      <id>3001</id>
      <uuid>ade93c5d-12f6-4929-8e94-3132c1507f57</uuid>
      <name>
        <![CDATA[My First Custom Rule]]>
      </name>
      <description>
        <![CDATA[checking servers]]>
      </description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </owner>
      <created>2017-05-17T11:55:30Z</created>
      <createdBy>
```
<id>3988443</id>
<username>john_doe</username>
<firstName><John></firstName>
<lastName><Doe></lastName>
</createdBy>
<updated>2017-05-17T11:55:30Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName><John></firstName>
  <lastName><Doe></lastName>
</updatedBy>
<conditions>
  <RuleCondition>
    <subject>
      <![CDATA[request.body.parameter]]>
    </subject>
    <key><![CDATA[blague]]></key>
    <operator>EQUAL</operator>
    <value><![CDATA[toto]]></value>
  </RuleCondition>
</conditions>
<action>
  <log>true</log>
  <block/>
</action>
</CustomRule>
</data>
</ServiceResponse>
Update Custom Rule

Update a custom rule identified by its identifier with given parameters. You can update all fields except tag ID and tag name.

**URL:**
https://<baseurl>/qps/rest/2.0/update/waf/customrule/<id>

**Methods allowed:** POST

**Input**
The "id" (Long) element is required. This identifies the custom rule you want to update.

Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: Custom Rules for descriptions of these elements.

<table>
<thead>
<tr>
<th>name (Text)</th>
<th>tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>description (Text)</td>
<td></td>
</tr>
<tr>
<td>conditions</td>
<td></td>
</tr>
<tr>
<td>action</td>
<td></td>
</tr>
</tbody>
</table>

**Permissions**
User must have the WAF module enabled
User must have "API Access" and "Update Patch/Exception Rule" permission
Asset must be within user's scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/update/waf/customrule < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
<data>
<CustomRule>
  <name>Update custom rule</name>
<description>update single custom rule</description>
<conditions>
  <RuleCondition>
    <subject>request.header</subject>
    <key>blague</key>
    <operator>EQUAL</operator>
    <value>toto</value>
  </RuleCondition>
</conditions>
<action>
  <log>true</log>
  <allow/>
</action>
</CustomRule>
</data>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://lqualysapi.qualys.com/qps/xsd/2.0/waf/customrule.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomRule>
      <id>3001</id>
      <uuid>ade93c5d-12f6-4929-8e94-3132c1507f57</uuid>
      <name>
        <![CDATA[Update custom rule]]>
      </name>
      <description>
        <![CDATA[update single custom rule]]>
      </description>
      <owner>
        <id>3988443</id>
        <username>john_doe</username>
        <firstName>John</firstName>
        <lastName>Doe</lastName>
      </owner>
    </CustomRule>
  </data>
</ServiceResponse>
<created>2017-05-17T11:55:30Z</created>
<createdBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</createdBy>
<updated>2017-05-17T12:09:39Z</updated>
<updatedBy>
  <id>3988443</id>
  <username>john_doe</username>
  <firstName>John</firstName>
  <lastName>Doe</lastName>
</updatedBy>
<conditions>
  <RuleCondition>
    <subject>
      <![CDATA[request.header]]>
    </subject>
    <key>
      <![CDATA[blague]]>
    </key>
    <operator>EQUAL</operator>
    <value>
      <![CDATA[toto]]>
    </value>
  </RuleCondition>
</conditions>
&action>
  <log>true</log>
  <allow/>
</action>
</CustomRule>
</data>
</ServiceResponse>
Update Custom Rules (bulk)

Update custom rules identified by a search with given parameters. You can update all fields except tag ID and tag name.

URL: https://<baseurl>/qps/rest/2.0/update/waf/customrule

Methods allowed: POST

Input

All elements for the search operation are supported. See Search Custom Rules.

Allowed input elements are listed below. The associated data type for each element appears in parentheses. All elements are optional. See Reference: Custom Rules

<table>
<thead>
<tr>
<th>name (Text)</th>
<th>tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>description (Text)</td>
<td></td>
</tr>
<tr>
<td>conditions</td>
<td></td>
</tr>
<tr>
<td>action</td>
<td></td>
</tr>
</tbody>
</table>

Allowed Operators

<table>
<thead>
<tr>
<th>Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integer</td>
<td>EQUALS, NOT EQUALS,</td>
</tr>
<tr>
<td>Text</td>
<td>CONTAINS, EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Date</td>
<td>EQUALS, NOT EQUALS, GREATER, LESSER</td>
</tr>
<tr>
<td>Keyword</td>
<td>EQUALS, NOT EQUALS, IN</td>
</tr>
<tr>
<td>Boolean</td>
<td>(true/false) EQUALS, NOT EQUALS</td>
</tr>
</tbody>
</table>

Permissions

User must have the WAF module enabled
User must have "API Access" and "Update Patch/Exception Rule" permission
Asset must be within user's scope
Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
hitsapsapi.qualys.com/qps/rest/2.0/update/waf/customrule <
file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="CONTAINS">rule</Criteria>
  </filters>
  <data>
    <CustomRule>
      <description>for qualys</description>
    </CustomRule>
  </data>
</ServiceRequest>

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/customrule.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>2</count>
  <data>
    <CustomRule>
      <id>2001</id>
      <uuid>84561e39-bc85-45cd-b8f1-c483c913750a</uuid>
      <name>
        <![CDATA[rule1]]>
      </name>
      <description>
        <![CDATA[for qualys]]>
      </description>
      <owner>
<id>3988443</id><username>john_doe</username><firstName>John</firstName><lastName>Doe</lastName></owner><created>2017-04-19T07:45:17Z</created><createdBy><id>3988443</id><username>john_doe</username><firstName>John</firstName><lastName>Doe</lastName></createdBy><updated>2017-05-17T12:16:34Z</updated><updatedBy><id>3988443</id><username>john_doe</username><firstName>John</firstName><lastName>Doe</lastName></updatedBy><conditions><RuleCondition><subject><![CDATA[request.body.charset]]></subject><operator>EQUAL</operator><value><![CDATA[UTF-16]]></value></RuleCondition><RuleCondition><subject><![CDATA[server.ip.address]]></subject><operator>EQUAL</operator><value><![CDATA[1.1.1.1]]></value></RuleCondition><RuleCondition><subject><![CDATA[request.header]]></subject>
<subject>
<key>
<![[CDATA[Qualys]]>
</key>
<operator>EQUAL</operator>
[value>
<![[CDATA[Rox!]]>]
</value>
</RuleCondition>
</conditions>
<action>
<redirect/>
<code>302</code>
<url>http://example.com</url>
</action>
</CustomRule>
<CustomRule>
{id>3001</id>
<uuid>ade93c5d-12f6-4929-8e94-3132c1507f57</uuid>
<name>
<![[CDATA[my test rule updated]]>
</name>
<description>
<![[CDATA[for qualys]]>
</description>
<owner>
{id>3988443</id>
<username>john_doe</username>
<firstName>John</firstName>
<lastName>Doe</lastName>
</owner>
<created>2017-05-17T11:55:30Z</created>
<createdBy>
{id>3988443</id>
<username>john_doe</username>
<firstName>John</firstName>
<lastName>Doe</lastName>
</createdBy>
</CustomRule>
<username>john_doe</username>
<firstName>John</firstName>
<lastName>Doe</lastName>
</updatedBy>
<conditions>
  <RuleCondition>
    <subject><![CDATA[request.header]]></subject>
    <key><![CDATA[blague]]></key>
    <operator>EQUAL</operator>
    <value><![CDATA[toto]]></value>
  </RuleCondition>
</conditions>
<action>
  <log>true</log>
  <allow/>
</action>
</CustomRule>
</data>
</ServiceResponse>
Delete Custom Rules

Delete an existing custom rule identified by its identifier.

URL: https://<baseurl>/qps/rest/2.0/delete/waf/customrule/<id>

Input

The element "id" (Integer) is required, where "id" identifies the custom rule of interest.

Permissions

User must have the WAF module enabled
User must have "API Access" and "Delete Patch/Exception Rule" permission
Asset must be within user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/customrule/1001

Response

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/customrule.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <CustomRule>
      <id>1001</id>
    </CustomRule>
  </data>
</ServiceResponse>
```
Delete Custom Rules (bulk)

Delete custom rules identified by a search with given parameters.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/customrule/

**Methods allowed:**
POST

**Input**

All elements for the search operation are supported. See [Search Custom Rules](#).

**Permissions**

User must have the WAF module enabled
User must have "API Access" and "Delete Patch/Exception Rule" permission
Asset must be within user's scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/customrule/<file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" encoding="UTF-8"?><ServiceRequest><filters><Criteria field="name" operator="CONTAINS">rule</Criteria></filters></ServiceRequest>
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/customrule.xsd">
  <responseCode>SUCCESS</responseCode>
</ServiceResponse>
```
Delete Custom Rules (bulk)

ServiceResponse
  <count>2</count>
  <data>
    <CustomRule>
      <id>2001</id>
    </CustomRule>
    <CustomRule>
      <id>3001</id>
    </CustomRule>
  </data>
</ServiceResponse>
# Reference: Custom Rules

A reference of all custom rules elements is provided below.

<p>| Element   | Description                                                                 |
|-----------|----------------------------------------------------------------------------|---|
| id        | (Long) Custom rule identifier on Qualys Cloud Platform.                    |
| uuid      | (UUID) Custom rule identifier within the Qualys Cloud WAF Service.         |
| name      | (Text) The name of the custom rule as defined by a user. This is unique in subscription. Valid action: Update |
| description | (Text) A description of the custom rule.                                     |
| conditions | Used to specify the conditions to meet for this rule to be activated.       |
| action    | The action to execute if the rule conditions were met. Can be block, allow or redirect, or block with custom page. Additionally a log element can be added with a value of true to enable logging of the action. In case of redirect a code and url element needs to be provided with a valid HTTP Redirect code (302, 301) and a valid HTTP URL. In case of block with custom page, provide the ID, UUID or name element to identify the custom page to be associated. |
| owner     | (Text) The user for Qualys Cloud Platform who owns this custom rule.        |
| owner.id  | (Long) The user ID of the custom rule owner.                               |
| owner.username | (Text) The user name of the custom rule owner.                           |
| owner.firstname | (Text) The first name of the custom rule owner.                            |
| owner.lastname | (Text) The last name of the custom rule owner.                            |
| created   | (Date) The date/time when the custom rule was created.                     |
| createdBy.id | (Long) The user ID who created the custom rule.                            |
| createdBy.username | (Text) The user name who created the custom rule.                        |
| createdBy.firstname | (Text) The first name of the user who created the custom rule.            |
| createdBy.lastname | (Text) The last name of the user who created the custom rule.            |
| updated   | (Date) The date/time when the custom rule was last updated.               |</p>
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>updatedBy.id</td>
<td>(Long) The user ID who last updated the custom rule.</td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>(Text) The user name who last updated the custom rule.</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>(Text) The first name of the user who updated the custom rule.</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>(Text) The last name of the user who updated the custom rule.</td>
</tr>
<tr>
<td>tags</td>
<td>(Text) List of tags (identifier and name).</td>
</tr>
<tr>
<td>tags.tag.id</td>
<td>(Long) A tag identifier in tag list of that custom rule.</td>
</tr>
<tr>
<td>tags.tag.name</td>
<td>(Text) A tag name in tag list of that custom rule.</td>
</tr>
</tbody>
</table>
Clusters API

Use these API functions to manage WAF clusters in the user’s subscription.

- Current cluster count
- Get details on a cluster
- Search clusters
- Create cluster
- Update cluster
- Update clusters (bulk)
- Delete cluster
- Delete clusters (bulk)

Reference: Clusters
Current cluster count

Returns the total number of WAF clusters in the user’s account.

URL: https://<baseurl>/qps/rest/2.0/count/waf/cluster
Methods allowed: GET

Input

No input elements are available.

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Manage WAFs" permission
Count includes clusters licensed for WAF and in the user’s scope

Example

Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml" https://qualysapi.qualys.com/qps/rest/2.0/count/waf/cluster

Response
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="https://qualysapi.qualys.com/qps/xsd/2.0/waf/cluster.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>52</count>
</ServiceResponse>
Get details on a cluster

Returns details about a specific WAF cluster in the user’s account. Want to find a cluster ID to use as input? See Search clusters.

URL: https://<baseurl>/qps/rest/2.0/get/waf/cluster/<id>
Methods allowed: GET

Input
The element "id" (Integer) is required, where "id" identifies the cluster ID of interest.

Permissions
User must have WAF module enabled
User must have "API ACCESS" permission
User must have "Manage WAFs" permission
Output includes cluster for WAF and within the user’s scope

Example
Request:
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml" https://qualysapi.qualys.com/qps/rest/2.0/get/waf/cluster/25401

Response
<?xml version="1.0" encoding="UTF-8"?>
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Cluster>
      <id>25401</id>
      <uuid>7de38215-a9e3-4855-8566-d9b369a92756</uuid>
      <name><![CDATA[Site1]]></name>
      <description><![CDATA[Manage all Appliances from Montlegun]]></description>
    </Cluster>
  </data>
</ServiceResponse>
<url>https://www.domain.com/resturl.html</url>
<status>302</status>
</redirect>
</errorResponse>
<trustedIPs>
<string>1.2.3.5</string>
<string>1.2.3.6/23</string>
</trustedIPs>
<updateSchedule>
<enabled>true</enabled>
<weekDays>MON, THU</weekDays>
<startTime>23</startTime>
<timezone>
<code>Asia/Aden</code>
<offset>+03:00</offset>
</timezone>
<freezeEndDate>2018-02-28</freezeEndDate>
<nextUpgradeDate>2018-03-01T23:00:00Z</nextUpgradeDate>
</updateSchedule>
<owner>
<id>3989443</id>
<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</owner>
<created>2017-07-25T09:35:01Z</created>
<createdBy>
<id>3989443</id>
<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</createdBy>
<updated>2017-07-25T10:11:13Z</updated>
<updatedBy>
<id>3989442</id>
<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</updatedBy>
<token><![CDATA[C190F3A8-08FA-46DD-BB16-EB14ECC47606]]></token>
<syncDate>2017-07-25T10:22:30Z</syncDate>
<status>DEGRADED</status>
<deploymentStatus>SUCCESS</deploymentStatus>
<deployed>2017-10-23T09:22:34Z</deployed>
<appliances>
  <Appliance>
    <id>15802</id>
    <uuid>580c5d0f-b80b-452d-a345-0ee1526fe88f</uuid>
    <name><![CDATA[580C5D0F-B80B-452D-A345-0EE1526FE88F]]></name>
  </Appliance>
  <Appliance>
    <id>15803</id>
    <uuid>6c5844e9-86f5-418e-90d5-b0201123bf13</uuid>
    <name><![CDATA[6C5844E9-86F5-418E-90D5-B0201123BF13]]></name>
  </Appliance>
</appliances>
<tags>
  <Tag>
    <id>10256057</id>
    <name><![CDATA[KIDS]]></name>
  </Tag>
</tags>
</Cluster>
</data>
</ServiceResponse>
Search clusters

Finds WAF clusters in the user’s account matching the search criteria.

**URL:** https://<baseurl>/qps/rest/2.0/search/waf/cluster

**Methods allowed:** POST

### Input

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: Clusters for descriptions of these elements.

<table>
<thead>
<tr>
<th>id (Long)</th>
<th>token (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>uuid (UUID)</td>
<td>syncDate (Date)</td>
</tr>
<tr>
<td>name (Text)</td>
<td>status (Text)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>deploymentStatus (Text)</td>
</tr>
<tr>
<td>tags.tag.id (Long)</td>
<td>deployed (Date)</td>
</tr>
<tr>
<td>tags.tag.name (Text)</td>
<td>errorResponse.action</td>
</tr>
<tr>
<td>owner.id (Long)</td>
<td>errorResponse.customPage.id (Long)</td>
</tr>
<tr>
<td>owner.username (Text)</td>
<td>errorResponse.customPage.uid (UUID)</td>
</tr>
<tr>
<td>owner.firstname (Text)</td>
<td>errorResponse.customPage.name (Text)</td>
</tr>
<tr>
<td>owner.lastname (Text)</td>
<td>errorResponse.redirect.url (Text)</td>
</tr>
<tr>
<td>created (Date)</td>
<td>errorResponse.redirect.status (Long)</td>
</tr>
<tr>
<td>createdBy.id (Long)</td>
<td>appliances.appliance.id (Long)</td>
</tr>
<tr>
<td>createdBy.username (Text)</td>
<td>appliances.appliance.uid (UUID)</td>
</tr>
<tr>
<td>createdBy.firstname (Text)</td>
<td>appliances.appliance.name (Text)</td>
</tr>
<tr>
<td>createdBy.lastname (Text)</td>
<td>webApps.webApp.id (Long)</td>
</tr>
<tr>
<td>updated (Date)</td>
<td>webApps.webApp.uid (UUID)</td>
</tr>
<tr>
<td>updatedBy.id (Long)</td>
<td>webApps.webApp.name (Text)</td>
</tr>
<tr>
<td>updatedBy.username (Text)</td>
<td>trustedIPs.string (Text)</td>
</tr>
<tr>
<td>updatedBy.firstname (Text)</td>
<td>updateSchedule.enabled (Boolean)</td>
</tr>
<tr>
<td>updatedBy.lastname (Text)</td>
<td></td>
</tr>
</tbody>
</table>
Chapter 10 — Clusters API

Search clusters

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Manage WAFs" permission
Output includes clusters licensed for WAF and within the user's scope

Example

Request:

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @- "https://qualysapi.qualys.com/qps/rest/2.0/search/waf/cluster" < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:

<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="name" operator="EQUALS">Site1</Criteria>
    <Criteria field="errorResponse.redirect.url" operator="CONTAINS">site1</Criteria>
  </filters>
</ServiceRequest>

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/cluster.xsd">
  <responseCode>SUCCESS</responseCode>
</ServiceResponse>

Allowed Operators

Long
  EQUALS, NOT EQUALS
  Note: The elements createdBy.id and updatedBy.id only support EQUALS.

UUID
  EQUALS, NOT EQUALS

Text
  CONTAINS, EQUALS, NOT EQUALS

Date
  EQUALS, NOT EQUALS, GREATER, LESSER
<count>1</count>
<hasMoreRecords>false</hasMoreRecords>
<data>
  <Cluster>
    <id>25401</id>
    <uuid>7de38215-a9e3-4855-8566-d9b369a92756</uuid>
    <name><![CDATA[Site1]]></name>
    <description><![CDATA[Manage all Appliances from Site1]]></description>
    <errorResponse>
      <redirect>
        <url>https://www.domain.com/resturl.html</url>
        <status>302</status>
      </redirect>
    </errorResponse>
    <trustedIPs>
      <string>1.2.3.5</string>
      <string>1.2.3.6/23</string>
    </trustedIPs>
    <updateSchedule>
      <enabled>true</enabled>
      <weekDays>MON,THU</weekDays>
      <startTime>23</startTime>
      <timezone>
        <code>Asia/Aden</code>
        <offset>+03:00</offset>
      </timezone>
      <freezeEndDate>2018-02-28</freezeEndDate>
      <nextUpgradeDate>2018-03-01T23:00:00Z</nextUpgradeDate>
    </updateSchedule>
    <owner>
      <id>3989443</id>
      <username>john_doe</username>
      <firstname>John</firstname>
      <lastname>Doe</lastname>
    </owner>
    <created>2017-07-25T09:35:01Z</created>
    <createdBy>
      <id>3989443</id>
      <username>john_doe</username>
      <firstname>John</firstname>
    </createdBy>
  </Cluster>
</data>
Search clusters

<lastname>Doe</lastname>
</createdBy>
<updated>2017-07-25T10:11:13Z</updated>
<updatedBy>
  <id>3989442</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<token><![CDATA[C190F3A8-08FA-46DD-BB16-EB14ECC47606]]></token>
<syncDate>2017-07-25T10:52:30Z</syncDate>
$status>DEGRADED</status>
<deploymentStatus>SUCCESS</deploymentStatus>
<deployed>2017-10-23T09:22:34Z</deployed>
<appliances>
  <Appliance>
    <id>15802</id>
    <uuid>580c5d0f-b80b-452d-a345-0ee1526fe88f</uuid>
    <name><![CDATA[580C5D0F-B80B-452D-A345-0EE1526FE88F]]></name>
  </Appliance>
  <Appliance>
    <id>15803</id>
    <uuid>6c5844e9-86f5-418e-90d5-b0201123bf13</uuid>
    <name><![CDATA[6C5844E9-86F5-418E-90D5-B0201123BF13]]></name>
  </Appliance>
</appliances>
<tags>
  <Tag>
    <id>10256057</id>
    <name><![CDATA[KIDS]]></name>
  </Tag>
</tags>
</Cluster>
</data>
</ServiceResponse>
<tags>
  <Tag>
    <id>7530430</id>
    <name><![CDATA[Cloud Agent]]></name>
  </Tag>
</tags>
</Cluster>
</data>
</ServiceResponse>
Create cluster

Create a WAF cluster.

**URL:**  
https://<baseurl>/qps/rest/2.0/create/waf/cluster

**Methods allowed:**  
POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. See Reference: Clusters for descriptions of these elements.

<table>
<thead>
<tr>
<th>Required Elements</th>
<th>Optional Elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>description (Text)</td>
</tr>
<tr>
<td>updateSchedule.enabled (Boolean)</td>
<td>errorResponse</td>
</tr>
<tr>
<td>Required only when using the updateSchedule element.</td>
<td>errorResponse.block</td>
</tr>
<tr>
<td></td>
<td>errorResponse.redirect.url (Text)</td>
</tr>
<tr>
<td></td>
<td>errorResponse.redirect.status (Long)</td>
</tr>
<tr>
<td></td>
<td>errorResponse.customPage.id (Long)</td>
</tr>
<tr>
<td></td>
<td>updateSchedule.weekDays (Text)</td>
</tr>
<tr>
<td></td>
<td>updateSchedule.startTime (Integer)</td>
</tr>
<tr>
<td></td>
<td>updateSchedule.timezone.code (Text)</td>
</tr>
<tr>
<td></td>
<td>updateSchedule.timezone.offset (Text)</td>
</tr>
<tr>
<td></td>
<td>updateSchedule.freezeEndDate (Date)</td>
</tr>
<tr>
<td></td>
<td>tags</td>
</tr>
<tr>
<td></td>
<td>tags.tag.id (Long)</td>
</tr>
<tr>
<td></td>
<td>tags.tag.name (Text)</td>
</tr>
<tr>
<td></td>
<td>trustedIPs.string (Text)</td>
</tr>
</tbody>
</table>

Use Web Applications API to associate a cluster to a web application.

**Permissions**

User must have the WAF module enabled  
User must have "API ACCESS" permission  
User must have "Manage WAFs" permission
Example

Request:

Note: "file.xml" contains the request POST data.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <data>
    <Cluster>
      <name>Site1</name>
      <description>Manage all Appliances from Site1</description>
      <errorResponse>
        <redirect>
          <url>https://www.domain.com/resturl.html</url>
          <status>302</status>
        </redirect>
      </errorResponse>
      <trustedIPs>
        <string>1.2.3.5</string>
        <string>1.2.3.6/23</string>
      </trustedIPs>
      <updateSchedule>
        <enabled>true</enabled>
        <weekDays>MON,THU</weekDays>
        <startTime>23</startTime>
        <timezone>
          <code>Asia/Aden</code>
          <offset>+03:00</offset>
        </timezone>
        <freezeEndDate>2018-02-28</freezeEndDate>
      </updateSchedule>
    </Cluster>
  </data>
</ServiceRequest>
Response

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/cluster.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Cluster>
      <id>25401</id>
      <uuid>7de38215-a9e3-4855-8566-d9b369a92756</uuid>
      <name><![CDATA[Site1]]></name>
      <description><![CDATA[Manage all Appliances from Site1]]></description>
      <errorResponse>
        <redirect>
          <url>https://www.domain.com/resturl.html</url>
          <status>302</status>
        </redirect>
      </errorResponse>
      <trustedIPs>
        <string>1.2.3.5</string>
        <string>1.2.3.6/23</string>
      </trustedIPs>
      <updateSchedule>
        <enabled>true</enabled>
        <weekDays>MON,THU</weekDays>
        <startTime>23</startTime>
        <timezone>
          <code>Asia/Aden</code>
          <offset>+03:00</offset>
        </timezone>
        <freezeEndDate>2018-02-28</freezeEndDate>
        <nextUpgradeDate>2018-03-01T23:00:00Z</nextUpgradeDate>
      </updateSchedule>
      <owner>
        <id>3989443</id>
        <username>john_doe</username>
        <firstname>John</firstname>
        <lastname>Doe</lastname>
      </owner>
    </Cluster>
  </data>
</ServiceResponse>
```
<owner>
<created>2017-07-25T09:35:01Z</created>
<createdBy>
  <id>3989443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updated>2017-07-25T09:35:01Z</updated>
<updatedBy>
  <id>3989443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<token><![CDATA[C190F3A8-08FA-46DD-BB16-EB14ECC47606]]></token>
<status>NO_SENSORS</status>
</Cluster>
</data>
</ServiceResponse>
Chapter 10 — Clusters API

Update cluster

Update a WAF cluster in the user’s account. You can update all fields except tag ID and tag name.

URL: https://<baseurl>/qps/rest/2.0/update/waf/cluster/<id>

Methods allowed: POST

Input

The "id" (Long) element is required. This identifies the cluster you want to update.

Optional input elements are listed below. The associated data type for each element appears in parentheses. See Reference: Clusters for descriptions of these elements.

<table>
<thead>
<tr>
<th>name (Text)</th>
<th>errorResponse.customPage.name (Text)</th>
</tr>
</thead>
<tbody>
<tr>
<td>description (Text)</td>
<td>updateSchedule.enabled (Boolean)</td>
</tr>
<tr>
<td>errorResponse</td>
<td>updateSchedule.weekDays (Text)</td>
</tr>
<tr>
<td>errorResponse:block</td>
<td>updateSchedule.startTime (Integer)</td>
</tr>
<tr>
<td>errorResponse.redirect.url</td>
<td>updateSchedule.timezone.code (Text)</td>
</tr>
<tr>
<td>trustedIPs.string (Text)</td>
<td>updateSchedule.timezone.offset (Text)</td>
</tr>
<tr>
<td>errorResponse.redirect.status</td>
<td>updateSchedule.freezeEndDate (Date)</td>
</tr>
<tr>
<td>errorResponse.customPage.id</td>
<td>tags</td>
</tr>
<tr>
<td>errorResponse.customPage.uuid</td>
<td>(UUID)</td>
</tr>
</tbody>
</table>

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Manage WAFs" permission
Cluster must be licensed in user’s subscription
Cluster must be within the user’s scope
Example

Request:

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/cluster/25401" < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:

<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
<data>
<Cluster>
 <name>Site1 Updated</name>
 <tags><Tag><name>ABC</name></Tag></tags>
</Cluster>
</data>
</ServiceRequest>

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/cluster.xsd">
 <responseCode>SUCCESS</responseCode>
 <count>1</count>
 <data>
  <Cluster>
   <id>25401</id>
   <uuid>7de38215-a9e3-4855-8566-d9b369a92756</uuid>
   <name> <![CDATA[Site1 Updated]]> </name>
   <description> <![CDATA[Manage all Appliances from Site1]]> </description>
   <errorResponse>
    <redirect>
     <url>https://www.domain.com/resturl.html</url>
     <status>302</status>
    </redirect>
   </errorResponse>
  </Cluster>
 </data>
</ServiceResponse>
<trustedIPs>
  <string>1.2.3.5</string>
  <string>1.2.3.6/23</string>
</trustedIPs>
<updateSchedule>
  <enabled>true</enabled>
  <weekDays>MON,THU</weekDays>
  <startTime>23</startTime>
  <timezone>
    <code>Asia/Aden</code>
    <offset>+03:00</offset>
  </timezone>
  <freezeEndDate>2018-02-28</freezeEndDate>
  <nextUpgradeDate>2018-03-01T23:00:00Z</nextUpgradeDate>
</updateSchedule>
<owner>
  <id>3989443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</owner>
<created>2017-07-25T09:35:01Z</created>
<createdBy>
  <id>3989443</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</createdBy>
<updatedBy>
  <id>3989444</id>
  <username>john_doe</username>
  <firstname>John</firstname>
  <lastname>Doe</lastname>
</updatedBy>
<token><![CDATA[C190F3A8-08FA-46DD-BB16-EB14ECC47606]]></token>
<syncDate>2017-07-25T12:22:30Z</syncDate>
<status>DEGRADED</status>
<appliances>
  <Appliance>
<id>15802</id>
<uuid>580c5d0f-b80b-452d-a345-0ee1526fe88f</uuid>
<name><![CDATA[580C5D0F-B80B-452D-A345-0EE1526FE88F]]></name>
</Appliance>
<Appliance>
  <id>15803</id>
  <uuid>6c5844e9-86f5-418e-90d5-b0201123bf13</uuid>
  <name><![CDATA[6C5844E9-86F5-418E-90D5-B0201123BF13]]></name>
</Appliance>
</appliances>
<tags>
  <Tag>
    <id>10256057</id>
    <name><![CDATA[ABC]]></name>
  </Tag>
  <Tag>
    <id>10256057</id>
    <name><![CDATA[ABC]]></name>
  </Tag>
</tags>
</Cluster>
</data>
</ServiceResponse>
Chapter 10 — Clusters API

Update clusters (bulk)

Update multiple WAF clusters in the user’s account. You can update all fields except tag ID and tag name.

**URL:** https://<baseurl>/qps/rest/2.0/update/waf/cluster

**Methods allowed:** POST

### Input

All elements for the search operation are supported. See Search clusters.

Allowed input elements for bulk update are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. See Reference: Clusters for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name (Text)</td>
<td>errorResponse.customPage.name (Text)</td>
</tr>
<tr>
<td>description (Text)</td>
<td>updateSchedule.enabled (Boolean)</td>
</tr>
<tr>
<td>errorResponse</td>
<td>updateSchedule.weekDays (Text)</td>
</tr>
<tr>
<td>errorResponse.block</td>
<td>updateSchedule.startTime (Integer)</td>
</tr>
<tr>
<td>errorResponse.redirect.url (Text)</td>
<td>updateSchedule.timezone.code (Text)</td>
</tr>
<tr>
<td>trustedIPs.string (Text)</td>
<td>updateSchedule.timezone.offset (Text)</td>
</tr>
<tr>
<td>errorResponse.redirect.status (Long)</td>
<td>updateSchedule.freezeEndDate (Date)</td>
</tr>
<tr>
<td>errorResponse.customPage.id (Long)</td>
<td>tags</td>
</tr>
<tr>
<td>errorResponse.customPage.uuid (UUID)</td>
<td></td>
</tr>
</tbody>
</table>

**Allowed Operators**

- **Long**: EQUALS, NOT EQUALS
- **UUID**: EQUALS, NOT EQUALS
- **Text**: CONTAINS, EQUALS, NOT EQUALS
- **Date**: EQUALS, NOT EQUALS, GREATER, LESSER
Permissions

- User must have the WAF module enabled
- User must have "API ACCESS" permission
- User must have "Manage WAFs" permission
- Cluster must be licensed in user's subscription
- Cluster must be within the user's scope

Example

Request:

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/waf/cluster" < file.xml

Note: "file.xml" contains the request POST data.

Request POST Data:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="createdBy.firstname" operator="EQUALS">John</Criteria>
  </filters>
  <data>
    <Cluster>
      <tags><Tag><name>XYZ</name></Tag></tags>
    </Cluster>
  </data>
</ServiceRequest>
```

Response:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/cluster.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Cluster>
      <id>25401</id>
    </Cluster>
  </data>
</ServiceResponse>
```
<uuid>7de38215-a9e3-4855-8566-d9b369a92756</uuid>
<name><![CDATA[Montlegun Updated]]></name>
<description><![CDATA[Manage all Appliances from Montlegun]]></description>
<errorResponse>
 <redirect>
  <url>https://www.domain.com/resturl.html</url>
  <status>302</status>
 </redirect>
</errorResponse>
<trustedIPs>
 <string>1.2.3.5</string>
 <string>1.2.3.6/23</string>
</trustedIPs>
<updateSchedule>
 <enabled>true</enabled>
 <weekDays>MON,THU</weekDays>
 <startTime>23</startTime>
 <timezone>
  <code>Asia/Aden</code>
  <offset>+03:00</offset>
 </timezone>
 <freezeEndDate>2018-02-28</freezeEndDate>
 <nextUpgradeDate>2018-03-01T23:00:00Z</nextUpgradeDate>
</updateSchedule>
<owner>
 <id>3989443</id>
 <username>john_doe</username>
 <firstname>John</firstname>
 <lastname>Doe</lastname>
</owner>
<created>2017-07-25T09:35:01Z</created>
<createdBy>
 <id>3989443</id>
 <username>john_doe</username>
 <firstname>John</firstname>
 <lastname>Doe</lastname>
</createdBy>
<updated>2017-07-25T12:30:24Z</updated>
<updatedBy>
 <id>3989444</id>

<username>john_doe</username>
<firstname>John</firstname>
<lastname>Doe</lastname>
</updatedBy>
<token><![CDATA[C190F3A8-08FA-46DD-BB16-EB14ECC47606]]></token>
<syncDate>2017-07-25T12:27:31Z</syncDate>
<status>DEGRADED</status>
<appliances>
<Appliance>
  <id>15802</id>
  <uuid>580c5d0f-b80b-452d-a345-0ee1526fe88f</uuid>
  <name><![CDATA[580C5D0F-B80B-452D-A345-0EE1526FE88F]]></name>
</Appliance>
<Appliance>
  <id>15803</id>
  <uuid>6c5844e9-86f5-418e-90d5-b0201123bf13</uuid>
  <name><![CDATA[6C5844E9-86F5-418E-90D5-B0201123BF13]]></name>
</Appliance>
</appliances>
<tags>
<Tag>
  <id>10256059</id>
  <name><![CDATA[XY2]]></name>
</Tag>
</tags>
</Cluster>
</data>
</ServiceResponse>
Delete cluster

Delete a WAF cluster in the user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/cluster/<id>

**Methods allowed:** POST

**Input**
The "id" (Long) element is required. This identifies the cluster asset you want to delete.

**Permissions**
User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Manage WAFs" permission
Cluster must be licensed in user’s subscription
Cluster must be within the user’s scope

**Example**

**Request:**
curl -u "USERNAME:PASSWORD" -X "POST" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/cluster/122801

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Cluster>
      <id>122801</id>
    </Cluster>
  </data>
</ServiceResponse>
```
Delete clusters (bulk)

Delete multiple WAF clusters in the user’s account.

**URL:**
https://<baseurl>/qps/rest/2.0/delete/waf/cluster

**Methods allowed:**
POST

**Input**

All elements for the search operation are supported. See [Search clusters](#).

**Permissions**

- User must have the WAF module enabled
- User must have "API ACCESS" permission
- User must have "Manage WAFs" permission
- Cluster must be licensed in user’s subscription
- Cluster must be within the user’s scope

**Example**

**Request**:

curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST" --data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/cluster" < file.xml

Note: "file.xml" contains the request POST data.

**Request POST Data:**

```xml
<?xml version="1.0" ?>
<ServiceRequest>
  <filters>
    <Criteria field="tags.tag.name" operator="EQUALS">XYZ</Criteria>
  </filters>
</ServiceRequest>
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
```

Qualys Web Application Firewall 261
<Chapter 10 — Clusters API>

Delete clusters (bulk)

```xml
<xsi:noNamespaceSchemaLocation="http://qualysapi.qualys.com/qps/xsd/2.0/waf/cluster.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Cluster>
      <id>25401</id>
    </Cluster>
  </data>
</ServiceResponse>
```
Reference: Clusters

A reference of all cluster elements is provided below.

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>(Long) Cluster identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid</td>
<td>(UUID) Cluster identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name</td>
<td>(Text) The name of the cluster as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>description</td>
<td>(Text) The description of the WAF cluster as defined by a user. Valid action: Update</td>
</tr>
<tr>
<td>owner</td>
<td>(Text) The user for Qualys Cloud Platform who owns this cluster.</td>
</tr>
<tr>
<td>owner.id</td>
<td>(Long) The user ID of the WAF cluster owner.</td>
</tr>
<tr>
<td>owner.username</td>
<td>(Text) The user name of the WAF cluster owner.</td>
</tr>
<tr>
<td>owner.firstname</td>
<td>(Text) The first name of the WAF cluster owner.</td>
</tr>
<tr>
<td>owner.lastname</td>
<td>(Text) The last name of the WAF cluster owner.</td>
</tr>
<tr>
<td>created</td>
<td>(Date) The date/time when the WAF cluster was created.</td>
</tr>
<tr>
<td>createdBy.id</td>
<td>(Long) The user ID who created the WAF cluster.</td>
</tr>
<tr>
<td>createdBy.username</td>
<td>(Text) The user name who created the WAF cluster.</td>
</tr>
<tr>
<td>createdBy.firstname</td>
<td>(Text) The first name of the user who created the WAF cluster.</td>
</tr>
<tr>
<td>createdBy.lastname</td>
<td>(Text) The last name of the user who created the WAF cluster.</td>
</tr>
<tr>
<td>updated</td>
<td>(Date) The date/time when the WAF cluster was last updated.</td>
</tr>
<tr>
<td>updatedBy</td>
<td>(Text) The user for Qualys Cloud Platform who updates this WAF cluster.</td>
</tr>
<tr>
<td>updatedBy.id</td>
<td>(Long) The user ID who last updated the WAF cluster.</td>
</tr>
<tr>
<td>updatedBy.username</td>
<td>(Text) The user name who last updated the WAF cluster.</td>
</tr>
<tr>
<td>updatedBy.firstname</td>
<td>(Text) The first name of the user who updated the WAF cluster.</td>
</tr>
<tr>
<td>updatedBy.lastname</td>
<td>(Text) The last name of the user who updated the WAF cluster.</td>
</tr>
<tr>
<td>token</td>
<td>(Text) The registration token for the WAF cluster.</td>
</tr>
<tr>
<td>syncDate</td>
<td>(Date) The last synchronization date of assets (between Qualys Cloud platform database and WAF database).</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>status</td>
<td>(Text) Status information: NO_SENSORS (No appliances are connected to the WAF cluster) INACTIVE (All appliances connected to the WAF cluster are up and ready for service, but the configuration is yet to be deployed) DEGRADED (At least one appliance is connected to the WAF cluster, but other appliances are disconnected) ACTIVE (All appliances connected to the WAF cluster are up and ready for service) ERROR (All appliances connected to the WAF cluster are disconnected)</td>
</tr>
<tr>
<td>deploymentStatus</td>
<td>(Text) Deployment Status information: SUCCESS (Configuration deployment was successful on all appliances in a cluster) PENDING_DEPLOY (Configuration deployment has been requested but not yet started) FAILURE (Configuration deployment has failed on all appliances in a cluster) PARTIAL (Configuration deployment was successful on some appliances but failed on other appliances in a cluster) IN_PROGRESS (Configuration deployment is in progress on all appliances in a cluster) UNUSED (Web application is not deployed on any WAF cluster, or the Web application is deployed on a cluster having no appliances registered to it)</td>
</tr>
<tr>
<td>deployed</td>
<td>(Date) The last deployment date of the cluster.</td>
</tr>
<tr>
<td>errorResponse.action</td>
<td>Error Response behavior (block, redirect, custom response) when a request is not routable.</td>
</tr>
<tr>
<td>errorResponse.block</td>
<td>Enables display of the default WAF error page.</td>
</tr>
<tr>
<td>errorResponse.redirect.url</td>
<td>(Text) Empty or a valid URL used to redirect web client when request cannot be routed to any known web application.</td>
</tr>
<tr>
<td>errorResponse.redirect.status</td>
<td>(Long) A valid redirection HTTP code (301/302) used to redirect web client (see redirect.url)</td>
</tr>
<tr>
<td>errorResponse.customPage.id</td>
<td>(Long) The ID of the custom response page to assign to the cluster.</td>
</tr>
<tr>
<td>errorResponse.customPage.uuid</td>
<td>(UUID) The UUID of the custom response page to assign to the cluster.</td>
</tr>
<tr>
<td>errorResponse.customPage.name</td>
<td>(Text) The name of the custom response page to assign to the cluster.</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>webApps.webApp.id</td>
<td>(Long) The ID of the Web Application this WAF cluster is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.uuid</td>
<td>(UUID) The UUID of the Web Application this WAF cluster is associated with.</td>
</tr>
<tr>
<td>webApps.webApp.name</td>
<td>(Text) The name of the Web Application this WAF cluster is associated with.</td>
</tr>
<tr>
<td>trustedIPs.string</td>
<td>(Text) Provide the IP address/range/network of trusted origin proxies or load balancers configured in full-proxy mode. Enter an IP address between 1.0.0.0 - 223.255.255.254 excluding 127.x.x.x. Optionally, you can provide a range for the IP address between 1 - 32. For example, 1.2.3.6/23 If the request is not from a trusted source the X-Forwarded-For header values are automatically discarded. If you do not provide IP addresses for trusted origin proxies or load balancers, then IP addresses as per RFC1918 are trusted.</td>
</tr>
<tr>
<td>updateSchedule.enabled</td>
<td>(Boolean) Schedule Auto-Update enabled or disabled. By default this is enabled.</td>
</tr>
<tr>
<td>updateSchedule.weekDays</td>
<td>(Text) Comma separated list of days the auto-update is allowed. By default enabled for all days: MON,TUE,WED,THU,FRI,SAT,SUN.</td>
</tr>
<tr>
<td>updateSchedule.startTime</td>
<td>(Integer) Hour of the day the auto-update may occur. Default is 0. Specify the hour in 24 hour format.</td>
</tr>
<tr>
<td>updateSchedule.timezone.code</td>
<td>(Text) The timezone. For example, Pacific Standard Time. Default is UTC.</td>
</tr>
<tr>
<td>updateSchedule.timezone.offset</td>
<td>(Text) The timezone offset (hours and minutes). Default is +00:00.</td>
</tr>
<tr>
<td>updateSchedule.freezeEndDate</td>
<td>(Date) If set the auto-update process will not be executed before the specified date (e.g. 2018-03-01). Disabled by default.</td>
</tr>
<tr>
<td>tags.id</td>
<td>(Long) The ID of a tag associated with the WAF cluster.</td>
</tr>
<tr>
<td>tags.name</td>
<td>(Text) The name, defined by a user, of a tag associated with the WAF cluster.</td>
</tr>
<tr>
<td>appliances</td>
<td>The list of appliances that are connected to the WAF cluster. Each appliance entry contains only basic information allowing item identification.</td>
</tr>
<tr>
<td>appliance.id</td>
<td>(Long) The ID of the appliance associated with the WAF cluster on Qualys Cloud Platform. (see Reference: Appliances)</td>
</tr>
</tbody>
</table>
### Element (Data Type) | Description
--- | ---
appliance.uuid | (UUID) The UUID of the appliance within WAF module. (see Reference: Appliances)
appliance.name | (Text) Appliance name, so far UUID in upper case as far name is not a field stored in database. (see Reference: Appliances)
Appliance API

Use these API functions to manage WAF appliances in the user’s subscription.

- Current appliance count
- Get details on appliance
- Search appliances
- Delete appliance

Reference: Appliances
Current appliance count

Returns the total number of WAF appliances in the user’s account.

**URL:** https://<baseurl>/qps/rest/2.0/count/waf/appliance

**Methods allowed:** GET

**Input**

No input elements are available.

**Permissions**

User must have the WAF module enabled
User must have “API ACCESS” permission
Count includes appliances licensed for WAF and in the user’s scope

**Example**

**Request:**

```sh
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/count/waf/appliance/
```

**Response**

```xml
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="https://qualysapi.qualys.com/qps/xsd/2.0/waf/appliance.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>4</count>
</ServiceResponse>
```
Get details on appliance

Returns details about a specific WAF appliance in the user’s account. Want to find an appliance ID to use as input? See Reference: Appliances.

**URL:**

https://<baseURL>/qps/rest/2.0/get/waf/appliance/:id

**Methods allowed:**

GET

**Input**

The element "id" (Integer) is required, where "id" identifies the appliance ID of interest.

**Permissions**

User must have WAF module enabled
User must have "API ACCESS" permission
Output includes appliance licensed for WAF and within the user’s scope

**Example**

Request:

curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/appliance/15804

Response

<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xs="http://qualysapi.qualys.com/qps/xsd/2.0/waf/appliance.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Appliance>
      <id>15804</id>
      <uuid>9276b0ed-53e4-4c43-b27e-463851990ee3</uuid>
    </Appliance>
  </data>
</ServiceResponse>
<status>INACTIVE</status>
<pollStatus>NOTPolling</pollStatus>
<systemOs><![CDATA[Linux 30c1def71a9d 4.4.0-83-generic #106-Ubuntu SMP Mon Jun 26 17:54:43 UTC 2017 x86_64]]></systemOs>
<systemRam>12284530688</systemRam>
<systemType><![CDATA[other]]></systemType>
<systemCpusCount>1</systemCpusCount>
<systemCpusCores>2</systemCpusCores>
<systemCpusSpeed>2600.406</systemCpusSpeed>
<systemCpusModel><![CDATA[Intel(R) Core(TM) i7-5600U CPU @ 2.60GHz]]></systemCpusModel>
<configVersion><![CDATA[2017-07-25T13:07:59.244Z]]></configVersion>
<configGenerated>2017-07-25T13:07:59Z</configGenerated>
<ip><![CDATA[172.17.0.2]]></ip>
<cluster>
  <id>25603</id>
  <uuid>d817eb34-2320-4044-9c0c-8b0b2523a75a</uuid>
  <name><![CDATA[Montlegun]]></name>
</cluster>
</Appliance>
</data>
</ServiceResponse>
Search appliances

Finds WAF appliances in the user’s account matching search criteria.

**URL:**
https://<baseurl>/qps/rest/2.0/search/waf/appliance

**Methods allowed:** POST

**Input**

Allowed input elements are listed below. The associated data type for each element appears in parentheses. These elements are optional and act as filters. When multiple elements are specified, parameters are combined using a logical AND. All dates must be entered in UTC date/time format. See Reference: Appliances for descriptions of these elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Long</td>
</tr>
<tr>
<td>uuid</td>
<td>UUID</td>
</tr>
<tr>
<td>name</td>
<td>Text</td>
</tr>
<tr>
<td>hostname</td>
<td>Text</td>
</tr>
<tr>
<td>lastPollDate</td>
<td>Date</td>
</tr>
<tr>
<td>applianceCreated</td>
<td>Date</td>
</tr>
<tr>
<td>applianceVersion</td>
<td>Text</td>
</tr>
<tr>
<td>status</td>
<td>Long</td>
</tr>
<tr>
<td>pollStatus</td>
<td>Long</td>
</tr>
<tr>
<td>heartbeatGenerated</td>
<td>Date</td>
</tr>
<tr>
<td>heartbeatProcessed</td>
<td>Date</td>
</tr>
<tr>
<td>systemOs</td>
<td>Text</td>
</tr>
<tr>
<td>systemRam</td>
<td>Long</td>
</tr>
<tr>
<td>systemType</td>
<td>Text</td>
</tr>
<tr>
<td>systemEc2InstanceId</td>
<td>Text</td>
</tr>
<tr>
<td>systemEc2InstanceType</td>
<td>Text</td>
</tr>
<tr>
<td>systemEc2Amild</td>
<td>Text</td>
</tr>
<tr>
<td>systemCpusCount</td>
<td>Long</td>
</tr>
<tr>
<td>systemCpusCores</td>
<td>Long</td>
</tr>
<tr>
<td>systemCpusSpeed</td>
<td>Float</td>
</tr>
<tr>
<td>systemCpusModel</td>
<td>Text</td>
</tr>
<tr>
<td>configRulesVersion</td>
<td>Text</td>
</tr>
<tr>
<td>configVersion</td>
<td>Text</td>
</tr>
<tr>
<td>configGenerated</td>
<td>Date</td>
</tr>
<tr>
<td>cluster.id</td>
<td>Long</td>
</tr>
<tr>
<td>cluster.uuid</td>
<td>UUID</td>
</tr>
<tr>
<td>cluster.name</td>
<td>Text</td>
</tr>
</tbody>
</table>

**Allowed Operators**

<table>
<thead>
<tr>
<th>Type</th>
<th>Operators</th>
</tr>
</thead>
<tbody>
<tr>
<td>Long</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>SystemRam, systemCpusCount, systemCpusCores</td>
<td>EQUALS, IN, NOT EQUALS, GREATER, LESSER</td>
</tr>
<tr>
<td>UUID</td>
<td>EQUALS, NOT EQUALS</td>
</tr>
<tr>
<td>Text</td>
<td>CONTAINS, EQUALS, NOT EQUALS</td>
</tr>
</tbody>
</table>

**Note:** The element ip only supports EQUALS.
Chapter 11 — Appliance API

Search appliances

Permissions

User must have the WAF module enabled
User must have "API ACCESS" permission
Output includes appliances licensed for WAF and within the user's scope

Example

Request:
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/search/waf/appliance" < file.xml

Note: "file.xml" contains the request POST data.
The request POST data is optional. If you leave it empty all appliances in the user's scope are returned.

Request POST Data:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
  <filters>
    <Criteria field="ip" operator="EQUALS">172.17.0.2</Criteria>
  </filters>
</ServiceRequest>

Response:
<?xml version="1.0" encoding="UTF-8"?>
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://<server.host>:<server.port>/qps/xsd/2.0/waf/appliance.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <hasMoreRecords>false</hasMoreRecords>
  <data>
    <Appliance>
      <id>15804</id>
      <uuid>9276b0ed-53e4-4c43-b27e-463851990ee3</uuid>
    </Appliance>
  </data>
</ServiceResponse>
<name><![CDATA[9276BOED-53E4-4C43-B27E-463851990EE3]]></name>
<hostname><![CDATA[30c1def71a9d]]></hostname>
<lastPollDate>2017-07-25T13:26:30Z</lastPollDate>
<applianceCreated>2017-07-25T12:51:03Z</applianceCreated>
<applianceVersion><![CDATA[1.4.0]]></applianceVersion>
<status>ACTIVE</status>
<pollStatus>POLLING</pollStatus>
<heartbeatGenerated>2017-07-25T13:26:00Z</heartbeatGenerated>
<heartbeatProcessed>2017-07-25T13:26:00Z</heartbeatProcessed>
<systemOs><![CDATA[Linux 30c1def71a9d 4.4.0-83-generic #106-Ubuntu SMP Mon Jun 26 17:54:43 UTC 2017 x86_64]]></systemOs>
<systemRam>12284530688</systemRam>
<systemType><![CDATA[other]]></systemType>
<systemCpusCount>1</systemCpusCount>
<systemCpusCores>2</systemCpusCores>
<systemCpusSpeed>2600.406</systemCpusSpeed>
<systemCpusModel><![CDATA[Intel(R) Core(TM) i7-5600U CPU @ 2.60GHz]]></systemCpusModel>
<configGenerated>2017-07-25T13:18:27Z</configGenerated>
<ip><![CDATA[172.17.0.2]]></ip>
<cluster>
  <id>25603</id>
  <uuid>d817eb34-2320-4044-9c0c-8b0b2523a75a</uuid>
  <name><![CDATA[Montlegun]]></name>
</cluster>
</Appliance>
</data>
</ServiceResponse>
Delete appliance

Delete a WAF appliance in the user’s account.

**URL:** https://<baseurl>/qps/rest/2.0/delete/waf/appliance/<id>

**Methods allowed:** POST

**Input**

The "id" (Long) element is required. This identifies the appliance you want to delete.

**Permissions**

User must have the WAF module enabled
User must have "API ACCESS" permission
User must have "Manage WAFs" permission
Appliance must be licensed in user’s subscription
Appliance must be within the user’s scope

**Example**

**Request:**

curl -u "USERNAME:PASSWORD" -X "POST" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/delete/waf/appliance/15804

**Response**

```xml
<ServiceResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="https://qualysapi.qualys.com/qps/xsd/2.0/waf/appliance.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Appliance>
      <id>15804</id>
    </Appliance>
  </data>
</ServiceResponse>
```
## Reference: Appliances

A reference of all appliance elements is provided below.

<table>
<thead>
<tr>
<th>Element (Data Type)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>(Long) WAF appliance identifier on Qualys Cloud Platform.</td>
</tr>
<tr>
<td>uuid</td>
<td>(UUID) WAF appliance identifier within the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>name</td>
<td>(Text) The name of the WAF appliance as defined by a user. This is unique in subscription. Valid action: Update</td>
</tr>
<tr>
<td>hostname</td>
<td>(Text) The hostname of the appliance (retrieved asynchronously).</td>
</tr>
<tr>
<td>lastPollDate</td>
<td>(Date) The last poll date of appliance. If the appliance does not poll for the commands, connectivity error or failure status is detected.</td>
</tr>
<tr>
<td>applianceCreated</td>
<td>(Long) Appliance creation date in the Qualys Cloud WAF Service.</td>
</tr>
<tr>
<td>applianceVersion</td>
<td>(Text) Software version of Qualys WAF Service running on the appliance.</td>
</tr>
<tr>
<td>status</td>
<td>(Text) The appliance status information: ACTIVE, INACTIVE, DEPROVISIONING (appliance is currently cleaning configuration before shutdown for appliance deletion), DISABLED (appliance is removed from any cluster and is been destroyed. No appliance records should be retrieved with that status which is temporary).</td>
</tr>
<tr>
<td>pollStatus</td>
<td>(Text) The appliance polling status computed from lastPollDate. The appliance is considered as unreachable with a 5 minutes timeout (Timeout can be changed later). The valid values are: POLLING: last poll date is greater than the timeout (5 minutes). NOT_POLLING: the last poll date is less than timeout (5 minutes).</td>
</tr>
<tr>
<td>heartbeatGenerated</td>
<td>(Date) Date when the WAF appliance generates a new Heartbeat message. It is not updated directly, but retrieved asynchronously and is synchronized when Qualys Cloud platform/WAF module updates their respective records. The Heartbeat message contains system information such as systemOs, systemRam, and systemType.</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>heartbeatProcessed</td>
<td>(Date) Date when the WAF server processed the Heartbeat message (stored and available). It is not updated directly, but retrieved asynchronously and is synchronized when Qualys Cloud platform/WAF module updates their respective records. The Heartbeat message contains system information such as systemOs, systemRam, and systemType.</td>
</tr>
<tr>
<td>systemOs</td>
<td>(Text) The operating system detected on the WAF appliance.</td>
</tr>
<tr>
<td>systemRam</td>
<td>(Long) Total RAM detected on the WAF appliance.</td>
</tr>
<tr>
<td>systemType</td>
<td>(Text) System deployment type.</td>
</tr>
<tr>
<td></td>
<td>EC2: for AWS</td>
</tr>
<tr>
<td></td>
<td>Other: for undefined virtual platforms</td>
</tr>
<tr>
<td>systemEc2InstanceId</td>
<td>(Text) Instance identifier of the system on the WAF appliance. For EC2, unique AWS virtual machine identifier. Valid only for EC2 systems.</td>
</tr>
<tr>
<td>systemEc2InstanceType</td>
<td>(Text) Type of instance. For EC2 type appliance, gives AWS virtual machine scheme. (such as 'x-large') Valid only for EC2 systems</td>
</tr>
<tr>
<td>systemEc2Amild</td>
<td>(Text) For EC2 type appliance, the virtual machine image identifier used to create AWS virtual machine. Valid only for EC2 systems.</td>
</tr>
<tr>
<td>systemCpusCount</td>
<td>(Long) Number of CPU (socket) physical or virtual.</td>
</tr>
<tr>
<td>systemCpusCores</td>
<td>(Long) Number of core(s) per CPU (socket).</td>
</tr>
<tr>
<td>systemCpusSpeed</td>
<td>(Float) CPU frequency in KHz such as 2678.9 for a 2.6 GHz CPU.</td>
</tr>
<tr>
<td>systemCpusModel</td>
<td>(Text) The type of CPU model on the WAF appliance. Simple text identifier that describes CPU virtualized (Such as &quot;Intel Bi-Xeon xxxx&quot;)</td>
</tr>
<tr>
<td>configRulesVersion</td>
<td>(Text) The version of security rules (rules provided by Qualys) that are available on the WAF appliance.</td>
</tr>
<tr>
<td>configVersion</td>
<td>(Text) Generated configuration version. The version is not updated directly, but is retrieved asynchronously and is synchronized when WAF module/Qualys Cloud platform update their respective records.</td>
</tr>
<tr>
<td>configGenerated</td>
<td>(Date) Date when the configuration is generated on the Qualys Cloud platform.</td>
</tr>
<tr>
<td>Element (Data Type)</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>ip</td>
<td>(Text) The IP address of the WAF appliance. This field may not be configured depending on appliance code, version, and network settings. The appliance may not be able to detect its own IP address, if multiple network devices exist.</td>
</tr>
<tr>
<td>cluster.id</td>
<td>(Long) Cluster asset identifier within Qualys Cloud platform identifying appliance's group.</td>
</tr>
<tr>
<td>cluster.uuid</td>
<td>(UUID) Cluster asset identifier within WAF module identifying the appliance's group.</td>
</tr>
<tr>
<td>cluster.name</td>
<td>(Text) Cluster asset name within WAF module identifying the appliance's group.</td>
</tr>
</tbody>
</table>