Qualys Cloud Suite API Release Notes

Version 2.30

Qualys Cloud Suite API gives you many ways to integrate your programs and API calls with Qualys capabilities. You'll find all the details in our user guides, available at the time of release. Just log in to your Qualys account and go to Help > Resources.

What's New

Host Asset Management API - new Amazon EC2 instance metadata

Tag Management API - dynamic tag support for Amazon EC2 metadata

New WAF API v2

New Admin - User API

WAS API - removed support for NOT EQUALS in delete request

URL to the Qualys API Server

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

Account Location	API Server URL
Qualys US Platform 1	https://qualysapi.qualys.com
Qualys US Platform 2	https://qualysapi.qg2.apps.qualys.com
Qualys US Platform 3	https://qualysapi.qg3.apps.qualys.com
Qualys EU Platform 1	https://qualysapi.qualys.eu
Qualys EU Platform 2	https://qualysapi.qg2.apps.qualys.eu
Qualys India Platform 1	https://qualysapi.qg1.apps.qualys.in
Qualys Private Cloud Platform	https://qualysapi.<customer_base_url>

The Qualys API documentation and sample code use the API server URL for the Qualys US Platform 1. If your account is located on another platform, please replace this URL with the appropriate server URL for your account.

Host Asset Management API - new Amazon EC2 instance metadata

The Host Asset Management API now adds additional metadata of Amazon EC2 hosts when inventoried using the Qualys EC2 Connector. The API output schema is changed to add new tags for EC2 metadata, populated with respective values for EC2 assets and empty for non EC2 assets.

API Request:

```
curl -n -u "USERNAME:PASSWORD"
"https://qualysapi.qualys.com/qps/rest/2.0/get/am/hostasset/709838"
```

XML Output:

New tags for EC2 assets appear in the **<Ec2AssetSourceSimple>** element, shown in bold below.

```
<?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/a
m/hostasset.xsd">
    <responseCode>SUCCESS</responseCode>
    <count>1</count>
    <data>
        <HostAsset>
            <id>709838</id>
            <name>my-ec2-target</name>
            <created>2017-07-27T18:14:28Z</created>
            <modified>2017-07-27T18:21:31Z</modified>
            <type>HOST</type>
            <tags>
                <list/>
            </tags>
            <sourceInfo>
                <list>
                    <Ec2AssetSourceSimple>
                        <firstDiscovered>2017-07-
27T18:14:28Z</firstDiscovered>
                        <lastUpdated>2017-07-27T19:51:03Z</lastUpdated>
                        <assetId>709838</assetId>
                        <ec2InstanceTags>
                            <tags>
                                <list>
                                    <EC2Tags>
                                        <key>Department</key>
                                        <value>Security</value>
                                    </EC2Tags>
```

```
                                <EC2Tags>
                                    <key>Owner</key>
                                    <value>Jason Kim</value>
                                </EC2Tags>
                                <EC2Tags>
                                    <key>Email</key>
                                <value>jkim@acme.com</value>
                                </EC2Tags>
                                <EC2Tags>
                                    <key>JIRA</key>
                                    <value>POR-6719</value>
                                </EC2Tags>
                                <EC2Tags>
                                    <key>Name</key>
                                    <value>my-ec2-target</value>
                                </EC2Tags>
                                <EC2Tags>
                                    <key>Lifecycle</key>
                                    <value>20171231</value>
                                </EC2Tags>
                            </list>
                        </tags>
                    </ec2InstanceTags>
                    <availabilityZone>us-east-1e</availabilityZone>
                    <instanceId>i-023b166432b1c7afc</instanceId>
                    <instanceType>t2.medium</instanceType>
                    <createdDate>2017-07-27T19:58:34Z</createdDate>
                    <instanceState>STOPPED</instanceState>
                    <groupId>sg-6b619117</groupId>
                    <groupName>default</groupName>
                    <spotInstance>true</spotInstance>
                    <accountId>205767712438</accountId>
                    <subnetId>subnet-7bbbcd56</subnetId>
                    <vpcId>vpc-2da7154b</vpcId>
                    <region>us-east-1</region>
                    <zone>VPC</zone>
                    <imageId>ami-22ce4934</imageId>
                    <publicIpAddress>127.0.0.1</publicIpAddress>
                    <privateIpAddress>10.97.15.117</privateIpAddress>
                    <monitoringEnabled>false</monitoringEnabled>
                </Ec2AssetSourceSimple>
            </list>
        </sourceInfo>
        <qwebHostId>12864</qwebHostId>
        <os>Linux</os>
        <address>10.97.15.117</address>
        <trackingMethod>INSTANCE_ID</trackingMethod>
        <openPort>
            <list/>
```

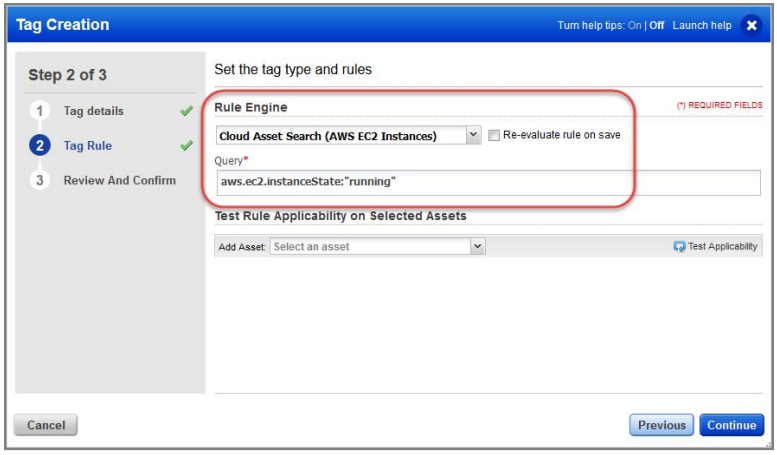
```
                </openPort>
                <software>
                    <list/>
                </software>
                <vuln>
                    <list/>
                </vuln>
                <processor>
                    <list/>
                </processor>
                <volume>
                    <list/>
                </volume>
                <account>
                    <list/>
                </account>
                <networkInterface>
                    <list>
                        <HostAssetInterface>
                            <interfaceId>eni-09f901fe</interfaceId>
                            <interfaceName>Primary network
interface</interfaceName>
                            <type>PRIVATE</type>
                            <address>10.97.15.117</address>
                        </HostAssetInterface>
                    </list>
                </networkInterface>
            </HostAsset>
        </data>
    </ServiceResponse>
```

Tag Management API - dynamic tag support for Amazon EC2 metadata

We've introduced a new dynamic tag rule type that allows you to tag your EC2 instances based on EC2 metadata attributes as collected by the EC2 Connector. For each tag rule you'll provide a search query with EC2 instance information like the public and private DNS name, image ID, VPC ID, instance state, instance type and more. You can also create tags in Qualys that are based on tags defined for your EC2 instances in AWS.

Tag Creation in the UI

In AssetView, go to Assets > Tags > New Tag. Under Tag Rule, choose "Cloud Asset Search (AWS EC2 Instances)" and enter your query. It's easy - start typing in the Query field and we'll show you the EC2 attributes you can search.



Check out these sample queries:

Find running EC2 instances:
`aws.ec2.instanceState:"running"`

Find EC2 instances with type "t2.medium" in the region "US West (N. California)":
`aws.ec2.instanceType:"t2.medium" and aws.ec2.region:"US West (N. California"`

Find EC2 instances with AWS tag key "department" and value "stage":
`aws.tags.key:department and aws.tags.value:stage`

Find EC2 instances created from pre-approved AMIs (ami-1231231 and ami-8790707):
`aws.ec2.imageId:ami-1231231 and aws.ec2.imageId:ami-8970707`

Find EC2 instances with specific criteria for scanning:
`aws.ec2.region:"US East (N. Virginia)" and aws.ec2.VPCid: [vpc-12321213, vpc-342342] and aws.ec2.instanceState:"running" and aws.tags.key:environment and aws.tags.value:build`

API Changes

The Tag Management API provides a suite of API functions for managing tags. We've made these changes to support the new Cloud Asset rule type:

1) Create Cloud Asset tags using the create tag operation. Specify CLOUD_ASSET for the ruleType and your search query as the ruleText in the request POST data.

2) CLOUD_ASSET is now a valid value when filtering by ruleType. Tag operations that support filtering include search tags, count tags and evaluate tag.

3) CLOUD_ASSET appears in the XML output when a Cloud Asset tag is returned by a tag operation. For example, you'll see rule type CLOUD_ASSET when you get or evaluate a Cloud Asset tag.

API Samples

Sample 1 - Create Cloud Asset Tag

API Request:

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

Request POST data (file.xml):

```
<?xml version="1.0" encoding="UTF-8" ?>
<ServiceRequest>
  <data>
    <Tag>
      <name>EC2 Running</name>
      <ruleType>CLOUD_ASSET</ruleType>
      <ruleText>aws.ec2.instanceState:running;</ruleText>
      <color>#FFFFFF</color>
    </Tag>
  </data>
</ServiceRequest>
```

XML Output:

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

```
d/2.0/am/tag.xsd">
  <responseCode>SUCCESS</responseCode>
  <count>1</count>
  <data>
    <Tag>
      <id>1589217</id>
      <name>EC2 Running</name>
      <created>2017-08-22T19:04:20Z</created>
      <modified>2017-08-22T19:04:20Z</modified>
      <color>#FFFFFF</color>
      <ruleText>aws.ec2.instanceState:running;</ruleText>
      <ruleType>CLOUD_ASSET</ruleType>
      <children>
        <list/>
      </children>
    </Tag>
  </data>
</ServiceResponse>
```

Sample 2 - Get Tag Info

API Request:

```
curl -n -u "USERNAME:PASSWORD"
"https://qualysapi.qualys.com/qps/rest/2.0/get/am/tag/9000238"
```

XML Output:

```
<?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/a
m/tag.xsd">
   <responseCode>SUCCESS</responseCode>
   <count>1</count>
   <data>
       <Tag>
           <id>9000238</id>
           <name>My EC2 Tag</name>
           <created>2017-08-24T00:44:54Z</created>
           <modified>2017-08-24T21:22:24Z</modified>
           <color>#FFFFFF</color>
           <ruleText>aws.tags.value:eng*;</ruleText>
           <ruleType>CLOUD_ASSET</ruleType>
           <children>
               <list/>
           </children>
       </Tag>
   </data>
</ServiceResponse>
```

Sample 3 - Update Tag

API Request:

```
curl -u "USERNAME:PASSWORD" -H "Content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/update/am/tag/9002049"
< file.xml
```

Request POST data (file.xml):

```
<?xml version="1.0" encoding="UTF-8" ?>
<ServiceRequest>
   <data>
       <Tag>
           <name>My EC2 Tag</name>
           <ruleType>CLOUD_ASSET</ruleType>
           <ruleText>aws.ec2.instanceState:stopped;</ruleText>

       </Tag>
   </data>
</ServiceRequest>
```

XML Output:

```
<?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/a
m/tag.xsd">
   <responseCode>SUCCESS</responseCode>
   <count>1</count>
   <data>
       <Tag>
           <id>9002049</id>
       </Tag>
   </data>
</ServiceResponse>
```

Sample 4 - Evaluate Tags (filter by ruleType)

API Request:

```
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/2.0/evaluate/am/tag"
```

Request POST data (file.xml):

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

```
  <filters>
    <Criteria field="ruleType"
operator="EQUALS">CLOUD_ASSET</Criteria>
    </filters>
</ServiceRequest>
```

XML Output:

```
<?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/a
m/tag.xsd">
   <responseCode>SUCCESS</responseCode>
   <count>1</count>
   <data>
       <Tag>
           <id>9000238</id>
           <name>My EC2 Tag</name>
           <created>2017-08-24T00:44:54Z</created>
           <modified>2017-08-24T21:22:24Z</modified>
           <color>#FFFFFF</color>
           <ruleText>aws.tags.value:eng*;</ruleText>
           <ruleType>CLOUD_ASSET</ruleType>
           <children>
               <list/>
           </children>
       </Tag>
   </data>
</ServiceResponse>
```

New WAF API v2

The new WAF API v2 allows you to integrate your programs and APIs with Web Application Firewall (WAF) capabilities and WAF data in your Qualys account. The new WAF API v2 is a significant upgrade from the previous WAF API version.

The new WAF API v2 supports many function calls as follows.

API	Supported calls
Web Applications API	count, get (details), search, create, update, bulk update, delete, bulk delete
Web Servers API	count, get (details), search, create, update, bulk update, delete, bulk delete
Healthchecks API	count, get (details), search, create, update, bulk update, delete, bulk delete
SSL Certificates API	count, get (details), search, create, update, bulk update, delete, bulk delete
Custom Response Pages API	count, get (details), search, create, update, bulk update, delete, bulk delete
Security Policies API	count, get (details), search, create, update, bulk update, delete, bulk delete
HTTP Profiles API	count, get (details), search, create, update, bulk update, delete, bulk delete
Custom Rules API	count, get (details), search, create, update, bulk update, delete, bulk delete
Clusters API	count, get (details), search, create, update, bulk update, delete, bulk delete
Appliance API	count, get (details), search, delete

Qualys WAF API v2 User Guide will be provided at the time of release. This guide has complete details and samples to help you with using the WAF API v2.

Sample 1 - Create Web Application

Request:

```
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 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>
            <securityPolicy><id>30682</id></securityPolicy>
            <httpProfile><id>1001</id></httpProfile>
        </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/wa
f/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>
                <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>
                <urls/>
                <webServer>
                    <id>1001</id>
                    <uuid>315cc797-3c73-4721-ba42-263e7e7b6cbb</uuid>
                    <name>
                        <![CDATA[First Pool]]>
                    </name>
                </webServer>
                <persistencyEnabled>false</persistencyEnabled>
                <blockingMode>false</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>
                <status>DOWN</status>
                <sslEnabled>false</sslEnabled>
                <deploymentStatus>PENDING_DEPLOY</deploymentStatus>
                <deployed>2017-06-01T09:22:47Z</deployed>
            </WebApp>
        </data>
    </ServiceResponse>
```

Sample 2 - Get Web Application Details

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/wa
f/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>
            <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>
        <url>https://site01.xfuentes-docker</url>
        <urls/>
        <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>
        <failureResponseCode>503</failureResponseCode>
        <certificate>
            <id>1</id>
            <uuid>a21b4a1b-de54-45e8-9d29-204444cef5bb</uuid>
            <name>
```

```
                        <![CDATA[Site01 Cert]]>
                    </name>
                </certificate>
                <sslProtocols>
                    <![CDATA[SSLV3,TLS10,TLS11,TLS12]]>
                </sslProtocols>
                <sslCiphers>
                    <![CDATA[ECDHE-RSA-AES256-GCM-SHA384,ECDHE-ECDSA-AES256-
GCM-SHA384,ECDHE-RSA-AES256-SHA384,ECDHE-ECDSA-AES256-SHA384,ECDHE-RSA-
AES256-SHA,ECDHE-ECDSA-AES256-SHA,DHE-DSS-AES256-GCM-SHA384,...]]>
                </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>
                        <name>
                            <![CDATA[Test custom rule]]>
                        </name>
                    </CustomRule>
                    <CustomRule>
                        <id>2001</id>
                        <uuid>c64c3008-c1af-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>
```

Sample 3 - Search security policies

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/wa
```

```
f/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>
            <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>
            <updated>2017-03-27T13:18:47Z</updated>
            <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>
            <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>
        <updated>2017-03-27T13:18:47Z</updated>
        <system>1</system>
        <applicationSecurity>
            <commandExecution>
                <confidence>MEDIUM</confidence>
            </commandExecution>
            <crossSiteScripting>
```

```
                        <confidence>MEDIUM</confidence>
                    </crossSiteScripting>
                    <directoryTraversal>
                        <confidence>MEDIUM</confidence>
                    </directoryTraversal>
                    <formatStringAttacks>
                        <confidence>MEDIUM</confidence>
                    </formatStringAttacks>
                    <informationLeakage>
                        <confidence>MEDIUM</confidence>
                    </informationLeakage>
                    <ldapInjection>
                        <confidence>MEDIUM</confidence>
                    </ldapInjection>
                    <lfiAttacks>
                        <confidence>MEDIUM</confidence>
                    </lfiAttacks>
                    <pathTraversal>
                        <confidence>MEDIUM</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>25</loggingThreshold>
                    <blockingThreshold>75</blockingThreshold>
                </threatLevel>
            </SecurityPolicy>
        </data>
    </ServiceResponse>
```

New Admin - User API

We have introduced a new API (https://<baseurl>/qps/rest/1.0/{action}/admin/user) that will give the list of users along with their tags to the authorized user. Currently, we support three actions for the users: search, count, and get details of a user.

URL: https://qualysapi.qualys.com/qps/rest/1.0/{action}/admin/user

Method allowed: POST, GET

Search Users (POST)

You can search for users by using different filters for user ID, username, email, tags, and module names. If no filter is specified, all users in the user's scope are listed.

New XSD: User.XSD

API Request:

```
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/1.0/search/admin/user" < file.xml
Note: “file.xml” contains the request POST data.
```

Request POST Data (file.xml):

```
<ServiceRequest>
    <filters>
        <Criteria field="username" operator="CONTAINS">10</Criteria>
    </filters>
</ServiceRequest>
```

XML 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/1.0/a
dmin/user.xsd">
    <responseCode>SUCCESS</responseCode>
    <count>1</count>
    <hasMoreRecords>false</hasMoreRecords>
    <data>
        <User>
            <id>3989626</id>
            <username>user_js10</username>
                <firstName><![CDATA[John]]></firstName>
                <lastName><![CDATA[Smith]]></lastName>
            <emailAddress>john.smith@afco.com</emailAddress>
```

```
            <tags>
                <count>1</count>
                <list>
                    <Tag>
                        <id>8721654</id>
                        <name>
                            <![CDATA[Unassigned Business Unit]]>
                        </name>
                    </Tag>
                </list>
            </tags>
            <modules>
                <count>5</count>
                <list>
                    <Module>QWEB_PCI</Module>
                    <Module>WAS</Module>
                    <Module>ADMIN</Module>
                    <Module>ASSET_MANAGEMENT</Module>
                    <Module>QWEB_VM</Module>
                </list>
            </modules>
        </User>
    </data>
</ServiceResponse>
```

Count Users (POST)

Returns the total number of users in the user's scope.

New XSD: User.XSD

API Request:

```
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/1.0/count/admin/user" < file.xml

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

Request POST Data (file.xml):

```
<ServiceRequest>
    <filters>
        <Criteria field="username" operator="CONTAINS">10</Criteria>
    </filters>
</ServiceRequest>
```

XML 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/1.0/a
dmin/user.xsd">
    <responseCode>SUCCESS</responseCode>
    <count>1</count>
</ServiceResponse>
```

Get details of a User

View details for a user in the user's scope. You can use search action to find a user ID to use as input.

New XSD: User.XSD

API Request:

```
curl -u "USERNAME:PASSWORD" " -X GET -H "Content-type: text/xml"
"https://qualysapi.qualys.com/qps/rest/1.0/get/admin/user/3989626" <
```

XML 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/1.0/a
dmin/user.xsd">
    <responseCode>SUCCESS</responseCode>
    <count>1</count>
    <data>
        <User>
            <id>3989626</id>
             <username>user_js10</username>
                <firstName><![CDATA[John]]></firstName>
                <lastName><![CDATA[Smith]]></lastName>
            <emailAddress>john.smith@afco.com</emailAddress>
            <tags>
                <count>1</count>
                <list>
                    <Tag>
                        <id>8721654</id>
                        <name>
                            <![CDATA[Unassigned Business Unit]]>
                        </name>
                    </Tag>
                </list>
            </tags>
            <modules>
                <count>5</count>
                <list>
                    <Module>WAS</Module>
                    <Module>ADMIN</Module>
```

```
                        <Module>QWEB_PCI</Module>
                        <Module>ASSET_MANAGEMENT</Module>
                        <Module>QWEB_VM</Module>
                    </list>
                </modules>
            </User>
        </data>
    </ServiceResponse>
```

WAS API - removed support for NOT EQUALS in delete request

Using the WAS API the NOT EQUALS operator is no longer supported in delete requests. This change applies to making delete requests using the following APIs:

1) Web Application API

2) Authentication API

3) Scan API

4) Schedule API

5) Report API

6) Option Profile API

7) Finding API