Qualys Cloud Platform v2.x

API Release Notes

Version 2.35

November 21, 2018

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

Dynamic tagging for AWS, AZURE, GCP

New Application Security Categories added in Security Policies

New Conditions Added to Custom Rule

Added Support for Response Headers to Custom Rule

Schedule Reactivation for Ignored Finding

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.

Dynamic tagging for AWS, AZURE, GCP

APIs affected	/qps/rest/2.0/create/am/tag /qps/rest/2.0/update/am/tag /qps/rest/2.0/get/am/tag /qps/rest/2.0/search/am/tag /qps/rest/2.0/count/am/tag /qps/rest/2.0/evaluate/am/tag
New or Updated APIs	Updated
DTD or XSD changes	Yes

The Asset Management and Tagging API has been updated to allow dynamic tagging for AWS (EC2), AZURE, and GCP assets. You can now group your cloud assets according to the cloud provider they belong to. Tags are applied to assets found by cloud agents (AWS, AZURE, GCP) and EC2 connectors (AWS).

Input Parameters

You'll need to provide the cloud provider information in the service request of an API call.

Sample 1 - Create Cloud Asset Tag for AWS

Sample 2 - Create Cloud Asset Tag for Azure

Sample 3 - Create Cloud Asset Tag for GCP

Sample 4 - Update Cloud Asset Tag ruleText

Sample 5 - Get Cloud Asset Tag

Sample 6 - Search Cloud Asset Tag

Sample 7 - Count Cloud Asset Tags

Sample 8 - Evaluate Cloud Asset Tag

Sample 1 - Create Cloud Asset Tag for AWS

/qps/rest/2.0/create/am/tag

[POST]

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
```

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

Request POST Data:

```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <data>
        <Tag>
            <name>AWS Tag</name>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>EC2</provider>
            <ruleText>aws.ec2.region.code:us-east-1</ruleText>
         </Tag>
    </data>
</ServiceRequest>
```

Note: For Cloud Asset Tag, the <provider> node is mandatory. For other rule types, <provider> node is optional. <provider> node accepts values: EC2, AZURE, GCP.

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>101335613</id>
            <name>AWS Tag</name>
            <created>2018-10-30T18:20:23Z</created>
            <modified>2018-10-30T18:20:23Z</modified>
            <ruleText>aws.ec2.region.code: us-east-1</ruleText>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>EC2</provider>
        </Tag>
    </data>
</ServiceResponse>
```

Sample 2 - Create Cloud Asset Tag for Azure

/qps/rest/2.0/create/am/tag

[POST]

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
```

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

Request POST Data:

```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <data>
        <Tag>
            <name>Azure Tag</name>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>AZURE</provider>
            <ruleText>azure.vm.location:westus</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>101335413</id>
            <name>Azure Tag</name>
            <created>2018-10-30T18:32:14Z</created>
            <modified>2018-10-30T18:32:14Z</modified>
            <ruleText>azure.vm.location:westus</ruleText>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>AZURE</provider>
        </Tag>
    </data>
</ServiceResponse>
```

Sample 3 - Create Cloud Asset Tag for GCP

/qps/rest/2.0/create/am/tag

[POST]

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
```

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

Request POST Data:

```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <data>
        <Tag>
            <name>GCP Tag</name>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>GCP</provider>
            <ruleText>gcp.compute.zone:us-east1-d</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>101335414</id>
            <name>GCP Tag</name>
            <created>2018-10-30T18:34:21Z</created>
            <modified>2018-10-30T18:34:21Z</modified>
            <ruleText>gcp.compute.zone:us-east1-d</ruleText>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>GCP</provider>
        </Tag>
    </data>
</ServiceResponse>
```

Sample 4 - Update Cloud Asset Tag ruleText

/qps/rest/2.0/update/am/tag

[POST]

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/101335414 <
file.xml
```

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

Request POST Data:

```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <data>
        <Tag>
         <ruleText>gcp.compute.publicIpAddress:104.196.75.216</ruleText>
        </Tag>
    </data>
</ServiceRequest>
```

Note: You cannot change value of the <provider> node during update.

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>101335414</id>
            <name>GCP Tag</name>
            <created>2018-10-30T18:34:21Z</created>
            <modified>2018-10-30T18:34:21Z</modified>
          <ruleText>gcp.compute.publicIpAddress:104.196.75.216</ruleText>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>GCP</provider>
        </Tag>
    </data>
</ServiceResponse>
```

Sample 5 - Get Cloud Asset Tag

/qps/rest/2.0/get/am/tag

[GET]

API request:

```
curl -n -u "USERNAME:PASSWORD"
"https://qualysapi.qualys.com/qps/rest/2.0/get/am/tag/101335414"
```

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>101335414</id>
            <name>GCP Tag</name>
            <created>2018-10-30T18:34:21Z</created>
            <modified>2018-10-30T18:34:21Z</modified>
            <ruleText>gcp.compute.zone:us-east1-d</ruleText>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>GCP</provider>
        </Tag>
    </data>
</ServiceResponse>
```

Sample 6 - Search Cloud Asset Tag

/qps/rest/2.0/search/am/tag

[POST]

API request:

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

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

Request POST Data:

```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <filters>
       <Criteria field="ruleType" operator="EQUALS">CLOUD_ASSET</Criteria>
        <Criteria field="provider" operator="EQUALS">GCP</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>
    <hasMoreRecords>false</hasMoreRecords>
    <data>
        <Tag>
            <id>101335414</id>
            <name>GCP Tag</name>
            <created>2018-10-30T18:34:21Z</created>
            <modified>2018-10-30T18:34:21Z</modified>
            <ruleText>gcp.compute.zone:us-east1-d</ruleText>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>GCP</provider>
        </Tag>
    </data>
</ServiceResponse>
```

Sample 7 - Count Cloud Asset Tags

/qps/rest/2.0/count/am/tag

[POST]

API request:

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

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

Request POST Data:

```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <filters>
       <Criteria field="ruleType" operator="EQUALS">CLOUD_ASSET</Criteria>
        <Criteria field="provider" operator="EQUALS">AZURE</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>2</count>
</ServiceResponse>
```

Sample 8 - Evaluate Cloud Asset Tag

/qps/rest/2.0/evaluate/am/tag

[POST]

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/101335414 <
file.xml
```

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

Request POST Data:

```
<?xml version="1.0" encoding="UTF-8"?>
<ServiceRequest>
</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>101335414</id>
            <name>GCP Tag</name>
            <created>2018-10-30T18:34:21Z</created>
            <modified>2018-10-30T18:34:21Z</modified>
            <ruleText>gcp.compute.zone:us-east1-d</ruleText>
            <ruleType>CLOUD_ASSET</ruleType>
            <provider>GCP</provider>
        </Tag>
    </data>
</ServiceResponse>
```

XSD update

Following element is added to /qps/xsd/2.0/am/tag.xsd

```
...
<element name="provider" type="string" minOccurs="0"/>
...
```

New Application Security Categories added in Security Policies

API affected	/qps/rest/2.0/get/waf/securitypolicy/<id> /qps/rest/2.0/search/waf/securitypolicy /qps/rest/2.0/create/waf/securitypolicy /qps/rest/2.0/update/waf/securitypolicy
New or Updated APIs	Updated
DTD or XSD changes	Yes

We have added support for four new application security categories. Add the new categories as elements under the applicationSecurity parameter and set confidence values for them.

Removed API support for the following application security categories: directoryTraversal, formatStringAttacks and pathTraversal.

Input Parameters

New elements for input parameter is described below.

Parameter	Description
applicationSecurity	Specify the event confidence by name (DISABLED, LOW, MEDIUM or HIGH) or by value (20-80) for the following new application security categories: -rpo -xmlInjection -elInjection -codeInjection RPO stands for relative path overwrite and elInjection is Expression level injection.

Sample 1 - Get Security Policy Profile

The response XML shows the new security categories along with their confidence values in the fetched security policy.

API request:

```
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/securitypolicy/3
3481
```

XML 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/wa
f/securitypolicy.xsd">
    <responseCode>SUCCESS</responseCode>
    <count>1</count>
    <data>
        <SecurityPolicy>
            <id>33481</id>
            ...
            <applicationSecurity>
                ...
                <rpo>
                    <confidence>HIGH</confidence>
                </rpo>
                <xmlInjection>
                    <confidence>HIGH</confidence>
                </xmlInjection>
                <elInjection>
                    <confidence>HIGH</confidence>
                </elInjection>
                <codeInjection>
                    <confidence>HIGH</confidence>
                </codeInjection>
                ...>
            </applicationSecurity>
            ...
    </data>
</ServiceResponse>
```

Sample 2 - Create security policy

Let us create a customized security policy with new security categories and set their confidence values.

API request:

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

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

Request POST Data:

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

```
                    ...
                    <rpo>
                        <confidence>HIGH</confidence>
                    </rpo>
                    <xmlInjection>
                        <confidence>HIGH</confidence>
                    </xmlInjection>
                    <elInjection>
                        <confidence>HIGH</confidence>
                    </elInjection>
                    <codeInjection>
                        <confidence>HIGH</confidence>
                    </codeInjection>
                    ...
                </applicationSecurity>
                <threatLevel>
                    <loggingThreshold>35</loggingThreshold>
                    <blockingThreshold>65</blockingThreshold>
                </threatLevel>
            </SecurityPolicy>
        </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/w
    af/securitypolicy.xsd">
        <responseCode>SUCCESS</responseCode>
        <count>1</count>
        <data>
            <SecurityPolicy>
                <id>33482</id>
                ...
                <applicationSecurity>
                    ...
                    <rpo>
                        <confidence>HIGH</confidence>
                    </rpo>
                    <xmlInjection>
                        <confidence>HIGH</confidence>
                    </xmlInjection>
                    <elInjection>
                        <confidence>HIGH</confidence>
                    </elInjection>
                    <codeInjection>
                        <confidence>HIGH</confidence>
                    </codeInjection>
```

```
                ...
            </applicationSecurity>
            ...
    </data>
</ServiceResponse>
```

Sample 3 - Update security policy

Let us update a customized security policy with new security categories and set their confidence values.

API 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>
    <data>
        <SecurityPolicy>
            ...
            <applicationSecurity>
                ...
                <rpo>
                    <confidence>HIGH</confidence>
                </rpo>
                <xmlInjection>
                    <confidence>HIGH</confidence>
                </xmlInjection>
                <elInjection>
                    <confidence>HIGH</confidence>
                </elInjection>
                <codeInjection>
                    <confidence>HIGH</confidence>
                </codeInjection>
                ...
            </applicationSecurity>
            ...
        </SecurityPolicy>
    </data>
</ServiceRequest>
```

XML output:

```
...
    <data>
        <SecurityPolicy>
        <id>33481</id>
            ...
            <applicationSecurity>
                ...
                <rpo>
                    <confidence>HIGH</confidence>
                </rpo>
                <xmlInjection>
                    <confidence>HIGH</confidence>
                </xmlInjection>
                <elInjection>
                    <confidence>HIGH</confidence>
                </elInjection>
                <codeInjection>
                    <confidence>HIGH</confidence>
                </codeInjection>
                ...
            </applicationSecurity>
            ...
```

XSD update

Changes in securitypolicy.xsd (qps/xsd/2.0/waf/securitypolicy.xsd)

```
...
    <xs:complexType name="SecurityPolicy">
            ...
            <xs:element name="applicationSecurity" minOccurs="0">
                <xs:complexType>
                    <xs:sequence>
                        ...
                        <xs:element name="rpo" type="Confidence"
minOccurs="0" />
                       <xs:element name="xmlInjection" type="Confidence"
minOccurs="0" />
                        <xs:element name="elInjection" type="Confidence"
minOccurs="0" />
                      <xs:element name="codeInjection" type="Confidence"
minOccurs="0" />
                    ...
                    </xs:sequence>
                ...
</xs:schema>
```

New Conditions Added to Custom Rule

API affected	/qps/rest/2.0/get/waf/customrule/<id> /qps/rest/2.0/search/waf/customrule /qps/rest/2.0/create/waf/customrule /qps/rest/2.0/update/waf/customrule
New or Updated APIs	Updated
DTD or XSD changes	No

Custom Rule API now supports the following new conditions and operators for custom rule.

Key	Supported Operators
client.ssl.session.timeout	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL, LOWER, NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
server.ssl.cipher	EQUAL, NOT.EQUAL, MATCH, NOT.MATCH
server.ssl.protocol	EQUAL, NOT.EQUAL
server.ssl.session.timeout	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL, LOWER, NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
server.tcp.port	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL, LOWER, NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
request.body.length	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL, LOWER, NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
response.code	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL, LOWER, NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
response.date	EQUAL, NOT.EQUAL, BETWEEN, NOT.BETWEEN
response.duration	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL,LOWER, NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
response.header	EQUAL, NOT.EQUAL, MATCH, NOT.MATCH, DETECT
response.header.content-length	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL, LOWER, NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
response.header.content-type	EQUAL, NOT.EQUAL, MATCH, NOT.MATCH, DETECT

Key	Supported Operators
response.header.cookie	EQUAL, NOT.EQUAL, MATCH, NOT.MATCH, DETECT
response.header.cookie.name	EQUAL, NOT.EQUAL, MATCH, NOT.MATCH, DETECT
response.header.cookie.value	EQUAL, NOT.EQUAL, MATCH, NOT.MATCH, DETECT
response.header.line.length	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL, LOWER, NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
response.header.name	EQUAL, NOT.EQUAL, MATCH, NOT.MATCH, DETECT
response.header.value	EQUAL, NOT.EQUAL, MATCH, NOT.MATCH, DETECT
response.protocol	EQUAL, NOT.EQUAL
response.time	EQUAL, NOT.EQUAL, BETWEEN, NOT.BETWEEN
transaction.day	EQUAL, NOT.EQUAL
transaction.duration	EQUAL, NOT.EQUAL, IN-RANGE, NOT.IN-RANGE, GREATER, NOT.GREATER, GREATER-EQUAL, NOT.GREATER-EQUAL, LOWER,NOT.LOWER, LOWER-EQUAL, NOT.LOWER-EQUAL
transaction.time	EQUAL, NOT.EQUAL, BETWEEN, NOT.BETWEEN

Sample 1 - Get details on a Custom Rule

The response XML shows the new conditions in the fetched custom rule.

API request:

```
curl -u "USERNAME:PASSWORD" -X "GET" -H "Content-Type: text/xml"
https://qualysapi.qualys.com/qps/rest/2.0/get/waf/customrule/271201
```

XML output:

```
...
<RuleCondition>
  <subject><![CDATA[server.ssl.cipher]]></subject>
  <operator>EQUAL</operator>
  <value><![CDATA[RC4]]></value>
</RuleCondition>
<RuleCondition>
  <subject><![CDATA[server.ssl.protocol]]></subject>
  <operator>EQUAL</operator>
  <value><![CDATA[sslv2]]></value>
</RuleCondition>
....
```

Sample 2 - Create custom rule with new conditions

API 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 version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <data>
        <CustomRule>
            ...
            <conditions>
                ...
                <RuleCondition>
                    <subject>server.ssl.cipher</subject>
                    <operator>EQUAL</operator>
                    <value>RC4</value>
                </RuleCondition>
                <RuleCondition>
                    <subject>server.ssl.protocol</subject>
                    <operator>EQUAL</operator>
                    <value>sslv2</value>
                    </RuleCondition>
                </RuleCondition>
                ...
            </conditions>
            ...
        </CustomRule>
    </data>
</ServiceRequest>
```

XML output:

```
            ...
            <RuleCondition>
              <subject><![CDATA[server.ssl.cipher]]></subject>
              <operator>EQUAL</operator>
              <value><![CDATA[RC4]]></value>
            </RuleCondition>
            <RuleCondition>
              <subject><![CDATA[server.ssl.protocol]]></subject>
              <operator>EQUAL</operator>
              <value><![CDATA[sslv2]]></value>
            </RuleCondition>
            ...
```

Added Support for Response Headers to Custom Rule

API affected	/qps/rest/2.0/get/waf/customrule/<id> /qps/rest/2.0/search/waf/customrule /qps/rest/2.0/create/waf/customrule /qps/rest/2.0/update/waf/customrule
New or Updated APIs	Updated
DTD or XSD changes	Yes

We have added three new actions: insertHeader, rewriteHeader and stripHeader to the Custom Rule API. You can configure these actions to insert, modify or remove HTTP headers in responses when the conditions for the actions are met.

Input Parameters

New elements for input parameter is described below.

Parameter	Description
action	The action to execute if the rule conditions were met. Following are the new elements added in the action: -insertHeader - We will add an HTTP header to the response. You can add a security header which instructs the browser exactly how to behave when it handles your website's content and data. An example of a security header could be an XFO header to mitigate clickjacking attacks: x-frame-options: SAMEORIGIN. -rewriteHeader - We will set/modify an HTTP header present in the response. -stripHeader - We will delete an HTTP header present in the response.

Sample 1: Sample XML for inserting an HTTP header

API 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 version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <data>
           ...
            <conditions>
                <RuleCondition>
                    <subject>request.body.parameter</subject>
                    <key>blague</key>
                    <operator>EQUAL</operator>
                    <value>toto</value>
                </RuleCondition>
            </conditions>
            <action>
                <log>true</log>
                <insertHeader>
                        <name>abc</name>
                        <value>123</value>
                </insertHeader>
            </action>
        </CustomRule>
    </data>
</ServiceRequest>
```

XML output:

```
          ...
          <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>
          <insertHeader>
            <name><![CDATA[abc]]></name>
            <value><![CDATA[123]]></value>
          </insertHeader>
        </action>
      </CustomRule>
    ...
```

Sample 2: Sample XML for setting/modifying an HTTP header

API 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 version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <data>
            ...
            <conditions>
                <RuleCondition>
                    <subject>request.body.parameter</subject>
                    <key>blague</key>
                    <operator>EQUAL</operator>
                    <value>toto</value>
                </RuleCondition>
            </conditions>
            <action>
                <log>true</log>
                <rewriteHeader>
                        <name>abcd</name>
                        <value>1234</value>
                </rewriteHeader>
            </action>
        </CustomRule>
    </data>
</ServiceRequest>
```

XML output:

```
            ...
            <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>
            <rewriteHeader>
              <name><![CDATA[abcd]]></name>
              <value><![CDATA[1234]]></value>
```

```
          </rewriteHeader>
        </action>
      ...
```

Sample 3: Sample XML for deleting an HTTP header

API 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 version="1.0" encoding="UTF-8"?>
<ServiceRequest>
    <data>
        <CustomRule>
            <name>test_rule1</name>
            <description>for qualys</description>
            <conditions>
                <RuleCondition>
                    <subject>request.body.parameter</subject>
                    <key>blague</key>
                    <operator>EQUAL</operator>
                    <value>toto</value>
                </RuleCondition>
            </conditions>
            <action>
                <log>true</log>
                <stripHeader>
                        <name>abcd</name>
                </stripHeader>
            </action>
        </CustomRule>
    </data>
</ServiceRequest>
```

XML output:

```
          ...
          <RuleCondition>
            <subject><![CDATA[request.body.parameter]]></subject>
            <key><![CDATA[blague]]></key>
            <operator>EQUAL</operator>
            <value><![CDATA[toto]]></value>
          </RuleCondition>
        </conditions>
        <action>
```

```
        <log>true</log>
        <stripHeader>
          <name><![CDATA[abcd]]></name>
        </stripHeader>
      </action>
    ...
```

XSD update

Changes in securitypolicy.xsd (qps/xsd/2.0/waf/customrule.xsd).

```
    ...
    <xs:sequence>
       <xs:element name="insertHeader">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="name" minOccurs="1" maxOccurs="1"
type="Cdata" />
                    <xs:element name="value" minOccurs="1" maxOccurs="1"
type="Cdata" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:sequence>
       <xs:element name="rewriteHeader">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="name" minOccurs="1" maxOccurs="1"
type="Cdata" />
                    <xs:element name="value" minOccurs="1" maxOccurs="1"
type="Cdata" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    <xs:sequence>
       <xs:element name="stripHeader">
            <xs:complexType>
                <xs:sequence>
                    <xs:element name="name" minOccurs="1" maxOccurs="1"
type="Cdata" />
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
    ...
```

Schedule Reactivation for Ignored Finding

API affected	/qps/rest/3.0/ignore/was/finding
New or Updated APIs	Updated
DTD or XSD changes	Yes

You can now schedule a date or the number of days to reactivate an ignored finding. With two new parameters: reactivateDate and reactivateIn, you can let us know when an ignored finding should be reactivated again.

Input Parameters

New elements for input parameter is described below.

Parameter	Description
reactivateDate= {date}	Specify the date after which the ignored finding should be re-activated. The date/time is specified in YYYY-MM-DD format.
reactivateIn ={value}	Specify the number of days after which the ignored finding should be reactivated.
	Note: reactivateDate and reactivateIn are mutually exclusive parameters and cannot be used together. You can use only either of them for a finding.

Sample 1 - Specify a date for reactivation of an ignored finding

API request:

```
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/3.0/ignore/was/finding/"
```

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

Request POST Data:

```
<ServiceRequest>
 <data>
   <Finding>
      <id>927823</id>
      <ignoredReason>FALSE_POSITIVE</ignoredReason>
      <ignoredComment>test</ignoredComment>
      <reactivateDate>2018-11-14</reactivateDate>
   </Finding>
  </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/3.0/
was/finding.xsd">
    <responseCode>SUCCESS</responseCode>
    <count>1</count>
    <data>
        <Finding>
            <id>927823</id>
        </Finding>
    </data>
</ServiceResponse>
```

Sample 2 - Specify days for reactivation of an ignored finding

API request:

```
curl -u "USERNAME:PASSWORD" -H "content-type: text/xml" -X "POST"
--data-binary @-
"https://qualysapi.qualys.com/qps/rest/3.0/ignore/was/finding/"
```

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

Request POST Data:

```
<ServiceRequest>
   <data>
     <Finding>
      <id>927913</id>
      <ignoredReason>FALSE_POSITIVE</ignoredReason>
      <ignoredComment>test</ignoredComment>
      <reactivateIn>1</reactivateIn>
    </Finding>
  </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/3.0/w
as/finding.xsd">
    <responseCode>SUCCESS</responseCode>
    <count>1</count>
    <data>
```

```
        <Finding>
            <id>927913</id>
        </Finding>
    </data>
</ServiceResponse>
```

Sample 3 - Get details of an ignored finding

Let us fetch the details of an ignored finding to know if it has been scheduled for reactivation or not.

API request:

```
curl -u "USERNAME:PASSWORD"
"https://qualysapi.qualys.com/qps/rest/3.0/get/was/finding/927913
```

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/3.0/w
as/finding.xsd">
    <responseCode>SUCCESS</responseCode>
<count>1</count>
<data>
    <Finding>
        <id>927913</id>
        <qid>150117</qid>
        <name><![CDATA[Path-Based Cross-Site Scripting (XSS)]]></name>
            <type>VULNERABILITY</type>
            <findingType>QUALYS</findingType>
....
<ignoredDate>2018-11-14T05:30:18Z</ignoredDate>
<ignoredComment>
     <![CDATA[test]]>
</ignoredComment>
     <reactivateIn>1</reactivateIn>
     <reactivateDate>2018-11-15T00:00:00Z</reactivateDate>
   <retest/>
  </Finding>
 </data>
</ServiceResponse>
```

XSD update

Changes in finding.xsd (/qps/xsd/3.0/was/finding.xsd)

```
...
    <xs:complexType name="Finding">
        <xs:all>
            <xs:element name="id" type="xs:long" minOccurs="0"/>
```

```
            ...
            ...
             <!-- Following element will be available only if finding is
ignored -->
            <element name="reactivateIn" type="xs:int" minOccurs="0"/>
            <!-- Following element will be available only if finding is
ignored -->
            <element name="reactivateDate" type="xs:dateTime"
minOccurs="0"/>
            ...
            ...
        </xs:all>
</xs:complexType>
```