Table of Contents

About this Guide ......................................................................................................................... 4
About Qualys ................................................................................................................................. 4
Qualys Support .............................................................................................................................. 4

Accessing the APIs ..................................................................................................................... 5
Permissions required to use APIs .................................................................................................. 5
Qualys API URLs .......................................................................................................................... 5
Authentication for Gateway URLs ............................................................................................... 5
Online API Guide .......................................................................................................................... 6

Configurations ............................................................................................................................ 7
Get all configurations in your account .......................................................................................... 7
Get details for a specific configuration ........................................................................................ 8
Create a configuration .................................................................................................................. 8
Update a configuration .................................................................................................................. 10

Containers .................................................................................................................................. 12
Get runtime details of a container ................................................................................................. 12
Get runtime profile for a container ............................................................................................... 13
Build a security policy based on a container’s behavior ............................................................ 14
Assign a configuration to a container .......................................................................................... 14

Images ....................................................................................................................................... 16
Instrument image with Qualys instrumentation ........................................................................... 16
Get CRS configuration of an image with instrumentation .......................................................... 17
Get configuration ID for instrumented image ............................................................................... 18
Assign configuration to an image ................................................................................................. 19

Events ....................................................................................................................................... 20
Get all events in your account .................................................................................................... 20

Policies ....................................................................................................................................... 24
Get all policies in your account .................................................................................................... 24
Get details for a specific policy .................................................................................................... 26
Create a new security policy ........................................................................................................ 28
Update a security policy ............................................................................................................... 31
Delete a security policy ................................................................................................................ 34
About this Guide

This user guide is intended for application developers who will use the APIs for Container Runtime Security (CRS).

CRS provides runtime visibility and protection for containers. This is achieved by instrumenting images with Container Security components that gather functional-level behavioral data about the processes running within a container. This behavioral data is used by Container Security to visualize process activity. You can create and apply security policies that provide custom security controls based on the container’s activity.

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 founding member of the Cloud Security Alliance (CSA). For more information, please visit www.qualys.com

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 online support information at www.qualys.com/support/.
Accessing the APIs

Several features of Container Runtime Security are available through REST APIs.

Permissions required to use APIs

- User must have the Container Security (CS) module enabled
- User must have API ACCESS permission

Qualys API URLs

Container Security supports the Qualys API gateway for API requests.

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

Click here to identify your Qualys platform and get the API Gateway URL.

Authentication for Gateway URLs

You must authenticate to the Qualys Cloud Platform using Qualys account credentials (user name and password) and get the JSON Web Token (JWT) before you can start using the Gateway URLs. Use the Qualys Authentication API to get the JWT.

For example:

```
```

where gateway.qg1.apps.qualys.com is the base URL to the Qualys API server where your account is located.

- **username** and **password** are the credentials of the user account for Container Security
- **token** should be true
- **Content-Type** should be "application/x-www-form-urlencoded"

The Authentication API returns a JSON Web Token (JWT) which you can use for authentication during Container Security API calls. The token expires in 4 hours. You must regenerate the token to continue using the Container Security API.
Online API Guide

You can directly access an online API guide from the following URL

http://<QualysGatewayURL>/apidocs/csapi/v1.3/runtime

For example, if your account is on US Platform 1

https://gateway.qgl.apps.qualys.com/apidocs/csapi/v1.3/runtime
Configurations

Here is the list of the APIs we currently support for instrumentation configurations:

<table>
<thead>
<tr>
<th>API Objective</th>
<th>Operator</th>
<th>API Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get all configurations in your account</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/configs</td>
</tr>
<tr>
<td>Get details for a specific configuration</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/configs/{configId}</td>
</tr>
<tr>
<td>Create a new configuration</td>
<td>POST</td>
<td>/csapi/v1.3/runtime/configs</td>
</tr>
<tr>
<td>Update a configuration</td>
<td>PUT</td>
<td>/csapi/v1.3/runtime/configs/{configId}</td>
</tr>
</tbody>
</table>

Samples for various operations on configurations:

- Get all configurations in your account
- Get details for a specific configuration
- Create a configuration
- Update a configuration

Get all configurations in your account

/csapi/v1.3/runtime/configs

[GET]

API request:

header 'Authorization: Bearer <token>'

Response:

```
[
  {
    "id": "5e18c86e4e08ce0001368941",
    "created": "2020-01-10T18:54:38.82Z",
    "updated": "2020-01-10T18:54:38.82Z",
    "policyId": "5e18c86e4e08ce0001368940",
    "logMode": "POLICY_MONITOR_DENY",
    "isDefaultConfig": true,
    "name": "default config"
  },
  {
    "id": "5e1ae506b06e090001bf8741",
    "created": "2020-01-10T18:54:38.82Z",
    "updated": "2020-06-04T13:03:53.676Z",
    "policyId": "5e2587fd6bee78001c5625e",
    "logMode": "POLICY_MONITOR_DENY",
    "isDefaultConfig": false,
  }
]```
Get details for a specific configuration

/csapi/v1.3/runtime/configs/{configId}

[GET]

**Input Parameters:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>configId</td>
<td>(Required) Specify the ID of the configuration you want to get details on.</td>
</tr>
</tbody>
</table>

**API request:**

curl --location --request GET 'https://gateway.qgl.apps.qualys.com/csapi/v1.3/runtime/configs/5e284a064b80630001437f4e' \  --header 'Authorization: Bearer <token>'

**Response:**

```
{
  "id": "5e284a064b80630001437f4e",
  "created": "2020-06-08T10:03:43.507Z",
  "updated": "2020-06-08T10:07:44.249Z",
  "policyId": "5e18c86e4e08ce0001368940",
  "logMode": "ALL",
  "isDefaultConfig": false,
  "name": "example configuration"
}
```

Create a configuration

/csapi/v1.3/runtime/configs

[POST]

**Input Parameters:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specify a name for the new configuration. Enter a maximum of 256 characters.</td>
</tr>
<tr>
<td>policyId</td>
<td>(Required) Specify the ID of the security policy for this container. Sample value: 59c2dc5dc0f870001548489 A valid policy ID must be provided, and the specified policy must be present for the user.</td>
</tr>
</tbody>
</table>
### Configurations

**Create a configuration**

#### logMode

| (Required) For API v1.3, specify logMode with a string value to indicate which policy hits (rule matches) get logged. Possible values: NONE, POLICY_MONITOR, POLICY_DENY, POLICY_MONITOR_DENY, POLICY_ALLOW, POLICY_ALL, BEHAVIOR, ALL. Values are case sensitive.
| For API v1.2, specify LogMode with the numeric value. Possible values:
| None: 0
| Policy Monitor: 1
| Policy Deny: 2
| Policy Monitor Deny: 3
| Policy Allow: 4
| Policy All: 7
| Behavior: 8
| All: 15

#### isDefaultConfig

| (Required) For API v1.3, use isDefaultConfig. For API v1.2, use Default. Set to false by default. Specify true to make this the default configuration for group.

---

**API request:**

```bash
--header 'Authorization: Bearer <token>' 
--header 'Content-Type: text/plain' 
--data-raw '{
"name": "example configuration",
"policyId": "59c2dc5dc07f870001548489",
"logMode": "POLICY_MONITOR",
"isDefaultConfig": false
}'
```

**Response:**

```json
{
"id": "5ede0cfff42b100001905d58",
"created": "2020-06-08T10:03:43.507Z",
"updated": "2020-06-08T10:03:43.507Z",
"policyId": "59c2dc5dc07f870001548489",
"logMode": "POLICY_MONITOR",
"isDefaultConfig": false,
"name": "example configuration"
}
```
Update a configuration

/csapi/v1.3/runtime/configs/{configId}

[PUT]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>configId</td>
<td>(Required) The ID of the configuration to update.</td>
</tr>
<tr>
<td>name</td>
<td>Specify a name for the configuration. Enter a maximum of 256 characters.</td>
</tr>
<tr>
<td>policyId</td>
<td>(Required) Specify the ID of the security policy for this container. Sample value: 59c2dc5dc07f870001548489. A valid policy ID must be provided, and the specified policy must be present for the user.</td>
</tr>
<tr>
<td>logMode</td>
<td>(Required) For API v1.3, specify logMode with a string value to indicate which policy hits (rule matches) get logged. Possible values: NONE, POLICY_MONITOR, POLICY_DENY, POLICY_MONITOR_DENY, POLICY_ALLOW, POLICY_ALL, BEHAVIOR, ALL. Values are case sensitive. For API v1.2, specify LogMode with the numeric value. Possible values: None: 0, Policy Monitor: 1, Policy Deny: 2, Policy Monitor Deny: 3, Policy Allow: 4, Policy All: 7, Behavior: 8, All: 15.</td>
</tr>
<tr>
<td>isDefaultConfig</td>
<td>(Required) For API v1.3, use isDefaultConfig. For API v1.2, use Default. Set to false by default. Specify true to make this the default configuration for group.</td>
</tr>
</tbody>
</table>

API request:

```bash
curl --location --request PUT
'https://gateway.qgl.apps.qualys.com/csapi/v1.3/runtime/configs/5ede0cfff42b100001905d58'
--header 'Authorization: Bearer <token>'
--header 'Content-Type: text/plain'
--data-raw '{
    "name": "example configuration",
    "policyId": "59c2dc5dc07f870001548489",
    "logMode": "ALL",
    "isDefaultConfig": false,
}'
```
Response:

{
    "id": "5ede0cfff42b100001905d58",
    "created": "2020-06-08T10:03:43.507Z",
    "updated": "2020-06-08T10:07:44.249Z",
    "policyId": "5e18c86e4e08ce0001368940",
    "logMode": "ALL",
    "isDefaultConfig": false,
    "name": "example configuration"
}
Containers

Here is the list of the APIs we currently support for containers:

<table>
<thead>
<tr>
<th>API Objective</th>
<th>Operator</th>
<th>API Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get runtime details of a container</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/containers/{containerSha}</td>
</tr>
<tr>
<td>Get runtime profile for a container</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/containers/{containerSha}/runtimeprofile</td>
</tr>
<tr>
<td>Build a security policy based on a container's behavior</td>
<td>POST</td>
<td>/csapi/v1.3/runtime/containers/{containerSha}/template</td>
</tr>
<tr>
<td>Assign instrumentation configuration to container</td>
<td>POST</td>
<td>/csapi/v1.3/runtime/containers/{containerSha}/configs/{configId}</td>
</tr>
</tbody>
</table>

Samples for various operations on containers:

- Get runtime details of a container
- Get runtime profile for a container
- Build a security policy based on a container's behavior
- Assign a configuration to a container

Get runtime details of a container

/csapi/v1.3/runtime/containers/{containerSha}

[GET]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>containerSha={value}</td>
<td>(Required) Specify the SHA value of the container for which you want to get runtime details.</td>
</tr>
</tbody>
</table>

API request:

curl --location  --request GET  
'https://gateway.qgl.apps.qualys.com/csapi/v1.3/runtime/containers/7113e5aa32875169d41d168a871ca17a510663a6c0ea0e3a9ba03d0eea00cff6' \
--header 'Authorization: Bearer <token>'

Response:

```
{
  "containerSha": "7113e5aa32875169d41d168a871ca17a510663a6c0ea0e3a9ba03d0eea00cff6",
  "configId": "5e7df4f14b89300001cde5cb"
}
```
Get runtime profile for a container
/csapi/v1.3/runtime/containers/{containerSha}/runtimeprofile

[GET]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>containerSha={value}</td>
<td>(Required) Specify the SHA value of the container for which you want to get the runtime profile.</td>
</tr>
</tbody>
</table>

API request:

curl --location --request GET
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/containers/7113e5aa32875169d41d168a871ca17a510663a6c0ea0e3a9ba03d0eea00cff6/runtimeprofile'
  --header 'Authorization: Bearer <token>'

Response:

```json
{
  "files": [
    "/etc/ld.so.cache",
    "/etc/resolv.conf",
    "/lib/x86_64-linux-gnu/libacl.so.1",
    "/lib/x86_64-linux-gnu/libacl.so.1.1.0",
    "/lib/x86_64-linux-gnu/libattr.so.1",
    "/lib/x86_64-linux-gnu/libattr.so.1.1.0",
    "/lib/x86_64-linux-gnu/libc-2.19.so",
    "/lib/x86_64-linux-gnu/libc.so.6",
    "/lib/x86_64-linux-gnu/libdl-2.19.so",
    "/lib/x86_64-linux-gnu/libdl.so.2",
    "/lib/x86_64-linux-gnu/libpcre.so.3",
    "/lib/x86_64-linux-gnu/libpcre.so.3.13.1",
    "/lib/x86_64-linux-gnu/libpthread-2.19.so",
    "/lib/x86_64-linux-gnu/libpthread.so.0",
    "/lib/x86_64-linux-gnu/libselinux.so.1"
  ],
  "programs": [
    "/bin/cat",
    "/bin/ls",
    "/bin/sh"
  ],
  "ports": null,
  "ips": null
}
```
Build a security policy based on a container's behavior
/csapi/v1.3/runtime/containers/{containerSha}/template

[POST]

Important - The container you specify must have Behavior logs present in order to successfully create a template policy based on the container’s behavior.

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>containerSha</td>
<td>(Required) Specify the SHA value of the container for which you want to create a new custom security policy based on the recorded activities of the specified container.</td>
</tr>
</tbody>
</table>

API request:

```bash
curl --location --request POST
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/containers/7113e5aa32875169d41d68a871ca17a510663a6c0e0e3a9ba03d0eaa00c0f6/template' \
--header 'Authorization: Bearer <token>'
```

Response:

```json
{
  "policyId": "5ede15a34b23720001a75560"
}
```

Assign a configuration to a container
/csapi/v1.3/runtime/containers/{containerSha}/configs/{configId}

[POST]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>containerSha</td>
<td>(Required) Specify the SHA value of the container that you’re assigning the configuration to.</td>
</tr>
<tr>
<td>configId</td>
<td>(Required) Specify the ID of the configuration you want to assign to the container.</td>
</tr>
</tbody>
</table>

API request:

```bash
curl --location --request POST
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/containers/7113e5aa32875169d41d68a871ca17a510663a6c0e0e3a9ba03d0eaa00c0f6/configs/5e7df4f14b8930001cde5cb' \
--header 'Authorization: Bearer <token>'
```
Response:
  response code 200
Images

Here is the list of the APIs we currently support for images:

<table>
<thead>
<tr>
<th>API Objective</th>
<th>Operator</th>
<th>API Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instrument image with Qualys instrumentation</td>
<td>POST</td>
<td>/csapi/v1.1/images/{imageld}/instrument</td>
</tr>
<tr>
<td>Get CRS configuration of an image with instrumentation</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/images/{imageSha}/agentconfig</td>
</tr>
<tr>
<td>Get configuration ID of an instrumented image</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/images/{imageSha}/config</td>
</tr>
<tr>
<td>Assign instrumentation configuration to an image</td>
<td>POST</td>
<td>/csapi/v1.3/runtime/images/{imageSha}/configs/{configId}</td>
</tr>
</tbody>
</table>

Samples for various operations on images:

Instrument image with Qualys instrumentation
Get CRS configuration of an image with instrumentation
Get configuration ID for instrumented image
Assign configuration to an image

Instrument image with Qualys instrumentation

Once the instrumenter service is up and running in your environment, you can instrument your images. Note that you can only instrument images that have been scanned by a registry scan job (registry sensor). For this API endpoint, you’ll use the Container Security API. To learn more about using Container Security APIs, please refer to the Container Security API User Guide.

/csapi/v1.1/images/{imageld}/instrument

[POST]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageld</td>
<td>(Required) Specify the ID or SHA value of the image that you want to instrument.</td>
</tr>
<tr>
<td>pullRegistryUuid</td>
<td>The UUID of the registry where the image is located.</td>
</tr>
<tr>
<td>pullRepository</td>
<td>Name of the repository where the image is located.</td>
</tr>
<tr>
<td>pullTag</td>
<td>Tag associated with the image.</td>
</tr>
<tr>
<td>pushRegistryUuid</td>
<td>The UUID of the registry where you want to put the instrumented image.</td>
</tr>
</tbody>
</table>
Get CRS configuration of an image with instrumentation

/csapi/v1.3/runtime/images/[imageSha]/agentconfig

[GET]

Use this API to return the compiled policy for a base image that has instrumentation added in.

**Input Parameters:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageSha</td>
<td>(Required) Specify the SHA value of the image for which you want to get the CRS agent configuration.</td>
</tr>
<tr>
<td>raw</td>
<td>Default value is false. Specify true to get the response format output in raw format.</td>
</tr>
</tbody>
</table>

**API request:**

```
curl --location --request GET
 'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/images/d6c91042580a703530b92f77f96e891993f154927073/agentconfig' \
 --header 'Authorization: Bearer <token>'
```

**Response:**

```
"0,8,\n---
\n1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1
```

Images

Get configuration ID for instrumented image

/csapi/v1.3/runtime/images/{imageSha}/config

[GET]

Use this API to return the configuration ID assigned to an instrumented image. Once you have the configId, you’ll be able to use it in other API calls that require it.

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageSha={value}</td>
<td>(Required) Specify the SHA value of an instrumented image for which you want to get the configuration ID.</td>
</tr>
</tbody>
</table>

Sample

In this sample, we’ll get the configId for the specified image.

API request:

curl --location --request GET
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/images/6124ce09d13b89e2f26f01f8/config' --header 'Authorization: Bearer <token>'

Response:

"ConfigID": "6124ce09d13b89e2f26f01f8"
Assign configuration to an image

/csapi/v1.3/runtime/images/{imageSha}/configs/{configId}

[POST]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>imageSha</td>
<td>(Required) Specify the SHA value of the image for which you want to assign the CRS agent configuration.</td>
</tr>
<tr>
<td>configId</td>
<td>(Required) Specify the ID of the configuration you want to assign to the specified image.</td>
</tr>
</tbody>
</table>

API request:

```bash
curl --location --request POST
'https://gateway.qgl.apps.qualys.com/csapi/v1.3/runtime/images/d6c910425801a703530b92f943575b8ea9dab520f77f96e891993f1549a27073/configs/5e7df4f14b8930001cde5cb'
--header 'Authorization: Bearer <token>'
```

Response:

response code 200
Events

Here is the list of the APIs we currently support for events:

<table>
<thead>
<tr>
<th>API Objective</th>
<th>Operator</th>
<th>API Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get all events</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/events</td>
</tr>
</tbody>
</table>

Samples for various operations on events:

Get all events in your account

Get all events in your account

/csapi/v1.3/runtime/events

[GET]

There are several options for filtering the events returned in the output. For example, you can only get events created after a certain date, before a certain date or within a date range. You can also filter the list to get events for a particular container or with a certain action type. See all options below.

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eventType</td>
<td>(Required) Specify the type of logs you want to return. Possible values are: STANDARD, BEHAVIOR.</td>
</tr>
<tr>
<td>startTime</td>
<td>Specify a starting date/time to get events created after this date. Specify the date in the format ['YYYY'-'MM'-'DD'T'hh':'mm':'ss'].</td>
</tr>
<tr>
<td>endTime</td>
<td>Specify an ending date/time to get events created before this date. Specify the date in the format ['YYYY'-'MM'-'DD'T'hh':'mm':'ss'].</td>
</tr>
<tr>
<td>filter</td>
<td>Specify a string value for a search query to filter the list of events returned in the output. In the search query you can include any value that appears in the response body like action, system, systemCall, containerSha, uuid, etc. For example, filter events with a string like this: filter=action:ALLOW AND containerSha:dc58cab81c9a1edc8dc39d34a8a61942c56dc1d4ad276d68684be4169d4f0ce53</td>
</tr>
<tr>
<td>pageNumber</td>
<td>The page to be returned. Page numbers start with 1.</td>
</tr>
<tr>
<td>pageSize</td>
<td>The number of records per page to be included in the response. When not specified you’ll get 10 events.</td>
</tr>
</tbody>
</table>
Sample for returning all events with Standard type
You’ll get up to 10 events in the output by default.

API request:
```
curl --location --request GET 'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/events?eventType=STANDARD' \
--header 'Authorization: Bearer <token>'
```

Response:
```
[

{
  "customerUuid": "6e0afd12-479c-db0d-822a-793a56bfe353",
  "containerSha": "3368ab5ebbccb9d17d45cf6f6fa289edade4af81ef5a94e044406a1904175d",
  "eventType": "STANDARD",
  "uuid": "70b0dd00-cde7-11ea-8000-a130bd09cb71",
  "created": 1595620450000,
  "action": "DENY",
  "bindAddress": null,
  "bindPort": 0,
  "fileName": "/etc/passwd",
  "openMode": 0,
  "processId": 42,
  "processName": "/usr/bin/cat",
  "seen": 1,
  "system": "amd64",
  "systemCall": 2,
  "systemCallName": "sys_open"
},
{
  "customerUuid": "6e0afd12-479c-db0d-822a-793a56bfe353",
  "containerSha": "3368ab5ebbccb9d17d45cf6f6fa289edade4af81ef5a94e044406a1904175d",
  "eventType": "STANDARD",
  "uuid": "70b0dd00-cde7-11ea-8000-51fe233a28cb",
  "created": 1595620450000,
  "action": "DENY",
  "bindAddress": null,
  "bindPort": 0,
  "fileName": "/etc/passwd",
  "openMode": 0,
  "processId": 43,
  "processName": "/usr/bin/cat",
  "seen": 1,
  "system": "amd64",
  "systemCall": 2,
  "systemCallName": "sys_open"
}
...
]
More Samples

Try these additional samples in your account.

**Sample with Page Number and Page Size specified**

In this sample we've specified the number of events to show in the output.

**API request:**

```bash
curl --location --request GET
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/events?eventType=STANDARD&pageNumber=1&pageSize=5' \
--header 'Authorization: Bearer <token>'
```

**Sample to get events with certain action**

In this sample the filter parameter is used to get events with the ALLOW action. Be sure to specify the action value in all caps (ALLOW, DENY, MONITOR).

**API request:**

```bash
curl --location --request GET
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/events?eventType=BEHAVIOR&filter=action:ALLOW' \
--header 'Authorization: Bearer <token>'
```

**Sample to get events created within a particular date range**

In this sample we'll get events created between June 30, 2020 and July 1, 2020.

**API request:**

```bash
curl --location --request GET
--header 'Authorization: Bearer <token>'
```

**Samples using filter string as input**

In this sample we'll only get events for the specified container.

**API request:**

```bash
curl --location --request GET
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/events?eventType=BEHAVIOR&filter=containerSha:dc58cab81c9a1ed8b8cd39d34a8a61942c56d1d4ad27668684be4169d4f0cec5' \
--header 'Authorization: Bearer <token>'
```
In this sample we'll only get events with the ALLOW action for the specified container.

**API request:**

```bash
curl --location --request GET 'https://gateway.qgl.apps.qualys.com/csapi/v1.3/runtime/events?eventType=BEHAVIOR&filter=action:ALLOW AND containerSha:dc58cab81c9a1edb8cd39d34a8a61942c56dc1d4ad27668684be4169d4f0cec5' \
--header 'Authorization: Bearer <token>'
```
Policies

Here is the list of the APIs we currently support for policies:

<table>
<thead>
<tr>
<th>API Objective</th>
<th>Operator</th>
<th>API Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get all policies in your account</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/policies</td>
</tr>
<tr>
<td>Get details for a specific policy</td>
<td>GET</td>
<td>/csapi/v1.3/runtime/policies/{policyId}</td>
</tr>
<tr>
<td>Create a new security policy</td>
<td>POST</td>
<td>/csapi/v1.3/runtime/policies</td>
</tr>
<tr>
<td>Update a security policy</td>
<td>PUT</td>
<td>/csapi/v1.3/runtime/policies/{policyId}</td>
</tr>
<tr>
<td>Delete a security policy</td>
<td>DELETE</td>
<td>/csapi/v1.3/runtime/policies/{policyId}</td>
</tr>
</tbody>
</table>

Samples for various operations on policies:

Get all policies in your account
Get details for a specific policy
Create a new security policy
Update a security policy
Delete a security policy

Get all policies in your account
/csapi/v1.3/runtime/policies

[GET]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pageNumber</td>
<td>The page to be returned. The default value is 1.</td>
</tr>
<tr>
<td>pageSize</td>
<td>The number of records per page to be included in the response. When not specified, you’ll get 50 records.</td>
</tr>
<tr>
<td>filter</td>
<td>Specify a string value for a search query to filter the list of policies returned in the output. Only name is supported in filter query. For example, enter filter=name:Default to return policies with &quot;Default&quot; in the name. The search is case sensitive. Double quotes can be used when your search value contains more than one word, such as filter=name:&quot;Test Policy&quot;.</td>
</tr>
</tbody>
</table>

Sample - Get all policies

API request:

curl --location --request GET
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/policies' \
--header 'Authorization: Bearer <token>'
Response:

[

{}

{
"id": "5e171bef8530d7000151408e",
"name": "Default Policy",
"created": "2020-01-09T12:26:23.496Z",
"updated": "2020-01-09T12:26:23.496Z",
"description": "Default group policy",
"policyMode": "ACTIVE"
},
{
"id": "5e171c738530d70001514091",
"name": "Prevent tampering to hosts file",
"created": "2020-01-09T12:28:35.761Z",
"updated": "2020-01-09T12:28:35.761Z",
"description": "Modifications to 'hosts' and 'resolve.conf' file can result in resolution of Domain name to malicious IP. This policy checks for the 'Write' event on either of the specified files",
"policyMode": "ACTIVE"
},
{
"id": "5e81cf5df1286000129938c",
"name": "Deny access in etc v11 Updating With PUT",
"created": "2020-05-29T04:54:16.432Z",
"updated": "2020-05-29T04:54:16.432Z",
"description": "Deny access in /etc dir for important files",
"policyMode": "INACTIVE"
},
...
]

Sample - Filter policies list

In this sample, we’re filtering the list of policies to only show policies with “Deny” in the policy name.

API request:

```
```

Response:

[

{}

{ 
"id": "5e171bef8530d7000151408e",
"name": "Deny Write Static Website Files",
"created": "2020-01-09T12:26:23.496Z",
"updated": "2020-01-09T12:26:23.496Z",
"description": "This sample policy prevents static website files..." 
}
Get details for a specific policy
/csapi/v1.3/runtime/policies/{policyId}

[GET]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policyId</td>
<td>(Required) Specify the ID of a specific policy for which you want to get details.</td>
</tr>
</tbody>
</table>

API request:

curl --location --request GET
'https://gateway.qgl.apps.qualys.com/csapi/v1.3/runtime/policies/5eba6fef2c79c40001e23488'
   --header 'Authorization: Bearer <token>'

Response:

{  
  "id": "5eba6fef2c79c40001e23488",
  "name": "My CRS Policy",
  "created": "2020-05-12T09:44:15.315Z",
  "updated": "2020-05-12T09:44:15.315Z",
  "defaultNetworkAction": "ALLOW",
  "defaultExecuteAction": "ALLOW",
  "defaultFileAction": "ALLOW",
  "rules": [  
    {  
      "id": "5fa25442e677eb00012916b7",
      "name": "Static file modification deny",
      "created": "2020-05-12T09:44:15.315Z",
      "updated": "2020-05-12T09:44:15.315Z",
      "inactive": false,
      "ruleType": "WRITE",
    }
  ]
}
"program": ",",
"action": "DENY",
"file": "/var/www/html/",
"port": 0,
"ipAddress": ",",
"syscall": ",",
"arg1": ",",
"arg2": ",",
"arg3": ","
},
{
"id": "5fa2512de677eb00012916b5",
"name": "Deny_Hosts_Write_Attempt",
"created": "2020-05-12T09:44:15.315Z",
"updated": "2020-05-12T09:44:15.315Z",
"inactive": false,
"ruleType": "READ",
"program": "/bin/cat",
"action": "DENY",
"file": "/etc/hosts",
"port": 0,
"ipAddress": ",",
"syscall": ",",
"arg1": ",",
"arg2": ",",
"arg3": ","
},
{
"id": "5fa24e78e677eb00012916b3",
"name": "Deny_Outbound",
"created": "2020-05-12T09:44:15.315Z",
"updated": "2020-05-12T09:44:15.315Z",
"inactive": false,
"ruleType": "NETWORK_OUTBOUND",
"program": ",",
"action": "DENY",
"file": ",",
"port": 22,
"ipAddress": "1.1.1.1",
"syscall": ",",
"arg1": ",",
"arg2": ",",
"arg3": ","
},
{
"id": "5fa25442e677eb00012916bc",
"name": "Block_sshd_communication",
"created": "2001-01-01T00:00:00Z",
"updated": "2001-01-01T00:00:00Z",
"inactive": false,
"ruleType": "NETWORK_INBOUND",
"program": ",",
"action": "DENY",
"file": ","
Create a new security policy

/csapi/v1.3/runtime/policies

[POST]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Specify a name for the policy.</td>
</tr>
<tr>
<td>description</td>
<td>Provide a description of your policy.</td>
</tr>
<tr>
<td>policyMode</td>
<td>(Required) For API v1.3, use policyMode to specify the policy mode using a string value. Possible values: ACTIVE, INACTIVE, PERMISSIVE. Values are case sensitive. For API v1.2, use Mode to specify the policy mode using the numeric value. Possible values: Active: 0 (the default) Inactive: 1 Permissive: 2</td>
</tr>
<tr>
<td>defaultNetworkAction</td>
<td>(Required) The default action when ruleType is NETWORK_OUTBOUND or NETWORK_INBOUND. Possible values: ALLOW or DENY. Values are case sensitive.</td>
</tr>
<tr>
<td>defaultExecuteAction</td>
<td>(Required) The default action when ruleType is SYSCALL. Possible values: ALLOW or DENY. Values are case sensitive.</td>
</tr>
<tr>
<td>defaultFileAction</td>
<td>(Required) The default action when ruleType is READ or WRITE. Possible values: ALLOW or DENY. Values are case sensitive.</td>
</tr>
</tbody>
</table>
ignoredSyscalls  
(Supported only with API v1.3) Define a list of system call names to ignore for this policy. No events will be created for ignored system calls even if there’s a policy rule match. Only valid system call names are allowed. Enter a list of values like this: ["sys_read", "sys_write"]

rules  
Policy rules defining controls for this policy specified within an array. See Rule Parameters below.

### Rule Parameters

Specify rules within an array. These rules will define control for the policy.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>(Required) Specify a name for the rule.</td>
</tr>
<tr>
<td>inactive</td>
<td>Specify whether the rule is inactive. Specify false (the default) if the rule is active. Specify true if the rule is not active.</td>
</tr>
<tr>
<td>ruleType</td>
<td>(Required) Specify the type of rule. Possible values: READ, WRITE, NETWORK_OUTBOUND, NETWORK_INBOUND, SYSCALL. Values are case sensitive.</td>
</tr>
<tr>
<td>ipAddress</td>
<td>(Required when ruleType is NETWORK_OUTBOUND or NETWORK_INBOUND) Specify the IP address this rule applies to.</td>
</tr>
<tr>
<td>port</td>
<td>(Optional when ruleType is NETWORK_OUTBOUND or NETWORK_INBOUND) Specify the network protocol that this rule applies to.</td>
</tr>
<tr>
<td>program</td>
<td>Specify the path to program that this rule applies to. Wildcards are allowed. The default value is &quot;.&quot;.</td>
</tr>
<tr>
<td>file</td>
<td>(Required when ruleType is READ or WRITE) Specify the path to the file that the rule applies to.</td>
</tr>
<tr>
<td>syscall</td>
<td>(Required when ruleType is SYSCALL) The system call provided must be a valid system call name.</td>
</tr>
<tr>
<td>arg1</td>
<td>(Required when ruleType is SYSCALL) Variable argument. Usage differs depending on rule type. Used only in syscall rules.</td>
</tr>
<tr>
<td>arg2</td>
<td>(Optional) Variable argument. Usage differs depending on rule type. Used only in syscall rules.</td>
</tr>
<tr>
<td>arg3</td>
<td>(Optional) Variable argument. Usage differs depending on rule type. Used only in syscall rules.</td>
</tr>
<tr>
<td>action</td>
<td>(Required) Specify the action that should be taken if this rule is matched. Possible values: ALLOW, DENY, MONITOR. Values are case sensitive.</td>
</tr>
<tr>
<td>created</td>
<td>Timestamp for when object was created in the format ['YYYY'-'MM'-'DD'T'hh':'mm':'ss'.'sss'Z].</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>updated</td>
<td>Timestamp for when object was last updated in the format ['YYYY'-'MM'-'DD'T'hh':'mm':'ss'.'sss'Z].</td>
</tr>
</tbody>
</table>

**Policies**

Create a new security policy

**API request:**

```bash
curl --location --request POST 'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/policies' \
  --header 'Authorization: Bearer <token>' \
  --header 'Content-Type: text/plain' \
  --data-raw '{
    "name": "Prevent Shadow Access To User",
    "created": "2020-11-10T08:14:22.509Z",
    "updated": "2020-11-10T08:14:22.509Z",
    "defaultNetworkAction": "ALLOW",
    "defaultExecuteAction": "ALLOW",
    "defaultFileAction": "ALLOW",
    "rules": [
      {
        "id": "5faa4bdeeda7de00015142c0",
        "name": "Deny access in cat /etc/shadow",
        "created": "0001-01-01T00:00:00Z",
        "updated": "0001-01-01T00:00:00Z",
        "inactive": false,
        "ruleType": "SYSCALL",
        "program": "*/cat",
        "action": "DENY",
        "file": "/etc/shadow",
        "port": 0,
        "ipAddress": ",
        "syscall": "sys_open",
        "arg1": "/etc/shadow",
        "arg2": ",
        "arg3": ","
      }
    ],
    "ignoredSyscalls": [],
    "policyMode": "ACTIVE",
    "description": "Example policy denies access to /etc/shadow from program cat"
  }
}'
```

**Response:**

Response Code 200 :

Response Message :

```json
{
  "id": "5fb5e21f5caea20000f027ce",
  "name": "Prevent Shadow Access To User",
  "created": "2020-11-19T03:10:23.36Z",
  "updated": "2020-11-19T03:10:23.36Z",
}
```
"defaultNetworkAction": "ALLOW",
"defaultExecuteAction": "ALLOW",
"defaultFileAction": "ALLOW",
"rules": [
{
   "id": "5faa4bdeeda7de00015142c0",
   "name": "Deny access in cat /etc/shadow",
   "created": "0001-01-01T00:00:00Z",
   "updated": "0001-01-01T00:00:00Z",
   "inactive": false,
   "ruleType": "SYSCALL",
   "program": "*/cat",
   "action": "DENY",
   "file": "/etc/shadow",
   "port": 0,
   "ipAddress": "",
   "syscall": "sys_open",
   "arg1": "/etc/shadow",
   "arg2": "",
   "arg3": ""
}
]
"ignoredSyscalls": [],
"policyMode": "ACTIVE",
"description": "Example policy denies access to /etc/shadow from program cat"
}

Update a security policy
/csapi/v1.3/runtime/policies/{policyId}

[PUT]

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policyId</td>
<td>(Required) Specify the ID of the policy to update.</td>
</tr>
<tr>
<td>name</td>
<td>Specify a name for the policy.</td>
</tr>
<tr>
<td>description</td>
<td>Provide a description of your policy.</td>
</tr>
<tr>
<td>policyMode</td>
<td>(Required) For API v1.3, use policyMode to specify the policy mode using a string value. Possible values: ACTIVE, INACTIVE, PERMISSIVE. Values are case sensitive. For API v1.2, use Mode to specify the policy mode using the numeric value. Possible values: Active: 0 (the default) Inactive: 1 Permissive: 2</td>
</tr>
</tbody>
</table>

31
defaultNetworkAction

(Required) The default action when ruleType is NETWORK_OUTBOUND or NETWORK_INBOUND. Possible values: ALLOW or DENY. Values are case sensitive.

defaultExecuteAction

(Required) The default action when ruleType is SYSCALL. Possible values: ALLOW or DENY. Values are case sensitive.

defaultFileAction

(Required) The default action when ruleType is READ or WRITE. Possible values: ALLOW or DENY. Values are case sensitive.

ignoredSyscalls

(Supported only with API v1.3) Define a list of system call names to ignore for this policy. No events will be created for ignored system calls even if there's a policy rule match. Only valid system call names are allowed. Enter a list of values like this: ["sys_read", "sys_write"]

rules

Policy rules defining controls for this policy specified within an array. See Rule Parameters in previous section.

API request:

curl --location --request PUT
'thttps://gateway.qgl.apps.qualys.com/csapi/v1.3/runtime/policies/5fb5e21f5caea20001fd27ce'

--header 'Authorization: Bearer <token>'

--header 'Content-Type: text/plain'

--data-raw '{
  "name": "Updated Policy Prevent Shadow Access To User",
  "created": "2020-11-10T08:14:22.509Z",
  "updated": "2020-11-10T08:14:22.509Z",
  "defaultNetworkAction": "ALLOW",
  "defaultExecuteAction": "ALLOW",
  "defaultFileAction": "ALLOW",
  "rules": [
    {
      "id": "5faa4bdeeda7de00015142c0",
      "name": "Deny access in cat /etc/shadow",
      "created": "0001-01-01T00:00:00Z",
      "updated": "0001-01-01T00:00:00Z",
      "inactive": false,
      "ruleType": "SYSCALL",
      "program": "/cat",
      "action": "DENY",
      "file": "/etc/shadow",
      "port": 0,
      "ipAddress": "",
      "syscall": "",
      "arg1": "*/cat",
      "arg2": "",
      "arg3": ""
    }
  ]
}'
Policies
Update a security policy

Response:

```json
{
    "id": "5fb5e21f5caea20001fd27ce",
    "name": "Updated Policy Prevent Shadow Access To User",
    "created": "2020-11-19T03:10:23.36Z",
    "updated": "2020-11-19T03:13:03.083Z",
    "defaultNetworkAction": "ALLOW",
    "defaultExecuteAction": "ALLOW",
    "defaultFileAction": "ALLOW",
    "rules": [
        {
            "id": "5faa4bdeeda7de00015142c0",
            "name": "Deny access in cat /etc/shadow",
            "created": "0001-01-01T00:00:00Z",
            "updated": "0001-01-01T00:00:00Z",
            "inactive": false,
            "ruleType": "SYSCALL",
            "program": "*/cat",
            "action": "DENY",
            "file": "/etc/shadow",
            "port": 0,
            "ipAddress": "",
            "syscall": "sys_open",
            "arg1": "/etc/shadow",
            "arg2": "",
            "arg3": ""
        }
    ],
    "ignoredSyscalls": [],
    "policyMode": "ACTIVE",
    "description": "Example policy denies access to /etc/shadow from program cat"
}
```
Delete a security policy

/csapi/v1.3/runtime/policies/{policyId}

[DELETE]

Note that you can only delete a policy that is not currently associated with any instrumented images/containers.

Input Parameters:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>policyId</td>
<td>(Required) Specify the ID of the policy to delete.</td>
</tr>
</tbody>
</table>

API request:

curl -X DELETE --header 'Accept: text/plain' --header 'Authorization: Basic VVNFUk5BTUU6UEFTU1dPUkQ='
'https://gateway.qg1.apps.qualys.com/csapi/v1.3/runtime/policies/5fa97660f19b060001e8ab6f'

Response:

response code 200