Qualys.

Qualys Container Security v1.x
API Release Notes

Version 1.8

December 16, 2020

Qualys Container Security API gives you many ways to integrate your programs and API calls with Qualys capabilities.

What's New

Introducing Policy Compliance Support

Qualys API URL

Container Security supports both API server URLs and API gateway URLs for API requests.

The Qualys API server or 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 URL

This documentation uses the API URL for Qualys US Platform 2 (https://gateway.qg2.apps.qualys.com) in sample API requests. If you're on another platform, please replace this URL with the appropriate server URL for your account.

Introducing Policy Compliance Support

Qualys now supports compliance scanning of running containers and images. This support is part of Container Security Sensor 1.7.0 and Container Security 1.8.0. With these releases, you can perform Policy Compliance (PC) checks and configuration assessments on your Docker hosts. All you need is the new PC manifest which contains the commands for checking against the CIS Docker Benchmark.

Interested in Policy Compliance for Container Security? Please reach out to your Technical Account Manager or Qualys Support to have the Container Security PC feature enabled for your subscription. Once enabled, the PC manifest will be sent to the sensor along with the VM manifest. There's nothing else you need to do to start collecting compliance data. This is done automatically with the new manifest.

API Changes

New Container Security APIs allow you to fetch compliance posture for an image or container, and list controls and control details. We've also updated existing APIs for images and containers to include compliance details in the response.

Compliance APIs are available through the new Qualys API gateway endpoint (https://gateway.qg2.apps.qualys.com/csapi/v1.3). This new gateway provides a token (JWT) based authentication mechanism that is more secure than the authentication mechanisms used in the previous API.

New APIs

We now support the following APIs for compliance:

API Objective	Operator	API Path
Show PC compliance posture for an image	GET	/csapi/v1.3/images/{imageSha}/compliance
Fetch PC compliance posture for a container	GET	/csapi/v1.3/containers/{containerSha}/compliance
Fetch details for a control	GET	/csapi/v1.3/controls/{controlId}
Fetch a list of controls	GET	/csapi/v1.3/controls

Updated APIs

The following image and container APIs have been updated to include compliance details in the API response when using the new v1.3 API endpoints:

API Objective	Operator	API Path
Show a list of images in your account	GET	/csapi/v1.3/images

Show details of a single image	GET	/csapi/v1.3/images/{imageSha}
Show a list of containers in your account	GET	/csapi/v1.3/containers
Show details of a container	GET	/csapi/v1.3/containers/{containerSha}

Summary of changes to existing APIs:

- The gateway API URL is now used (https://gateway.qg2.apps.qualys.com/csapi/v1.3).

- For list images/containers, the response will show the number of controls with a status of passed, failed and error, as well as the last compliance scan date. Note that pageNo is not supported as an input using the new v1.3 API. pageNumber is supported.

- For fetch image/container details, the response will show the last compliance scan date. Note that only imageSha and containerSha are supported when fetching details using the new v1.3 API. You cannot specify imageId or containerId like in the previous API version.

- For fetch image/container details, we removed the following vulnerability details from the response since they are not supported: ageInDays, fixed, os, nonRunningKernel, nonExploitableConfig and runningService.

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:

```
curl -X POST 'https://gateway.qg2.apps.qualys.com/auth' -H 'Content-Type:
application/x-www-form-urlencoded' --data-urlencode 'username=Value' --
data-urlencode 'password=Value' --data-urlencode 'token=true' --
dataurlencode 'permissions=true'
```

where gateway.qg2.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.

API Samples

See more details, including samples, in the sections that follow.

Fetch compliance posture for an image

/v1.3/images/{imageSha}/compliance

[GET]

Input Parameters:

Parameter	Description
imageSha	Specify the SHA value of a specific image in the user's scope.

API request:

```
curl
'https://gateway.qg2.apps.qualys.com/csapi/v1.3/images/c64844065dcbc3d0a9
0c365c1f56421766a5cebf05f7ecbd3377af410fff09fd/compliance' --header
'Authorization: Bearer <token>'
```

Response:

```
{
  "uuid": "5d48f83b-cddb-33ac-8fad-e8452dd116b1",
  "sha":
"c64844065dcbc3d0a90c365c1f56421766a5cebf05f7ecbd3377af410fff09fd",
  "customerUuid": "192cc974-1e44-cb6c-806e-f78f6441cb0d",
  "created": "1603477517000",
  "updated": "1605017537578",
  "controls": [
    {
      "controlId": 10826,
      "policyUuid": "e07da90a-dc32-48e0-9bbb-2eae68012333",
      "technologyId": 0,
      "criticality": "MEDIUM",
      "posture": "SETTING_NOT_FOUND",
      "lastEvaluated": "1605017382800",
      "datapoints": [
        {
          "key": "dockersensor00.image.healthcheck",
          "value": "161803399999999"
        }
      ]
    }
  ],
  "lastComplianceScanned": "1605017537578"
}
```

Fetch compliance posture for a container

/v1.3/containers/{containerSha}/compliance

[GET]

Input Parameters:

Parameter	Description
containerSha	Specify the SHA value of a specific container in the user's scope.

API request:

```
curl
'https://gateway.qg2.apps.qualys.com/csapi/v1.3/containers/b87b645dffda05
e59bb80ac20678b1c1f051c4a1286bafe8c55a58e523d49af5/compliance' --header
'Authorization: Bearer <token>'
```

Response:

```
{
  "uuid": "fd011da7-9fed-314f-9884-76713eb66156",
  "sha":
"b87b645dffda05e59bb80ac20678b1c1f051c4a1286bafe8c55a58e523d49af5",
  "customerUuid": "192cc974-1e44-cb6c-806e-f78f6441cb0d",
  "created": "1604869118000",
  "updated": "1604922275091",
  "controls": [
    {
      "controlId": 10808,
      "policyUuid": "e18b623d-3f07-485b-a754-5a1c31727df3",
      "technologyId": 0,
      "criticality": "CRITICAL",
      "posture": "SETTING_NOT_FOUND",
      "lastEvaluated": "1604869185266",
      "datapoints": [
        {
          "key": "dockersensor00.container.capdrop",
          "value": "161803399999999"
        }
      ]
    },
    {
      "controlId": 10716,
      "policyUuid": "e18b623d-3f07-485b-a754-5a1c31727df3",
      "technologyId": 0,
      "criticality": "SERIOUS",
      "posture": "SETTING_NOT_FOUND",
      "lastEvaluated": "1604869185266",
      "datapoints": [
        {
          "key": "dockersensor00.container.pidmode",
          "value": "161803399999999"
        }
      ]
    }
  ],
  "lastComplianceScanned": "1604869185266"
}
```

Fetch control details

/v1.3/controls/{controlId}

[GET]

Input Parameters:

Parameter	Description
controlId	Specify the ID of a compliance control for which you want to get details.

API request:

```
curl 'https://gateway.qg2.apps.qualys.com/csapi/v1.3/controls/10808' --
header 'Authorization: Bearer <token>'
```

Response:

```
{
  "id": 10808,
  "statement": "Status of the 'cap-drop' flag settings on Docker
containers on the host system",
  "criticality": "CRITICAL",
  "comments": "STMT: Status of the 'cap-drop' flag settings on Docker
containers on the host system\n\nRAT: Linux Capabilities allows dividing
privileges associated with superuser into distinct group of smaller units,
known as capabilities.  By default, Docker containers are started with a
restricted set of capabilities where each one can be independently enabled
and disabled.  This enables the processes running inside a container to
perform almost all the specific areas where root privileges are usually
needed without having to have them run as root.  Unrestricted Linux
capabilities could allow unauthorized access to containers which could
potentially lead to attacks such as privilege escalation exploits.  Linux
Capabilities on Docker containers should be restricted as appropriate to
the needs of the business to have only those that are required for the
containers to perform their
function.\n\nCIS_Docker_1.11.0_Benchmark_v1.0.0: 5.3 Restrict Linux
Kernel Capabilities within
containers\nCIS_Docker_1.12.0_Benchmark_v1.0.0: 5.3 Restrict Linux Kernel
Capabilities within containers",
  "deprecated": "Control is not deprecated",
  "category": "Access Control Requirements",
  "subCategory": "Authorization (Single-user ACL/role)",
  "technologies": [
    {
      "technologyId": 283,
      "technologyName": "Docker Containers/Images",
      "rational": "Linux Capabilities allows dividing privileges
associated with superuser into distinct group of smaller units, known as
capabilities.  By default, Docker containers are started with a restricted
set of capabilities where each one can be independently enabled and
disabled.  This enables the processes running inside a container to
perform almost all the specific areas where root privileges are usually
needed without having to have them run as root.  Unrestricted Linux
capabilities could allow unauthorized access to containers which could
potentially lead to attacks such as privilege escalation exploits.  Linux
Capabilities on Docker containers should be restricted as appropriate to
the needs of the business to have only those that are required for the
containers to perform their function.",
      "remediation": "Run the following command to verify that the added
and dropped Linux Kernel Capabilities are in line with the ones needed for
container process for each container instance.\n# docker ps --quiet |
xargs docker inspect --format '}: CapAdd=} CapDrop=}'\n\nRun the following
command to add needed capabilities:\n# docker run --cap-add={\"Capability
1\",\"Capability 2\"} <run-arguments> <container-image-name-or-ID>
<command>\n\nFor example,\n# docker run --interactive --tty --cap-
add={\"NET_ADMIN\",\"SYS_ADMIN\"} centos:latest /bin/bash\n\nTo drop
unneeded capabilities, run the following command:\n# docker run --cap-
drop={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-
image-name-or-ID> <command>\n\nFor example, \ndocker run --interactive --
tty --cap-drop={\"SETUID\",\"SETGID\"} centos:latest
/bin/bash\n\nAlternatively, drop all capabilities and add only add only
those that are the needed:\n# docker run --cap-drop=all --cap-
add={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-image-
name-or-ID> <command>\n\nFor example, \ndocker run --interactive --tty --
cap-drop=all --cap-add={\"NET_ADMIN\",\"SYS_ADMIN\"} centos:latest
/bin/bash"
    }
  ]
}
```

Fetch a list of controls

/v1.3/controls

[GET]

Input Parameters:

Parameter	Description
filter	Filter the controls list by providing a query using Qualys syntax. Refer to the “How to Search” topic in the online help for assistance with creating your query
pageNumber	(Required) The page to be returned. The default is 1.
pageSize	(Required) The number of records per page to be included in the response. The default value is 50.
sort	Sort the results using a Qualys token. For example **`created:desc`**. Refer to the “Sortable tokens” topic in the online help for more information.

API request:

```
curl
'https://gateway.qg2.apps.qualys.com/csapi/v1.3/controls?pageNumber=1&pag
eSize=50&sort=created%3Adesc' --header 'Authorization: Bearer <token>'
```

Response:

```
{
  "data": [
    {
      "id": 18990,
      "statement": "Status of the host devices and their exposure",
      "criticality": "SERIOUS",
      "comments": "STMT: Status of the host devices and their
exposure\n\nRAT: Host devices can be directly exposed to containers at
runtime. Do not directly expose host devices to containers, especially to
containers that are not trusted. The --device option exposes host devices
to containers and as a result of this, containers can directly access
these devices. The  container would not need to run in privileged mode to
access and manipulate them, as by default, the container is granted this
type of access. Additionally, it would possible for containers to remove
block devices from the host. You therefore should not expose host devices
to containers directly. If for some reason you wish to expose the host
device to a container you should consider which sharing permissions you
wish to use on a case by case base as appropriate to your
organization.\n\n\n----\nOld mappings before granular NIST was added: New
control\n\nGranular mapping:\nAC-1, ACCESS CONTROL POLICY AND
```

```
PROCEDURES\n\nDelete Old Mapping:N/A\nGranular Mapping: Yes\nGranular
Mapping Review: No",
      "deprecated": "Control is not deprecated",
      "category": "OS Security Settings",
      "subCategory": "System Settings (OSI layers 6-7)",
      "technologies": [
        {
          "id": 181,
          "name": "Docker CE/EE",
          "rational": "Host devices can be directly exposed to containers
at runtime. Do not directly expose host devices to containers, especially
to containers that are not trusted. The --device option exposes host
devices to containers and as a result of this, containers can directly
access these devices. The  container would not need to run in privileged
mode to access and manipulate them, as by default, the container is
granted this type of access. Additionally, it would possible for
containers to remove block devices from the host. You therefore should not
expose host devices to containers directly. If for some reason you wish to
expose the host device to a container you should consider which sharing
permissions you wish to use on a case by case base as appropriate to your
organization.",
          "remediation": "You should not directly expose host devices to
containers. If you do need to expose host devices to containers, you
should use granular permissions as appropriate to your organization:\nFor
example, do not start a container using the command below:\ndocker run --
interactive --tty --device=/dev/tty0:/dev/tty0:rwm --
device=/dev/temp_sda:/dev/temp_sda:rwm centos bash\nYou should only share
the host device using appropriate permissions:\ndocker run --interactive -
-tty --device=/dev/tty0:/dev/tty0:rw --
device=/dev/temp_sda:/dev/temp_sda:r centos bash"
        },
        {
          "id": 283,
          "name": "Docker Containers/Images",
          "rational": "Host devices can be directly exposed to containers
at runtime. Do not directly expose host devices to containers, especially
to containers that are not trusted. The --device option exposes host
devices to containers and as a result of this, containers can directly
access these devices. The  container would not need to run in privileged
mode to access and manipulate them, as by default, the container is
granted this type of access. Additionally, it would possible for
containers to remove block devices from the host. You therefore should not
expose host devices to containers directly. If for some reason you wish to
expose the host device to a container you should consider which sharing
permissions you wish to use on a case by case base as appropriate to your
organization.",
          "remediation": "You should not directly expose host devices to
containers. If you do need to expose host devices to containers, you
should use granular permissions as appropriate to your organization:\nFor
```

```
example, do not start a container using the command below:\ndocker run --
interactive --tty --device=/dev/tty0:/dev/tty0:rwm --
device=/dev/temp_sda:/dev/temp_sda:rwm centos bash\nYou should only share
the host device using appropriate permissions:\ndocker run --interactive -
-tty --device=/dev/tty0:/dev/tty0:rw --
device=/dev/temp_sda:/dev/temp_sda:r centos bash"
        }
      ]
    },
    {
      "id": 10808,
      "statement": "Status of the 'cap-drop' flag settings on Docker
containers on the host system",
      "criticality": "CRITICAL",
      "comments": "STMT: Status of the 'cap-drop' flag settings on Docker
containers on the host system\n\nRAT: Linux Capabilities allows dividing
privileges associated with superuser into distinct group of smaller units,
known as capabilities.  By default, Docker containers are started with a
restricted set of capabilities where each one can be independently enabled
and disabled.  This enables the processes running inside a container to
perform almost all the specific areas where root privileges are usually
needed without having to have them run as root.  Unrestricted Linux
capabilities could allow unauthorized access to containers which could
potentially lead to attacks such as privilege escalation exploits.  Linux
Capabilities on Docker containers should be restricted as appropriate to
the needs of the business to have only those that are required for the
containers to perform their
function.\n\nCIS_Docker_1.11.0_Benchmark_v1.0.0: 5.3 Restrict Linux
Kernel Capabilities within
containers\nCIS_Docker_1.12.0_Benchmark_v1.0.0: 5.3 Restrict Linux Kernel
Capabilities within containers",
      "deprecated": "Control is not deprecated",
      "category": "Access Control Requirements",
      "subCategory": "Authorization (Single-user ACL/role)",
      "technologies": [
        {
          "id": 112,
          "name": "Docker 1.x",
          "rational": "Linux Capabilities allows dividing privileges
associated with superuser into distinct group of smaller units, known as
capabilities.  By default, Docker containers are started with a restricted
set of capabilities where each one can be independently enabled and
disabled.  This enables the processes running inside a container to
perform almost all the specific areas where root privileges are usually
needed without having to have them run as root.  Unrestricted Linux
capabilities could allow unauthorized access to containers which could
potentially lead to attacks such as privilege escalation exploits.  Linux
Capabilities on Docker containers should be restricted as appropriate to
the needs of the business to have only those that are required for the
```

```
containers to perform their function.",
          "remediation": "Run the following command to verify that the
added and dropped Linux Kernel Capabilities are in line with the ones
needed for container process for each container instance.\n# docker ps --
quiet | xargs docker inspect --format '}: CapAdd=} CapDrop=}'\n\nRun the
following command to add needed capabilities:\n# docker run --cap-
add={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-image-
name-or-ID> <command>\n\nFor example,\n# docker run --interactive --tty -
-cap-add={\"NET_ADMIN\",\"SYS_ADMIN\"} centos:latest /bin/bash\n\nTo drop
unneeded capabilities, run the following command:\n# docker run --cap-
drop={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-
image-name-or-ID> <command>\n\nFor example, \ndocker run --interactive --
tty --cap-drop={\"SETUID\",\"SETGID\"} centos:latest
/bin/bash\n\nAlternatively, drop all capabilities and add only add only
those that are the needed:\n# docker run --cap-drop=all --cap-
add={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-image-
name-or-ID> <command>\n\nFor example, \ndocker run --interactive --tty --
cap-drop=all --cap-add={\"NET_ADMIN\",\"SYS_ADMIN\"} centos:latest
/bin/bash"
        },
        {
          "id": 181,
          "name": "Docker CE/EE",
          "rational": "Linux Capabilities allows dividing privileges
associated with superuser into distinct group of smaller units, known as
capabilities.  By default, Docker containers are started with a restricted
set of capabilities where each one can be independently enabled and
disabled.  This enables the processes running inside a container to
perform almost all the specific areas where root privileges are usually
needed without having to have them run as root.  Unrestricted Linux
capabilities could allow unauthorized access to containers which could
potentially lead to attacks such as privilege escalation exploits.  Linux
Capabilities on Docker containers should be restricted as appropriate to
the needs of the business to have only those that are required for the
containers to perform their function.",
          "remediation": "Run the following command to verify that the
added and dropped Linux Kernel Capabilities are in line with the ones
needed for container process for each container instance.\n# docker ps --
quiet | xargs docker inspect --format '}: CapAdd=} CapDrop=}'\n\nRun the
following command to add needed capabilities:\n# docker run --cap-
add={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-image-
name-or-ID> <command>\n\nFor example,\n# docker run --interactive --tty -
-cap-add={\"NET_ADMIN\",\"SYS_ADMIN\"} centos:latest /bin/bash\n\nTo drop
unneeded capabilities, run the following command:\n# docker run --cap-
drop={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-
image-name-or-ID> <command>\n\nFor example, \ndocker run --interactive --
tty --cap-drop={\"SETUID\",\"SETGID\"} centos:latest
/bin/bash\n\nAlternatively, drop all capabilities and add only add only
those that are the needed:\n# docker run --cap-drop=all --cap-
```

```
add={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-image-
name-or-ID> <command>\n\nFor example, \ndocker run --interactive --tty --
cap-drop=all --cap-add={\"NET_ADMIN\",\"SYS_ADMIN\"} centos:latest
/bin/bash"
        },
        {
          "id": 283,
          "name": "Docker Containers/Images",
          "rational": "Linux Capabilities allows dividing privileges
associated with superuser into distinct group of smaller units, known as
capabilities.  By default, Docker containers are started with a restricted
set of capabilities where each one can be independently enabled and
disabled.  This enables the processes running inside a container to
perform almost all the specific areas where root privileges are usually
needed without having to have them run as root.  Unrestricted Linux
capabilities could allow unauthorized access to containers which could
potentially lead to attacks such as privilege escalation exploits.  Linux
Capabilities on Docker containers should be restricted as appropriate to
the needs of the business to have only those that are required for the
containers to perform their function.",
          "remediation": "Run the following command to verify that the
added and dropped Linux Kernel Capabilities are in line with the ones
needed for container process for each container instance.\n# docker ps --
quiet | xargs docker inspect --format '}: CapAdd=} CapDrop=}'\n\nRun the
following command to add needed capabilities:\n# docker run --cap-
add={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-image-
name-or-ID> <command>\n\nFor example,\n# docker run --interactive --tty -
-cap-add={\"NET_ADMIN\",\"SYS_ADMIN\"} centos:latest /bin/bash\n\nTo drop
unneeded capabilities, run the following command:\n# docker run --cap-
drop={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-
image-name-or-ID> <command>\n\nFor example, \ndocker run --interactive --
tty --cap-drop={\"SETUID\",\"SETGID\"} centos:latest
/bin/bash\n\nAlternatively, drop all capabilities and add only add only
those that are the needed:\n# docker run --cap-drop=all --cap-
add={\"Capability 1\",\"Capability 2\"} <run-arguments> <container-image-
name-or-ID> <command>\n\nFor example, \ndocker run --interactive --tty --
cap-drop=all --cap-add={\"NET_ADMIN\",\"SYS_ADMIN\"} centos:latest
/bin/bash"
        }
      ]
    }
  ],
  "count": "26"
}
```

Fetch a list of containers in your account

/v1.3/containers

[GET]

We updated the response to show counts for the number of controls with a status of passed, failed and error, as well as the last compliance scan date/time.

Input Parameters:

Parameter	Description
filter	Filter the containers list by providing a query using Qualys syntax. Refer to the "How to Search" topic in the online help for assistance with creating your query.
pageNumber	(Optional) The page to be returned. Page numbers start with 1.
pageSize	(Required) The number of records per page to be included in the response.
sort	Sort the results using a Qualys token. For example **`created:desc`**. Refer to the "Sortable tokens" topic in the online help for more information.

API request:

```
curl
'https://gateway.qg2.apps.qualys.com/csapi/v1.3/containers?pageNumber=1&p
ageSize=50&sort=created%3Adesc' --header 'Authorization: Bearer <token>'
```

Response:

```
{
  "data": [
    {
      "uuid": "2c8d6485-0c6d-3d7b-a2bd-86616ff78205",
      "sha":
"27723ada671ea9f25624a9ffdadde273038e4a48373bada6133371cb3bd7a9d3",
      "imageId": "27723ada671e",
      "repo": [
        {
          "registry": "docker.io",
          "repository": "qualysdemo/checkoutapp",
          "tag": "demo"
        }
      ],
      "repoDigests": [
        {
          "registry": "docker.io",
          "repository": "qualysdemo/checkoutapp",
          "digest":
"6d0f0a22ba1768ebed30dabba5d856fc5536609ec12fc2b23e7bec7aa79ccd9b"
        }
      ],
      "created": 1507592726000,
      "updated": 1603767703217,
      "associatedContainersCount": 0,
      "associatedHostsCount": 1,
      "lastScanned": null,
      "size": 718071042,
      "vulnerabilities": {
        "severity5Count": 0,
        "severity4Count": 0,
        "severity3Count": 0,
        "severity2Count": 0,
        "severity1Count": 0
      },
      "registryUuid": null,
      "source": [
        "GENERAL"
      ],
      "isDockerHubOfficial": false,
      "isInstrumented": false,
      "instrumentedFrom": null,
      "instrumentationState": null,
      "instrumentationErrors": null,
      "scanType": null,
      "scanErrorCode": null,
      "scanStatus": null,
      "lastFoundOnHost": {
        "sensorUuid": "cb9fa762-b161-43bb-9268-ebae5fc606af",
        "uuid": null,
        "hostname": "Test HostName",
        "ipAddress": "254.254.254.254",
        "lastUpdated": "2020-03-19T11:01:08.907Z"
      },
      "compliance": {
        "passCount": 3,
        "failCount": 1,
        "errorCount": 1
      },
      "lastComplianceScanned": 1507592707000
    }
  ],
  "groups": {}
}
```

Fetch container details

/v1.3/containers/{containerSha}

[GET]

You'll now see the last compliance scan date/time in the API response.

Input Parameters:

Parameter	Description
containerSha	Specify the SHA value of a specific container in the user's scope.

API request:

```
curl
'https://gateway.qg2.apps.qualys.com/csapi/v1.3/containers/c64844065dcbc3
d0a90c365c1f56421766a5cebf05f7ecbd3377af410fff09fd' --header
'Authorization: Bearer <token>'
```

Response:

```
{
  "created": "1603477517000",
  "updated": "1605017537578",
  "author": "Couchbase Docker Team <docker@couchbase.com>",
  "repo": [
    {
      "registry": "docker.io",
      "tag": "latest",
      "repository": "couchbase"
    }
  ],
  "repoDigests": [
    {
      "registry": "docker.io",
      "digest":
"1d811b3c382893f70f0cc0f2371a12d3671c1d5175bcc67e8c2a5c0bf4c8f976",
      "repository": "couchbase"
    }
  ],
  "label": null,
  "uuid": "5d48f83b-cddb-33ac-8fad-e8452dd116b1",
  "sha":
"c64844065dcbc3d0a90c365c1f56421766a5cebf05f7ecbd3377af410fff09fd",
  "operatingSystem": "Ubuntu Linux 16.04.7",
  "customerUuid": "192cc974-1e44-cb6c-806e-f78f6441cb0d",
  "dockerVersion": "18.09.7",
    "size": 1183790011,
    "layers": [
      {
        "size": "130553983",
        "createdBy": "ADD
file:c1f3147c7b6710af5affd417ff822ee28df872d716003858d3d2e23d2277c981 in
/ ",
        "created": "1603474388000",
        "comment": "",
        "id": null,
        "sha": null,
        "tags": null
      },
      {
        "size": "0",
        "createdBy": "rm -rf /var/lib/apt/lists/*",
        "created": "1603474389000",
        "comment": "",
        "id": null,
        "sha": null,
        "tags": null
      },
      {
        "size": "1930",
        "createdBy": "COPY
file:d816a67f62bfba76d2812cefbe92252afa13f3852775c3e68599df7741e90cb7 in
/ ",
        "created": "1603477517000",
        "comment": "",
        "id": null,
        "sha": null,
        "tags": null
      }
    ],
    "host": [
      {
        "sensorUuid": "fed79006-2fa9-4b67-8f5a-272b4e02f084",
        "hostname": "host.qualys.com",
        "ipAddress": "10.44.29.40",
        "uuid": "6ba5be85-2758-4f44-814a-b690c9ed23ee",
        "lastUpdated": "2020-11-10T14:10:29.218Z"
      }
    ],
    "architecture": "amd64",
    "imageId": "c64844065dcb",
    "lastScanned": "1605017537578",
    "registryUuid": null,
    "source": [
      "GENERAL"
```

```
  ],
  "totalVulCount": "0",
  "users": [
    "root"
  ],
  "isDockerHubOfficial": null,
  "isInstrumented": null,
  "instrumentedFrom": null,
  "instrumentationState": null,
  "scanType": "DYNAMIC",
  "scanErrorCode": null,
  "scanStatus": "SUCCESS",
  "lastFoundOnHost": {
    "sensorUuid": "fed79006-2fa9-4b67-8f5a-272b4e02f084",
    "hostname": "host.qualys.com",
    "ipAddress": "10.44.29.40",
    "uuid": "6ba5be85-2758-4f44-814a-b690c9ed23ee",
    "lastUpdated": "2020-11-10T14:10:29.218Z"
  },
  "softwares": [
    {
      "name": "libncursesw5:amd64",
      "version": "6.0+20160213-1ubuntu1",
      "fixVersion": null,
      "vulnerabilities": null
    },
    {
      "name": "libgpg-error0:amd64",
      "version": "1.21-2ubuntu1",
      "fixVersion": null,
      "vulnerabilities": null
    }
  ],
  "vulnerabilities": [],
  "lastComplianceScanned": "1603477517000"
}
```

Fetch a list of images in your account

/v1.3/images

[GET]

We updated the response to show counts for the number of controls with a status of passed, failed and error, as well as the last compliance scan date/time.

Input Parameters:

Parameter	**Description**
filter	Filter the containers list by providing a query using Qualys syntax. Refer to the “How to Search” topic in the online help for assistance with creating your query.
pageNumber	(Optional) The page to be returned. Page numbers start with 1.
pageSize	(Required) The number of records per page to be included in the response.
sort	Sort the results using a Qualys token. For example **`eventOccurred:desc`**. Refer to the “Sortable tokens” topic in the online help for more information.

API request:

```
curl
'https://gateway.qg2.apps.qualys.com/csapi/v1.3/images?pageNumber=1&pageS
ize=50&sort=created%3Adesc' --header 'Authorization: Bearer <token>'
```

Response:

```
{
  "data": [
    {
      "uuid": "2c8d6485-0c6d-3d7b-a2bd-86616ff78205",
      "sha":
"27723ada671ea9f25624a9ffdadde273038e4a48373bada6133371cb3bd7a9d3",
      "imageId": "27723ada671e",
      "repo": [
        {
          "registry": "docker.io",
          "repository": "qualysdemo/checkoutapp",
          "tag": "demo"
        }
      ],
      "repoDigests": [
        {
          "registry": "docker.io",
          "repository": "qualysdemo/checkoutapp",
          "digest": 
"6d0f0a22ba1768ebed30dabba5d856fc5536609ec12fc2b23e7bec7aa79ccd9b"
        }
      ],
      "created": 1507592726000,
      "updated": 1603767703217,
      "associatedContainersCount": 0,
      "associatedHostsCount": 1,
      "lastScanned": null,
      "size": 718071042,
      "vulnerabilities": {
        "severity5Count": 0,
        "severity4Count": 0,
        "severity3Count": 0,
        "severity2Count": 0,
        "severity1Count": 0
      },
      "registryUuid": null,
      "source": [
        "GENERAL"
      ],
      "isDockerHubOfficial": false,
      "isInstrumented": false,
      "instrumentedFrom": null,
      "instrumentationState": null,
      "instrumentationErrors": null,
      "scanType": null,
      "scanErrorCode": null,
      "scanStatus": null,
      "lastFoundOnHost": {
        "sensorUuid": "cb9fa762-b161-43bb-9268-ebae5fc606af",
        "uuid": null,
        "hostname": "Test HostName",
        "ipAddress": "254.254.254.254",
        "lastUpdated": "2020-03-19T11:01:08.907Z"
      },
      "compliance": {
        "passCount": 3,
        "failCount": 1,
        "errorCount": 1
      },
      "lastComplianceScanned": 1507592707000
    }
  ],
  "groups": {}
}
```

Fetch image details

/v1.3/images/{imageSha}

[GET]

You'll now see the last compliance scan date/time in the API response.

Input Parameters:

Parameter	Description
imageSha	Specify the SHA value of a specific image in the user's scope.

API request:

```
curl
'https://gateway.qg2.apps.qualys.com/csapi/v1.3/images/c64844065dcbc3d0a9
0c365c1f56421766a5cebf05f7ecbd3377af410fff09fd' --header 'Authorization:
Bearer <token>'
```

Response:

```
{
  "created": "1603477517000",
  "updated": "1605017537578",
  "author": "Couchbase Docker Team <docker@couchbase.com>",
  "repo": [
    {
      "registry": "docker.io",
      "tag": "latest",
      "repository": "couchbase"
    }
  ],
  "repoDigests": [
    {
      "registry": "docker.io",
      "digest":
"1d811b3c382893f70f0cc0f2371a12d3671c1d5175bcc67e8c2a5c0bf4c8f976",
      "repository": "couchbase"
    }
  ],
  "label": null,
  "uuid": "5d48f83b-cddb-33ac-8fad-e8452dd116b1",
  "sha":
"c64844065dcbc3d0a90c365c1f56421766a5cebf05f7ecbd3377af410fff09fd",
  "operatingSystem": "Ubuntu Linux 16.04.7",
  "customerUuid": "192cc974-1e44-cb6c-806e-f78f6441cb0d",
  "dockerVersion": "18.09.7",
  "size": 1183790011,
  "layers": [
    {
      "size": "130553983",
      "createdBy": "ADD
file:c1f3147c7b6710af5affd417ff822ee28df872d716003858d3d2e23d2277c981 in
/ ",
      "created": "1603474388000",
      "comment": "",
      "id": null,
      "sha": null,
      "tags": null
    },
    {
      "size": "0",
      "createdBy": "rm -rf /var/lib/apt/lists/*",
      "created": "1603474389000",
      "comment": "",
      "id": null,
      "sha": null,
      "tags": null
    },
    {
      "size": "1930",
      "createdBy": "COPY
file:d816a67f62bfba76d2812cefbe92252afa13f3852775c3e68599df7741e90cb7 in
/ ",
      "created": "1603477517000",
      "comment": "",
      "id": null,
      "sha": null,
      "tags": null
    }
  ],
  "host": [
    {
      "sensorUuid": "fed79006-2fa9-4b67-8f5a-272b4e02f084",
      "hostname": "host.qualys.com",
      "ipAddress": "10.44.29.40",
      "uuid": "6ba5be85-2758-4f44-814a-b690c9ed23ee",
      "lastUpdated": "2020-11-10T14:10:29.218Z"
    }
  ],
  "architecture": "amd64",
  "imageId": "c64844065dcb",
  "lastScanned": "1605017537578",
  "registryUuid": null,
  "source": [
    "GENERAL"
    ],
    "totalVulCount": "0",
    "users": [
      "root"
    ],
    "isDockerHubOfficial": null,
    "isInstrumented": null,
    "instrumentedFrom": null,
    "instrumentationState": null,
    "scanType": "DYNAMIC",
    "scanErrorCode": null,
    "scanStatus": "SUCCESS",
    "lastFoundOnHost": {
      "sensorUuid": "fed79006-2fa9-4b67-8f5a-272b4e02f084",
      "hostname": "host.qualys.com",
      "ipAddress": "10.44.29.40",
      "uuid": "6ba5be85-2758-4f44-814a-b690c9ed23ee",
      "lastUpdated": "2020-11-10T14:10:29.218Z"
    },
    "softwares": [
      {
        "name": "libncursesw5:amd64",
        "version": "6.0+20160213-1ubuntu1",
        "fixVersion": null,
        "vulnerabilities": null
      },
      {
        "name": "libgpg-error0:amd64",
        "version": "1.21-2ubuntu1",
        "fixVersion": null,
        "vulnerabilities": null
      }
    ],
    "vulnerabilities": [],
    "lastComplianceScanned": "1603477517000"
  }
```