



Qualys Container Scanning Connector for Jenkins

User Guide
Version 1.7

February 15, 2024

Copyright 2018-2024 by Qualys, Inc. All Rights Reserved.

Qualys and the Qualys logo are registered trademarks of Qualys, Inc. All other trademarks are the property of their respective owners.

Qualys, Inc.
919 E Hillsdale Blvd
4th Floor
Foster City, CA 94404
1 (650) 801 6100



Table of Contents

About this Guide	4
About Qualys	4
Qualys Support	4
About Container Security Documentation	4
Container Security Overview	5
What data does Container Security collect?	6
About the Container Scanning Connector	6
Getting started	7
What you'll need	7
Install the Plugin	8
Scanning CI/CD images	9
Start Using the Plugin	11
Define container image IDs	11
Using the WebHook	14
Configuration Details	17
Qualys API Server URL	18
Encapsulating API user credentials in job specific configuration	18
View Your Qualys Report.....	20
Debugging and Troubleshooting	21
Where are the logs?	21
HTTP codes in API response	21
Plugin times out, no report seen	22
No space left on device	22
Incorrect checksum of retrieved jar	23
Upgrade Issues	24
Want to contact Support?	24

About this Guide

Welcome to Qualys Container Security! We'll help you get acquainted with the Qualys solutions for securing your Container environments like Images, Containers and Docker Hosts using the Qualys Cloud Security Platform.

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/.

About Container Security Documentation

This document provides information about using the Qualys Container Scanning Connector for Jenkins.

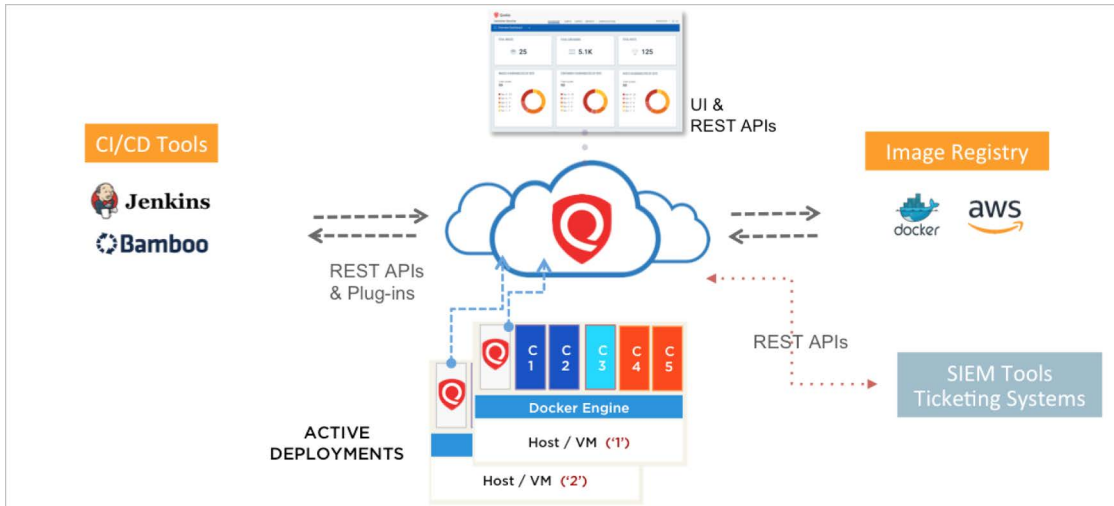
For information on using the Container Security UI to monitor vulnerabilities in Images, Containers, and Registries, refer to the [Qualys Container Security User Guide](#).

For information on deploying the sensor on MAC, CoreOS, and various orchestrators and cloud environments, refer to the [Qualys Container Sensor Deployment Guide](#).

For information on using the Container Security API, refer to the [Qualys Container Security API Guide](#).

Container Security Overview

Qualys Container Security provides discovery, tracking, and continuously protecting container environments. This addresses vulnerability management for images and containers in their DevOps pipeline and deployments across cloud and on-premise environments.



With this version, Qualys Container Security supports

- Discovery, inventory, and near-real time tracking of container environments
- Vulnerability analysis for images and containers
- Vulnerability analysis for registries
- Integration with CI/CD pipeline using APIs (DevOps flow)
- Uses new 'Container Sensor' – providing native container support, distributed as container image

Upon installation, the sensor does automatic discovery of Images and Containers on the deployed host, provides a vulnerability analysis of them, and additionally it monitors and reports on the docker related events on the host. The Container Security Jenkins plugin currently supports monitoring and reporting on the Containerd events as well. The sensor lists and scans registries for vulnerable images. The sensor container runs in non-privileged mode. It requires a persistent storage for storing and caching files.

Currently, the sensor only scans Images and Containers. For getting a vulnerability posture on the Host, you would require Qualys Cloud Agents or a scan through Qualys Virtual Scanner Appliance.

What data does Container Security collect?

The Qualys Container Security sensor fetches the following information about Images and Containers in your environment:

- **Inventory of Images and Containers** in your environment from commands such as **docker ps** that lists all containers.
- **Metadata information** about Images and Containers from commands such as **docker inspect** and **docker info** that fetches low level information on docker objects.
- **Event information** about Images and Containers from the docker host for docker events like created, started, killed, push, pull, etc.
- **Vulnerabilities** found on Images and Containers. This is the output of the vulnerability management manifests run for identifying vulnerability information in Images and Containers. This is primarily software package listing, services running, ports, etc.

For example, package manager outputs like **rpm -qa**, **npm**. This is supported across various Linux distributions (CentOS, Ubuntu, CoreOS, etc) and across images like Python, NodeJS, Ruby, and so on.

About the Container Scanning Connector

Qualys Container Security provides a plugin for Jenkins to get the security posture for the container images built via the tool. The plugin can be configured to fail or pass the container image builds based on the vulnerabilities detected.

Getting started

Follow the steps to get started with Container Scanning Connector for Jenkins.

What you'll need

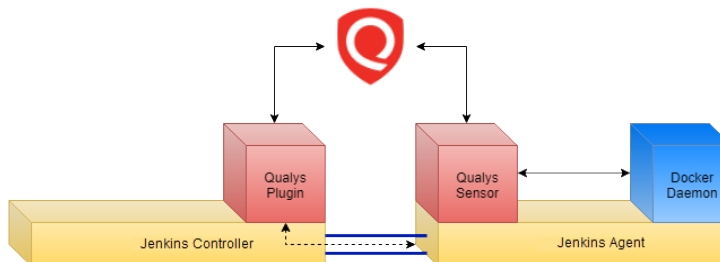
- A valid Qualys subscription with the Container Security application activated.
- Access to Qualys Container Security application API endpoint from your build host.
- Requires the container sensor for CI/CD environment to be installed on the Jenkins build host. Refer to Qualys Container Security Sensor Deployment Guide for instructions on installing the container cicd sensor. You must pass the following parameter while deploying the sensor for CI/CD environment `--cicd-deployed-sensor` or `-c`.
- Internet connection for agent to be able to connect to the Qualys Cloud Platform. Install sensor with proxy option if agent is running behind proxy.
- The Jenkins controller and agent nodes should have an open connection to the Qualys Cloud Platform in order to get data from the Qualys Cloud Platform for vulnerability reporting.

Jenkins plugin automatically tags images built out of CI/CD pipeline with the tag **qualys_scan_target:<image-sha>** to mark them for scanning and only those images are scanned for vulnerabilities. Once the scanning is over, Qualys Container Sensor will remove the tag. However, if an image has no other tag applied to it other than 'qualys_scan_target:<image-sha>', the sensor will retain the tag to avoid removal of the image from the host.

Recommended setup for controller-agent deployment

Container Scanning Connector should be deployed on the Jenkins controller. Qualys Container Security Sensor should be installed where the Docker daemon is running. If the Docker daemon is running on Jenkins agent, install the Sensor on Jenkins agent. If the Docker daemon is running on a remote host, install the sensor over there.

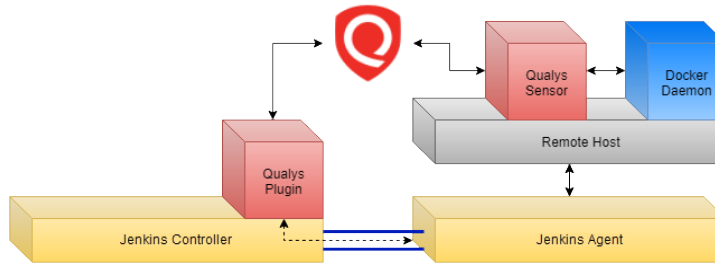
Following figure shows the Docker daemon running on Jenkins agent.



Docker Host on Jenkins Agent

i.e. Docker daemon is directly running on Jenkins agent machine.
Docker CLI talks to Docker daemon on Unix socket.

Following figure shows the docker daemon running on a remote host.



Docker Daemon on Remote Host

i.e. Docker images are built on 3rd machine other than Jenkins controller and agent. Agent has Docker CLI which talks to Docker daemon on TCP.

Note that the plugin will make a call to check if the sensor container is running on the host and if it is installed in the CI/CD mode.

For the ContainerD scan running in the clustered environments, the Container Security Sensor and the Jenkins Controller should be installed in the CI/CD mode for the clustered environment as per the given instruction. The Container Security Connector v1.7.x can further be installed as the plugin on the Jenkins which supports the image scan for both Dockerd and Containerd.

Install the Plugin

You can install the Container Scanning Connector in two ways: Install the plugin from within Jenkins, or download the plugin from the Qualys Cloud Platform and then upload the plugin to your Jenkins instance.

Note: If you have an existing plugin installed from the Qualys Cloud Platform, it won't be automatically upgraded to the Jenkins plugin available on the Jenkins Update Center. You must uninstall the existing plugin and then re-install the latest plugin from the Jenkins Update Center. After the plugin re-installation you must reconfigure the Jenkins jobs. This reconfiguration is required only for first time installation of the plugin from the Jenkins Update Center. Subsequent plugin updates through the Jenkins Update Center do not require reconfiguration of the jobs.

Installing the Plugin from Jenkins

- 1) To install the plugin from Jenkins, login to your Jenkins instance and then click Manage Jenkins.
- 2) On the Manage Jenkins page, click Manage Plugins.
- 3) If you are installing the plugin for the first time, click the Available tab and search for Qualys Container Scanning Connector using the Filter field. Select the plugin and click **Install without restart**.

After the plugin is installed, it will be listed on the Installed tab.

If the plugin is already installed in Jenkins and you want to update the plugin, go to the Updates tab, search for Qualys Container Security and then click **Install without restart**.

Note that the plugin is also listed on the Jenkins plugin store at <https://plugins.jenkins.io/>.

Installing the Plugin when Jenkins update center is not accessible from Jenkins Server instance

- 1) Open your browser and go to <http://updates.jenkins-ci.org/download/plugins/qualys-cs/>. You will see the available versions of plugin for download along with their respective sha-256. Click a plugin version to download the plugin installation file (.hpi file).
- 3) Go to Manage Jenkins > Manage Plugins > Advanced.
- 4) Go to the Upload Plugin section, click Choose File and select the downloaded .hpi plugin file.
- 5) Click Upload.
- 6) Restart Jenkins server instance if needed.

Installing the Plugin from Qualys Cloud Platform

- 1) On Qualys Cloud Platform, go to Configurations > Integrations to download the Jenkins plugin.
- 2) Upload the plugin to your Jenkins tool. Go to Manage Jenkins > Manage Plugins > Advanced > Upload Plugin, then restart it.

Upgrade the Plugin

Follow the instructions below to upgrade the CS Jenkins Plugin to respective versions.

Version 1.7x

If you are upgrading the plugin to v1.7.x, then ensure that the Jenkins version is 2.387.1 or greater and the Container Security (CS) sensor is latest.

If you are using a pipeline job in your Jenkins, you must reconfigure the pipeline snippet for 'Qualys Container Scanning Connector' plugin task in your pipeline. Failure to do will result in the plugin breaking after upgrade. This is a one time activity post-upgrade for existing pipelines to run smoothly.

Note: This change does not affect the 'Freestyle' jobs in Jenkins.

How to reconfigure the Pipeline Snippet?

With version 1.7.x, the Qualys CS Plugin accepts platform information with a text field replacing the previous dropdown selection. The text field accepts the Qualys supported platform URLs as listed [here](#).

This change in input field also changes the field key generated in the pipeline snippet.

Refer to the samples given below.

Sample pipeline snippet before 1.7.x

```
getImageVulnsFromQualys credentialsId: 'Qualys-Credentials',
imageIds: "java:latest", isExcludeConditions: true, excludeBy:
'qid', excludeList:qualysExclusionList , isSev4Vulns: true,
isSev5Vulns: true, platform: 'US_PLATFORM_3', pollingInterval: '30',
proxyCredentialsId:'', proxyPort: 3128, proxyServer: 'proxyserver.com',
useLocalConfig: true, useProxy: true, vulnsTimeout:'1200'
```

Sample pipeline snippet after latest version for Job Specific Configuration with Job specific "Docker URL/Nerdctl binary path"

```
getImageVulnsFromQualys apiServer:
'https://qualysapi.qg2.apps.qualys.com', credentialsId: '68e2a407-ed05-
411b-b2da-423106f1201b', dockerUrl: '/var/containerd_ctl', imageIds:
'java:latest', isSev1Vulns: true, isSev2Vulns: true, isSev3Vulns: true,
isSev4Vulns: true, isSev5Vulns: true, pollingInterval: '30',
useLocalConfig: true, vulnsTimeout: '600'
```

Sample pipeline snippet after latest version for Global Configuration with Job specific "Docker URL/Nerdctl binary path"

```
getImageVulnsFromQualys dockerUrl: '/var/containerd_ctl', imageIds:
'java:latest', useGlobalConfig: true
```

When an older pipeline snippet for the Qualys CS plugin runs with v1.7.x, the plugin breaks as the expected field key for the Qualys platform is missing. Ensure you reconfigure the pipeline snippet to avoid this scenario.

Version 1.6.2.1

If you are upgrading the plugin to v1.6.2.1, then ensure that the Jenkins version is 2.204.1 or greater and the Container Security (CS) version is 1.8.0.0 or greater.

Scanning CI/CD images

Configure the docker URL / socket path for the plugin to automatically tag CI/CD images with 'qualys_scan_target:<image-id>'.

To configure the following fields, navigate to 'Qualys Container Security' section in the path {Jenkins-Instance-url}/manage/configure> **Qualys Container Security > Advanced Settings**.

Docker URL/Nerdctl binary path: Configure this field as per your runtime environment.

For dockerd, the expected configuration is docker socket path eg. `unix://path_of_docker.sock` or `tcp://[host]:[port]`, in case of TLS, cert path should be provided in field 'Cert file path' For containerd, the expected configuration is nerdctl binary path.

Containerd Prerequisites

The Nerdctl binary path is required for the plugin to complete its activities. As such, it is advised to map the nerdctl binary path and the containerd.sock file path in the volumeMounts section of the Jenkins deployment.yaml.

Cert File Path (optional): If you are using remote server enabled https, you can provide a specific folder location which contains the files `ca.pem`, `cert.pem` and `key.pem`. For example, `/var/jenkins_home/certs`.

Note: A Job Specific (local) configuration will use the Docker URL and Cert File Path configured in global configuration for tagging CI/CD images.

Docker URLs (unix socket or TCP) to be used in various docker deployment scenarios

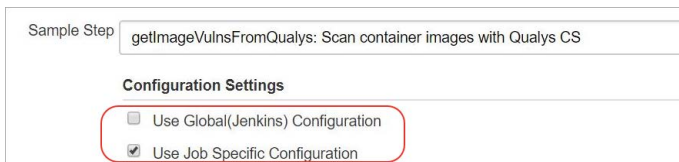
Deployment scenario	Sensor location	Docker URL to be used
Job executed by Jenkins controller AND Docker host == Jenkins controller	Jenkins controller	UNIX <code>unix:///var/run/docker.sock</code>
Job executed by Jenkins controller AND Docker Host == Remote docker host (any machine other than Jenkins controller or agent)	Remote docker host	TCP path of the Remote Docker host: <code>tcp://<ip_of_RDH>:<port></code> For example, <code>tcp://10.xxx.xx.xx:2375</code>
Job executed by Jenkins agent AND Docker host == Jenkins agent	Jenkins agent	UNIX <code>unix:///var/run/docker.sock</code>
Job executed by Jenkins agent AND Docker Host == Remote docker host (any machine other than Jenkins controller or agent)	Remote docker host	TCP path of the Remote Docker host: <code>tcp://<ip_of_RDH>:<port></code> For example, <code>tcp://10.xxx.xx.xx:2375</code>

Start Using the Plugin

Qualys recommends to set up the Jenkins Plugin after the container image is built, and before the image is pushed to the registry. Ensure that you do not delete the image before the plugin setup is complete.

While setting up the plugin you can either provide a **global configuration** or a **job specific configuration**. Global configuration can be set once and used for multiple projects: both Pipeline and Freestyle.

To set a global configuration, go to Manage Jenkins > Configure System, then scroll down to the Qualys Container Security section, and provide the configuration details listed below.



The screenshot shows a Jenkins configuration interface. At the top, there is a 'Sample Step' section with a text input field containing 'getImageVulnsFromQualys: Scan container images with Qualys CS'. Below this is a 'Configuration Settings' section with two radio button options. The first option, 'Use Global(Jenkins) Configuration', is unselected. The second option, 'Use Job Specific Configuration', is selected and highlighted with a red rounded rectangle.

If you want to set a job specific configuration, on the Pipeline Syntax > Snippet Generator page, select "getImageVulnsFromQualys: Scan container images with Qualys CS", and then select the Use Job Specific Configuration option.

Note: Selecting the "Use Global(Jenkins) Configuration" option here will let the job use the global configuration you have set under **Manage Jenkins > Configure System > Qualys Container Security**.

See [Configuration Details](#)

This plugin provides a build step and a post-build action. You can use it with pipeline type projects (for CI/CD pipeline) as well as freestyle projects. We'll describe both in the sections that follow.

- [Pipeline Project](#)
- [Freestyle Project](#)

Define container image IDs

For Docker Images

In the plugin, configuration there is a field called image IDs/Image Names. Set this to the container image IDs or names you want to report on. The plugin will only pull a report for the image IDs/names you specify.

Enter a single string value like imageIds: 'a1b2c3d4e5f6' or a comma-separated list like imageIds: 'a1b2c3d4e5f6, abcdef123456'. Specify an image name in the format repo:tag.

If you provide an image IDs / image names, the plugin fetches the corresponding image sha-256. The plugin tries to fetch the image sha using the docker socket path configured in global configuration. If your docker host is running locally to build tool/agent, the docker socket path is `unix:///var/run/docker.sock`; whereas if your docker host is running remotely, the docker socket path is the TCP URL to the remote docker host. See [Scanning CI/CD images](#).

You can also provide image ids through an environment variable. Get the image ids of the images programmatically created in earlier stages of the build and provide these ids in the 'imageIds' argument. For example, in pipeline script, you can get the image ids by executing shell script and store it in environment variable. And then use the same environment variable in 'ImageIds' argument to provide the image ids.

For Containerd Images

Containerd images can only be provided in the `repo:tag` format. Enter image name in the format `repo:tag` or comma-seperated images name as `repo:tag,repo-1:tag-1`. If you provide an image repo and tag, the plugin fetches the corresponding image sha-256. The plugin tries to fetch the image sha using the containerd socket path configured in global configuration. Containerd does not support image id/SHA format input for tagging the image. Hence, do not provide image id or sha256 directly as input in the plugin.

Pipeline Project

With pipeline projects, you provide the container image ID(s) to the plugin via a command argument. Use the Snippet Generator to generate this command, and then copy/paste it into your pipeline script (Jenkinsfile).

What are the steps?

- Use a specific tag to build a container image you wish to scan for vulnerabilities.
- Use that tag to get the ID of the container image and then store that image ID into an environment variable.
- Provide that environment variable to the Qualys Jenkins Plugin as input.

Sample Pipeline script

```
stage('Build container image') {  
    //Build the container image with a tag (qualys:sample  
in this case)  
    steps {  
        dir("dockerbuild") {  
            sh "docker build -t qualys:sample . >  
docker_output"  
        }  
    }  
  
    stage('Get Image id') {  
        //Use the same repo:tag (qualys:sample in this case)
```

```
combination with the grep command to get the same image id and save
the image id in an environment variable
    steps {
        script {
            def IMAGE_ID = sh(script: "docker images | grep
-E '^qualys.*sample' | head -1 | awk '{print \$3}'", returnStdout:
true).trim()

            env.IMAGE_ID = IMAGE_ID
        }
    }

    stage('Get Image Vulns - Qualys Plugin') {
        //Use the same environment variable(env.IMAGE_ID) as
an input to Qualys Plugin's step
        steps {
            getImageVulnsFromQualys useGlobalConfig:true,
imageIds: env.IMAGE_ID
        }
    }
}
```

Freestyle Project

You need to provide the container image IDs/Image Names on the plugin configuration page. When Jenkins executes the post-build steps the plugin only pulls a report for the image IDs/names you've specified. You can also provide image ids through an environment variable.

In Post-Build Actions, select "Scan container images with Qualys CS". This opens a form similar to the one shown for pipeline projects. Provide configuration details and test the connection to make sure it's successful. See [How to configure?](#)

Set the IMAGE_ID environment variable to the container image IDs you want to report on. IMAGE_ID can be a single string value like 'a1b2c3d4e5f6' or a comma-separated list like 'a1b2c3d4e5f6,abcdef123456'.

Note: The variable name must be defined correctly or the plugin does not work.

What are the steps?

- Use a specific tag to build a container image you wish to scan for vulnerabilities.
- Add a build step called "Execute shell" in which write a shell script to use that tag to get the ID of the container image and then store that image ID in the format:
IMAGE_ID=<image_id> into a file.

- Add the "Inject environment variables" step in which provide the filename of the file in which you stored the image ID.

The screenshot shows two configuration panels in Jenkins. The top panel, titled "Execute shell", contains a text area with the command: `echo IMAGE_ID=$(docker images | grep -E '^qualys.*sample' | head -1 | awk '{print $3}') > propsfile`. Below the text area is a link: "See the list of available environment variables". The bottom panel, titled "Inject environment variables", has a "Properties File Path" field containing "propsfile" and an empty "Properties Content" text area. Both panels have a red close button (X) and a help icon (?) in the top right corner.

Using the WebHook

You can forward Jenkins job results to a WebHook URL.

You can set a global WebHook URL under Manage Jenkins > Configure System > Qualys Container Security, or a Job Specific WebHook URL under Snippet Generator by selecting "getImageVulnsFromQualys - Scan container images with Qualys CS", and then clicking Advanced.

Note: WebHook URL specified under Snippet Generator, for a particular project, will always take preference over the global WebHook URL specified under Manage Jenkins > Configure System > Qualys Container Security.

WebHook data sample

```
{
  "buildNumber": "12",
  "jobName": "1600Freestyle",
  "jobUrl": "http://localhost:8080/job/1600Freestyle/",
  "buildStatus": "Failed",
  "failReason": [
    {
      "imageId": "1708c4cb79fd",
      "severity": {
        "2": {
          "configured": 0,
          "found": 2
        },
        "3": {
          "configured": 0,
          "found": 20
        }
      }
    }
  ]
}
```

```
,
  "4": {
    "configured": 0,
    "found": 4
  },
  "5": {
    "configured": 0,
    "found": 3
  }
},
"qid": {
  "configured": "256510-256570,176983",
  "found": "176983"
},
"cve": {
  "configured": "CVE-2019-3846,CVE-2019-5489,CVE-2019-9500,CVE-2019-9503,CVE-2019-10126,CVE-2019-11477,CVE-2019-11478,CVE-2019-11479,CVE-2019-11486,CVE-2019-11599,CVE-2019-11815,CVE-2019-11833,CVE-2019-11884",
  "found": "CVE-2019-3846,CVE-2019-5489,CVE-2019-9500,CVE-2019-9503,CVE-2019-10126,CVE-2019-11477,CVE-2019-11478,CVE-2019-11479,CVE-2019-11486,CVE-2019-11599,CVE-2019-11815,CVE-2019-11833,CVE-2019-11884"
},
"cvss": {
  "configured": 0,
  "found": 29,
  "version": 2
},
"software": {
  "configured": "python-urlgrabber, python3",
  "found": "python3=3.5.3-1"
}
},
"images": [
  {
    "imageId": "1708c4cb79fd",
    "uuid": "28bbxxxx-xxxx-xxxx-xxxx-xxxfd626af7a",
    "sha": "1708xxxxxxxxd6cb318xxxxxxxx23a36axxxxxcaxxxxxddcda8c9ac15c48e9de7b5",
    "size": 1242755393,
    "repo": [
      {
        "registry": "docker.io",
        "tag": "latest",
        "repository": "dock_jenkins_3"
      }
    ]
  },
  "operatingSystem": "Debian Linux 9.5",
  "layersCount": 62,
```



```
"dockerVersion": "1.13.1",
"architecture": "amd64",
"vulnerabilities": {
  "totalVulnerabilities": 29,
  "typeDetected": {
    "Confirmed": 24,
    "Potential": 5
  },
  "severity": {
    "Potential": {
      "1": 0,
      "2": 1,
      "3": 2,
      "4": 1,
      "5": 1
    },
    "Confirmed": {
      "1": 0,
      "2": 1,
      "3": 18,
      "4": 3,
      "5": 2
    }
  },
  "patchable": {
    "yes": 26,
    "no": 3
  }
}
]
```

Configuration Details

Provide the following configuration details:

- (1) the Qualys API Server URL. You can find the appropriate server URL from the API URLs section of <https://www.qualys.com/platform-identification/>
- (2) Click **Test Connection** to verify that the plugin can call the Qualys Container Security API.

Configuration Settings

☐ Use Global(Jenkins) Configuration

☒ Use Job Specific Configuration

1 API Login

Provide details for accessing the Qualys Container Security API.

API Server URL: ?
Example: <https://qualysapi.qualys.com>

Credentials ?

☒ Use Proxy Settings

Proxy Server: ?
Examples: 10.15.201.155, corp.proxyserver.company.com

Proxy Port: ?

Credentials ?

2 Test Connection

- (3) data collection frequency.

(4) build fail conditions.

3

Data Collection

Qualys vulnerability data will be collected per these settings. For each enter a value in seconds or an expression like 2*60*60 for 2 hours or 2*60 for 2 minutes.

Frequency

How often to check for data ?

30

seconds.

Timeout

How long to wait for data ?

600

seconds.

4

Configure Container Image Validation Policy

Set the conditions to fail the container image build job. The build will fail when ANY of conditions are met.

Failure Conditions

By Vulnerability Severity

?

☐ Fail with more than

0

severity 1

☐ Fail with more than

0

severity 2

☐ Fail with more than

0

severity 3

☐ Fail with more than

0

severity 4

☒ Fail with more than

0

severity 5

By Qualys Vulnerability Identifiers (QIDs)

?

☐ Fail with any of these QIDs:

By CVEs

?

☐ Fail with any of these CVE Ids:

By Software Names

?

☐ Fail with any of these Softwares:

By CVSS score

?

☒ Fail with

CVSSv2

 Base score

0.0

or above.

☐ Include the above conditions to Potential Vulnerabilities identified too ?

☐ Exclude Conditions

Configure either QIDs or CVEs in below fields which should be ignored while evaluating failure conditions.

?

(5) container image IDs / image names to check for vulnerabilities. We internally use corresponding image sha-256 of the image IDs / image names.

Note: If you are using containerd, the input is expected only in the image name format.

5 Container Images

List of container images to fetch and validate the vulnerability results for.

Image IDs/Image Names ?

(6) Provide the Docker URL/Nerdctl binary path.

6 Runtime Settings

Docker URL/Nerdctl binary path ?

Provide your docker socket path or containerd nerdctl binary file path

Cert file path

If you are using docker with TLS, cert path should be provided in this field.

(7) forward Jenkins job results to a WebHook URL.

7 Advanced Settings

Advanced ^

Webhook URL ?

Add post-build action ▾

Save Apply

When you're ready, click **Generate Pipeline Script** to get the script command.

If you are setting a global configuration, you can select a user from the Credential Store to authenticate to the API Server. In case of Job specific configuration, you can provide the credentials in the pipeline / freestyle script.

Note: Use global configuration for scanning images in CI/CD pipeline. See [Scanning CI/CD images](#).

Irrespective of the image, the plugin always fetches the latest scan result for a given image ID. The API checks the time stamp of the image data in your account, and if the timestamp falls in the time range calculated using the “last scanned” filter, then the API returns the image data to the plugin in the response.

Qualys API Server URL

The Qualys API 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. In the API URLs section, the API Gateway URL column in the table displays the gateway URLs for the corresponding Qualys platforms.

Qualys Container Scanning Connector uses gateway URL internally for both testing connectivity with Qualys platform and pulling vulnerability data from Qualys.

From Qualys Container Scanning Connector v1.6.1.0, the plugin internally translates the platform URL input given by the user to its respective gateway URL.

Example:

If the user with US POD 1 account has configured URL “https://qualysapi.qualys.com”, then while making API call, the plugin will translate this US POD 1 URL to its respective gateway URL that is “https://gateway.qg1.apps.qualys.com”.

If you are already using the plugin, then configure the gateway URL of the platform that has your account for the new jobs.

From Qualys Container Scanning Connector v1.7.x, the user should manually enter the API server URL for the global or job specific configuration. The dropdown to select POD is replaced with the interactive field where the required POD API server URL can be configured.

If the user wants to US POD1 the API server url to be entered in the configuration field should be “https://qualysapi.qualys.com”. This will further be converted into gateway url to generate the auth token and proceed further. The credentials for the same can be saved or selected (if existing already) from the below field.

Encapsulating API user credentials in job specific configuration

Currently, we do not support using API credentials provided in Global Configuration to be used as it is in a Job Specific Configuration, without the need of specifying the credentials for each Job Specific configuration. However, you can create variables to encapsulate the user credentials, and then use the variables instead of actual user credentials while creating a Job Specific Configuration.

To achieve this, on the Pipeline Syntax > Snippet Generator page, select “withCredentials: Bind credentials to variables”. Under Bindings, click Add and then select “Username and password (separated)”.

Specify variable names for Username and Password, select the appropriate credentials, and then click Generate Pipeline Script. See the sample script below:

```
withCredentials([usernamePassword(credentialsId: 'xxxxxx',
passwordVariable: 'PASS', usernameVariable: 'USER')])
{
//contents
}
```

Now, generate your Job Specific Configuration by providing the variables in place of actual username and password. Sample script:

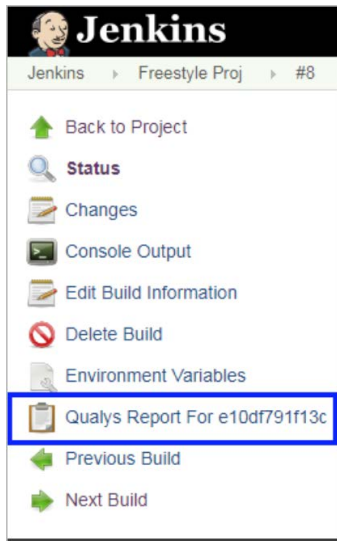
```
getImageVulnsFromQualys apiPass: PASS, apiServer:
'https://qualysguard.qualys.com', apiUser: USER, imageIds:
env.IMAGE_ID, isSev4Vulns: true, isSev5Vulns: true,
pollingInterval: '30', proxyPort: 3128, proxyServer:
'10.115.27.54', useLocalConfig: true, useProxy: true,
vulnsTimeout: '600'
```

Place the Job Specific Configuration script in the “WithCredentials” block:

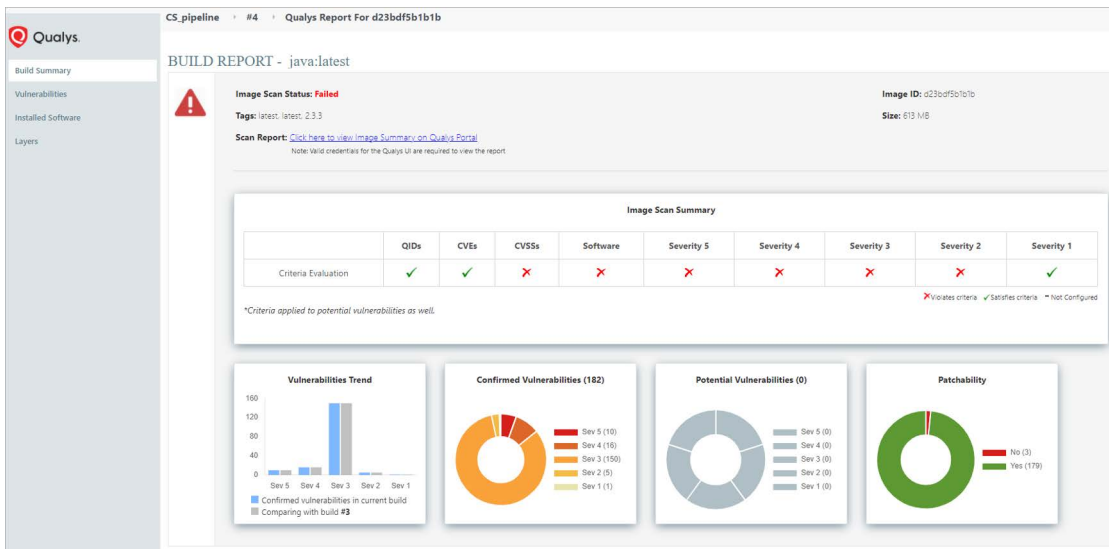
```
withCredentials([usernamePassword(credentialsId: 'xxxxxx',
passwordVariable: 'PASS', usernameVariable: 'USER')])
{
getImageVulnsFromQualys apiPass: PASS, apiServer:
'https://qualysguard.qualys.com', apiUser: USER, imageIds:
env.IMAGE_ID, isSev4Vulns: true, isSev5Vulns: true,
pollingInterval: '30', proxyPort: 3128, proxyServer:
'10.115.27.54', useLocalConfig: true, useProxy: true,
vulnsTimeout: '600'
}
```

View Your Qualys Report

In either case - pipeline project or freestyle project - the plugin will generate one report for each container image in the build. In other words, multiple image Ids will result in multiple report links.



Click any report link to see vulnerability details for the container image.



The Build Summary provides a dashboard view of your security posture. Go to Vulnerabilities for a list of detected QIDs, Installed Software to see software detected on the container image, and Layers to view a list of layers the image is made of.

Debugging and Troubleshooting

Where are the logs?

Windows:

Jenkins logs are located on Windows at JENKINS_HOME path.

For example,

Controller logs - C:\Program Files (x86)\Jenkins\jenkins.err

Agent logs - C:\Program Files (x86)\Jenkins\logs\agent\<agent_name>\agent_log_file

Linux:

Controller logs - /var/log/jenkins/jenkins.log

Agent logs - /var/jenkins/remoting/logs/remoting.log.0

HTTP codes in API response

All API calls and their responses are logged by the plugin and are visible in the Console Output. Here are the HTTP response codes you may see during plugin execution.

Code	Error	Description
204	No content	Qualys sensor is processing data. You can see 200 OK when complete.
200	OK	You would see this code in two situations: 1) You might have received partial data from Qualys where image details are available but vulnerability data is not available. 2) Vulnerability data is also available. This is usually the last call, after which the thread for that image Id stops.
500	Internal server error	Qualys service is down or there was an issue processing data.
400	Bad request	Qualys API server is unable to understand the request.
401	Unauthorized	The credentials used for Qualys API server are incorrect or the user does not have access to the APIs.

If you don't see any API calls being made...

Make sure you are correctly passing image Ids to the plugin. When the plugin starts the execution, it will print the image Ids provided and you can see this in the Console Output. Check that the container image Ids you provided are printed.

Plugin times out, no report seen

The plugin is designed to keep polling the Qualys API until the configured timeout period is reached. If it does not get vulnerability data from Qualys within this period, it stops. In this case, the plugin will fail the build only if you have set any fail-on conditions. Otherwise, it will not fail the build. You will not see any report links since the plugin could not get vulnerability data, and could not prepare a report.

How to fix this?

On the Qualys Cloud Platform, go to Container Security > Assets > Images and verify if the image for which you are checking the vulnerabilities is present in the Images list.

If the image is not present console logs have the following entry:

```
Get scan result API for image e0111ddfea06 returned code : 404;
HTTP Code: 404. Image: Not known to Qualys. Vulnerabilities: To be
processed.. API Response : {"errorCode":"CMS-2002","message":"Data not
available for given Image Id.", "timestamp":1554568122039}
```

Ensure that the Qualys Container Sensor is installed on the host where image is being built.

If the image is present console logs have the following entry:

```
Get scan result API for image cef4ca723229 returned code : 200;
Waiting for vulnerabilities data from Qualys for image id cef4ca723229
HTTP Code: 200. Image: known to Qualys. Vulnerabilities: To be processed.
```

Wait for the vulnerabilities data to be uploaded to the Qualys Cloud Platform.

No space left on device

Console logs have the following entry:

```
Apr 04, 2019 11:20:47 AM
hudson.remoting.JarCacheSupport$DownloadRunnable run
WARNING: Failed to resolve a jar 7cab007bbd804ea3a998084524ce236e
java.io.IOException: Failed to write to
/home/dockerAdmin/.jenkins/cache/jars/7C/AB007BBD804EA3A998084524CE236E.
jar
    at
hudson.remoting.FileSystemJarCache.retrieve(FileSystemJarCache.java:150)
    at
hudson.remoting.JarCacheSupport$DownloadRunnable.run(JarCacheSupport.jav
a:110)
    at
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)
    at java.util.concurrent.FutureTask.run(FutureTask.java:266)
    at
```

```
hudson.remoting.AtmostOneThreadExecutor$Worker.run (AtmostOneThreadExecut
or.java:112)
    at java.lang.Thread.run (Thread.java:748)
Caused by: java.nio.file.FileSystemException:
/home/dockerAdmin/.jenkins/cache/jars/7C: No space left on device
```

Possible reasons could be:

1) Agent machine is running out of disk space.

Use the command **df -h** to check the disk space. Observe 'Use%' and 'Available' space in the output of the command.

Ensure that you have provisioned enough disk space on the host. You may remove unnecessary files to increase free disk space.

2) iNodes are exhausted during the Jenkins plugin job.

Use the command **df -i** to check the iNodes. Observe 'IFree' count and 'IUse%' in the output of the command.

If IUse% is at 100 or near, then large number of small (cached) files are present on the host.

To delete the cached files from the host:

- Run the following command to list all the directories and files:

```
for i in /*; do echo $i; find $i | wc -l; done
```

- Locate the directory with unusually large number of small files, and then run the following command to see where exactly the files are:

For example, if the directory is /home:

```
for i in /home/*; do echo $i; find $i |wc -l; done
```

- Run the following command to remove those files:

```
sudo rm -rf <path of the directory with small files>
```

Incorrect checksum of retrieved jar

Console logs have the following entry:

```
WARNING: Failed to resolve a jar 4fd2a7e0ee23f360d678b2af7903dab0
java.io.IOException: Failed to write to
/home/dockerAdmin/.jenkins/cache/jars/4F/D2A7E0EE23F360D678B2AF7903DAB0.
jar
    at
hudson.remoting.FileSystemJarCache.retrieve (FileSystemJarCache.java:150)
    at
hudson.remoting.JarCacheSupport$DownloadRunnable.run (JarCacheSupport.jav
```

```
a:110)
  at
java.util.concurrent.Executors$RunnableAdapter.call(Executors.java:511)
  at java.util.concurrent.FutureTask.run(FutureTask.java:266)
  at
hudson.remoting.AtmostOneThreadExecutor$Worker.run(AtmostOneThreadExecut
or.java:112)
  at java.lang.Thread.run(Thread.java:748)
Caused by: java.io.IOException: Incorrect checksum of retrieved jar:
```

Jenkins writes the jar required for job execution in its cache on the agent host. This error comes in case the already existing jar in cache is corrupted.

To fix this, clear the cache. The logs show the location of the cache.

For example, in the above logs the cache is located at:

```
/home/dockerAdmin/.jenkins/cache/jars/4F/D2A7E0EE23F360D678B2AF7903DAB0.
jar
```

Execute the command **-ll** to verify that Jenkins has permission to write to the folder:

```
/home/dockerAdmin/.jenkins/
```

Ensure that the cache directory `/home/dockerAdmin/.jenkins/cache/` is writable to the user with which the agent is added to the Jenkins Server. To give permissions, add the respective user owner group OR change the owner to the user with which the Jenkins agent is configured.

Upgrade Issues

If the Qualys Container Scanning Connector is upgraded to version 1.6.2.5 or higher, it is recommended that you reconfigure your existing Qualys Container Scanning operations with Freestyle and/or Pipeline Jobs. Starting with version 1.6.2.5, we are providing a drop-down that contains a list of Qualys platforms to avoid manually entering the input of Qualys API URL. Prior to version 1.6.2.5, the Qualys API URL field was provided in the text box, so this change is not compatible with the previous version.

Want to contact Support?

Access online support information at www.qualys.com/support/

You may need to provide the following information for Qualys Jenkins plugin issues:

- Jenkins version
- Java version on which Jenkins is running
- Version of the Qualys Container Scanning Connector.
- Jenkins controller-agent topology

- Whether the Docker daemon is on Jenkins controller or Jenkins agent or remote host