# Table of Contents

**About this Guide** .................................................................................................................. 5  
About Qualys .......................................................................................................................... 5  
Qualys Support ...................................................................................................................... 5  
About Container Security Documentation ........................................................................... 5  

**Container Security Overview** ............................................................................................ 6  
Qualys Container Sensor ........................................................................................................ 6  
What data does Container Security collect? ......................................................................... 7  

**Get Started** .......................................................................................................................... 8  
Qualys Subscription and Modules required ........................................................................ 8  
System support ....................................................................................................................... 8  
Deploying Container Sensor .................................................................................................... 9  
Proxy Support ......................................................................................................................... 12  
Qualys URL your hosts need to access ............................................................................... 13  
Sensor network configuration ............................................................................................... 13  
Static scanning of Docker images ....................................................................................... 14  

**Installing the sensor on a MAC** ....................................................................................... 15  

**Installing the sensor on CoreOS** ................................................................................... 17  

**Installing the sensor from Docker Hub (Beta)** ............................................................. 19  
Deploying the container sensor on standalone docker host using docker compose .... 19  
Deploying the container sensor on standalone docker host using docker run .......... 23  
Deploying the container sensor using Docker Hub on Kubernetes ............................... 26  

**Installing the CI/CD Sensor in Docker-in-Docker Environment** .................. 32  
Step 1: Have the CS Sensor image inside a Docker-in-Docker Container ....................... 32  
Step 2: Launch the Container Security Sensor ................................................................. 33  

**Deploying sensor in Kubernetes** .................................................................................... 35  
Obtain the Container Sensor Image ........................................................................................ 35  
Deploy in Kubernetes - Docker Runtime ............................................................................ 36  
Deploy in Kubernetes - Containerd Runtime (Beta) ......................................................... 40  
Deploy in Kubernetes with Rancher - Docker Runtime .................................................... 45  
Update the sensor deployed in Kubernetes ...................................................................... 50  

**Deploying sensor in Docker Swarm** .............................................................................. 53  

**Deploying sensor in AWS ECS Cluster** ......................................................................... 57  

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 on deploying the sensor on MAC, CoreOS, and various orchestrators and cloud environments.

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 using the Container Security API, refer to the Qualys Container Security API Guide.

For information on deploying the sensor in CI/CD environments refer to:

- Qualys Vulnerability Analysis Plugin for Jenkins
- Qualys Vulnerability Analysis Plugin for Bamboo
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 ‘Container Sensor’ – providing native container support, distributed as a docker image

Qualys Container Sensor

The sensor from Qualys is designed for native support of Docker environments. Sensor is packaged and delivered as a Docker Image. Download the image and deploy it as a Container alongside with other application containers on the host.

The sensor is docker based, can be deployed on hosts in your data center or cloud environments like AWS ECS. Sensor currently is only supported on Linux Operating systems and requires docker daemon of version 1.12 and higher to be available.

Since they are docker based, the sensor can be deployed into orchestration tool environments like Kubernetes, Mesos or Docker Swarm just like any other application container.
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 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.
Get Started

Follow the steps to get started with Container Security.

Qualys Subscription and Modules required

You would require “Container Security” (CS) module enabled for your account. Additionally, in order to get vulnerabilities for the hosts that run the containers, you would need to enable Vulnerability Management (VM), either via Scanner Appliance or Cloud Agent.

System support

You can install the Container Security Sensor on the following systems running Docker version 1.12 or later:
- CentOS Linux 7.3
- CentOS Linux 7.4
- CentOS Linux 7.5
- CentOS Linux 7.6
- CentOS Linux 7.7
- Ubuntu 14.04
- Ubuntu 16.04
- Ubuntu 18.05
- Debian Linux 8 (Jessie)
- Red Hat Enterprise Linux 7.4
- Red Hat Enterprise Linux Atomic Host 7.5
- Red Hat Enterprise Linux Atomic Host 7.7
- Mac OS 10.13
- Fedora Release 28
- CoreOS 1855.4.0

The Container Security Sensor can scan container images based on the following Operating Systems:
- Red Hat Enterprise Linux Server
- CentOS
Deploying Container Sensor

The Container Security Sensor can be installed in either of the following ways:
- download the sensor tar file from Qualys Cloud Platform and then install it on the host.
- install the sensor from Docker Hub. See Installing the sensor from Docker Hub (Beta).

To download the sensor tar file from Qualys Cloud Platform, log into your Qualys portal with your user credentials. Select Container Security from the module picker.

As a first time user, you’ll land directly into the Home page.
Go to Configurations > Sensors, and then click Download to download the sensor tar file. You can see various sensor types:

- **General (Host) Sensor**: Scan any host other than registry / build (CI/CD).
- **Registry Sensor**: Scan images in a registry (public / private).
- **Build (CI/CD) Sensor**: Scan images on CI/CD pipeline (Jenkins / Bamboo).

For Registry you need to append the install command with `--registry-sensor` or `-r`

For CI/CD you need to append the install command with `--cicd-deployed-sensor` or `-c`

The General Sensor gets installed by default if the parameters for Registry or CI/CD are not provided.

Download the QualysContainerSensor.tar.xz file and run the commands generated directly from the screen on the docker host. Note the requirements for installing the sensor, the sensor needs a minimum of 1 GB persistent storage on the host.
A quick overview of the “installsensor.sh” script command line parameters options:

**ActivationId**: Activation Id for the container sensor, auto-generated based on your subscription.

**CustomerId**: Qualys subscription’s customerId, auto-generated based on your subscription.

**Storage**: Directory where the sensor would store the files. Default: /usr/local/qualys/sensor/data. Create it if not already available or you can specify a custom directory location.

**ImageFile**: Location of the Sensor ImageFile, defaults to the local directory [Optional]

**LogLevel**: Configuration to set the logging level for sensor, accepts 0 to 5. Default is 3 i.e., Information [Optional]

**HostIdSearchDir**: Directory to map the marker file created by Qualys Agent or Scanner appliance on the host, update if modified. Default is /etc/qualys [Optional]

**CpuUsageLimit**: CPU usage limit in percentage for sensor. Valid range is in between 0-100. Default is 0.2 i.e. 20% per core on the host [Optional]

**ConcurrentScan**: Number of docker/registry asset scans to run in parallel. Default is 4 [Optional]

**Proxy**: IPv4/IPv6 address or FQDN of the proxy server [Optional]

**ProxyCertFile**: Proxy certificate file path [Optional]

ProxyCertFile is applicable only if Proxy has valid certificate file. If this option is not provided then Sensor would try to connect to the server with given https Proxy settings only.

If only ProxyCertFile is provided without Proxy then Sensor would simply ignore the ProxyCertFile and it would try to connect to the server without any https proxy settings.

**--silent or -s**: Run installsensor.sh in non-interactive mode [Optional]

**--disable-auto-update**: Do not let sensor update itself automatically [Optional]

**--cicd-deployed-sensor or -c**: Run Sensor in CI/CD environment

**--registry-sensor or -r**: Run sensor to list and scan registry assets

**--enable-console-logs**: Print logs on console. These logs can be retrieved using the docker logs command.

**DockerHost**: IPv4 address or FQDN:Port#. The address on which the docker daemon is configured to listen. [optional]

**DockerSocketDirectory**: Docker socket directory path. Default is Default: /var/run [optional]

**--sensor-without-persistent-storage**: Run the sensor without using persistent storage on host.
--read-only : Run sensor in read-only mode. In this mode the sensor uses persistent storage on host.

Note: The sensor should be run either with “--sensor-without-persistent-storage” option or with “--read-only” option and not with both options enabled together. If you want to install the Sensor without persistent storage, exclude the “Storage” option, and include the “--sensor-without-persistent-storage” option in the installer script. It is recommended to use the “--enable-console-logs” option along with “--sensor-without-persistent-storage” to preserve the logs as data is not available on host but stored at the /usr/local.qualys/qpa/data folder relative to the Sensor.

Only few parameters have default values. These default values can be changed during sensor installation. However, the default values (e.g., LogLevel) once set may get overridden by a config update. If you want to change any default value post sensor installation, you must rerun the “installsensor.sh” script with new values.

For more information on installing the registry sensor, refer to the Qualys Container Sensor User Guide.

For information on installing the sensor from Docker Hub, see Installing the sensor from Docker Hub (Beta).

For information on deploying the sensor in CI/CD environments refer to:
- Qualys Vulnerability Analysis Plugin for Jenkins
- Qualys Vulnerability Analysis Plugin for Bamboo

See About Container Security Documentation.

Note: Your hosts must be able to reach your Qualys Cloud Platform (or the Qualys Private Cloud Platform) over HTTPS port 443. See Qualys URL your hosts need to access.

Proxy Support

The install script asks for proxy configuration. You need to provide the IP Address/FQDN and port number along with the proxy certificate file path. For example,

Do you want connection via Proxy [y/N]: y
Enter Https Proxy settings [<IP Address>:<Port #>]: 10.xxx.xx.xx:3xxx
Enter Https Proxy certificate file path: /etc/qualys/cloud-agent/cert/ca-bundle.crt

Your proxy server must provide access to the Qualys Cloud Platform (or the Qualys Private Cloud Platform) over HTTPS port 443. See Qualys URL your hosts need to access.
Qualys URL your hosts need to access

The Qualys URL you use depends on the Qualys platform where your account is located.

<table>
<thead>
<tr>
<th>PLATFORM</th>
<th>PLATFORM URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>EU1</td>
<td><a href="https://cmsqagpublic.qg1.apps.qualys.eu/ContainerSensor">https://cmsqagpublic.qg1.apps.qualys.eu/ContainerSensor</a></td>
</tr>
<tr>
<td>EU2</td>
<td><a href="https://cmsqagpublic.qg2.apps.qualys.eu/ContainerSensor">https://cmsqagpublic.qg2.apps.qualys.eu/ContainerSensor</a></td>
</tr>
<tr>
<td>IN1</td>
<td><a href="https://cmsqagpublic.qg1.apps.qualys.in/ContainerSensor">https://cmsqagpublic.qg1.apps.qualys.in/ContainerSensor</a></td>
</tr>
</tbody>
</table>

Sensor network configuration

The sensor is pre-configured with the Qualys URL and the subscription details it needs to communicate to. In order for the sensor to communicate to Qualys, the network configuration and firewall needs to provide accessibility to Qualys domain over port 443. See Qualys URL your hosts need to access.

After successful installation of the Sensor, the sensor is listed under Configurations > Sensors where you can see its version, status, etc. and access details.

Additionally, you can Download the sensor from the link under Configurations > Sensors.
Static scanning of Docker images

The sensor will perform static scanning for docker images as a fallback mechanism to current dynamic scanning in case docker images does not have a shell. Static scanning will not be performed on Docker container or Docker images having a shell.

Static scanning collects the list of installed software from the Docker image file system to find vulnerabilities in the Docker images. The installed software list is retrieved from the Package manager metadata files. Package managers supported are RPM, DPKG and Alpine.

Static scanning for Docker images without shell is disabled by default and you can enable it by contacting Qualys Support.
Installing the sensor on a MAC

You can install the Qualys Container Sensor on a MAC.

Here are the steps:

Download the QualysContainerSensor.tar.xz file using the “Download and Install Qualys Container Sensor” link on the Get Started page or from the Configurations > Sensors tab on Qualys Cloud Platform.

Copy the file to the target MAC host. Then run the following commands in sequence.

This command extracts the tar file:

```bash
sudo tar -xvf QualysContainerSensor.tar.xz
```

This command creates the directory where the sensor data like configuration, manifest, logs, and setup is stored:

```bash
sudo mkdir -p /tmp/qualys/sensor/data
```

This command provides required permissions to the directory to run the installer script:

```bash
sudo chmod -R 777 /tmp/qualys/sensor/data
```

If you want to specify a custom location for storage, ensure that the Docker’s File Sharing is enabled for the same. On your MAC host, go to Docker > Preferences > File Sharing, add the custom path e.g. /usr/local/qualys/sensor/data, then click Apply & Restart.

Enabling file sharing is required only if the custom location is NOT from /Users, /Volumes, /private or /tmp.

To avoid this step, we recommend using Storage=/tmp/qualys/sensor/data and HostIdSearchDir=/private/etc/qualys during sensor install.

That way you can leverage the existing shared location with docker, without the need of additional configuration to launch the CS Sensor.

If you are using a custom location, provide permissions to the directory to run the installer script. For example:
sudo chmod -R 777 /usr/local/qualys/sensor/data

The following commands install the sensor. Notice that the command includes the Activation ID and your Customer ID, both generated based on your subscription. The Storage parameter specifies where to install the sensor. Ensure that the HostIdSearchDir exists, otherwise the installer script will throw an error.

Use the following command to install a General Sensor:

```
./installsensor.sh ActivationId=d5814d5f-5fd2-44ec-8969-e03cc58a4ef5
CustomerId=6f35826e-4430-d75e-8356-c444a0abbb31
HostIdSearchDir=/private/etc/qualys Storage=/tmp/qualys/sensor/data -s
```

Use the following command to install a Registry Sensor:

```
./installsensor.sh ActivationId=d5814d5f-5fd2-44ec-8969-e03cc58a4ef5
CustomerId=6f35826e-4430-d75e-8356-c444a0abbb31
HostIdSearchDir=/private/etc/qualys Storage=/tmp/qualys/sensor/data -s -registry-sensor
```

Use the following command to install a CI/CD Sensor:

```
./installsensor.sh ActivationId=d5814d5f-5fd2-44ec-8969-e03cc58a4ef5
CustomerId=6f35826e-4430-d75e-8356-c444a0abbb31
HostIdSearchDir=/private/etc/qualys Storage=/tmp/qualys/sensor/data -s -cicd-deployed-sensor
```

Note: If you want to install the Sensor without persistent storage, exclude the "Storage" option, and include the "--sensor-without-persistent-storage" option in the installer script. It is recommended to use the "--enable-console-logs" option along with "--sensor-without-persistent-storage" to preserve the logs as data is not available on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.
Installing the sensor on CoreOS

You can install the Qualys Container Sensor on CoreOS.

Here are the steps:

Download the QualysContainerSensor.tar.xz file using the “Download and Install Qualys Container Sensor” link on the Get Started page or from the Configurations > Sensors tab on Qualys Cloud Platform.

Copy the file to the target host. Then run the following commands in sequence.

This command extracts the tar file:

```
sudo tar -xvf QualysContainerSensor.tar.xz
```

This command creates the directory where the sensor data like configuration, manifest, logs, and setup is stored:

```
sudo mkdir -p /var/opt/qualys/sensor/data
```

Note: You need to set the directory path `/var/opt/qualys/sensor/data` to Storage which is writable on CoreOS.

This command provides required permissions to the directory to run the installer script:

```
sudo chmod -R 777 /var/opt/qualys/sensor/data
```

The following commands install the sensor. Notice that the command includes the Activation ID and your Customer ID, both generated based on your subscription. The Storage parameter specifies where to install the sensor.

Use the following command to install a General Sensor:

```
Sudo ./installsensor.sh ActivationId=d5814d5f-5fd2-44ec-8969-e03cc58a4ef5 CustomerId=6f35826e-4430-d75e-8356-c444a0abbb31 Storage=/var/opt/qualys/sensor/data/ -s
```

Use the following command to install a Registry Sensor:

```
Sudo ./installsensor.sh ActivationId=d5814d5f-5fd2-44ec-8969-e03cc58a4ef5 CustomerId=6f35826e-4430-d75e-8356-c444a0abbb31 Storage=/var/opt/qualys/sensor/data/ -s --registry_sensor
```

Use the following command to install a CI/CD Sensor:

```
Sudo ./installsensor.sh ActivationId=d5814d5f-5fd2-44ec-8969-e03cc58a4ef5 CustomerId=6f35826e-4430-d75e-8356-c444a0abbb31 Storage=/var/opt/qualys/sensor/data/ -s --cicd_deployed_sensor
```
Note: To install the Sensor without persistent storage, exclude the “Storage” option, and include the “--sensor-without-persistent-storage” option in the installer script. It is recommended to use the “--enable-console-logs” option along with “--sensor-without-persistent-storage” to preserve the logs as data is not available on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.
Installing the sensor from Docker Hub (Beta)

This document provides information on deploying the sensor through Docker Hub:
- On standalone docker host

Deploying the container sensor on standalone docker host using docker compose

Deploying the container sensor on standalone docker host using docker run
- On Kubernetes

Deploying the container sensor using Docker Hub on Kubernetes

The Container Security Sensor on Docker Hub is available as:

```
qualys/qcs-sensor:<tag>
qualys/qcs-sensor:latest
```

Current tag is 1.3.1-22

Deploying the container sensor on standalone docker host using docker compose

Prerequisites:
- Docker engine version: 1.13.0+
- Docker-compose version: 1.14.0+
- Docker-compose file format version: 2.2
- Docker host should be able to communicate with the Docker Hub

Create a new yml file containing the following information. You can name the file `qualys_cs_sensor_docker_compose.yml`.

**Note:** The field alignment in the yml file is very important. Please make sure to honor the formatting provided in the below template.

```
version: '2.2'
services:
  cs_sensor:
    container_name: qualys-container-sensor
    image: qualys/qcs-sensor:latest
    restart: on-failure
    # Uncomment the below security option if SELinux is enabled with enforcing mode on docker host
    # security_opt:
    #   - label:disable

    # Enable the flag if you want to launch CS sensor in read-only mode.
```
Installing the sensor from Docker Hub (Beta)
Deploying the container sensor on standalone docker host using docker compose

```yaml
#   read_only: true
network_mode: host
cpus: 0.2
command: ["--scan-thread-pool-size", "4"]
environment:
  - ACTIVATIONID=<Activation id>
  - CUSTOMERID=<Customer id>
  - POD_URL=<provide POD URL>

# define the proxy if required
#   - qualys_https_proxy= <IP address or FQDN>:<Port#>

# Specify in case sensor should talk to remote docker daemon
#   - DOCKER_HOST=<Remote docker daemon address with port>

volumes:
# Provide host Id search directory path
  - /etc/qualys:/usr/local/qualys/qpa/data/conf/agent-data
# mount volume for docker socket
  - /var/run:/var/run
# mount volume for persistent storage
  - /usr/local/qualys/sensor/data:/usr/local/qualys/qpa/data
# mount volume proxy certificate if required
#   - <Proxy certificate path on host>:/etc/qualys/qpa/cert/custom-ca.crt

Parameters used in the yml file

container_name
set to qualys-container-sensor

image_name
set to qualys/qcs-sensor:<tag>
OR
set to qualys/qcs-sensor:latest

The image will get pulled from the Docker Hub by docker-compose.

restart
Defines the sensor restart policy and should be set to on-failure.

security_opt
This parameter should be used only when SELinux is enabled with enforcing mode on the docker host.

security_opt:
  - label:disable
read-only
Set to true when launching the sensor in read-only mode.

network_mode
Set to host specifying that the sensor needs to be launched with host's network stack.

cpus
Restrict the cpu usage to a certain value.

cpus: 0.2 # Default CPU usage limit (20% of one core/processor on the host).

For example, for limiting the cpu usage to 5%, set cpus: 0.05. This limits the cpu usage to 5% of one core/processor on the host.

If there are multiple processors on a node, setting the cpus value applies the CPU limit to one core/processor only. For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, set cpus value to 0 or remove/comment it out.

Note: If docker host's kernel does not support setting the CPU limit on running containers, disable CPU usage limit, otherwise the sensor won't get launched.

command
If you want to deploy the sensor for CI/CD environment provide the command value as:

command: ["--cicd-deployed-sensor"]

If you want to deploy a Registry Sensor provide the command value as:

command: ["--registry-sensor"]

Note: The General Sensor gets installed by default if the parameters for Registry or CI/CD are not provided.

Additional values you can provide in the command parameter:

"--enable-console-logs" to print logs on console. These logs can be retrieved using the docker logs command.

"--log-level" to set the logging level for sensor, accepts 0 to 5. Default is 3 (Information).

"--scan-thread-pool-size" to launch the sensor with scan thread value. Default is 4.

"--sensor-without-persistent-storage" to run the sensor without using persistent storage on host. In this case do not provide persistent storage mapping under volumes. It is recommended to use the "--enable-console-logs" option along with "--sensor-without-persistent-storage" to preserve the logs as data is not available on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.
Example,

command: 
```yaml
["--cicd-deployed-sensor", "--sensor-without-persistent-storage", "--enable-console-logs"]
```

volumes:

# mount volume for persistent storage
# -/usr/local/qualys/qpa/data

evironment

Provide the ACTIVATIONID, CUSTOMERID, and POD_URL from your subscription. To get
the Activation ID and Customer ID, login to the Container Security UI, go to Configurations
> Sensors, click Download, and then click any sensor type. The installation command on
the Installation Instructions screen contains your Activation ID and Customer ID.
Activation ID is like a password, do not share it.

Note: Your hosts must be able to reach your Qualys Cloud Platform (or the Qualys Private
Cloud Platform) over HTTPS port 443. See Qualys URL your hosts need to access.

Specify the DOCKER_HOST if the Docker daemon is communicating over TCP port. In this
case DO NOT provide mapping for Docker Unix socket file under volumes.

```yaml
DOCKER_HOST=<IPv4 address or FQDN>:<Port#>
```

volumes:

# mount volume for docker socket
# - /var/run:/var/run

Specify qualys_https_proxy if a proxy is required for the sensor to communicate with the
Qualys Cloud Platform.

- qualys_https_proxy=<IP/ address or FQDN>:<Port#>

volumes

Specify the persistent storage mapping to launch the sensor with persistent storage. The
persistent storage directory is automatically created if doesn’t exist.

```yaml
volumes:
# mount volume for persistent storage
-/usr/local/qualys/sensor/data:/usr/local/qualys/qpa/data
```

Specify hostid directory location if you want to use the same hostid used in the previous
installation.

```
# Provide host Id search directory path
-/etc/qualys:/usr/local/qualys/qpa/data/conf/agent-data
```

Map the Unix socket file to sensor file system if the Docker daemon on the Docker host is
communicating over Unix socket.

```
# mount volume for docker socket
-/var/run:/var/run
```
Note: If the Docker daemon is communicating over TCP port specify the DOCKER_HOST parameter under environment and DO NOT provide mapping for docker unix socket file under volumes.

Specify the proxy certificate (if required):
- `<Proxy certificate path on host>:/etc/qualys/qpa/cert/custom-ca.crt`

**Launching the sensor**

Once the yml file is created, use the following command to launch the sensor:

```
docker-compose -f <path to qualys_cs_sensor_docker_compose.yml file> up -d
```

**Upgrading the sensor**

The Qualys Container Sensor image hosted on Docker Hub do not support auto update. Perform the following steps to update the sensor installed from Docker Hub:

1. Update the image name in the yml file:
   
   Set to `qualys/qcs-sensor:<tag>`
   OR
   Set to `qualys/qcs-sensor`

2. Run the command to recreate the sensor:

   ```
docker-compose -f <path to qualys_cs_sensor_docker_compose.yml file> up -d
```

**Removing the sensor**

Run the following command to remove the sensor:

```
docker-compose -f <path to qualys_cs_sensor_docker_compose.yml file> rm -s
```

Note: The docker-compose does not provide an option to delete the persistent storage. You must delete the persistent storage files manually.

**Deploying the container sensor on standalone docker host using docker run**

Prerequisites: Docker engine version: 1.13.0+

Run the following commands to install the sensor. Provide the ACTIVATIONID, CUSTOMERID, and POD_URL from your subscription. To get the Activation ID and Customer ID, login to the Container Security UI, go to Configurations > Sensors, click Download, and then click any sensor type. The installation command on the Installation Instructions screen contains your Activation ID and Customer ID. Activation ID is like a password, do not share it.
Installing the sensor from Docker Hub (Beta)
Deploying the container sensor on standalone docker host using docker run

**General Sensor**

docker run -d --restart on-failure -v /var/run:/var/run -v /usr/local/qualys/sensor/data:/usr/local/qualys/qpa/data -e ACTIVATIONID=<Activation id> -e CUSTOMERID=<Customer id> -e POD_URL=<POD URL> --net=host --name qualys-container-sensor qualys/qcs-sensor:latest

**Registry Sensor**

docker run -d --restart on-failure -v /var/run:/var/run -v /usr/local/qualys/sensor/data:/usr/local/qualys/qpa/data -e ACTIVATIONID=<Activation id> -e CUSTOMERID=<Customer id> -e POD_URL=<POD URL> --net=host --name qualys-container-sensor qualys/qcs-sensor:latest --registry-sensor

**CI/CD Sensor**

docker run -d --restart on-failure -v /var/run:/var/run -v /usr/local/qualys/sensor/data:/usr/local/qualys/qpa/data -e ACTIVATIONID=<Activation id> -e CUSTOMERID=<Customer id> -e POD_URL=<POD URL> --net=host --name qualys-container-sensor qualys/qcs-sensor:latest --cicd-deployed-sensor

**Volumes used in the above commands:**

/\var/\/\var/\ run - mounts the Docker socket directory to the sensor file system. This is mandatory unless user specifies the DOCKER_HOST environment variable if docker daemon is running on TCP port.

/usr/local/\qualys/\sensor/\data:/usr/\loca//\qualys/\qpa/\data - provides persistent storage for the sensor container. This mapping is mandatory unless "--sensor-without-persistent-storage" option is used. You may change the storage directory. The directory is automatically created if doesn't exist.

Additional environment variables/volumes can be provided:

1. If proxy is used to communicate with Qualys Cloud Platform, specify

- e qualys_https_proxy=<IP/ address or FQDN>:<Port#>

2. If the proxy cert is required, mount volume for proxy certificate by adding

- v <Proxy_File_Path>/etc/qualys/qpa/cert/custom-ca.crt

3. If the Docker daemon is running on TCP port, specify - e DOCKER_HOST=<TCP socket>. Ensure that you remove the Docker socket mount volume(-v /var/run:/var/run -) in this case.

4. /etc/qualys:/usr/\loca//\qualys/\qpa/\data/\conf/agent-data - HostID search directory to map the marker file created by Qualys Agent or Scanner appliance on the host.
Optional Parameters

--cpus
Restrict the cpu usage to a certain value.

--cpus=0.2 # Default CPU usage limit (20% of one core/processor on the host).

For example, for limiting the cpu usage to 5%, set --cpus=0.05. This limits the cpu usage to 5% of one core/processor on the host.

If there are multiple processors on a node, setting the cpus value applies the CPU limit to one core/processor only. For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, do not specify the option.

Note: If docker host’s kernel does not support setting the CPU limit on running containers, disable CPU usage limit, otherwise the sensor won’t get launched.

--enable-console-logs
Print logs on console. These logs can be retrieved using the docker logs command.

--sensor-without-persistent-storage
Run the sensor without using persistent storage on host. In this case do not provide persistent storage mapping under volumes. It is recommended to use the "--enable-console-logs" option along with "--sensor-without-persistent-storage" to preserve the logs as data is not available on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.

--log-level
Set the logging level for sensor, accepts 0 to 5. Default is 3 (Information).

--scan-thread-pool-size
Launch the sensor with scan thread value. Default is 4.

--read-only
Run sensor in read-only mode. In this mode the sensor uses persistent storage on host.

Note: The sensor should be run either with "--sensor-without-persistent-storage" option or with "--read-only" option and not with both options enabled together.
Deploying the container sensor using Docker Hub on Kubernetes

Prerequisites:
- Kubernetes setup should be up and running
- K8S nodes should be able to communicate with the Docker hub/private registry
- The container sensor image should be available in the private registry if you are installing from there.

Modify the cssensor-ds.yml file

Create a new yaml file containing the following information and name it `cssensor-ds.yml`
or download the yaml file directly from `https://github.com/Qualys/cs_sensor`

Note: The field alignment in the yaml file is very important. Please make sure to honor the formatting provided in the below template.

```yaml
apiVersion: extensions/v1beta1
category: DaemonSet
metadata:
  name: qualys-container-sensor
  namespace: kube-system
  labels:
    k8s-app: qualys-cs-sensor
spec:
  selector:
    matchLabels:
      name: qualys-container-sensor
  updateStrategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        name: qualys-container-sensor
    spec:
      containers:
        - name: qualys-container-sensor
          image: <CS Sensor image name in the docker hub/private registry>
          resources:
            limits:
              cpu: 0.2 # Default CPU usage limit on each node for sensor.
          args: ["--k8s-mode"]
          env:
            - name: ACTIVATIONID
              value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXXX
            - name: CUSTOMERID
```
value: XXXXXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXXXX
- name: POD_URL
  value:
- name: qualys_https_proxy
  value: proxy.qualys.com:3128
volumeMounts:
- mountPath: /var/run
  name: socket-volume
- mountPath: /usr/local/qualys/qpa/data
  name: persistent-volume
- mountPath: /usr/local/qualys/qpa/data/conf/agent-data
  name: agent-volume
- mountPath: /etc/qualys/qpa/cert/custom-ca.crt
  name: proxy-cert-path
securityContext:
  allowPrivilegeEscalation: false
volumes:
- name: socket-volume
  hostPath:
    path: /var/run
    type: Directory
- name: persistent-volume
  hostPath:
    path: /usr/local/qualys/sensor/data
    type: DirectoryOrCreate
- name: agent-volume
  hostPath:
    path: /etc/qualys
    type: DirectoryOrCreate
- name: proxy-cert-path
  hostPath:
    path: /root/cert/proxy-certificate.crt
    type: File
hostNetwork: true

In order for the yml file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Ensure that from all Kubernetes nodes the Docker hub/private registry (where the CS Sensor image is published) is accessible.

containers:
  - name: qualys-container-sensor
    image: <CS Sensor image name in the private/docker hub registry>
    args: ['--k8s-mode']
If you want to deploy the sensor for CI/CD environment provide the args value as:

```yaml
args: [--k8s-mode, --cicd-deployed-sensor]
```

If you want to deploy a Registry Sensor provide the args value as:

```yaml
args: [--k8s-mode, --registry-sensor]
```

Note: General Sensor is installed if neither value is specified.

If you want to print logs on the console, provide `--enable-console-logs` as an additional value in args.

```yaml
args: [--k8s-mode, --enable-console-logs]
```

If you want to launch the sensor without persistent storage, provide `--sensor-without-persistent-storage` as an additional value in args.

```yaml
args: [--k8s-mode, --sensor-without-persistent-storage]
```

In this case, data is not stored on the host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor. It is recommended to use the `--enable-console-logs` option along with `--sensor-without-persistent-storage` to preserve the logs.

Under `volumeMounts` remove/comment the persistent-volume section.

```yaml
volumeMounts:
  - mountPath: /usr/local/qualys/qpa/data
    name: persistent-volume
```

Under `volumes` remove/comment the persistent-volume section.

```yaml
volumes:
  - name: persistent-volume
    hostPath:
      path: /usr/local/qualys/sensor/data
      type: DirectoryOrCreate
```

If you want to change the log level, provide `--log-level", "<a number between 0 and 5>"` as an additional value in args, e.g., if you want logs in trace mode provide 5:

```yaml
args: [--k8s-mode, --log-level, "5"]
```

If you want to launch the sensor with scan thread value other than default 4, provide `--scan-thread-pool-size", "<number of threads>"` as an additional value in arg.

```yaml
args: [--k8s-mode, --scan-thread-pool-size, "6"]
```
To restrict the cpu usage to a certain value, change the following: (Optional)
Under **resources** specify the following:

```yaml
resources:
  limits:
    cpu: 0.2 # Default CPU usage limit (20% of one core on the host).
```

For example, for limiting the cpu usage to 5%, set `resources:limits:cpu: 0.05`. This limits the cpu usage to 5% of one core on the host.

If there are multiple processors on a node, setting the `resources:limits:cpu` value applies the CPU limit to one core only.

For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, set `resources:limits:cpu` value to 0.

Under **env** specify the following:

**Activation ID (Required)**

- name: ACTIVATIONID
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

**Customer ID (Required)**

- name: CUSTOMERID
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

**POD_URL (Required)**

- name: POD_URL
  value: <Specify POD URL>

Specify proxy information, or remove if not required:

- name: qualys_https_proxy
  value: proxy.localnet.com:3128

Under **volumes** specify the proxy cert path, or remove if not required:

- name: proxy-cert-path
  hostPath:
    path: /root/cert/proxy-certificate.crt
    type: File
Provide the ACTIVATIONID, CUSTOMERID, and POD_URL from your subscription. To get the Activation ID and Customer ID, login to the Container Security UI, go to Configurations > Sensors, click Download, and then click any sensor type. The installation command on the Installation Instructions screen contains your Activation ID and Customer ID. Activation ID is like a password, do not share it.

If you are using a proxy, ensure that all Kubernetes nodes have a valid certificate file for the sensor to communicate with the Qualys Cloud Platform.

If you are not using a proxy and you have removed the above mentioned parts, you can remove the following part from volumeMounts as well:

```
- mountPath: /etc/qualys/qpa/cert/custom-ca.crt
  name: proxy-cert-path
```

**Remove hostNetwork: true when IP identification is not necessary**

Remove/comment out the `hostNetwork: true` option to follow the security best practice of least privilege, where host IP address identification is not necessary.

```
hostNetwork: true
```

The `hostNetwork: true` option provides the sensor the ability to detect the Docker host’s IP address. The user can remove/comment out the `hostNetwork: true` line in the yaml file, however a drawback is that the UI will show the docker bridge network IP address as the host IP address. To aid with host identification, the hostname appears in the UI so customers can identify the Docker host by hostname. We also recommend that the hostname be unique within the environment.

**Deploying the Container Sensor DaemonSet**

Once you have created the `cssensor-ds.yml` file, run the following command on Kubernetes master to create a DaemonSet:

```
kubectl create -f cssensor-ds.yml
```

**Removing the Container Sensor DaemonSet**

If you need to uninstall Qualys Container Sensor, run the following command on Kubernetes master:

```
kubectl delete -f cssensor-ds.yml
```

**Upgrading the Container Sensor DaemonSet**

Perform the following steps on Kubernetes master for updating the Container Sensor DaemonSet to a new version.

Note: Ensure that the Container Sensor DaemonSet is running in the Kubernetes environment.
1. Get the name of the Container Sensor DaemonSet
   
   `kubectl get ds -n kube-system`

2. Update the image for the DaemonSet to use new qualys/sensor image
   
   `kubectl set image ds/<daemonset-name> -n kube-system <container-name>=<container-new-image>`

3. Monitor Rollout status - this may take a while depending on the number of nodes
   
   `kubectl rollout status daemonset <daemonset-name> -n kube-system`

4. If something goes wrong during rollout, rollback the DaemonSet to use the last image
   
   `kubectl rollout undo daemonset <daemonset-name> -n kube-system`
Installing the CI/CD Sensor in Docker-in-Docker Environment

In this section we’ll describe how to install the CS Sensor in a CI/CD pipeline build for a Docker-in-Docker environment. This will allow you to scan images inside the Docker-in-Docker container.

Step 1: Have the CS Sensor image inside a Docker-in-Docker Container

There are two ways to do this: 1) You can pull the CS Sensor image from the registry and launch the sensor when the container is spun up, or 2) you can bake the Docker-in-Docker container image with the CS Sensor tar in it.

Pull the CS Sensor image from the registry and launch the sensor

Benefits:
- No need to have pre-baked Docker-in-Docker container image with CS Sensor image/tar.
- You can easily use CS sensor image hosted on Docker hub registry

Disadvantages:
- All Docker-in-Docker containers need to have access to the registry.
- An image is pulled each time a Docker-in-Docker container is spun up and that would be overhead.

Pre-baked Docker-in-Docker container image with CS sensor tar in it

Benefits:
- No need to have access to the registry from Docker-in-Docker container.
- The execution of a few commands and installsensor.sh script is enough to launch the CS Sensor.

Disadvantages:
- The Docker-in-Docker container image size will be increased.
- You’ll need to re-bake the Docker-in-Docker image for each new sensor release.
Step 2: Launch the Container Security Sensor

There are two ways to do this: 1) You can launch the sensor when the Docker-in-Docker container boots up, or 2) launch the sensor from a build job. We’ll describe both methods.

Launch sensor on Docker-in-Docker container bootup

Benefits:
- No need to modify the build pipeline configuration to launch the CS Sensor.

Disadvantages:
- The credentials (Activation ID/Customer ID) need to be stored in the init script.
- Only predefined persistent storage path can be provided.

Launch init script inside the Docker-in-Docker container

Use the init script to launch the sensor. The init script will have following command:

docker run -d --restart on-failure --cpus=0.2 -v /etc/qualys:/usr/local/qualys/qpa/data/conf/agent-data -v /var/run:/var/run - v /usr/local/qualys/sensor/data:/usr/local/qualys/qpa/data -e ACTIVATIONID=<Activation id> -e CUSTOMERID=<Customer id> -e POD_URL=<POD URL> --net=host --name qualys-container-sensor <Qualys CS Sensor image name from registry> --scan-thread-pool-size 4 --cicd-deployed-sensor

Use installsensor.sh script

Use the installsensor.sh script to launch the sensor on Docker-in-Docker bootup.

tar –xvf QualysContainerSensor.tar.xz

docker load -i qualys-sensor.tar

./installsensor.sh ActivationId=<Activation id> CustomerId=<Customer id> HostIdSearchDir=/private/etc/qualys Storage=/tmp/qualys/sensor/data --cicd-deployed-sensor -s

Launch sensor from build job

Benefits:
- Credentials (AI/CI) and sensor parameters can be passed from build job configuration.
- The persistent storage can be defined during launch.
- It’s easy to have a unique directory for each job (using Job ID) and using it as persistent storage.

Disadvantages:
- You’ll need to modify the build job configuration to launch the CS Sensor.
Installing the CI/CD Sensor in Docker-in-Docker Environment
Step 2: Launch the Container Security Sensor

Launch CS Sensor using docker run command to pull image from registry
Launch the CS Sensor using the docker run command in order to pull the CS Sensor image from the registry.

docker run -d --restart on-failure --cpus=0.2 -v /etc/qualys:/usr/local/qualys/qpa/data/conf/agent-data -v /var/run:/var/run -v /usr/local/qualys/sensor/data:/usr/local/qualys/qpa/data -e ACTIVATIONID=<Activation id> -e CUSTOMERID=<Customer id> -e POD_URL=<POD URL> --net=host --name qualys-container-sensor <Qualys CS Sensor image name from registry> --scan-thread-pool-size 4 --cicd-deployed-sensor

Launch CS Sensor as part of a build job using pre-baked Docker-in-Docker image
This command will launch the CS Sensor as part of a build job using a pre-baked Docker-in-Docker container image with the CS Sensor tar in it. It will launch the CS Sensor as part of the job.

<path>/installsensor.sh ActivationId=<Activation id> CustomerId=<Customer id> HostIdSearchDir=/private/etc/qualys Storage=/tmp/qualys/sensor/data --cicd-deployed-sensor -s

Persistent storage for CS sensor running in Docker-in-Docker build container
Please provide the appropriate persistent storage for CS sensor so that the logs can be retrieved in case of CS sensor failure or container image scan failure.
Deploying sensor in Kubernetes

This section provides steps for deploying the container sensor in Kubernetes.

Jump to a section:
- Obtain the Container Sensor Image
- Deploying the container sensor using Docker Hub on Kubernetes
- Deploy in Kubernetes - Docker Runtime
- Deploy in Kubernetes - Containerd Runtime (Beta)
- Deploy in Kubernetes with Rancher - Docker Runtime
- Update the sensor deployed in Kubernetes

Obtain the Container Sensor Image

The first step for any Kubernetes deployment is to obtain the sensor image. You can download QualysContainerSensor.tar.xz from the Container Security UI. Or you can use the latest Qualys Container Sensor image - qualys/qcs-sensor:latest - from Docker hub.

Download from UI

Download the QualysContainerSensor.tar.xz file from the UI on a Linux computer with Docker installed on it.

Note: Containerd runtime is only supported by the general (host) sensor. It is not supported by registry sensor or build CI/CD sensor.
After downloading the file, untar the sensor package using this command:

```
sudo tar -xvf QualysContainerSensor.tar.xz
```

Push the Qualys sensor image to a repository common to all nodes in the Kubernetes cluster using these commands:

```
sudo docker load -i qualys-sensor.tar
sudo docker tag <IMAGE NAME/ID> <URL to push image to the repository>
sudo docker push <URL to push image to the repository>
```

For example:

```
sudo docker load -i qualys-sensor.tar
sudo docker tag c3fa63a818df mycloudregistry.com/container-sensor:qualys-sensor-xxx
sudo docker push mycloudregistry.com/container-sensor:qualys-sensor-xxx
```

**Note:** Do not use these examples as is. Replace the registry/image path with your own.

**Get image from docker hub**

Use the latest Qualys Container Sensor image - qualys/qcs-sensor:latest - from Docker hub. The Container Security Sensor on Docker Hub is available as:

- qualys/qcs-sensor:<tag>
- qualys/qcs-sensor:latest

*Current tag is 1.3.1-22*

The Docker hub Qualys Container Sensor image can either be pushed to your private registry or used directly. Ensure that from all Kubernetes nodes the Docker hub/private registry (where the CS Sensor image is published) is accessible.

**Deploy in Kubernetes - Docker Runtime**

This section assumes you have the sensor image: Obtain the Container Sensor Image

Integrate the Container Sensor into the DaemonSet like other application containers and set the replication factor to 1 to ensure there is always a sensor deployed on the Docker Host. This information is applicable for Amazon Elastic Container Service for Kubernetes (Amazon EKS), Google Kubernetes Engine (GKE), and Azure Kubernetes Service (AKS).

Perform the following steps for creating a DaemonSet for the Qualys sensor to be deployed in Kubernetes.

**Note:** Ensure that the Container Sensor has read and write access to the persistent storage and the docker daemon socket.
Modify the cssensor-ds.yml file

Modify the cssensor-ds.yml file (extracted from QualysContainerSensor.tar.xz) to provide values for the following parameters. In order for the yml file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Note that you can download the yml file directly from https://github.com/Qualys/cs_sensor

Ensure all Kubernetes nodes have the latest Qualys sensor image from the URL provided.

```yaml
containers:
  - name: qualys-container-sensor
    image: mycloudregistry.com/container-sensor:qualys-sensor-xxx
    args: ["--k8s-mode"]
```

If you want to deploy the sensor for CI/CD environment provide the `args` value as:

```yaml
args: ["--k8s-mode", "--cicd-deployed-sensor"]
```

If you want to deploy a Registry Sensor provide the `args` value as:

```yaml
args: ["--k8s-mode", "--registry-sensor"]
```

If you want print logs on the console, provide `--enable-console-logs` as an additional value in `args`.

To restrict the cpu usage to a certain value, change the following: (Optional)

Under `resources` specify the following:

```yaml
resources:
  limits:
    cpu: "0.2" # Default CPU usage limit (20% of one core on the host).
```

For example, for limiting the cpu usage to 5%, set `resources:limits:cpu: "0.05"`. This limits the cpu usage to 5% of one core on the host.

If there are multiple processors on a node, setting the `resources:limits:cpu` value applies the CPU limit to one core only. For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, set `resources:limits:cpu` value to 0.

Under `env` specify the following:

Activation ID (Required)

```yaml
- name: ACTIVATIONID
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
```

Customer ID (Required)

```yaml
- name: CUSTOMERID
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
```
Specify proxy information, or remove if not required:

- name: qualys_https_proxy
  value: proxy.localnet.com:3128

Under **volumes** specify the proxy cert path, or remove if not required:

- name: proxy-cert-path
  hostPath:
    path: /root/cert/proxy-certificate.crt
    type: File

Activation ID and Customer ID are required. Use the Activation ID and Customer ID from your subscription.

If you are using a proxy, ensure that all Kubernetes nodes have a valid certificate file for the sensor to communicate with the Container Management Server.

If you are not using a proxy and you have removed the above mentioned parts, you can remove the following part from **volumeMounts** as well:

- mountPath: /etc/qualys/qpa/cert/custom-ca.crt
  name: proxy-cert-path

Once you have modified the **cssensor-ds.yml** file, run the following command on Kubernetes master to create a DaemonSet:

```bash
kubectl create -f cssensor-ds.yml
```

If you need to uninstall Qualys Container Sensor, run the following command on Kubernetes master:

```bash
kubectl delete -f cssensor-ds.yml
```

**Using Persistent Volume Claims**

You can use PersistentVolumeClaim (PVC) to request for storage of specific size from the gross Persistent Volume you have specified.

**Modify the cssensor-ds_pv_pvc.yml file**

Modify the **cssensor-ds_pv_pvc.yml** file (extracted from QualysContainerSensor.tar.xz) to provide values for the following parameters. In order for the yml file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Note that you can download the yml file directly from https://github.com/Qualys/cs_sensor

```yaml
kind: PersistentVolume
apiVersion: v1
metadata:
  name: qualys-sensor-pv-volume
labels:
  type: local
```
Deploying sensor in Kubernetes

Deploy in Kubernetes - Docker Runtime

```yaml
spec:
  storageClassName: manual
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: "/mnt/data"

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: qualys-sensor-pv-claim
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

Here a PVC of 1Gi is made on a Persistent Volume to 5Gi. Click here for a list of supported Persistent Volume Types.

Add the name of the PVC under volumes:

```yaml
- name: persistent-volume
  persistentVolumeClaim:
    claimName: qualys-sensor-pv-claim
```

**Launch sensor without persistent storage**

You can run the sensor without using persistent storage on host. In this case data is not stored on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.

To launch sensor without persistent storage, modify the `cssensor-ds.yml` file and provide "--sensor-without-persistent-storage" as an additional value in `args`.

```bash
args: ["--k8s-mode","--sensor-without-persistent-storage"]
```

It is recommended to use the "--enable-console-logs" option along with "--sensor-without-persistent-storage" to preserve the logs.

Under `volumeMounts` remove/comment the persistent-volume section.

```yaml
volumeMounts:
- mountPath: /usr/local/qualys/qpa/data
  name: persistent-volume
```
Under **volumes** remove/comment the persistent-volume section.

```yaml
volumes:
- name: persistent-volume
  hostPath:
    path: /usr/local/qualys/sensor/data
    type: DirectoryOrCreate
```

### Deploy in Kubernetes - Containerd Runtime (Beta)

This section assumes you have the sensor image: [Obtain the Container Sensor Image](#).

**Note:** Containerd runtime is only supported by the general (host) sensor. It is not supported by registry sensor or build CI/CD sensor.

**Modify the cssensor-containerd-ds.yml file**

Below is the Kubernetes DaemonSet deployment template that can be used to deploy the Qualys Container Sensor. For customers’ convenience this template is available in the QualysContainerSensor.tar.xz. Note that you can download the yml file directly from [https://github.com/Qualys/cs_sensor](https://github.com/Qualys/cs_sensor).

**IMPORTANT:** The field alignment in the yml file is very important. Please make sure to honor the formatting provided in the template.

```yaml
kind: List
apiVersion: v1
items:
  # Service Account
  - kind: ServiceAccount
    apiVersion: v1
    metadata:
      name: qualys-service-account
      namespace: kube-system
  # ClusterRole for read permission to whole cluster
  - kind: ClusterRole
    apiVersion: rbac.authorization.k8s.io/v1beta1
    metadata:
      name: qualys-cluster-reader-role
    rules:
      - apiGroups: ["*"]
        resources: ["*"]
        verbs: ["get", "list", "watch", "create", "delete", "deletecollection"]
    # RoleBinding to assign permissions in qualys-cluster-reader-role to qualys-service-account
    - kind: ClusterRoleBinding
      apiVersion: rbac.authorization.k8s.io/v1beta1
```

---

40
metadata:
  name: qualys-cluster-reader-rb
subjects:
- kind: ServiceAccount
  name: qualys-service-account
  namespace: kube-system
roleRef:
  kind: ClusterRole
  name: qualys-cluster-reader-role
  apiGroup: rbac.authorization.k8s.io

# Qualys Container Sensor pod with
- kind: DaemonSet
  apiVersion: apps/v1
metadata:
  name: qualys-container-sensor
  namespace: kube-system
  labels:
    k8s-app: qualys-cs-sensor
spec:
  selector:
    matchLabels:
      name: qualys-container-sensor
  updateStrategy:
    type: RollingUpdate
  template:
    metadata:
      labels:
        name: qualys-container-sensor
    spec:
      serviceAccountName: qualys-service-account
      containers:
      - name: qualys-container-sensor
        image: <CS Sensor image name in the private/docker hub registry>
        resources:
          limits:
            cpu: "0.2" # Default CPU usage limit on each node for sensor.
        args: ["--k8s-mode", "--container-runtime", "containerd"]
        env:
          - name: ACTIVATIONID
            value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
          - name: CUSTOMERID
            value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
          - name: POD_URL
Deploy in Kubernetes - Containerd Runtime (Beta)

Value: {{POD_URL}}
- name: QUALYS_SCANNING_CONTAINER_LAUNCH_TIMEOUT
  value: "10"
- name: qualys_https_proxy
  value: {{PROXY_URL}}

volumeMounts:
- mountPath: /var/run
  name: socket-volume
- mountPath: /usr/local/qualys/qpa/data
  name: persistent-volume
- mountPath: /usr/local/qualys/qpa/data/conf/agent-data
  name: agent-volume
- mountPath: /etc/qualys/qpa/cert/custom-ca.crt
  name: proxy-cert-path

securityContext:
  allowPrivilegeEscalation: false

volumes:
- name: socket-volume
  hostPath:
    path: /var/run
    type: Directory
- name: persistent-volume
  hostPath:
    path: /usr/local/qualys/sensor/data
    type: DirectoryOrCreate
- name: agent-volume
  hostPath:
    path: /etc/qualys
    type: DirectoryOrCreate
- name: proxy-cert-path
  hostPath:
    path: /root/cert/proxy-certificate.crt
    type: File
hostNetwork: true

Qualys Container Sensor DaemonSet should be deployed as a part of ServiceAccount with adequate permission to communicate with Kubernetes API Server. ClusterRole and ClusterRoleBindings are used to assign the necessary permissions to the ServiceAccount.

You’ll need these permissions:

get, list, watch - to monitor the resources to be scanned for vulnerabilities
create, delete, deletecollection - to spawn containers for image vulnerability assessment and then clean up after itself

**Note:** Ensure that the Container Sensor has read and write access to the persistent storage and the containerd daemon socket.
Modify parameters in the yaml file

Copy the cssensor-containerd-ds.yml file to Kubernetes cluster’s master node then modify it by providing values for the following parameters. In order for the yaml file to work properly, ensure you only update the parameters/sections specified below. Note that you can download the yml file directly from https://github.com/Qualys/cs_sensor

containers:
  - name: qualys-container-sensor
    image: <CS Sensor image name in the private/docker hub registry>
    args: ["--k8s-mode", "--container-runtime", "containerd"]

General Sensor is assumed here. As mentioned earlier, containerd is only supported by the general (host) sensor. It is not supported by registry sensor or build CI/CD sensor.

If you want to change the log level, provide "--log-level","<a number between 0 and 5>" as an additional value in args, e.g if you want logs in trace provide:

    args: ["--k8s-mode", "--container-runtime", "containerd", "--log-level", "5"]

If you want to launch the sensor with scan thread value other than default 4, provide "--scan-thread-pool-size","<number of threads>" as an additional value in arg.

    args: ["--k8s-mode", "--container-runtime", "containerd", "--scan-thread-pool-size", "6"]

Under resources specify the following:

    resources:
      limits:
        cpu: 0.2 # Default CPU usage limit (20% of one core on the host).

For example, for limiting the cpu usage to 5%, set resources:limits:cpu: "0.05". This limits the cpu usage to 5% of one core on the host.

If there are multiple processors on a node, setting the resources:limits:cpu value applies the CPU limit to one core only. For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, set resources:limits:cpu value to 0.

Under env specify the following:

    Activation ID (Required)
    - name: ACTIVATIONID
      value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXX

    Customer ID (Required)
    - name: CUSTOMERID
      value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXXX
Specify POD_URL when using docker hub image. Otherwise, remove it.

- name: POD_URL
  value: <Specify POD URL>

Specify the scanning container launch timeout in minutes. If this env variable is not present, then 10 minutes is the default.

- name: QUALYS_SCANNING_CONTAINER_LAUNCH_TIMEOUT
  value: "10"

Specify proxy information, or remove if not required:

- name: qualys_https_proxy
  value: <PROXY_URL>

Under volumes specify the proxy cert path, or remove if not required:

- name: proxy-cert-path
  hostPath:
    path: /root/cert/proxy-certificate.crt
    type: File

Activation ID and Customer ID are required. Use the Activation ID and Customer ID from your subscription. To get the Activation ID and Customer ID, login to the Container Security UI, go to Configurations > Sensors, click Download, and then click any sensor type. The installation command on the Installation Instructions screen contains your Activation ID and Customer ID. Activation ID is like a password, do not share it.

If you are using an https proxy, ensure that all Kubernetes nodes have a valid certificate file for the sensor to communicate with the Container Management Server.

If you are not using a proxy and you have removed the above mentioned parts, you can remove the following part from volumeMounts as well:

- mountPath: /etc/qualys/qpa/cert/custom-ca.crt
  name: proxy-cert-path

**How to deploy the Container Sensor DaemonSet**

Once you have modified the cssensor-containerd-ds.yml file, run the following command on Kubernetes master to create a DaemonSet:

```
kubectl create -f cssensor-containerd-ds.yml
```

**How to remove the Container Sensor DaemonSet**

To uninstall Qualys Container Sensor, run the following command on Kubernetes master:

```
kubectl delete -f cssensor-containerd-ds.yml
```
Note: The persistent storage will need to be removed manually on each worker node.

Note: When the sensor operation is interrupted you may end up with some dangling PODs in kube-system namespace starting with “qcs”. Those will need to be manually removed on the master node.

Deploy in Kubernetes with Rancher - Docker Runtime

This section assumes you have the sensor image: Obtain the Container Sensor Image

Modify the cssensor-ds.yml file

Modify the cssensor-ds.yml file (extracted from QualysContainerSensor.tar.xz) to provide values for the following parameters. In order for the yml file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Note that you can download the yml file directly from https://github.com/Qualys/cs_sensor

Ensure all Kubernetes nodes have the latest Qualys sensor image from the URL provided.

containers:
  - name: qualys-container-sensor
    image: mycloudregistry.com/container-sensor:qualys-sensor-xxx
    args: ["--k8s-mode"]

If you want to deploy the sensor for CI/CD environment provide the args value as:

    args: ["--k8s-mode","--cicd-deployed-sensor"]

If you want to deploy a Registry Sensor provide the args value as:

    args: ["--k8s-mode","--registry-sensor"]

If you want print logs on the console, provide "--enable-console-logs" as an additional value in args.

To restrict the cpu usage to a certain value, change the following: (Optional)

Under resources specify the following:

resources:
  limits:
    cpu: "0.2" # Default CPU usage limit(20% of one core on the host).

For example, for limiting the cpu usage to 5%, set resources:limits:cpu: "0.05". This limits the cpu usage to 5% of one core on the host. If there are multiple processors on a node, setting the resources:limits:cpu value applies the CPU limit to one core only.

For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, set resources:limits:cpu value to 0.
Under `env` specify the following:

### Activation ID (Required)
- name: ACTIVATIONID  
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

### Customer ID (Required)
- name: CUSTOMERID  
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX

Specify proxy information, or remove if not required:
- name: qualys_https_proxy  
  value: proxy.localnet.com:3128

Under volumes specify the proxy cert path, or remove if not required:
- name: proxy-cert-path  
  hostPath:  
    path: /root/cert/proxy-certificate.crt  
    type: File

Activation ID and Customer ID are required. Use the Activation ID and Customer ID from your subscription.

If you are using a proxy, ensure that all Kubernetes nodes have a valid certificate file for the sensor to communicate with the Container Management Server.

If you are not using a proxy and you have removed the above-mentioned parts, you can remove the following part from `volumeMounts` as well:
- mountPath: /etc/qualys/qpa/cert/custom-ca.crt  
  name: proxy-cert-path

Once you have modified the `cssensor-ds.yml` file, save it.
Create Qualys sensor DaemonSet in Rancher UI

Log in to the Rancher UI to create a Qualys sensor DaemonSet. Use the credentials that were set during the creation setup.

From the top menu select the Cluster and Project under which the DaemonSet for the Qualys sensor is to be deployed in Rancher.

You will be navigated to the Resources tab. Click the "Import YAML" button.
Click the "Read from a file" button, then browse for and select the `cssensor-ds.yml` file that you’ve modified.

Click the Import button.

On the Workloads page under kube-system namespace ensure that the qualys-container-sensor DaemonSet is deployed and active.
If you need to uninstall the Qualys Container Sensor, then under kube-system namespace, select the check box next to qualys-container-sensor and click Delete.

**Launch sensor without persistent storage**

You can run the sensor without using persistent storage on host. In this case data is not stored on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.

To launch sensor without persistent storage, modify the `cssensor-ds.yml` file and provide "--sensor-without-persistent-storage" as an additional value in `args`.

```
args: ["--k8s-mode","--sensor-without-persistent-storage"]
```

It is recommended to use the "--enable-console-logs" option along with "--sensor-without-persistent-storage" to preserve the logs.

Under `volumeMounts` remove/comment the persistent-volume section.

```
volumeMounts:
- mountPath: /usr/local/qualys/qpa/data
  name: persistent-volume
```

Under `volumes` remove/comment the persistent-volume section.

```
volumes:
- name: persistent-volume
  hostPath:
    path: /usr/local/qualys/sensor/data
    type: DirectoryOrCreate
```
Deploying sensor in Kubernetes

Update the sensor deployed in Kubernetes

You can update the Container Sensor DaemonSet to the latest version in Kubernetes. This information is applicable for Amazon Elastic Container Service for Kubernetes (Amazon EKS), Google Kubernetes Engine (GKE), and Azure Kubernetes Service (AKS).

Ensure that the Container Sensor has read and write access to the persistent storage and the docker daemon socket.

Perform the following steps on Kubernetes master for updating the Container Sensor.

Note: Ensure the Container Sensor DaemonSet is running in the Kubernetes environment.

Download the `QualysContainerSensor.tar.xz` file from Qualys Cloud Portal on Kubernetes master.

Untar the sensor package:

```
    sudo tar -xvf QualysContainerSensor.tar.xz
```

Copy the Sensor version from the `version-info` file (extracted from QualysContainerSensor.tar.xz)

Modify the `cssensor-ds.yml` file

Modify the `cssensor-ds.yml` file (extracted from QualysContainerSensor.tar.xz) to provide values for the following parameters. In order for the yml file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Note that you can download the yml file directly from https://github.com/Qualys/cs_sensor

Ensure all Kubernetes nodes have the latest Qualys sensor image from the URL provided.

```yaml
containers:
  - name: qualys-container-sensor
    image: mycloudregistry.com/qualys/sensor:1.2.3-63
    args: ['--k8s-mode']
```

The image value must be in the format:

```
registryurl/qualys/sensor:<version-info>
```

If you want to deploy the sensor for CI/CD environment provide the `args` value as:

```
args: ['--k8s-mode','--cicd-deployed-sensor']
```

If you want to deploy a Registry Sensor provide the `args` value as:

```
args: ['--k8s-mode','--registry-sensor']
```

If you want print logs on the console, provide "--enable-console-logs" as an additional value in `args`. 
To restrict the cpu usage to a certain value, change the following: (Optional)

Under **resources** specify the following:

```yaml
resources:
  limits:
    cpu: "0.2" # Default CPU usage limit (20% of one core on the host).
```

For example, for limiting the cpu usage to 5%, set `resources:limits:cpu: "0.05"`. This limits the cpu usage to 5% of one core on the host.

If there are multiple processors on a node, setting the `resources:limits:cpu` value applies the CPU limit to one core only.

For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, set `resources:limits:cpu` value to 0.

Under **env** specify the following:

**Activation ID** (Required: Use the same Activation ID provided in the existing Container Sensor DaemonSet that you are upgrading)

```yaml
- name: ACTIVATIONID
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
```

**Customer ID** (Required: Use the same Customer ID provided in the existing Container Sensor DaemonSet that you are upgrading)

```yaml
- name: CUSTOMERID
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
```

Specify proxy information, or remove if not required:

```yaml
- name: qualys_https_proxy
  value: proxy.localnet.com:3128
```

Under **volumes** specify the proxy cert path, or remove if not required:

```yaml
- name: proxy-cert-path
  hostPath:
    path: /root/cert/proxy-certificate.crt
    type: File
```

Activation ID and Customer ID are required. Use the Activation ID and Customer ID from your subscription.

If you are using a proxy, ensure that all Kubernetes nodes have a valid certificate file for the sensor to communicate with the Container Management Server.
If you are not using a proxy and you have removed the above mentioned parts, you can remove the following part from `volumeMounts` as well:

```yaml
- mountPath: /etc/qualys/qpa/cert/custom-ca.crt
  name: proxy-cert-path
```

Once you have modified `cssensor-ds.yml`, save the file, and then perform docker login to the registry on Kubernetes master before running the update script (`k8s-rolling-update.sh`).

For example,

```
docker login mycloudregistry.com
```

The registry should be accessible from all Kubernetes nodes and the Kubernetes master from where the update is being performed.

To update the Container Sensor DaemonSet to the latest version, run the following command on Kubernetes master:

```
./k8s-rolling-update.sh Registry.Url=mycloudregistry.com
```

Note: `k8s-rolling-update.sh` will do docker load, docker tag and docker push to the registry.
Deploying sensor in Docker Swarm

Integrate the Container Sensor into the DaemonSet like other application containers and set the replication factor to 1 to ensure there is always a sensor deployed on the Docker Host.

Perform the following steps for creating a DaemonSet for the Qualys sensor to be deployed in Docker Swarm.

Download the `QualysContainerSensor.tar.xz` file from Qualys Cloud Portal on a Linux computer.

Untar the sensor package:

```
sudo tar -xvf QualysContainerSensor.tar.xz
```

Use the following commands to push the qualys sensor image to a repository common to all nodes in the Docker Swarm cluster:

```
sudo docker load -i qualys-sensor.tar
sudo docker tag <IMAGE NAME/ID> <URL to push image to the repository>
sudo docker push <URL to push image to the repository>
```

For example:

```
sudo docker load -i qualys-sensor.tar
sudo docker tag c3fa63a818df myregistry.com/qualys_sensor:xxx
sudo docker push myregistry.com/qualys_sensor:xxx
```

**Note:** Do not use the examples as is. Replace the registry/image path with your own.

**Modify the cssensor-swarm-ds.yml file**

Modify the `cssensor-swarm-ds.yml` file (extracted from QualysContainerSensor.tar.xz) to provide values for the following parameters. In order for the yml file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Note that you can download the yml file directly from https://github.com/Qualys/cs_sensor

Ensure that all master and worker nodes have the latest Qualys sensor image from the URL provided.

```
qualys-container-sensor:
  image: myregistry.com/qualys_sensor:xxx
  deploy:
    mode: global  # Deploy 1 container on each node == DaemonSet
    command: ['--swrm-mode']
```

If you want to deploy the sensor for CI/CD environment provide the `command` value as:

```
command: ['--swrm-mode', '--cicd-deployed-sensor']
```
If you want to deploy a Registry Sensor provide the `command` value as:

```bash
command: ["--swrm-mode","--registry-sensor"]
```

If you want print logs on the console, provide "--enable-console-logs" as an additional value in `command`.

To restrict the cpu usage to a certain value, change the following: (Optional)

Under `deploy` specify the following:

```yaml
mode: global  # Deploy 1 container on each node == DaemonSet
resources:
  limits:
    cpus: '0.20' # Default CPU usage limit(20% of one core on the host.
```

For example, for limiting the cpu usage to 5%, set `deploy:resources:limits:cpus: "0.05"`. This limits the cpu usage to 5% of one core on the host.

If there are multiple processors on a node, setting the `deploy:resources:limits:cpus` value applies the CPU limit to one core only.

For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, set `deploy:resources:limits:cpus` value to 0.

Under `environment` specify the following:

```yaml
environment:
  ACTIVATIONID: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
  CUSTOMERID: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
  qualys_https_proxy: proxy.qualys.com:3128
```

Activation ID and Customer ID are required. Use the Activation ID and Customer ID from your subscription. You can remove the proxy information if not required.

Under `volumes` ensure that you provide the following information:

```yaml
volumes:
- type: bind
  source: /var/run/
  target: /var/run/
- type: volume
  source: persistent-volume
  target: /usr/local/qualys/qpa/data/
- type: bind
  source: /etc/qualys  # Must exist!
  target: /usr/local/qualys/qpa/data/conf/agent-data
```
Keep source as "persistent-volume". This ensures that the source directory in volume mapping is set to docker swarm root directory (i.e. /data/docker/volumes).

/etc/qualys directory must exist on all masters and worker nodes for successful volume mapping.

volumes:
  persistent-volume:

Under configs ensure that you provide the following information:

configs:
  proxy-cert-path:
    file: /root/cert/proxy-certificate.crt

If you are using a proxy, ensure that all masters and worker nodes have a valid certificate file for the sensor to communicate with the Container Management Server.

If you are not using a proxy and you have removed qualys_https_proxy from environment, you can remove the following parts as well:

configs:
  - source: proxy-cert-path
    target: /etc/qualys/qpa/cert/custom-ca.crt

configs:
  proxy-cert-path:
    file: /root/cert/proxy-certificate.crt

Once you have modified the cssensor-swarm-ds.yml file, run the following command on docker swarm master/leader to create a stack:

docker stack deploy -c cssensor-swarm-ds.yml qualys-container-sensor

If you need to uninstall Qualys Container Sensor, run the following command on docker swarm master/leader:

docker stack rm qualys-container-sensor

**Launch sensor without persistent storage**

You can run the sensor without using persistent storage on host. In this case data is not stored on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.

To launch sensor without persistent storage, modify the cssensor-swarm-ds.yml file and provide "--sensor-without-persistent-storage" as an additional value in command.

command: ["--swrm-mode","--sensor-without-persistent-storage"]
It is recommended to use the "--enable-console-logs" option along with "--sensor-without-persistent-storage" to preserve the logs.

Under volumes (outside services) remove/comment the persistent-volume section.

```yaml
volumes:
  persistent-volume:
```

Under volumes (inside services) remove/comment the persistent-volume section.

```yaml
services:
  volumes:
    - type: volume
      source: persistent-volume
      target: /usr/local/qualys/qpa/data/
```
Deploying sensor in AWS ECS Cluster

Perform the following steps to deploy Qualys Container Sensor as a daemon service in Amazon ECS cluster.

**Prerequisites**: AWS ECS Cluster should be up and running.

Download the `QualysContainerSensor.tar.xz` file from Qualys Cloud Portal on a Linux computer.

Untar the sensor package:

```
sudo tar -xvf QualysContainerSensor.tar.xz
```

Use the following commands to push the qualys sensor image to a repository common to all nodes in the cluster:

```
sudo docker load -i qualys-sensor.tar
sudo docker tag <IMAGE NAME/ID> <URL to push image to the repository>
sudo docker push <URL to push image to the repository>
```

For example:

```
sudo docker load -i qualys-sensor.tar
sudo docker tag c3fa63a818df 20576712438.dr.ecr.us-east-1.amazonaws.com/container-sensor:qualys-sensor-xxx
sudo docker push 20576712438.dr.ecr.us-east-1.amazonaws.com/container-sensor:qualys-sensor-xxx
```

**Note**: Do not use the examples as is. Replace the registry/image path with your own.

**Modify the cssensor-aws-ecs.json file**

Modify the `cssensor-aws-ecs.json` file (extracted from QualysContainerSensor.tar.xz) to provide values for the following parameters. In order for the json file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Note that you can download the json file directly from https://github.com/Qualys/cs_sensor

```
"containerDefinitions": [ 
  {
    "name": "qualys-container-sensor",
    "image": "20576712438.dr.ecr.us-east-1.amazonaws.com/container-sensor:qualys-sensor-xxx",
    "cpu": 10,
    "memory": 512,
    "essential": true,
    "command": [ 
      "--ecs-mode"
    ],
  }
]
```

Specify appropriate values for **cpu** (no. of vcpu) and **memory** (size in MB).
If you want to deploy the sensor for CI/CD environment provide the `command` value as:

```
"command": [
    "--ecs-mode",
    "--cicd-deployed-sensor"
],
```

If you want to deploy a Registry Sensor provide the `command` value as:

```
"command": [
    "--ecs-mode",
    "--registry-sensor"
],
```

If you want print logs on the console, provide "--enable-console-logs" as an additional value in `command`.

Under `environment` specify the following:

```
"environment": [
    {
        "name": "ACTIVATIONID",
        "value": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
    },
    {
        "name": "CUSTOMERID",
        "value": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX"
    },
    {
        "name": "qualys_https_proxy",
        "value": "proxy.qualys.com:3128"
    }
]
```

Activation ID and Customer ID are required. Use the Activation ID and Customer ID from your subscription. Specify proxy information, or remove the section if not required. If you remove the proxy section, ensure that json indentation is correct.

If you are not using a proxy and you have removed `qualys_https_proxy` from `environment`, you can remove the following parts from `mountPoints` and `volumes`:

```
configs:
  - source: proxy-cert-path
    target: /etc/qualys/qpa/cert/custom-ca.crt

configs:
  proxy-cert-path:
    file: /root/cert/proxy-certificate.crt
```
Deploying sensor in AWS ECS Cluster

If proxy section is removed from environment, then remove proxy-cert-path sections under mountPoints and volumes as well:

"mountPoints": [
    {
        "sourceVolume": "proxy-cert-path",
        "containerPath": "/etc/qualys/qpa/cert/custom-ca.crt"
    },
]
"volumes": [
    {
        "name": "proxy-cert-path",
        "host": {
            "sourcePath": "/root/cert/proxy-certificate.crt"
        }
    }
]

Under volumes, provide information for persistent_volume. If you specify a custom location for persistent_volume, it would get created if not already available on the Docker Host.

Once you are done with the changes, save the cssensor-aws-ecs.json file.

Import the json file into Amazon ECS UI to complete the sensor deployment

On the Amazon ECS UI, under Task Definitions, click Create New Task Definition.

Select the launch type compatibility as EC2 (Fargate is not supported). Provide the Task Definition name, and then provide Task Role, Network Mode, and Task Execution Role if applicable.

Scroll to the bottom of the page and select Configure via JSON option. Remove any existing content and then copy-paste the entire contents of the cssensor-aws-ecs.json file.

Click Create to create the Task Definition. Once created, it should get listed under Task Definitions.

Now go to Clusters, and click the cluster name on which you want to deploy the sensor.

Under Services tab, click Create.

Select the launch type as EC2. Select the Task Definition you created above and its revision, and then select a cluster. Provide the Service name, Service type as “DAEMON”, and then configure Network, Load Balancing, and Auto Scaling if applicable.

Review the provided information, and then click Create to create the Service. Once created, it should get listed under Services.

Verify that the service status is Active. In the tasks tab, verify that tasks are running on all ECS containers.
Deploying sensor in AWS ECS Cluster

Stopping Qualys sensor on Amazon ECS Cluster

If you want to stop the Qualys container sensor from running on all containers, simply delete the service from the Services tab. This will kill the qualys-container-sensor service, but will not remove the sensor from the AWS ECS instances.

Launch sensor without persistent storage

You can run the sensor without using persistent storage on host. In this case data is not stored on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.

To launch sensor without persistent storage, modify the cssensor-aws-ecs.json file and provide "--sensor-without-persistent-storage" as an additional value in command.

```
"command": [
    "--ecs-mode",
    "--sensor-without-persistent-storage"
],
```

It is recommended to use the "--enable-console-logs" option along with "--sensor-without-persistent-storage" to preserve the logs.

Under **mountPoints** remove the persistent-volume section.

```
"mountPoints": [
    {
    "sourceVolume": "persistent_volume",
    "containerPath": "/usr/local/qualys/qpa/data"
    },

Under **volumes** remove the persistent-volume section.

```
"volumes": [
    {
    "name": "persistent_volume",
    "host": {
        "sourcePath": "/usr/local/qualys/sensor/data"
    }
    },
```
Deploying sensor in Mesosphere DC/OS

Perform the following steps to deploy Qualys Container Sensor as an application in DC/OS Marathon.

**Prerequisites**: A running DC/OS cluster with the DC/OS CLI installed.

Download the `QualysContainerSensor.tar.xz` file from Qualys Cloud Portal on DC/OS master.

Untar the sensor package:

```bash
sudo tar -xvf QualysContainerSensor.tar.xz
```

Use the following commands to push the qualys sensor image to a repository common to all nodes in the cluster:

```bash
sudo docker load -i qualys-sensor.tar
sudo docker tag <IMAGE NAME/ID> <URL to push image to the repository>
sudo docker push <URL to push image to the repository>
```

For example:

```bash
sudo docker load -i qualys-sensor.tar
sudo docker tag c3fa63a818df myregistry.com/qualys_sensor:xxx
sudo docker push myregistry.com/qualys_sensor:xxx
```

**Note**: Do not use the examples as is. Replace the registry/image path with your own.

**Modify the cssensor-dcos.json file**

Modify the `cssensor-dcos.json` file (extracted from QualysContainerSensor.tar.xz) to provide values for the following parameters. In order for the json file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Note that you can download the json file directly from [https://github.com/Qualys/cs_sensor](https://github.com/Qualys/cs_sensor)

```json
    "id": "/qualys-container-sensor",
    "args": ["--dcos-mode"],
    "cpus": 1,
    "mem": 128,
    "disk": 0,
    "instances": 1,
    "acceptedResourceRoles": ["*"],
```

Specify appropriate values for `cpus` (no. of vcpu), `mem` (size in MiB) and `disk` (size in MiB). If you want to deploy the sensor for CI/CD environment provide the `args` value as:

```json
    "args": ["--dcos-mode","--cicd-deployed-sensor"],
```
Deploying sensor in Mesosphere DC/OS

If you want to deploy a Registry Sensor provide the **args** value as:

```
"args": ["--dcos-mode", "--registry-sensor"],
```

If you want print logs on the console, provide "--enable-console-logs" as an additional value in **args**.

Ensure that **instances** value is the number of nodes in the cluster. This ensures that the container Sensor runs on each cluster node.

```
"container": {
    "type": "DOCKER",
    "docker": {
        "forcePullImage": true,
        "image": "myregistry.com/qualys_sensor:xxx",
        "parameters": [],
        "privileged": false
    }
},
```

Under **env** specify the following:

```
"env": {
    "ACTIVATIONID": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "CUSTOMERID": "XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX",
    "qualys_https_proxy": "proxy.qualys.com:3128"
},
```

Activation ID and Customer ID are required. Use the Activation ID and Customer ID from your subscription. If you are using a proxy, ensure that all nodes have a valid certificate file for the sensor to communicate with the Container Management Server.

Under **volumes** specify the following:

```
"volumes": [
    {
        "containerPath": "/usr/local/qualys/qpa/data",
        "hostPath": "/usr/local/qualys/sensor/data",
        "mode": "RW"
    },
    {
        "containerPath": "/var/run",
        "hostPath": "/var/run",
        "mode": "RW"
    },
    {
        "containerPath": "/usr/local/qualys/qpa/data/conf/agent-data",
        "hostPath": "/etc/qualys",
        "mode": "RW"
    }
],
```
Deploying sensor in Mesosphere DC/OS

},
{
  "containerPath": "/etc/qualys/qpa/cert/custom-ca.crt",
  "hostPath": "/root/cert/proxy-certificate.crt",
  "mode": "RO"
}
]

The directories specified for the hostPath are automatically created if not already available on the nodes. Ensure to provide a valid proxy-certificate.crt file path if you want to deploy the Sensor using a proxy.

If you are not using a proxy and you have removed qualys_https_proxy from env, you can remove the following from volumes as well, while ensuring that json indentation is correct:

```json
{
  "containerPath": "/etc/qualys/qpa/cert/custom-ca.crt",
  "hostPath": "/root/cert/proxy-certificate.crt",
  "mode": "RO"
}
```

Under portDefinitions specify the following:

```
"portDefinitions": [
  {
    "port": 10000,
    "protocol": "tcp"
  }
]
```

Specify a valid port number. Replace port number 10000, if already in use.

Once you have modified the cssensor-dcos.json file, run the following command on DC/OS master to add the qualys-container-sensor application to Marathon:

dcos marathon app add cssensor-dcos.json

Use this command to verify that the application is added successfully:

dcos marathon app list

If you need to uninstall Qualys Container Sensor from Marathon, run the following command on DC/OS master:

dcos marathon app remove --force /qualys-container-sensor
Launch sensor without persistent storage

You can run the sensor without using persistent storage on host. In this case data is not stored on host but stored at the /usr/local/qualys/qpa/data folder relative to the Sensor.

To launch sensor without persistent storage, modify the cssensor-dcos.json file and provide "--sensor-without-persistent-storage" as an additional value in args.

```
"args": ["--dcos-mode","--sensor-without-persistent-storage"],
```

It is recommended to use the "--enable-console-logs" option along with "--sensor-without-persistent-storage" to preserve the logs.

Under volumes remove the volume mount with
"containerPath": "/usr/local/qualys/qpa/data".

```
"volumes": [ 
    {
      "containerPath": "/usr/local/qualys/qpa/data",
      "hostPath": "/usr/local/qualys/sensor/data",
      "mode": "RW"
    }
],
```
Deploying sensor in OpenShift

Integrate the Container Sensor into the DaemonSet like other application containers to ensure that there is always a Sensor deployed on the Docker Host.

Perform the following steps for creating a DaemonSet for the Qualys sensor to be deployed in OpenShift.

Note: Ensure that the Container Sensor has read and write access to the persistent storage and the docker daemon socket.

Download the QualysContainerSensor.tar.xz file from Qualys Cloud Portal on OpenShift master.

Untar the sensor package:

```bash
sudo tar -xvf QualysContainerSensor.tar.xz
```

Use the following commands to push the Qualys sensor image to a repository common to all nodes in the OpenShift cluster:

```bash
sudo docker load -i qualys-sensor.tar
docker tag <IMAGE NAME/ID> <URL to push image to the repository>
docker push <URL to push image to the repository>
```

For example:

```bash
sudo docker load -i qualys-sensor.tar
docker tag c3fa63a818df mycloudregistry.com/container-sensor:qualys-sensor-xxx
docker push mycloudregistry.com/container-sensor:qualys-sensor-xxx
```

**Note:** Do not use the examples as is. Replace the registry/image path with your own.

Modify the cssensor-openshift-ds.yml file

Modify the cssensor-openshift-ds.yml file (extracted from QualysContainerSensor.tar.xz) to provide values for the following parameters. In order for the yml file to work properly, ensure that you do not remove/comment the respective sections mentioned below. Note that you can download the yml file directly from https://github.com/Qualys/cs_sensor

```yaml
serviceAccountName:
  qualysuser

containers:
  - name: qualys-container-sensor
    image: mycloudregistry.com/container-sensor:qualys-sensor-xxx
    securityContext:
      privileged: true
```

Ensure that the serviceAccountName is provided in the pod declaration.
Deploying sensor in OpenShift

```
args: ["--k8s-mode"]
```

If you want to deploy the sensor for CI/CD environment provide the `args` value as:
```
args: ["--k8s-mode","--cicd-deployed-sensor"]
```

If you want to deploy a Registry Sensor provide the `args` value as:
```
args: ["--k8s-mode","--registry-sensor"]
```

If you want print logs on the console, provide "--enable-console-logs" as an additional value in `args`.

To restrict the cpu usage to a certain value, change the following: (Optional)
Under `resources` specify the following:
```
resources:
  limits:
    cpu: "0.2" # Default CPU usage limit(20% of one core on the host).
```

For example, for limiting the cpu usage to 5%, set `resources:limits:cpu: "0.05"`. This limits the cpu usage to 5% of one core on the host.

If there are multiple processors on a node, setting the `resources:limits:cpu` value applies the CPU limit to one core only.

For example, if you have 4 CPUs on the system and you want to set CPU limit as 20% of overall CPU capacity, then the CPU limit should be set to 0.8 i.e., 80% of one core only which becomes 20% of total CPU capacity.

To disable any CPU usage limit, set `resources:limits:cpu` value to 0.

Under `env` specify the following:

**Activation ID (Required)**
```
- name: ACTIVATIONID
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
```

**Customer ID (Required)**
```
- name: CUSTOMERID
  value: XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX
```

Specify proxy information, or remove if not required:
```
- name: qualys_https_proxy
  value: proxy.localnet.com:3128
```

Under `volumes` specify the proxy cert path, or remove if not required:
```
- name: proxy-cert-path
```
Deploying sensor in OpenShift

```
hostPath:
  path: /root/cert/proxy-certificate.crt
```

Activation ID and Customer ID are required. Use the Activation ID and Customer ID from your subscription.

If you are using a proxy, ensure that all OpenShift nodes have a valid certificate file for the sensor to communicate with the Container Management Server.

If you are not using a proxy and you have removed the above mentioned parts, you can remove the following part from `volumeMounts` as well:

```
- mountPath: /etc/qualys/qpa/cert/custom-ca.crt
  name: proxy-cert-path
```

Once you have modified the `cssensor-openshift-ds.yml` file, run the following command on OpenShift master to create a DaemonSet:

```
oc create -f cssensor-openshift-ds.yml
```

If you need to uninstall Qualys Container Sensor, run the following command on OpenShift master:

```
oc delete ds qualys-container-sensor -n kube-system
```

**Launch sensor without persistent storage**

You can run the sensor without using persistent storage on host. In this case data is not stored on host but stored at the `/usr/local/qualys/qpa/data` folder relative to the Sensor.

To launch sensor without persistent storage, modify the `cssensor-openshift-ds.yml` file and provide `"--sensor-without-persistent-storage"` as an additional value in `args`.

```
args: ["--k8s-mode","--sensor-without-persistent-storage"]
```

It is recommended to use the `"--enable-console-logs"` option along with `"--sensor-without-persistent-storage"` to preserve the logs.

Under `volumeMounts` remove/comment the persistent-volume section.

```
volumeMounts:
- mountPath: /usr/local/qualys/qpa/data
  name: persistent-volume
```

Under `volumes` remove/comment the persistent-volume section.

```
volumes:
- name: persistent-volume
  hostPath:
    path: /usr/local/qualys/sensor/data
```
Administration

Sensor updates

When an update is available you’ll see “Update Available” next to the sensor name.

Container sensor update is otherwise automatic, however if you are currently using the beta version of the sensor you need to update to the latest sensor version manually. Automatic update kicks off once you are on a version higher than the beta.

To manually update the sensor from beta to the latest version, download the `QualysContainerSensor.tar.xz` file from Qualys Cloud Portal and then run the following commands directly from the screen on the docker host.

Untar the sensor package:

```
sudo tar -xvf QualysContainerSensor.tar.xz
```

Launch the new sensor:

```
sudo ./installsensor.sh ActivationId=5e7e422a-a1ca-403f-9274-506622dc5b28 CustomerId=a8cf7043-0245-6f1d-82f8-97f784652b93 Storage=/usr/local/qualys/sensor/data -s
```

Note: If you want to install the Sensor without persistent storage, exclude the “Storage” option, and include the “--sensor-without-persistent-storage” option in the installer script. It is recommended to use the “--enable-console-logs” option along with “--sensor-without-persistent-storage” to preserve the logs as data is not available on host but stored at the `/usr/local/qualys/qpa/data` folder relative to the Sensor.

Enter Y at the prompt asking you to upgrade ‘Qualys-Container-Sensor’ from version x.x.x to x.x.x.

The install script asks for proxy configuration. If you want to configure proxy, see Proxy Support.
How to uninstall the sensor

The QualysContainerSensor.tar.xz file (which you download for sensor installation from Qualys Cloud Platform) has the script `uninstallsensor.sh` for uninstalling the sensor.

To uninstall a sensor:

If the docker host is configured to communicate over docker.sock, use the following command:

```
./uninstallsensor.sh -s
```

If the docker host is configured to communicate over TCP socket then provide the address on which docker daemon is configured to listen:

```
./uninstallsensor.sh DockerHost=<IPv4 address or FQDN>:<Port#> -s
```

For example,

```
./uninstallsensor.sh DockerHost=10.115.27.54:3128 -s
```

Follow the on-screen prompts to uninstall the sensor. Qualys recommends not to clear the persistent storage.
Troubleshooting

Check sensor logs

The sensor log file is located at (by default):

/usr/local/qualys/sensor/data/logs/qpa.log

Diagnostic script

Qualys provides a script to collect diagnostic information about the sensor. You must run
the script on the host on which you want to collect the diagnostic information from.

The diagnostic script is present in the QualysContainerSensor.tar.xz that you downloaded
for installing the sensor.

The script is called Sensor_Diagnostic_Script.py. You must have Python installed on the
host in order to run the script. The script collects the following information from the host
and puts it in a tar file called SensorDiagnostic.tar. You can send that file to Qualys
Support for further assistance.

The SensorDiagnostic.tar includes 'ScanInfo.json', 'qpa.log' of qualys-container-sensor
from given persistent storage, docker logs of qualys-container-sensor, and all information
described below in the 'SensorDiagnostic.log' file. If 'ScanInfo.json' and Sensor logs are not
available on the Docker host then this script creates empty 'ScanInfo.json' and qpa.log
files, and appends "File not found" to them.

- Operating System Information (Type of OS i.e. Linux or Mac and other details)
- Proxy Configuration (Type of proxy set e.g. system, docker, cloud-agent proxy)
- CPU Architecture (Details about model, CPUs, cores, etc)
- RAM Usage (Memory allocation and utilization on host)
- Docker Version (Docker version installed on host)
- Socket Configuration (Docker socket configuration on host e.g. TCP/unix domain)
- Number of docker images (Count of all docker images and their details)
- Number of docker containers (Count of all docker containers and their details)
- CPU and Memory usage of running containers (First result of all resource usage
  statistics)
Sensor crashes during upgrade

Use installsensor.sh to reinstall Qualys container sensor keeping the “Storage” value as it was for earlier Sensor. This will ensure that the new sensor will not be marked as another Sensor and will simply upgrade the existing one.

For help on install command, see Deploying Container Sensor.

Note: At any given point in time, DO NOT delete the persistent storage. Else, the sensor deployed thereafter will be marked as a new sensor.

What if sensor restarts?

The Sensor is designed to handle restart scenarios and will continue functioning normally after restart. No customer intervention is needed until the sensor crashes.

Note: The Qualys container sensor will fail to restart if it has exited due to a fatal error before the docker host/service restarts.

Duplicate Kubernetes containers

While searching for containers you may see duplicates of containers orchestrated by Kubernetes. This is because Kubernetes spins up a monitoring container for every service container it brings up. Qualys container sensor sees them as two different containers and reports and scans both of the containers.

To see results without duplicate containers add the following string to queries used for searching Kubernetes containers.

not label.key:POD

For example, use this query to find running containers in Kubernetes:

state:“RUNNING” and not label.key:POD