Qualys Container Security

Release Notes for CRS Instrumentation

Version 2.3.0
August 6, 2021

Here's what's new in Container Runtime Security (CRS) Instrumentation 2.3.0!

New Proxy Environment Variables for Instrumenter service

Note: This feature was previously introduced for instrumentation using CLI mode. Now, starting in this release, it is also supported for instrumentation using the Instrumenter service.

You'll need to provide proxy details if the instrumented container is running behind a proxy to allow the CRS instrumenter to talk to the Qualys backend. The instrumented container can be launched with any of following proxy environment variables. If multiple proxy environment variables are used, then they will be honored by the CRS instrumenter in the order shown below.

```
-e LI_HTTPS_PROXY=<proxy>
-e LI_HTTP_PROXY=<proxy>
-e HTTPS_PROXY=<proxy>
-e HTTP_PROXY=<proxy>
```

The following example uses the LI_HTTPS_PROXY environment variable:

```
docker run -itd -e LI_MQURL=https://<cmsqagpublic VIP>/crs/v1.2 -e
LI_MQSKIPVERIFYTLS=true -e LI_HTTPS_PROXY=<proxy> <your registry/repo:tag>
```

Daemon Logging Now Available for Instrumenter service

Note: This feature was previously introduced for instrumentation using CLI mode. Now, starting in this release, it is also supported for instrumentation using the Instrumenter service.

After you've successfully instrumented an image, if you need to enable logging for troubleshooting the daemon you have two options. Option 1 covers spawning a container from the instrumented image with specific logging config environment variables. Option 2 is to edit the daemon.toml file in an already instantiated container and provide the logging config, then restarting the daemon process. The logging config will enable additional daemon log levels.

Log levels

The following log levels are supported. Please note that log levels have a certain hierarchy as listed below. When you choose a log level, all levels below it are also included. For example, a level of "trace" includes all other levels since it's at the top of the hierarchy. A level of "error" includes fatal and panic but not warn, info, debug or trace.

Log levels:
- trace
-- debug
--- info
---- warn
----- error
------ fatal
------- panic

How to enable daemon logging

You can enable daemon logging using either of the options described below.

Option 1: Use environment variables

Use LI_LOGLEVEL to specify the log level you want, and LI_DAEMONLOG to specify the log file and path where the daemon should write logs.

Run the following command:

```
docker run -itd -e LI_MQSKIPVERIFYTLS=true -e LI_LOGLEVEL="<loglevel>" -e
LI_DAEMONLOG="<path/filename>" <repo:tag>
```

Example:

```
docker run -itd -e LI_MQSKIPVERIFYTLS=true -e LI_LOGLEVEL="debug" -e
LI_DAEMONLOG="/tmp/daemonlogs_new" my-repo:my-tag
```

Option 2: Edit the toml file

Go to /etc/layint and edit the daemon.toml configuration file. Append the following config options to specify the log level and file path:

```
logLevel = "<log-level>"
daemonLog = "<path/filename>"
```

Example:

```
logLevel = "debug"
daemonLog = "/tmp/daemonlogs_new"
```

Note: You will need to restart the daemon process for this change to take effect.
Note: A valid directory path must be present inside the container.

Non-root Google Distroless Image Instrumentation using Instrumenter service

Note: This feature was previously introduced for instrumentation using CLI mode. Now, starting in this release, it is also supported for instrumentation using the Instrumenter service.

Some users ran into issues with non-root Google distroless image instrumentation. To fix these issues, please note the following requirements when instrumenting a non-root Google distroless image and launching a container from it.

- The minimum Docker version required to instrument a non-root Google distroless image is v17.09.0-ce (or above).

- In order for the CRS instrumentation to work properly, only the user ID defined in the Docker file should be used to launch the instrumented container. Optionally, you can use a root user other than the user defined in the Docker file.

Additional Images Supported for CRS Instrumentation

Instrumentation is supported for container images with certain libc/glibc versions. With this release, we've added support for the following versions:

Operating System: Alpine 3.12
libc/glibc version: musl-1.1.24-r10.apk

Operating System: Alpine 3.12.1
libc/glibc version: musl-1.1.24-r9

Docker image version: ubuntu:18.04
libc/glibc version: glibc_2.27-3ubuntu1.4