Working with Docker Content Trust

Overview

Notary is Docker's platform to provide trusted delivery of content by signing images that are published. A content publisher can then provide the corresponding signing keys that allow users to verify that content when it is consumed. Artifactory fully supports working with Docker Notary to ensure that Docker images uploaded to Artifactory can be signed, and then verified when downloaded for consumption. When the Docker client is configured to work with Docker Notary, after pushing an image to Artifactory, the client notifies the Notary to sign the image before assigning it a tag.

Artifactory supports hosting signed images without the need for any additional configuration.

Page Contents

Configuring Docker Notary and Docker Client

There is no configuration needed in Artifactory in order to work with trusted Docker images. However, in the setup instructions below, we do recommend testing your configuration by signing Artifactory and running it in a container.

To configure the Docker Notary and client to work with Artifactory, execute the following main steps:

- Configure your hosts file
- Configure the Notary server and run it as a container
- Configure the Docker client

Configuring Your Hosts File

If you are not working with a DNS, add the following entries to your */etc/hosts* file:

```
sudo sh -c 'echo "<Host IP> <Notary Server Name>" >> /etc/hosts'
sudo sh -c 'echo "<Host IP> <Artifactory Server Name>" >> /etc/hosts'
```

Configuring the Notary Server

Create a directory for your Notary server. In the code snippets below we will use ***notarybox***.

Create a dockerfile with the following content:

```
FROM debian:jessie

ADD https://get.docker.com/builds/Linux/x86_64/docker-1.9.1 /usr/bin/docker
RUN chmod +x /usr/bin/docker \
          && apt-get update \
         && apt-get install -y \
         tree \
         vim \
         git \
          ca-certificates \
         --no-install-recommends

WORKDIR /root
RUN git clone https://github.com/docker/notary.git && \
        cp /root/notary/fixtures/root-ca.crt /usr/local/share/ca-certificates/root-ca.crt && \
        update-ca-certificates

ENTRYPOINT ["bash"]
```

Use a private certificate

This configuration runs with a public certificate. Any Docker client running with the same public certificate may be able to access your Notary server.

For a secure setup, we recommend replacing it with your organization's private certificate by replacing the public `root-ca.crt` certificate file with your private certificate under */root/notary/fixtures* on your Notary server, and under */usr/local/share/ca-certificates* on the machine running your Docker client.

Build the test image:

```
docker build -t [image name] [path to dockerfile]
```

If you are running the build in your dockerfile directory, you can just use "." as the path to the dockerfile

Start the Notary server:

To start the Notary server, you first need to have Docker Compose installed.

Then execute the following steps:

```
cd notarybox
git clone -b trust-sandbox https://github.com/docker/notary.git
cd notary
docker-compose build
docker-compose up -d
```

Configuring the Docker Client

To connect the Notary server to the Docker client you need to enable the Docker content trust flag and add the Notary server URL as follows:

```
export DOCKER_CONTENT_TRUST=1
export DOCKER_CONTENT_TRUST_SERVER=https://notaryserver:4443
```

Test Your Setup

The example below demonstrates setting up the Notary server and Docker client, signing an image and the pushing it to Artifactory, with the following assumptions:

- Notary server and Artifactory run on localhost (127.0.0.1)
- Notary server is in directory *notarybox*
- Working without a DNS (so we need to configure the *hosts* file)

- Notary server name is `notaryserver`
- Artifactory server name is `artifactory-registry`
- Docker Compose is installed

Set up the IP mappings

```
sudo sh -c 'echo "127.0.0.1 notaryserver" >> /etc/hosts'
sudo sh -c 'echo "127.0.0.1 artifactory-registry" >> /etc/hosts'
```

Create the Dockerfile

```
FROM debian:jessie

ADD https://get.docker.com/builds/Linux/x86_64/docker-1.9.1 /usr/bin/docker
RUN chmod +x /usr/bin/docker \
          && apt-get update \
         && apt-get install -y \
         tree \
         vim \
         git \
          ca-certificates \
         --no-install-recommends

WORKDIR /root
RUN git clone -b trust-sandbox https://github.com/docker/notary.git && \
        cp /root/notary/fixtures/root-ca.crt /usr/local/share/ca-certificates/root-ca.crt && \
        update-ca-certificates

ENTRYPOINT ["bash"]
```

Note that this example uses the public `root-ca.crt` certificate.

Navigate to the Dockerfile location and build the test image

```
docker build -t notarybox .
```

Run Artifactory as a container

```
docker pull jfrog-docker-reg2.bintray.io/jfrog/artifactory-registry:<version>
docker run -d --name artifactory-registry -p 80:80 -p 8081:8081 -p 443:443 -p 5000-5002:5000-5002 jfrog-docker-
reg2.bintray.io/jfrog/artifactory-registry:<version>
```

Access your Artifactory instance (at *`http://localhost:8081/artifactory`*) and activate it with an Artifactory Pro license.

For more details on running Artifactory as a Docker container, please refer to Installing with Docker.

Start your container

In this step you will start the container with the `dockerfile` you created earlier, and link it to your Notary server and Artifactory.

```
docker run -it -v /var/run/docker.sock:/var/run/docker.sock --link notary_server_1:notaryserver --link
artifactory-registry:artifactory-registry notarybox
```

Pull an image for testing

```
docker pull docker/trusttest
```

After you have pulled the image, you need to `docker login` to `artifactory-registry:5002/v2`

Configure the Docker client

```
export DOCKER_CONTENT_TRUST=1
export DOCKER_CONTENT_TRUST_SERVER=https://notaryserver:4443
```

Tag the image you pulled for testing and push it to Artifactory

```
docker tag docker/trusttest artifactory-registry:5002/test/trusttest:latest
docker push artifactory-registry:5002/test/trusttest:latest
```

You will be asked to enter the root key passphrase. This will be needed every time you push a new image while the DOCKER_CONTENT_TRUST flag is set.

The root key is generated at: */root/.docker/trust/private/root_keys*

You will also be asked to enter a new passphrase for the image. This is generated at */root/.docker/trust/private/tuf_keys/[registry name] /[imagepath]*