Venafi DevOps Integrations

Venafi Docker Key & Certificate Management Container

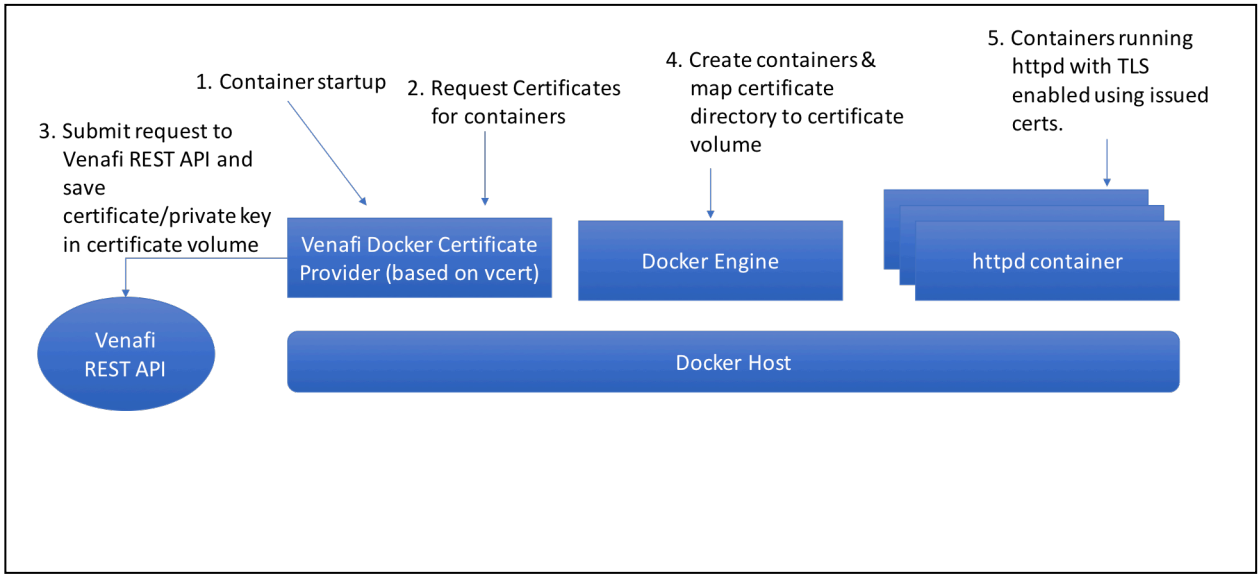
Version	Date	Description	Author
V1	March 20, 2017	Initial Version	Walter Goulet
V2	April 12, 2017	Updates per testing	Walter Goulet
V3	June 22, 2017	Trust Protection Platform additions	Ryan Treat

Introduction

The Venafi Docker Key & Certificate Management Container allows Docker users to seamlessly request and deploy certificates to Docker containers using a centralized, easy to use container that interfaces with Venafi REST APIs for certificate management operations..

Overview

The Venafi Docker Key & Certificate Management Container is a lightweight, dedicated purpose Docker container that generates key material and requests certificates from Venafi REST APIs. The certificates are securely exposed to other containers running on the same Docker host as the Venafi container. The



following diagram outlines the solution.

Installing the Venafi Docker Key & Certificate Management Container

The Venafi Docker Key & Certificate Management Container is published via Venafi's Docker Hub account. The image must be deployed on your target Docker host.

```
sudo docker login
sudo docker pull venaficloud/venafi-docker-key-management
```

Using the Venafi Docker Key & Certificate Management Container

---

NOTE: The Venafi Docker Key & Certificate Management container currently does not support encrypting private key material stored in the local docker volume. Ensure that the docker host is properly configured to restrict unauthorized access to the system.

Create an account with Venafi

The Venafi Docker Key and Certificate Management Container provides a function to allow users to register with the Venafi Cloud service to obtain test certificates. For production certificates, a Venafi Cloud administrator will need to configure your organization's Venafi Cloud account accordingly.

To register for an account with Venafi, run

```
docker run -ti --rm venaficloud/venafi-docker-key-management:production venafi register
```

and provide an email address when prompted.

Setup and running the service

Create a certificates volume using the following command:

```
docker volume create --name certificates
```

Start the service in the Venafi Docker Key & Certificate Management Container using:

(For Venafi Cloud)

```
docker run -ti --name vencloud --rm -v certificates:/certificates venaficloud/venafi-
docker-key-management:production venafi service -v --api-key XXX
```

(For Venafi Trust Protection Platform)

```
docker run -ti --name ventpp --rm -v certificates:/certificates venaficloud/venafi-docker-
key-management:production venafi service -v --tpp-user USERNAME --tpp-passwd PASSWORD --url
TPP_WEBSDK_URL
```

if you do not want to give your authentication information on the command line you can create a venafi.yaml file and mount that into the container using:

```
docker run -ti --name vencloud --rm -v certificates:/certificates -v
$(pwd)/venafi.yaml:/root/.venafi.yaml venaficloud/venafi-docker-key-management
venafi service -v
```

This command will create and run the container using the name 'vencloud' which is referenced in subsequent commands. Any name that is appropriate for your environment can be used instead.

The YAML file should have the following structure:

(For Venafi Cloud)

```
auth:
  api-key: XXXXX
```

(For Venafi Trust Protection Platform)

```
auth:
  tpp-user: USERNAME
  tpp-passwd:  PASSWORD
  url:  TPP_WEBSDK_URL
```

NOTE: For Venafi Trust Protection Platform, the CA certificate used to issue the TPP Operational Certificate must be trusted by the Venafi Docker container in order for the container to establish a connection to the TPP server. Once the container is running, execute the following Docker command to add the CA certificate to the container's trust store.

```
docker cp root-ca.pem vencloud:/usr/local/share/ca-certificates && docker
exec vencloud update-ca-certificates
```

Once the container is up and running, you can view the available commands supported by the container using the following:

```
docker exec vencloud venafi -h

A longer description that spans multiple lines and likely contains
examples and usage of using your application. For example:

Cobra is a CLI library for Go that empowers applications.
This application is a tool to generate the needed files
to quickly create a Cobra application.

Usage:
  venafi [command]

Available Commands:
  generate    Generate a new certificate, private key and certificate chain
  help        Help about any command
  register    Provide an existing API token or an email address and an API token will be
mailed to you
  service     Start the Venafi service
  setup       Helper script to configure your docker environment

Flags:
      --api-key string      Your api key for the Venafi cloud service
```

```
      --conf string         Location of the config file
  -h, --help                help for venafi
  -f, --log-format string   Logging format to use. Available options
(uicolor|text|json|none) (default "uicolor")
      --tpp-passwd string   Your password for the Venafi Trust Protection Platform
      --tpp-user string     Your username for the Venafi Trust Protection Platform
      --url string          URL for accessing the Venafi Trust Protection Platform or non-
production Venafi Cloud service
  -v, --verbose             Turn on verbose logging

Use "venafi [command] --help" for more information about a command.
```

Requesting a Certificate

Certificates are requested by invoking the 'generate' command on the Venafi container using docker exec. When this command is executed, the Venafi container will create a unique directory for the keypair and certificate. It will then submit a certificate request to the Venafi REST API. When the certificate is issued, the certificate will be stored in the directory.

This directory is output on the host when the enrollment process is complete. The directory is then specified as a volume when another container that uses the certificate is created.

NOTE: The Venafi Cloud service by default issues certificates intended for test and development purposes. In the default configuration, certificates that are issued will be valid only for the following test domains:

[subdomain].example.com
[subdomain].example.org
[subdomain].example.net
[subdomain].invalid
[subdomain].local
[subdomain].localhost
[subdomain].test

Where [subdomain] is the subdomain of the registered Venafi Cloud user's email address. For example, if the user registers with email address jdoe@mydemocorp.com, certificate requests submitted to the Venafi Cloud service must match one of the following patterns:

**.mydemocorp.example.com*
**.mydemocorp.example.org*
**.mydemocorp.example.net*
**.mydemocorp.invalid*
**.mydemocorp.local*
**.mydemocorp.localhost*
**.mydemocorp.test*

Contact Venafi at support@venafi.com if support for certificates for use in production environments is required.

Generate a Certificate & Key

The 'generate' command accepts several parameters as input:

Argument	Description
--chain (string)	Placement of root certificate within the chain root-first|root-last|ignore (default "root-last")
-c, --common-name (string)	Certificate Common Name field
--key-size (int)	Size of generated private key (default 2048)
--key-type (string)	Key type rsa|ecdsa (default "rsa")
--san-dns (string)	Specify one of more DNS SANs
--san-email (string)	Specify one of more Email SANs
--san-ip (string)	Specify one of more IP SANs
-z, --zone (string)	Zone which will process the certificate request. When interfacing with Trust Protection Platform this is the DN of the policy folder.
-v	Flag to indicate verbose output should be displayed.

An example run of the command is shown below:

```
sudo docker exec venprod venafi generate -c testcert2.vfidev.invalid --san-dns
testcert201.vfidev.com --key-size 2048 -v -z Default

DEBU[2017-03-20T19:34:11Z] Making RPC connection                          status=complete
DEBU[2017-03-20T19:34:23Z] Making Enroll Call
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb status=complete
DEBU[2017-03-20T19:34:33Z] Fetching certificate data
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb retry_in=1s status=pending
DEBU[2017-03-20T19:34:45Z] Fetching certificate data
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb retry_in=2s status=pending
DEBU[2017-03-20T19:34:57Z] Fetching certificate data
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb retry_in=4s status=pending
DEBU[2017-03-20T19:35:12Z] Fetching certificate data
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb retry_in=8s status=pending
DEBU[2017-03-20T19:35:31Z] Fetching certificate data
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb retry_in=16s status=pending
DEBU[2017-03-20T19:35:57Z] Fetching certificate data
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb retry_in=32s status=pending
DEBU[2017-03-20T19:36:40Z] Fetching certificate data
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb retry_in=1m4s status=pending
DEBU[2017-03-20T19:37:55Z] Fetching certificate data
pickup_id=3260ece0-0da4-11e7-9be2-891dab33d0eb retry_in=2m8s status=pending
INFO[2017-03-20T19:40:03Z] Fetching certificate data
completed_in=5m51.808547667s location="/certificates/1db895d6-b597-4266-a76e-0d9e223c24da"
status=complete
```