Fortanix

# Integration Guide

## USING FORTANIX CONFIDENTIAL COMPUTING MANAGER TO BUILD AND RUN HASHICORP ENTERPRISE VAULT

Fortanix Inc. | 800 West El Camino Real | Suite 180 | Mountain View, CA 94040 | United States of America | +1 (650) 943-2484 |
info@fortanix.com | www.fortanix.com

# 1.0 INTRODUCTION

This article describes how to build and deploy a **Hashicorp Vault** server within an enclave using **Fortanix Confidential Computing Manager (CCM)** and **Fortanix Enclave OS**.

**NOTE**: The following testing is done using Fortanix CCM CLI but the same can also be done using the Fortanix CCM SaaS UI https://ccm.fortanix.com/.

# 2.0 STEPS

## 2.1 AUTHENTICATE TO FORTANIX CCM

Before you can issue any requests, you first need to authenticate to Fortanix CCM using the following commands:

```
cpath=$(mktemp -p "/tmp" -t "fortanix_ccm_cookie.XXXXX")

curl -u <username>:<password> -c $cpath -X POST
https://ccm.fortanix.com/v1/sys/auth
```

where `<username>` and `<password>` need to be replaced with the email address and password of your Fortanix CCM account.

**NOTE**: Authentication session tokens are short-lived. If you ever get the response **{"message":"Forbidden","code":"FORBIDDEN"}** you can run the following command to refresh the session token.

```
curl -b $cpath -c $cpath -H "X-CSRF-Header:true" -X POST
https://ccm.fortanix.com/v1/sys/session/refresh
```

## 2.2 SELECT AN ACCOUNT USING AN API CALL

Once you have successfully authenticated to Fortanix CCM, you need to select an account. First, you can list all accounts available using the following command:

```
curl -b $cpath -c $cpath -H "X-CSRF-Header:true"
https://ccm.fortanix.com/v1/accounts
```

This command will return a JSON string of the following form:

```
{"name":"My account","acct_id":"26eaa328-5eb4-41c7-b09b-8a3e0a0f65c7",
...}, ...
```

To select an account, you need to copy the account ID of the account you are interested in (the string `26eaa328-5eb4-41c7-b09b-8a3e0a0f65c7` in the example above), let us call it `<account_id>`, and run:

```
curl -b $cpath -c $cpath -H "X-CSRF-Header:true" -X POST
https://ccm.fortanix.com/v1/sys/session/select_account/<account_id>
```

## 2.3 CREATE AN APPLICATION USING AN API CALL

Next, to create a new application, create a file called `app.json` whose contents are shown below. Make sure to replace the `output_image_name` with your private registry.

```
{
  "name": "Vault demo server",
  "description": "A hashicorp vault server demo",
  "input_image_name": "vault",
  "output_image_name": "<private-registry>/vault-sgx",
  "isvprodid": 1,
  "isvsvn": 1,
  "mem_size": 2048,
  "threads": 16,
  "advanced_settings": { "rw_dirs": ["/vault", "/home/vault"] }
}
```

**NOTE:**

Since the application needs to write to paths "`/vault`" and "`/home/vault`", these paths need to be marked as read/write (`rw_dirs`) because by default Enclave OS sets these paths to read-only.

Then, create an application using the following API call:

```
curl -b $cpath -c $cpath -H "X-CSRF-Header:true" -H "Content-Type:
application/json" -d @app.json -X POST https://ccm.fortanix.com/v1/apps
```

This will print information about the newly created application including its `<app_id>`:

```
{"name":"Vault demo server","app_id":"cc386097-dcf7-4813-880a-
ddacdafb48a2",...}
```

## 2.4 CREATING AN IMAGE USING AN API CALL

Once the application has been created, you can create an image similarly using the following steps. First, create a file called `build.json` as shown below. Replace `<app_id>` with the ID of your newly created application. The `<username>` and `<password>` are the credentials of the registry that you want the converted image to be stored at. This was specified above as `output_image_name`.

```
{
 "app_id":"<app_id>",
 "input_docker_version":"latest",
 "output_docker_version":"latest",
 "outputAuthConfig":{
 "username":"<username>",
 "password":"<password>"
 }
}
```

**NOTE:** See the Fortanix CCM Quickstart guide on how to set up registry credentials to avoid including credentials in this file.

Now you can create the image using the following command:

```
curl -b $cpath -c $cpath -H "X-CSRF-Header:true" -H "Content-Type:
application/json" -d @build.json -X POST
https://ccm.fortanix.com/v1/builds/convert-app
```

This returns the output that shows the <task_id> (f0d815b6-9520-4ce4-b4f4-6a82a718bb7e in this example), among other information:

```
{"build_name":"<private-registry>/vault-
sgx:latest","pending_task_id":"f0d815b6-9520-4ce4-b4f4-
6a82a718bb7e",...}
```

Finally, you can approve the image using its <task_id> and the following command:

```
curl -b $cpath -c $cpath -H "X-CSRF-Header:true" -H "Content-Type:
application/json" -d '{"status":"APPROVED"}' -X PATCH
https://ccm.fortanix.com/v1/tasks/<task_id>
```

## 3.0 RUN THE APPLICATION

Whether you chose to create your application using the UI or the API option, you should now have converted and whitelisted an application image and can run the application on an SGX compute node. Depending on the node agent attestation type, run the application using one of the following commands:

If the node attestation type is Enhanced Privacy ID (EPID), use the command:

```
docker run -it --device /dev/isgx:/dev/isgx --device
/dev/gsgx:/dev/gsgx -v
/var/run/aesmd/aesm.socket:/var/run/aesmd/aesm.socket -e
```

```
'VAULT_LOCAL_CONFIG={"listener": {"tcp": {"address": "127.0.0.1:8000",
"tls_disable": true}}, "disable_mlock": true}' -e
'VAULT_API_ADDR=http://127.0.0.1:8000' -e SKIP_SETCAP=1 --network=host
<private-registry>/vault-sgx
```

If the node attestation type is Data Center Attestation Primitives (DCAP), use the command:

```
docker run -it --device /dev/sgx/enclave:/dev/sgx/enclave -e
'VAULT_LOCAL_CONFIG={"listener": {"tcp": {"address": "127.0.0.1:8000",
"tls_disable": true}}, "disable_mlock": true}' -e
'VAULT_API_ADDR=http://127.0.0.1:8000' -e SKIP_SETCAP=1 --network=host
<private-registry>/vault-sgx
```

Where,

- `8000` is the port on which Hashicorp Vault listens to.
- `<private-registry>/vault-sgx:latest` is the converted app.
- `SKIP_SETCAP` environment variable: Skip the setcap call. Vault does this so it can use mlock for pages that contain secret information, so they do not get swapped to disk (where it is easier for an attacker to get them than memory). We skip these for multiple reasons: setcap probably will not work in enclave OS, mlock for this purpose is irrelevant in an enclave OS app (since all memory is protected, even when swapped out of EPC), and the additional forks are slow on SGX.
- Use `"disable_mlock": true` in the `VAULT_LOCAL_CONFIG` and do not use `--cap-add=IPC_LOCK`.

To verify that your vault server is running, use:

```
curl http://127.0.0.1:8000/v1/sys/init
```

Which should print:

```
"{"initialized":true}"
```

# 4.0 DOCUMENT INFORMATION

## 4.1 DOCUMENT LOCATION

The latest published version of this document is located at the URL:
https://support.fortanix.com/hc/en-us/articles/360059921872-Using-Fortanix-Confidential-Computing-Manager-to-Build-and-Run-Hashicorp-Enterprise-Vault

## 4.2 DOCUMENT UPDATES

This document will typically be updated on a periodic review and update cycle.

For any urgent document updates, please send an email to: support@fortanix.com