VeeaHub Toolkit R0.5 Guide

Contents

Preface

Information in this document is provided solely in connection with Veea Inc. and its affiliates (collectively “Veea”) products. Veeareserves the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

Use of Veea products and services is subject to the terms of use and/or separate agreements and warranties applicable to those products and services. Please visit www.veea.com/legal for these terms.

Evaluators (as defined in the Evaluation Agreement) are solely responsible for the choice, selection and use of the Veea products and services described herein, and Veea assumes no liability whatsoever relating to the choice, selection or use of the Veea products and services described herein.

Trademark Credits

Veea and all Veea related trademarks are owned by Veea Inc.

All other trademarks and tradenames are the property of their respective owners.

Copyright Information and Restrictions

Document Feedback

Veea welcomes your suggestions for improving our documentation. If you have comments, send your feedback to: support@veeahub.com

Document History

Issue	Issue Date	Author	Description
1	2019-10-30	RB	First published version for VeeaHub Toolkit release 0.5

Reference Documents

Document Ref.	Issue	Document Title
[RD/1]		Ubuntu Installation in VirtualBox (Veea Systems Ltd)
[RD/2]		VeeaHub ToolKit: Container Flow (Veea Systems Ltd)
[RD/3]		VeeaHub ToolKit: Container Methodology (Veea Systems Ltd)

Reference Software

Software Ref.	Release	Description/Title
[RS/1]	Test	Installation file: *https://c3templates.veeaplatform.net/0.5.0/setup-veeahub-dev-env_0.5.0.sh*

1. Introduction

1.1. Purpose of this Document

This document describes the installation and use of VeeaHub Toolkit and VeeaHub Client for creating applications for the VeeaHub.

1.2. Terminology

VH	VeeaHub
VHT	VeeaHub Toolkit
VHC	VeeaHub Client, included as part of VHT
Template	A zip archive containing a fully functional application that can be modified. A user downloads it and can usually run it immediately without modification.

1.3. VeeaHub Client: VHC

VeeaHub Client, or VHC, is a command-line utility designed to help developers easily create, build, and deploy applications to Veea's VeeaHub edge computing platforms.

VHC is currently supported only on Ubuntu Linux 18.04.3 and above.

1.4. Documentation

This is one of three documents describing VeeaHub Toolkit release 0.5.

[RD/2]describes the security policies of the VeeaHub platform, and the procedures for creating and deploying signed (secure) and unsigned containers.

[RD/3] describes guidelines for creating containers for the VeeaHub.

2. Installation and Setup

VHC requires Ubuntu Linux 18.04.3 and above. To install Ubuntu Linux on a virtual machine in VirtualBox on a Windows host, see [RD/1].

Important: If you are porting containers created under VHT release 0.4 or earlier, please follow the instructions in section 11. Do not install release 0.5 (or later) until instructed to do so.

To install VHC, download the installation script [RS/1]. You can use **wget**, **curl** or a similar method.

```
$ wget https://c3templates.veeaplatform.net/0.5.0/setup-veeahub-dev-env_0.5.0.sh
--2019-10-28 16:27:20--  https://c3templates.veeaplatform.net/0.5.0/setup-veeahub-dev-
env_0.5.0.sh
Resolving c3templates.veeaplatform.net (c3templates.veeaplatform.net)... 13.227.171.38,
13.227.171.92, 13.227.171.54, ...
Connecting to c3templates.veeaplatform.net (c3templates.veeaplatform.net)|13.227.171.38|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 3975 (3.9K) [text/x-shellscript]
Saving to: ‘setup-veeahub-dev-env_0.5.0.sh’

setup-veeahub-dev-e 100%[===================>]   3.88K  --.-KB/s    in 0s

2019-10-28 16:27:20 (771 MB/s) - ‘setup-veeahub-dev-env_0.5.0.sh’ saved [3975/3975]
```

Run the installation script.

```
$ ./setup-veeahub-dev-env_0.5.0.sh

VeeaHub Development Environment Setup Utility

Installing pre-requisites...
-- Found docker.io
-- containerd already installed
-- docker already installed
-- installing qemu
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  opensc-pkcs11
Use 'sudo apt autoremove' to remove it.
The following additional packages will be installed:
  qemu-block-extra qemu-system qemu-system-common qemu-user qemu-utils
Suggested packages:
  debootstrap
The following packages will be upgraded:
  qemu qemu-block-extra qemu-system qemu-system-common qemu-user qemu-utils
6 to upgrade, 0 to newly install, 0 to remove and 79 not to upgrade.
Need to get 9,160 kB of archives.
After this operation, 16.4 kB of additional disk space will be used.
Get:1 http://gb.archive.ubuntu.com/ubuntu bionic-updates/main amd64 qemu-utils amd64 1:2.11+dfsg-
1ubuntu7.19 [869 kB]
Get:2 http://gb.archive.ubuntu.com/ubuntu bionic-updates/main amd64 qemu-system-common amd64
1:2.11+dfsg-1ubuntu7.19 [672 kB]
Get:3 http://gb.archive.ubuntu.com/ubuntu bionic-updates/main amd64 qemu-block-extra amd64
1:2.11+dfsg-1ubuntu7.19 [39.5 kB]
Get:4 http://gb.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 qemu-system amd64
1:2.11+dfsg-1ubuntu7.19 [12.3 kB]
```

```
Get:5 http://gb.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 qemu-user amd64
1:2.11+dfsg-1ubuntu7.19 [7,352 kB]
Get:6 http://gb.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 qemu amd64 1:2.11+dfsg-
1ubuntu7.19 [215 kB]
Fetched 9,160 kB in 2s (4,181 kB/s)
(Reading database ... 203692 files and directories currently installed.)
Preparing to unpack .../0-qemu-utils_1%3a2.11+dfsg-1ubuntu7.19_amd64.deb ...
Unpacking qemu-utils (1:2.11+dfsg-1ubuntu7.19) over (1:2.11+dfsg-1ubuntu7.17) ...
Preparing to unpack .../1-qemu-system-common_1%3a2.11+dfsg-1ubuntu7.19_amd64.deb ...
Unpacking qemu-system-common (1:2.11+dfsg-1ubuntu7.19) over (1:2.11+dfsg-1ubuntu7.17) ...
Preparing to unpack .../2-qemu-block-extra_1%3a2.11+dfsg-1ubuntu7.19_amd64.deb ...
Unpacking qemu-block-extra:amd64 (1:2.11+dfsg-1ubuntu7.19) over (1:2.11+dfsg-1ubuntu7.17) ...
Preparing to unpack .../3-qemu-system_1%3a2.11+dfsg-1ubuntu7.19_amd64.deb ...
Unpacking qemu-system (1:2.11+dfsg-1ubuntu7.19) over (1:2.11+dfsg-1ubuntu7.17) ...
Preparing to unpack .../4-qemu-user_1%3a2.11+dfsg-1ubuntu7.19_amd64.deb ...
Unpacking qemu-user (1:2.11+dfsg-1ubuntu7.19) over (1:2.11+dfsg-1ubuntu7.17) ...
Preparing to unpack .../5-qemu_1%3a2.11+dfsg-1ubuntu7.19_amd64.deb ...
Unpacking qemu (1:2.11+dfsg-1ubuntu7.19) over (1:2.11+dfsg-1ubuntu7.17) ...
Setting up qemu-user (1:2.11+dfsg-1ubuntu7.19) ...
Setting up qemu-block-extra:amd64 (1:2.11+dfsg-1ubuntu7.19) ...
Setting up qemu-utils (1:2.11+dfsg-1ubuntu7.19) ...
Setting up qemu-system (1:2.11+dfsg-1ubuntu7.19) ...
Setting up qemu-system-common (1:2.11+dfsg-1ubuntu7.19) ...
Setting up qemu (1:2.11+dfsg-1ubuntu7.19) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
-- installing qemu-user-static
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  opensc-pkcs11
Use 'sudo apt autoremove' to remove it.
The following packages will be upgraded:
  qemu-user-static
1 to upgrade, 0 to newly install, 0 to remove and 78 not to upgrade.
Need to get 10.0 MB of archives.
After this operation, 3,072 B of additional disk space will be used.
Get:1 http://gb.archive.ubuntu.com/ubuntu bionic-updates/universe amd64 qemu-user-static amd64
1:2.11+dfsg-1ubuntu7.19 [10.0 MB]
Fetched 10.0 MB in 2s (4,485 kB/s)
(Reading database ... 203692 files and directories currently installed.)
Preparing to unpack .../qemu-user-static_1%3a2.11+dfsg-1ubuntu7.19_amd64.deb ...
Unpacking qemu-user-static (1:2.11+dfsg-1ubuntu7.19) over (1:2.11+dfsg-1ubuntu7.17) ...
Setting up qemu-user-static (1:2.11+dfsg-1ubuntu7.19) ...
Processing triggers for man-db (2.8.3-2ubuntu0.1) ...
-- qemu-user already installed
-- binfmt-support already installed
-- avahi-daemon already installed
-- avahi-discover already installed
-- avahi-utils already installed
-- python3-jsonschema already installed
-- usbutils already installed
-- gnutls-bin already installed
-- pcscd already installed
Getting opensc-pkcs11-veea package...
--2019-10-28 16:30:04--  https://c3templates.veeaplatform.net/tools/1/opensc-pkcs11-veea_0.19.0-
1_amd64.deb
Resolving c3templates.veeaplatform.net (c3templates.veeaplatform.net)... 143.204.192.25,
143.204.192.17, 143.204.192.11, ...
Connecting to c3templates.veeaplatform.net (c3templates.veeaplatform.net)|143.204.192.25|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 783520 (765K) [application/vnd.debian.binary-package]
Saving to: ‘/tmp/opensc-pkcs11-veea_0.19.0-1_amd64.deb’

opensc-pkcs11-veea_ 100%[===================>] 765.16K  4.61MB/s    in 0.2s
```

```
2019-10-28 16:30:09 (4.61 MB/s) - ‘/tmp/opensc-pkcs11-veea_0.19.0-1_amd64.deb’ saved
[783520/783520]

Installing opensc-pkcs11-veea package...
(Reading database ... 203692 files and directories currently installed.)
Preparing to unpack .../opensc-pkcs11-veea_0.19.0-1_amd64.deb ...
Unpacking opensc-pkcs11-veea:amd64 (0.19.0-1) over (0.19.0-1) ...
Setting up opensc-pkcs11-veea:amd64 (0.19.0-1) ...
Processing triggers for libc-bin (2.27-3ubuntu1) ...
Getting opensc-veea package...
--2019-10-28 16:30:10--  https://c3templates.veeaplatform.net/tools/1/opensc-veea_0.19.0-
1_amd64.deb
Resolving c3templates.veeaplatform.net (c3templates.veeaplatform.net)... 143.204.192.25,
143.204.192.31, 143.204.192.11, ...
Connecting to c3templates.veeaplatform.net (c3templates.veeaplatform.net)|143.204.192.25|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 201468 (197K) [application/vnd.debian.binary-package]
Saving to: ‘/tmp/opensc-veea_0.19.0-1_amd64.deb’

opensc-veea_0.19.0- 100%[===================>] 196.75K  --.-KB/s    in 0.03s

2019-10-28 16:30:10 (6.23 MB/s) - ‘/tmp/opensc-veea_0.19.0-1_amd64.deb’ saved [201468/201468]

Installing opensc-veea package...
(Reading database ... 203692 files and directories currently installed.)
Preparing to unpack .../opensc-veea_0.19.0-1_amd64.deb ...
Unpacking opensc-veea (0.19.0-1) over (0.19.0-1) ...
Setting up opensc-veea (0.19.0-1) ...
Getting libengine-pkcs11-openssl-veea package...
--2019-10-28 16:30:11--  https://c3templates.veeaplatform.net/tools/1/libengine-pkcs11-openssl-
veea_0.4.10-1_amd64.deb
Resolving c3templates.veeaplatform.net (c3templates.veeaplatform.net)... 143.204.192.25,
143.204.192.31, 143.204.192.11, ...
Connecting to c3templates.veeaplatform.net (c3templates.veeaplatform.net)|143.204.192.25|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 29932 (29K) [application/vnd.debian.binary-package]
Saving to: ‘/tmp/libengine-pkcs11-openssl-veea_0.4.10-1_amd64.deb’

libengine-pkcs11-op 100%[===================>]  29.23K  --.-KB/s    in 0.01s

2019-10-28 16:30:11 (2.34 MB/s) - ‘/tmp/libengine-pkcs11-openssl-veea_0.4.10-1_amd64.deb’ saved
[29932/29932]

Installing libengine-pkcs11-openssl-veea package...
(Reading database ... 203692 files and directories currently installed.)
Preparing to unpack .../libengine-pkcs11-openssl-veea_0.4.10-1_amd64.deb ...
Unpacking libengine-pkcs11-openssl-veea (0.4.10-1) over (0.4.10-1) ...
Setting up libengine-pkcs11-openssl-veea (0.4.10-1) ...
Getting VeeaHubToolkit package...
--2019-10-28 16:30:12--  https://c3templates.veeaplatform.net/0.5.0/VeeaHubToolkit_0.5.0-rel.deb
Resolving c3templates.veeaplatform.net (c3templates.veeaplatform.net)... 143.204.192.25,
143.204.192.31, 143.204.192.11, ...
Connecting to c3templates.veeaplatform.net (c3templates.veeaplatform.net)|143.204.192.25|:443...
connected.
HTTP request sent, awaiting response... 200 OK
Length: 7070576 (6.7M) [application/vnd.debian.binary-package]
Saving to: ‘/tmp/VeeaHubToolkit_0.5.0-rel.deb’

VeeaHubToolkit_0.5. 100%[===================>]   6.74M  3.87MB/s    in 1.7s

2019-10-28 16:30:14 (3.87 MB/s) - ‘/tmp/VeeaHubToolkit_0.5.0-rel.deb’ saved [7070576/7070576]

Installing VeeaHubToolkit package...
```

```
(Reading database ... 203692 files and directories currently installed.)
Preparing to unpack .../VeeaHubToolkit_0.5.0-rel.deb ...
Unpacking veeahubtoolkit (0.5.0-rel) over (0.4.0-1) ...
Removing user richard from group docker
Setting up veeahubtoolkit (0.5.0-rel) ...
```

If a previous version has already been installed, only the VHC binary will be replaced.

The installation includes the script for setting up the USB secure token, required for certifying applications for the VeeaHub. See [RD/2]

Because the user running the installation is added to the **docker** group during the procedure, following installation you must either log out and log in again or run the following for the group change to take effect:

```
$ su - $USER
```

Add the installation directory to your path, either in the current terminal or your **.bashrc**, **.profile**, or **.bash_profile** as appropriate (the last is recommended but requires opening a new terminal once complete):

```
$ export PATH=$PATH:/opt/veea/vht/bin
```

Validate that VHT installed correctly:

```
$ which vhc
/opt/veea/vht/bin/vhc
```

Validate that the Markdown files are included:

```
$ ls /opt/veea/vht/doc
README.md  secure.md  sideload.md
```

Dump the help for VHC:

```
$ vhc --help
Using vhc you can easily create new applications for the VeeaHub.

Usage:
  vhc [command]

Available Commands:
  build       Builds, packages, signs, and verifies applications for the VeeaHub.
  help        Help about any command
  hub         Used to load, start, and stop containers on and communicate with a VeeaHub.
  info        Sets and displays application information from a configuration file.
  list        Lists application templates, skeletons and assets.
  new         Sets up workspaces for VeeaHub applications.
  port        Adds/removes published ports to/from the container configuration.
  secure      Adds and removes entries to the secure portion of the configuration.
  update      Updates the index and assets for template and skeleton applications.
  version     Prints out version information for the VHC utility.

Flags:
  -h, --help   help for vhc

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

3. Using VeeaHub Client

All VHC commands have help associated with them and take the form of **vhc --help**, **vhc <command> --help**, or **vhc <command> sub-command> --help** etc., depending on the level of command/sub-command nesting.

Note: All VHC command arguments have both a short and long version, for example, **vhc hub -s** and **vhc hub --show**. This document uses the long format exclusively, as it is more descriptive of the option being used.

3.1. Version

To display the current version of the VHC, run:

```
$ vhc version
0.4.0
```

3.2. Download the VHC Index

The VHC index is used to inform VHC about the latest templates, hubs, base images, and skeletons. The VHC index is stored in a user's home directory under *~/.vhc/index.json* You can download the latest version by running:

```
$ vhc update
Updating Index... Done.
Updating Assets... Done.
```

3.3. List Templates

To list templates and skeleton projects, use the following command:

```
$ vhc list
```

You can list only the templates using:

```
$ vhc list --templates
+----------------------+-------------------------------------------------+
|         NAME         |                   DESCRIPTION                   |
+----------------------+-------------------------------------------------+
| vh_dbus              | Sample Python container that demonstrates a     |
|                      | simple DBUS interaction                         |
| vh_ext_serial        | Sample VHX09-10 container that demonstrates     |
|                      | the external serial port                        |
| vh_io_attach         | Sample container that demonstrates STDIO        |
|                      | websockets attaching                            |
| vh_ncsdk2_example    | Sample VHX09-10 container that demostrates      |
|                      | the Movidius Neural Compute Stick               |
| vh_nodejs_web        | Sample container that demonstrates a Node.js    |
|                      | web server                                      |
| vh_python            | Sample container that demonstrates running      |
|                      | Alpine with Python 3                            |
| vh_audio             | Sample container demonstrating audio playback   |
|                      | over HDMI                                       |
| vh_golang_web        | Sample golang web server created using          |
|                      | multistage build pattern to keep image small    |
|                      | i.e. ~6MB                                       |
| vh_hdmi              | Sample container demonstrating audio and        |
|                      | video playback through hdmi connection          |
| vh_persistent_storage | Sample container that demonstrates usage of a  |
```

```
|                       | persistent storage volume                      |
+-----------------------+------------------------------------------------+
```

Use the **update** and **list** commands to check for new templates regularly.

Note: Each template comes with documentation in the form of a Markdown file called *readme.md*

---

4. Creating a New Application from a Template

4.1. Downloading Templates

Template applications are pre-made applications that are ready to deploy on the VeeaHub. Using a Veea template application you can deploy an application to your VeeaHub in minutes. Templates can also be used as a jumping off point for building your own custom applications.

To download a template, you can use the VHC **new** command with the **--template** flag and the name of the template. We will use the *vh_nodejs_web* template as our example.

```
$ vhc new --template vh_nodejs_web
2019/09/24 16:45:33 Creating app from template vh_nodejs_web.
Using vh_nodejs_web
2019/09/24 16:45:33 Creating application directory: vh_nodejs_web
2019/09/24 16:45:38 vh_nodejs_web.zip downloaded
2019/09/24 16:45:38 Unzipping...
2019/09/24 16:45:38 Unzipped vh_nodejs_web.zip to vh_nodejs_web
```

This downloads the template archive and unzips it in the current directory. From here you can proceed to build and deploy your application.

Navigate to the new application *vh_nodejs_web* and see what's there:

```
$ cd  vh_nodejs_web
$ ls -R
.:
bin  config.yaml  Dockerfile  README.md  src

./bin:

./src:
package.json  server.js
```

Note the following files:

```
README.md file explaining the application and its specific details and run instructions.
config.yaml file containing build and run information for the application.
Dockerfile file containing container code for the application.
src directory containing the source code for the application.
bin directory used to store a tarball for each target's container.
```

All VHC applications must include *config.yaml* and a *Dockerfile*.

4.2. Application Information

To display information about your current application, from the base directory of the current application (that is, the directory containing *config.yaml*), run:

```
$ vhc info --show
+------------+----------------+
| PARAMETER  |     VALUE      |
+------------+----------------+
| Name       | vh nodejs web  |
| Language   | node           |
| Version    | 1              |
| Targets    | vhx09-10 vhc05 |
| Type       | app            |
```

```
| Deployment | single         |
+------------+----------------+
```

You may also specify a *config.yaml* from anywhere else by using the **--fname** option, for example, **vhc info --fname /path/to/config.yaml**.

Note the list of targets, as they will be used in other VHC commands. **vhx09-10** is used to represent both VHE09 and VHE10.

4.3. Generic Build Options

VHC supports generic user-specified Docker build options via the following method:

- All options should be placed in a file named *build-opts.txt* in the user's project directory.
- Each option should be on a single line, contain no trailing **** characters and no leading or trailing spaces.
- The format should follow one of the following two conventions:

  --<option> NAME=VALUE or

  --<option> NAME="$(shell <command-to-run>)".

  Note: **NAME** and **VALUE** may be optional depending on **<option>**.

For more information, see the Docker build options page:
https://docs.docker.com/engine/reference/commandline/build/

4.4. Ports

To publish a port at container run/start-time, run:

```
$ vhc port --add-publish 80:8080
Project configuration modified. Please rebuild image(s).
```

The format is **<host-port>:<container-port>**.

Verify that the port was set correctly:

```
$ vhc port --show
Published Ports:
+------+-----------+
| HOST | CONTAINER |
+------+-----------+
|   80 |      8080 |
| 9500 |      9500 |
+------+-----------+
```

Now, remove the port and verify that it is gone:

```
$ vhc port --remove-publish 80:8080
Project configuration modified. Please rebuild image(s).
$ vhc port --show
Published Ports:
+------+-----------+
| HOST | CONTAINER |
+------+-----------+
| 9500 |      9500 |
+------+-----------+
```

Notes:

- Any time you publish a port using the **port** command, you also need to make sure that the container port is present as an **EXPOSE** command in the source Dockerfile, for example, by editing the Dockerfile to add **- EXPOSE 8080**, otherwise an error will be returned when attempting to create the container.
- VHC does not currently support publishing a range of ports or specifying a protocol qualifier such as **tcp** or **udp**.

4.5. Secure Options

Please see the Container Flow document for information regarding creating secure containers. For now, the following two options need to be set:

1. Set the UUID. This should be done only **once** and not changed.
2. Enable the container to run unsigned.

To auto-generate a UUID for the container, run:

```
$ vhc secure --gen-uuid
Generated UUID: A1654E88-F683-43B5-8CE5-E09711CB6314
Project configuration modified. Please rebuild image(s).
```

To enable the container to run unsigned:

```
$ vhc secure --unauth-host
Project configuration modified. Please rebuild image(s).
```

Verify that both were set correctly:

```
$ vhc secure --show
Secure Options:
+-------------+---------------------------------------+
|    NAME     |                VALUE                  |
+-------------+---------------------------------------+
| unauth_host | true                                  |
| uuid        | A1654E88-F683-43B5-8CE5-E09711CB6314 |
+-------------+---------------------------------------+
```

Note: Please avoid changing the UUID unnecessarily as this can cause unintended consequences downstream!

4.6. Adding Devices

Before adding any devices, list what is available:

```
$ vhc secure device --list
Available Secure Devices:
+--------------------------+-----------+
|           NAME           | TARGET(S) |
+--------------------------+-----------+
| DEV:EXCLUSIVE:audio      |   vhc05   |
| DEV:EXCLUSIVE:bluetooth  |    all    |
| DEV:EXCLUSIVE:ext-serial0 | vhx09-10  |
| DEV:EXCLUSIVE:movidius   | vhx09-10  |
| DEV:EXCLUSIVE:video      |   vhc05   |
| DEV:EXCLUSIVE:webcam     |    all    |
```

```
| DEV:EXCLUSIVE:zigbee      |    all    |
| DEV:EXCLUSIVE:zwave       |    all    |
| DEV:SHARED:bluetooth      |    all    |
| DEV:SHARED:movidius       | vhx09-10  |
| DEV:SHARED:webcam         |    all    |
| DEV:SHARED:zigbee         |    all    |
| DEV:SHARED:zwave          |    all    |
+---------------------------+-----------+
```

To add the Zigbee device, for example, run:

```
$ vhc secure device --add DEV:EXCLUSIVE:zigbee
Project configuration modified. Please rebuild image(s).
```

Validate that the device was added correctly:

```
$ vhc secure device --show
Secure Devices:
+----------------------+
|         NAME         |
+----------------------+
| DEV:EXCLUSIVE:zigbee |
+----------------------+
```

To remove the Zigbee device, run:

```
$ vhc secure device --remove exclusive-zigbee
Project configuration modified. Please rebuild image(s).
```

Validate that the device was removed correctly:

```
$ vhc secure device --show
Secure Devices:
None.
```

The **Node JS Web** template does not require access to any devices.

4.7. Adding a Persistent Volume

To add a persistent volume to your container, run:

```
$ vhc secure volumes persist --add my-volume
Project configuration modified. Please rebuild image(s).
```

Validate that the volume was added correctly:

```
$ vhc secure volumes persist --show
Secure Persists:
+-----------+
|   NAME    |
+-----------+
| my-volume |
+-----------+
```

Notes:

- Persistent volumes are mapped into the container under */var/lib/veea/<volume-name>*.
- Information on accessing persistent volumes off-hub is presented below.

To remove a persistent volume from the configuration, run:

```
$ vhc secure volumes persist -remove my-volume
Project configuration modified. Please rebuild image(s).
```

Validate that the volume was removed correctly:

```
$ vhc secure volumes persist --show
Secure Persists:
None.
```

4.8. Building a Container

To build the container for the VHC05 target:

```
$ vhc build --target vhc05
Making vh_nodejs_web image for target vhc05...
Dockerfile.vhc05 does not exist.
Generating Dockerfile for Dockerfile.vhc05...
bin/vh_nodejs_web:vhc05.tar does not exist.
Building vh_nodejs_web image for vhc05...
Command Started
Sending build context to Docker daemon  19.97kB
Step 1/21 : FROM arm32v7/alpine:3.9
 ---> ea2ccc7da15e
Step 2/21 : RUN mkdir /app
 ---> Running in 8ae1736530d9
Removing intermediate container 8ae1736530d9
 ---> 036b98932643
Step 3/21 : COPY src/ /app/
 ---> 55cb69bc0a4d
Step 4/21 : WORKDIR /app
 ---> Running in 682f77a8ea41
Removing intermediate container 682f77a8ea41
 ---> a020aa52438b
Step 5/21 : ENV WELCOME_MESSAGE Welcome to the VeeaHub platform!
 ---> Running in f71652777435
Removing intermediate container f71652777435
 ---> 679c0779cebe
Step 6/21 : ENV PORT 9500
 ---> Running in 910a607af3a0
Removing intermediate container 910a607af3a0
 ---> f9e401e447b4
Step 7/21 : RUN apk update
 ---> Running in c9684cfa81a3
fetch http://dl-cdn.alpinelinux.org/alpine/v3.9/main/armv7/APKINDEX.tar.gz
fetch http://dl-cdn.alpinelinux.org/alpine/v3.9/community/armv7/APKINDEX.tar.gz
v3.9.4-178-g61f36dcfec [http://dl-cdn.alpinelinux.org/alpine/v3.9/main]
v3.9.4-178-g61f36dcfec [http://dl-cdn.alpinelinux.org/alpine/v3.9/community]
OK: 9561 distinct packages available
Removing intermediate container c9684cfa81a3
 ---> 3658630be4fe
Step 8/21 : RUN apk add --no-cache nodejs npm
 ---> Running in 74b75e107ffb
fetch http://dl-cdn.alpinelinux.org/alpine/v3.9/main/armv7/APKINDEX.tar.gz
fetch http://dl-cdn.alpinelinux.org/alpine/v3.9/community/armv7/APKINDEX.tar.gz
```

```
(1/8) Installing ca-certificates (20190108-r0)
(2/8) Installing c-ares (1.15.0-r0)
(3/8) Installing libgcc (8.3.0-r0)
(4/8) Installing http-parser (2.8.1-r0)
(5/8) Installing libstdc++ (8.3.0-r0)
(6/8) Installing libuv (1.23.2-r0)
(7/8) Installing nodejs (10.14.2-r0)
(8/8) Installing npm (10.14.2-r0)
Executing busybox-1.29.3-r10.trigger
Executing ca-certificates-20190108-r0.trigger
OK: 50 MiB in 22 packages
Removing intermediate container 74b75e107ffb
 ---> 93649f73cc28
Step 9/21 : RUN npm install
 ---> Running in 81fe6dece23e
qemu: Unsupported syscall: 384
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN vh-chat@1.0.0 No repository field.

added 95 packages from 58 contributors and audited 203 packages in 89.496s
found 0 vulnerabilities

Removing intermediate container 81fe6dece23e
 ---> e6a199cdf036
Step 10/21 : EXPOSE 9500
 ---> Running in 13ba1217e71f
Removing intermediate container 13ba1217e71f
 ---> ec497c2654bc
Step 11/21 : LABEL com.veea.vhc.target="vhc05"
 ---> Running in b87b675376d9
Removing intermediate container b87b675376d9
 ---> e40114603b5a
Step 12/21 : LABEL com.veea.vhc.version="0.5.0"
 ---> Running in e649c7ad2c27
Removing intermediate container e649c7ad2c27
 ---> 2ac6748ca2f9
Step 13/21 : LABEL com.veea.vhc.app.name="vh nodejs web"
 ---> Running in 1c5794acd40a
Removing intermediate container 1c5794acd40a
 ---> f8d8ae1e60cc
Step 14/21 : LABEL com.veea.vhc.app.version="1"
 ---> Running in d8f34a3f3b02
Removing intermediate container d8f34a3f3b02
 ---> a6e7e7eff7ca
Step 15/21 : LABEL com.veea.vhc.config.proj.version="1"
 ---> Running in 6d8113318ee4
Removing intermediate container 6d8113318ee4
 ---> 19ab554138d6
Step 16/21 : LABEL com.veea.vhc.config.user.version="1"
 ---> Running in 5e2d970c8f5a
Removing intermediate container 5e2d970c8f5a
 ---> e1e7c5ede1e7
Step 17/21 : ARG PARTNER_ID
 ---> Running in 99688fb87645
Removing intermediate container 99688fb87645
 ---> d866bea534ef
Step 18/21 : LABEL com.veea.image.persistent_uuid="$PARTNER_ID-778F-442C-AAC8-D440276CC664"
 ---> Running in cff93279ce4b
Removing intermediate container cff93279ce4b
 ---> 6bf8ec221925
Step 19/21 : LABEL com.veea.authorisation.allowOnUnauthenticatedHost="true"
 ---> Running in ce508b47d7a0
Removing intermediate container ce508b47d7a0
 ---> e0e893eb49f0
Step 20/21 : LABEL com.veea.authorisation.feature1="DEVELOPER"
 ---> Running in 0787a7e8994b
```

```
Removing intermediate container 0787a7e8994b
 ---> d39db3ac6fb6
Step 21/21 : CMD ["npm","start"]
 ---> Running in 873468cc15f7
Removing intermediate container 873468cc15f7
 ---> 28e30e47c4b3
Successfully built 28e30e47c4b3
Successfully tagged vh_nodejs_web:vhc05


Saving vh_nodejs_web image for vhc05.
```

This creates a target-specific Dockerfile in the application's base directory:

```
$ ls
bin  config.yaml  Dockerfile  Dockerfile.vhc05  README.md  src
```

which has the contents:

```
$ more Dockerfile.vhc05

#BEGIN vhc05
FROM arm32v7/alpine:3.9
#END

RUN mkdir /app
COPY src/ /app/
WORKDIR /app

ENV WELCOME_MESSAGE Welcome to the VeeaHub platform!
ENV PORT 9500

RUN apk update
RUN apk add --no-cache nodejs npm
RUN npm install
EXPOSE 9500

#BEGIN AUTO-GENERATED - DO NOT EDIT!!!
LABEL com.veea.vhc.target="vhc05"
LABEL com.veea.vhc.version="0.5.0"
LABEL com.veea.vhc.app.name="vh nodejs web"
LABEL com.veea.vhc.app.version="1"
LABEL com.veea.vhc.config.proj.version="1"
LABEL com.veea.vhc.config.user.version="1"
ARG PARTNER_ID
LABEL com.veea.image.persistent_uuid="$PARTNER_ID-778F-442C-AAC8-D440276CC664"
LABEL com.veea.authorisation.allowOnUnauthenticatedHost="true"
LABEL com.veea.authorisation.feature1="DEVELOPER"
#END AUTO-GENERATED - DO NOT EDIT!!!

CMD ["npm","start"]
```

It also creates a tarball of the container in the *bin* directory:

```
$ ls bin
vh_nodejs_web:vhc05.tar
```

5. Deploying Containers

5.1. Configuring the Hub

Note: Internally, VHC uses a few different methods to look up the hub's address, one of which is mDNS. If your hub cannot be located and you are using a virtual machine, please change the networking mode to Bridged and try again after locating the hub using **avahi-browse -all**.

Before we can deploy a container to a hub, we must configure the hub in VHC. If you don't know your hub's IPv4 address you can add a hub without it to start:

```
$ vhc hub --add-hub 1:C05BCB00C0A000001127
```

where the format is: **<id>:<serial-number>**. Substitute your hub's ID and serial number as necessary.

Verify that the hub was added correctly:

```
$ vhc hub --show
Hub Information:
+--------+---------------+----------------------+
| HUB ID |   PARAMETER   |        VALUE         |
+--------+---------------+----------------------+
| 1      |               |                      |
|        | IPv4 Address  |                      |
|        | Serial Number | C05BCB00C0A000001127 |
+--------+---------------+----------------------+
```

Otherwise, if you do know your hub's IP address you can add it as follows:

```
$ vhc hub --add-hub 1:C05BCB00C0A000001127:10.20.0.90
```

where the format is: **<id>:<serial-number>:<ipv4-addr>**. Substitute your hub's ID, serial number and IPv4 address as necessary.

Verify that the hub was added correctly:

```
$ vhc hub --show
Hub Information:
+--------+---------------+----------------------+
| HUB ID |   PARAMETER   |        VALUE         |
+--------+---------------+----------------------+
| 1      |               |                      |
|        | IPv4 Address  | 10.20.0.90           |
|        | Serial Number | C05BCB00C0A000001127 |
+--------+---------------+----------------------+
```

To update your hub's IPv4 address at any time, run:

```
$ vhc hub --update-hub 1:10.20.0.90
```

where the format is: **<id>:<ipv4-addr>**. Substitute your hub's ID and IPv4 address as necessary.

Verify that the hub was updated correctly:

```
$ vhc hub --show
Hub Information:
+--------+---------------+----------------------+
| HUB ID |   PARAMETER   |        VALUE         |
+--------+---------------+----------------------+
| 1      |               |                      |
|        | IPv4 Address  | 10.20.0.90           |
|        | Serial Number | C05BCB00C0A000001127 |
+--------+---------------+----------------------+
```

If only a single hub is configured then this hub will be used by default for all commands requiring **--hub-id**.

In the case where multiple hubs are configured, VHC supports the setting of an active hub that will be used in all subsequent **hub** commands:

```
$ vhc hub --active-hub 1
```

Verify that hub 1 is now active:

```
$ vhc hub --show
Hub Information:
+------------+---------------+----------------------+
|   HUB ID   |   PARAMETER   |        VALUE         |
+------------+---------------+----------------------+
| 1 (ACTIVE) |               |                      |
|            | IPv4 Address  | 10.20.0.90           |
|            | Serial Number | C05BCB00C0A000001127 |
+------------+---------------+----------------------+
```

Notes:

- At any time, you can override the active hub by specifying the **hub-id** parameter to any **hub** command.
- For all remaining **hub** commands in this document it is assumed that hub 1 is the only/active hub.

To remove a hub from the configuration, run:

```
$ vhc hub --remove-hub 1
```

where the format is: **<id>**. Substitute your hub's ID as necessary.

Verify that the hub was removed correctly:

```
$ vhc hub --show
No hub configurations available.
```

5.2. Pinging the Sideload Server

Ping the Sideload Server on the hub to make sure that VHC can talk to it:

```
$ vhc hub --ping
Located hub 10.20.0.90 via DNS.
Pinging C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/ping)...
Success
```

5.3. Checking the Partner ID

Check the partner ID:

```
$ vhc hub --get-partner-id
Retrieving Partner ID from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/partner/id)...
{
  "partner_id": "ffffffff"
}
```

At this point, it is important to note that any **hub** commands or sub-commands that return data from the hub will write the returned data in JSON format to the file *hub-results.json* in the directory in which VHC was run. In this case the file looks as follows:

```
$ more hub-results.json
{
  "partner_id": "ffffffff"
}
```

This allows for the scripting of VHC commands based on received data (for example, using **jq**).

5.4. Listing Hub Images

To get a list of what images are on the hub, run:

```
$ vhc hub image --get-images
Retrieving images from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/images)...
{
  "images": []
}
```

5.5. Creating an Image

To create an image, run:

```
$ vhc hub image --create bin/vh_nodejs_web\:vhc05.tar
Creating image push for file [bin/vh_nodejs_web:vhc05.tar] on C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/images/push)...
 54.27 MB / 54.27 MB [=============================================================]
100.00%
{"image_id": "c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74"}
```

Note: there is a delay from when the file transfer progress bar reaches 100% and the image ID is returned, due to processing of the image on the hub.

Check the list of images again:

```
$ vhc hub image --get-images
Retrieving images from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/images)...
{
  "images": [
    "c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74"
  ]
}
```

Note: VHC will prevent you from trying to load an image created for a different target, for example, loading a VHC05 image on a VHE09 target. In this case, you will see a message similar to the following:

```
ERROR: Image file target vhc05 does not match hub model prefix E09
```

5.6. Getting Image Information

To retrieve information about an image, run:

```
$ vhc hub image --get-info <image-id>
Retrieving image c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74 info from
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/images/c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74)
...
{
  "image": {
    "Containers": -1,
    "Created": 1569501010,
    "Id": "sha256:c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74",
    "Labels": {
      "com.veea.authorisation.allowOnUnauthenticatedHost": "true",
      "com.veea.authorisation.feature1": "DEVELOPER",
      "com.veea.image.persistent_uuid": "FFFFFFFF-F683-43B5-8CE5-E09711CB6314",
      "com.veea.vhc.app.name": "vh nodejs web",
      "com.veea.vhc.app.version": "1",
      "com.veea.vhc.config.proj.version": "1",
      "com.veea.vhc.config.user.version": "1",
      "com.veea.vhc.target": "vhc05",
      "com.veea.vhc.version": "0.5.0"
    },
    "ParentId": "",
    "RepoDigests": [
      "<none>@<none>"
    ],
    "RepoTags": [
      "<none>:<none>"
    ],
    "SharedSize": -1,
    "Size": 49467792,
    "VirtualSize": 49467792
  }
}
```

5.7. Listing Image Labels

To retrieve only the labels from an image, run:

```
$ vhc hub image --get-labels <image-id>
Retrieving image c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74 labels from
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/images/c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74/
labels)...
{
  "labels": {
    "com.veea.authorisation.allowOnUnauthenticatedHost": "true",
    "com.veea.authorisation.feature1": "DEVELOPER",
    "com.veea.image.persistent_uuid": "FFFFFFFF-F683-43B5-8CE5-E09711CB6314",
    "com.veea.vhc.app.name": "vh nodejs web",
    "com.veea.vhc.app.version": "1",
    "com.veea.vhc.config.proj.version": "1",
    "com.veea.vhc.config.user.version": "1",
```

```
    "com.veea.vhc.target": "vhc05",
    "com.veea.vhc.version": "0.5.0"
  }
}
```

5.8. Listing Hub Containers

To get a list of what containers are on the hub, run:

```
$ vhc hub container --get-containers
Retrieving containers from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/containers)...
{
  "containers": []
}
```

6. Running Containers

6.1. Creating a Container

Containers may be created with one of the following optional restart policy options using **--restart-policy**:

- **no**: Do not automatically restart the container when it exits.
- **unless-stopped**: Restart the container unless it is explicitly stopped.
- **always**: Always restart the container regardless of the exit status.

To create a container from the newly created image, run:

```
$ vhc hub image --create-container <image-id>
Creating container from image c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74 on
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/images/c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74/
create_container)...
 351 B / 351 B [============================================================================]
100.00%
{
  "container_id": "74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5"
}
```

Now, check the list of containers again:

```
$ vhc hub container --get-containers
Retrieving containers from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/containers)...
{
  "containers": [
    "74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5"
  ]
}
```

6.2. Getting Container Information

To retrieve information about a container, run:

```
$ vhc hub container --get-info <container-id>
Retrieving container 74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5 info from
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94
fc5)...
{
  "container": {
    "AppArmorProfile": "",
    "Args": [
      "start"
    ],
    "Config": {
      "ArgsEscaped": true,
      "AttachStderr": true,
      "AttachStdin": false,
      "AttachStdout": true,
      "Cmd": [
        "npm",
        "start"
      ],
      "Domainname": "",
      "Entrypoint": null,
```

```
      "Env": [
        "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
        "WELCOME_MESSAGE=Welcome to the VeeaHub platform!",
        "PORT=9500"
      ],
      "ExposedPorts": {
        "9500/tcp": {}
      },
      "Hostname": "74de3f949d4e",
      "Image": "c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74",
      "Labels": {
        "com.veea.authorisation.allowOnUnauthenticatedHost": "true",
        "com.veea.authorisation.feature1": "DEVELOPER",
        "com.veea.image.persistent_uuid": "FFFFFFFF-F683-43B5-8CE5-E09711CB6314",
        "com.veea.vhc.app.name": "vh nodejs web",
        "com.veea.vhc.app.version": "1",
        "com.veea.vhc.config.proj.version": "1",
        "com.veea.vhc.config.user.version": "1",
        "com.veea.vhc.target": "vhc05",
        "com.veea.vhc.version": "0.5.0"
      },
      "OnBuild": null,
      "OpenStdin": false,
      "StdinOnce": false,
      "Tty": false,
      "User": "u7777777wqnb3ldhf4clrds3dcq",
      "Volumes": null,
      "WorkingDir": "/app"
    },
    "Created": "2019-09-26T12:34:32.884055471Z",
    "Driver": "overlay2",
    "ExecIDs": null,
    "GraphDriver": {
      "Data": {
        "LowerDir":
"/vmrun/docker/overlay2/77598f15c5ddb8a1a51e6c156589e8eded3e77ec6e702b0301f4ffba1dce1a04-
init/diff:/vmrun/docker/overlay2/b7c66b487623640aa2023b93204450e86847810fd29e7e0fbf2c17fefe4b9a97
/diff:/vmrun/docker/overlay2/310dc275e5b044d997a45fe3e9fcd3abc7a3dde8f1015e50a9687e4bcf838dcf/dif
f:/vmrun/docker/overlay2/11914c852072feaa858570cfec323a8341453037755d4b06ef6bf2a5f791f0c7/diff:/v
mrun/docker/overlay2/0cd5b4e6246ce469eecd366244d7bcbf8a19cca4dafb4333cdfe9859e5c1bd10/diff:/vmrun
/docker/overlay2/9af7731f147407e896c0179d02112415898345878d3ce562309f2ff5d7811eff/diff:/vmrun/doc
ker/overlay2/8fefe3b72ef5e6e3f897dda9f3e74dca03411cd095c0140e88d29161876b2cb5/diff",
        "MergedDir":
"/vmrun/docker/overlay2/77598f15c5ddb8a1a51e6c156589e8eded3e77ec6e702b0301f4ffba1dce1a04/merged",
        "UpperDir":
"/vmrun/docker/overlay2/77598f15c5ddb8a1a51e6c156589e8eded3e77ec6e702b0301f4ffba1dce1a04/diff",
        "WorkDir":
"/vmrun/docker/overlay2/77598f15c5ddb8a1a51e6c156589e8eded3e77ec6e702b0301f4ffba1dce1a04/work"
      },
      "Name": "overlay2"
    },
    "HostConfig": {
      "AutoRemove": false,
      "Binds": [

"/var/run/hmd/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5/dev:/de
v",

"/var/run/hmd/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5/host:/h
ost",

"/var/run/hmd/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5/sys:/sy
s",

"/var/run/hmd/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5/persist
:/var/lib/veea",
```

```
        "/vmrun/bootstrap/configuration_data/containers/config/FFFFFFFF-F683-43B5-8CE5-
E09711CB6314:/usr/local/config/defaults",
        "/var/run/dbus/system_bus_socket:/var/run/dbus/system_bus_socket"
      ],
      "BlkioDeviceReadBps": null,
      "BlkioDeviceReadIOps": null,
      "BlkioDeviceWriteBps": null,
      "BlkioDeviceWriteIOps": null,
      "BlkioWeight": 0,
      "BlkioWeightDevice": null,
      "CapAdd": null,
      "CapDrop": null,
      "Cgroup": "",
      "CgroupParent": "",
      "ConsoleSize": [
        0,
        0
      ],
      "ContainerIDFile": "",
      "CpuCount": 0,
      "CpuPercent": 0,
      "CpuPeriod": 0,
      "CpuQuota": 0,
      "CpuRealtimePeriod": 0,
      "CpuRealtimeRuntime": 0,
      "CpuShares": 0,
      "CpusetCpus": "",
      "CpusetMems": "",
      "DeviceCgroupRules": null,
      "Devices": null,
      "DiskQuota": 0,
      "Dns": null,
      "DnsOptions": null,
      "DnsSearch": null,
      "ExtraHosts": null,
      "GroupAdd": null,
      "IOMaximumBandwidth": 0,
      "IOMaximumIOps": 0,
      "IpcMode": "shareable",
      "Isolation": "",
      "KernelMemory": 0,
      "Links": null,
      "LogConfig": {
        "Config": {
          "tag": "docker:{{.Name}}/{{.ID}}"
        },
        "Type": "syslog"
      },
      "MaskedPaths": [
        "/proc/asound",
        "/proc/acpi",
        "/proc/kcore",
        "/proc/keys",
        "/proc/latency_stats",
        "/proc/timer_list",
        "/proc/timer_stats",
        "/proc/sched_debug",
        "/proc/scsi",
        "/sys/firmware"
      ],
      "Memory": 0,
      "MemoryReservation": 0,
      "MemorySwap": 0,
      "MemorySwappiness": null,
      "NanoCpus": 0,
      "NetworkMode": "default",
```

```
      "OomKillDisable": false,
      "OomScoreAdj": 0,
      "PidMode": "",
      "PidsLimit": 0,
      "PortBindings": null,
      "Privileged": false,
      "PublishAllPorts": false,
      "ReadonlyPaths": [
        "/proc/bus",
        "/proc/fs",
        "/proc/irq",
        "/proc/sys",
        "/proc/sysrq-trigger"
      ],
      "ReadonlyRootfs": false,
      "RestartPolicy": {
        "MaximumRetryCount": 0,
        "Name": ""
      },
      "SecurityOpt": null,
      "ShmSize": 67108864,
      "UTSMode": "",
      "Ulimits": null,
      "UsernsMode": "",
      "VolumeDriver": "",
      "VolumesFrom": null
    },
    "HostnamePath": "",
    "HostsPath": "",
    "Id": "74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5",
    "Image": "sha256:c2c2f3571536ca991ba10f08b2bc053012cc319fd57699e41c8fcb83d7201e74",
    "LogPath": "",
    "MountLabel": "",
    "Mounts": [
      {
        "Destination": "/var/lib/veea",
        "Mode": "",
        "Propagation": "rprivate",
        "RW": true,
        "Source":
"/var/run/hmd/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5/persist
",
        "Type": "bind"
      },
      {
        "Destination": "/usr/local/config/defaults",
        "Mode": "",
        "Propagation": "rprivate",
        "RW": true,
        "Source": "/vmrun/bootstrap/configuration_data/containers/config/FFFFFFFF-F683-43B5-8CE5-
E09711CB6314",
        "Type": "bind"
      },
      {
        "Destination": "/var/run/dbus/system_bus_socket",
        "Mode": "",
        "Propagation": "rprivate",
        "RW": true,
        "Source": "/var/run/dbus/system_bus_socket",
        "Type": "bind"
      },
      {
        "Destination": "/dev",
        "Mode": "",
        "Propagation": "rprivate",
        "RW": true,
```

```
        "Source":
"/var/run/hmd/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5/dev",
        "Type": "bind"
      },
      {
        "Destination": "/host",
        "Mode": "",
        "Propagation": "rprivate",
        "RW": true,
        "Source":
"/var/run/hmd/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5/host",
        "Type": "bind"
      },
      {
        "Destination": "/sys",
        "Mode": "",
        "Propagation": "rprivate",
        "RW": true,
        "Source":
"/var/run/hmd/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5/sys",
        "Type": "bind"
      }
    ],
    "Name": "/priceless_ishizaka",
    "NetworkSettings": {
      "Bridge": "",
      "EndpointID": "",
      "Gateway": "",
      "GlobalIPv6Address": "",
      "GlobalIPv6PrefixLen": 0,
      "HairpinMode": false,
      "IPAddress": "",
      "IPPrefixLen": 0,
      "IPv6Gateway": "",
      "LinkLocalIPv6Address": "",
      "LinkLocalIPv6PrefixLen": 0,
      "MacAddress": "",
      "Networks": {
        "bridge": {
          "Aliases": null,
          "DriverOpts": null,
          "EndpointID": "",
          "Gateway": "",
          "GlobalIPv6Address": "",
          "GlobalIPv6PrefixLen": 0,
          "IPAMConfig": null,
          "IPAddress": "",
          "IPPrefixLen": 0,
          "IPv6Gateway": "",
          "Links": null,
          "MacAddress": "",
          "NetworkID": ""
        }
      },
      "Ports": {},
      "SandboxID": "",
      "SandboxKey": "",
      "SecondaryIPAddresses": null,
      "SecondaryIPv6Addresses": null
    },
    "Path": "npm",
    "Platform": "linux",
    "ProcessLabel": "",
    "ResolvConfPath": "",
    "RestartCount": 0,
    "State": {
```

```
      "Dead": false,
      "Error": "",
      "ExitCode": 0,
      "FinishedAt": "0001-01-01T00:00:00Z",
      "OOMKilled": false,
      "Paused": false,
      "Pid": 0,
      "Restarting": false,
      "Running": false,
      "StartedAt": "0001-01-01T00:00:00Z",
      "Status": "created"
    }
  }
}
```

6.3. Starting a Container

To start the container created above, run:

```
$ vhc hub container --start <container-id>
Starting container 74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94
fc5/start)...
Success
```

6.4. Testing the Application

At this point you should be able to use a web browser to view the page hosted by the "Hello, World" application. Substituting your hub's IP address as appropriate, in your browser navigate to:

```
http://10.20.0.90:9500
```

and you should see the following message displayed:

```
Welcome to the VeeaHub platform!
```

6.5. Attaching to STDIO

To attach to stdio on the container, run:

```
$ vhc hub container --attach <container-id> "/bin/sh -il"
Attaching to stdin/stdout/stderr on C05BCB00C0A000001127:9001...
Attaching to container 74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94
fc5/attach)...
Success
```

Specifying **"/bin/sh -il"** as the last argument command above drops VHC into a terminal-like state where the user can interact with the container, for example:

```
74de3f949d4e:/app$ /bin/sh: can't access tty; job control turned off
74de3f949d4e:/app$ ls
node_modules
package-lock.json
package.json
```

```
server.js
74de3f949d4e:/app$ pwd
/app
74de3f949d4e:/app$ exit
read tcp 10.211.55.3:53640->10.20.0.90:9001: use of closed network connection
read tcp 10.211.55.3:53638->10.20.0.90:9001: use of closed network connection
Detaching from container 74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94
fc5/detach)...
Success
```

To exit the "terminal" enter either **exit** or hit **Ctrl+C Ctrl+D**.

If you do not specify a command then the **attach** command will hook into the container's run-time stdio (that is, any run-time output to stdout or stderr). Exit this mode of attach in the same manner as the previous example.

Notes:

- The process of attaching can take up to around 20 to 25 seconds to complete when running a signed container.
- The “can't access tty” message above may be safely ignored.
- The “read tcp” messages above may be safely ignored.

6.6. Stopping a Container

To stop the container started above, run:

```
$ vhc hub container --stop <container-id>
74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5
Stopping container 74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94fc5 from
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/containers/74de3f949d4eb5251de623ed3e01fa35625b3d490564832b9aca627a2df94
fc5/stop)...
Success
```

7. Persistent Container and Hub Volumes

7.1. Volumes

To create a persistent volume in a container, use the **secure volumes persist** command as described in section 4.7. This command creates a volume that is accessible from the container under */var/lib/veea/<volume-name>* and persists across stopping and restarting the container.

If you need to populate data for the container before starting it, you must use the **hub volume --create** command in addition to **secure volumes persist** so that the volume is accessible prior to start.

To access hub volumes over a network using WebDAV, use the **hub volume --export** command.

These commands and more are detailed below.

7.2. Listing Hub Volumes

To list all volumes on the hub for all UUIDs, run:

```
$ vhc hub volume --get-all-volumes
Retrieving volumes from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/volumes)...
{
  "volumes": []
}
```

7.3. Creating a Volume on the Hub

To populate data in a container volume before starting the container, first create the persistent volume as described earlier using the **secure volumes persist** command and then run:

```
$ vhc hub volume --create my-volume
partnerId FFFFFFFF, uuid F683-43B5-8CE5-E09711CB6314
Creating volume FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume on C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/volumes/FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume)...
Success
```

Now, check the list of volumes again:

```
$ vhc hub volume --get-all-volumes
Retrieving volumes from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/volumes)...
{
  "volumes": [
    "FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume"
  ]
}
```

7.4. Listing Hub Volumes by UUID

You can also list all volumes on the hub for the current UUID:

```
$ vhc hub volume --get-uuid-volumes
partnerId FFFFFFFF, uuid F683-43B5-8CE5-E09711CB6314
Retrieving volumes from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/volumes/FFFFFFFF-F683-
43B5-8CE5-E09711CB6314)...
```

```
{
  "volumes": [
    "my-volume"
  ]
}
```

7.5. Exporting a Volume on the Hub

To export a volume for access via WebDAV, run:

```
$ vhc hub volume --export my-volume
partnerId FFFFFFFF, uuid F683-43B5-8CE5-E09711CB6314
Exporting volume FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume on C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/volumes/FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume/export)...
{
  "url": "https://C05BCB00C0A000001127.veeamesh.local:9002"
}
```

Next, install WebDAV:

```
$ sudo apt install -y davfs2
```

Now, mount the volume via WebDAV on a separate LINUX machine:

```
$ sudo mount -t davfs https://C05BCB00C0A000001127:9002 /mnt
Please enter the username to authenticate with server
https://C05BCB00C0A000001127:9002 or hit enter for none.
  Username:
Please enter the password to authenticate user  with server
https://C05BCB00C0A000001127:9002 or hit enter for none.
  Password:
/sbin/mount.davfs: the server certificate does not match the server name
/sbin/mount.davfs: the server certificate is not trusted
  issuer:      Veea, Iselin, New Jersey, US
  subject:     Veea, Iselin, New Jersey, US
  identity:    veea.com
  fingerprint: 8b:3b:d0:cb:a1:83:19:f8:8d:39:e5:e4:7a:6a:41:9c:14:f6:b8:8b
You only should accept this certificate, if you can
verify the fingerprint! The server might be faked
or there might be a man-in-the-middle-attack.
Accept certificate for this session? [y,N] y
/sbin/mount.davfs: warning: the server does not support locks
```

See what's there:

```
$ ls /mnt
lost+found
```

Create a file:

```
$ echo "What's all the Hubbub!" > file.txt
$ sudo cp file.txt /mnt
$ cat /mnt/file.txt
What's all the Hubbub!
```

See how big it is:

```
$ ls -l /mnt
total 1
-rw-r--r-- 1 root root 24 May 17 15:14 file.txt
drwx------ 2 root root  0 May 17 15:05 lost+found
```

Synchronize the file system:

```
$ sync
```

If you attach to the container, for example:

```
vhc hub container --attach <container-id> /bin/sh
```

you will be able to access the contents of the volume and file under */var/lib/veea/<volume-name>*. If you point a web browser at the given URL you should be able to see the contents.

Note: When modifying the contents of a persistent container volume via WebDAV or container attach, it is important to issue a **sync** command on the side performing the modification prior to accessing the files from the other side to ensure that both sides are properly synchronized.

7.6. Getting Information About a Volume

To get information about a volume, run:

```
$ vhc hub volume --get-info my-volume
partnerId FFFFFFFF, uuid F683-43B5-8CE5-E09711CB6314
Retrieving volume FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume info from
C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/volumes/FFFFFFFF-F683-43B5-8CE5-
E09711CB6314/my-volume)...
{
  "size": "8"
}
```

7.7. Unexporting a Volume on the Hub

Before unexporting a volume be sure to unmount the WebDAV on your separate LINUX machine from earlier:

```
$ sudo umount /mnt
/sbin/umount.davfs: waiting while mount.davfs (pid 29195) synchronizes the cache .. OK
$ ls /mnt
$
```

To turn off the WebDAV export the volume specified by the current UUID and volume name, run:

```
$ vhc hub volume --unexport my-volume
partnerId FFFFFFFF, uuid F683-43B5-8CE5-E09711CB6314
Unexporting volume FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume on C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/volumes/FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume/unexport)...
Success
```

7.8. Deleting a Volume on the Hub

To delete a volume of the hub, run:

```
$ vhc hub volume --delete my-volume
partnerId FFFFFFFF, uuid F683-43B5-8CE5-E09711CB6314
Deleting volume FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume from C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/volumes/FFFFFFFF-F683-43B5-8CE5-E09711CB6314/my-volume)...
```

Check the list of volumes again:

```
$ vhc hub volume --get-all-volumes
Retrieving volumes from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/volumes)...
{
  "volumes": []
}
```

8. Container Configuration

8.1. Overview

All containers with the same UUID access configuration files via the path */usr/local/config/defaults*. This location is Read-Only (RO) from the container's perspective but may be accessed Read/Write (RW) over the network using WebDAV in a similar manner to persistent container volumes described above. Note that the same considerations about synchronizing the filesystem apply here as well. Configuration options must be specified in JSON format. This mechanism is how container start-time options are handled.

Please see the Docker run options page (*https://docs.docker.com/engine/reference/run/*) and Docker Engine API page (*https://docs.docker.com/engine/api/v1.30/#*) for more information.

8.2. Exporting Configuration via WebDAV

To export the configuration directory for the current UUID so that it can be accessed via WebDAV, run:

```
$ vhc hub config --export
partnerId FFFFFFFF, uuid F683-43B5-8CE5-E09711CB6314
Exporting configuration FFFFFFFF-F683-43B5-8CE5-E09711CB6314 on C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/configuration/FFFFFFFF-F683-43B5-8CE5-E09711CB6314/export)...
{
  "url": "https://C05BCB00C0A000001127.veeamesh.local:9002"
}
```

The configuration export is used to give containers access to JSON configuration files, including the Control Center key and secret configuration when running untrusted.

8.3. Unexporting Configuration via WebDAV

To turn off the WebDAV export of the configuration directory for the specified UUID, run:

```
$ vhc hub config --unexport
partnerId FFFFFFFF, uuid F683-43B5-8CE5-E09711CB6314
Unexporting configuration FFFFFFFF-F683-43B5-8CE5-E09711CB6314 on C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/configuration/FFFFFFFF-F683-43B5-8CE5-E09711CB6314/unexport)...
Success
```

9. Deleting Containers and Images

9.1. Deleting a Container

Caution: This function does not offer an undo!

Before deleting a container be sure to stop it first (section 6.6). Containers and images cannot be deleted if running.

To delete a container, run:

```
$ vhc hub container --delete <container-id>
Deleting container e9fbf2efcf985e20de81b8eda1d1d0029e16ae94f97c4a05db0b8b92f3e9bcdb from
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/containers/e9fbf2efcf985e20de81b8eda1d1d0029e16ae94f97c4a05db0b8b92f3e9b
cdb)...
```

Make sure the container was deleted:

```
$ vhc hub container --get-containers
Retrieving containers from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/containers)...
{
  "containers": []
}
```

Using the flag **--delete-all** deletes all containers from the hub.

9.2. Deleting an Image

Caution: This function does not offer an undo!

To delete an image, run:

```
$ vhc hub image --delete <image-id>
Deleting image 9a315b5a44e18bb2c03a9216bb0f95680f9ae287a32869ac2fda0a1795d79b00 from
C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/images/9a315b5a44e18bb2c03a9216bb0f95680f9ae287a32869ac2fda0a1795d79b00)
...
```

Make sure the image was deleted:

```
$ vhc hub image --get-images
Retrieving images from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/images)...
{
  "images": []
}
```

10. Utilities

VHC offers some convenience utilities for maintaining your hub, including the commands **hub software** and **hub licenses**.

10.1. Hub Software **Command**

The **hub software** command is used to retrieve information about the software running on the hub, and to update it. This corresponds respectively to the options **--get-info** and **--upgrade**.

Notes:

For the examples below it is assumed that an active hub has been set.

The command **hub software --upgrade** requires an updated version of platform software that must first be installed by other means, for example, MAS.

To get information about the software currently running on the hub, run:

```
$ vhc hub software --get-info
Retrieving software information from C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/software/info)...
{
  "build_time": "July 09, 2019 02:48:38",
  "running_system": "Preferred system (A/B)",
  "version": "2.2.0-81.6.g089f6a7+jdoe.20190709.102900"
}
```

To update your hub software, run:

```
$ vhc hub software --upgrade /path/to/image/file
Software upgrade using file [/path/to/image/file] on C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/software/upgrade)...
 166.30 MB / 166.30 MB [============================================================================]
100.00%Success
```

10.2. Hub Licenses **Command**

The **hub licenses** command is used to retrieve from and add licenses to a hub, which corresponds to the options **--get** and **--add**.

Note: For the examples below it is assumed that an active hub has been set.

To retrieve the licenses currently installed on the hub, run:

```
$ vhc hub licenses --get
Retrieving licenses from C05BCB00C0A000001127:9000 (https://10.20.0.90:9000/licenses)...
{
  "CONTAINER-BOOTSTRAP-DEBUG": {
    "name": "CONTAINER-BOOTSTRAP-DEBUG"
  },
  "CONTAINER-EXEC": {
    "name": "CONTAINER-EXEC"
  },
  "CONTAINER-TTY": {
    "name": "CONTAINER-TTY"
  },
"DEVELOPER": {
    "data_b64": "",
    "name": "DEVELOPER"
  },
```

```
  "SERIAL-CONSOLE": {
    "name": "SERIAL-CONSOLE",
    "parameters": {
      "group": [
        "sudo",
        "docker",
        "ssh-lan"
      ],
      "groups": "sudo,docker",
      "shadowhash":
"$6$Y9LcnqfQlp7iWiWS$3UXF.L/dN5xFkGDK52n2fcdmaNbdCyMzuhvKdDKfThMVtDx8BFb0J/tft4w/ZUj9MXTq8CXVWhtcur5h0yciI1",
      "user": "veea"
    }
  },
  "SSH-LOGIN": {
    "name": "SSH-LOGIN",
    "parameters": {
      "group": [
        "wheel",
        "docker",
        "ssh-lan"
      ],
      "groups": "wheel,docker",
      "sshkey": "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCsF8h7cPF0CXChgkQpEOH/WQJ1hyb0asQnmeqRjdntWdqp6pEM2LXm9CR5r6pDrq28QJOdwNTi87f5KNavPdsn3k1UKx0AGJEJAR9W6GlXXJ559a1UJuM0CvG5bcPN2PuO7DcHnbkdMkJmij1SRd6qFeINbmnV1ZqPd9FKMSkkapPZxAocEIrdVYaJxT2JfcjfjO2lvlNyDGFYw+7Dj+JtnOZWxiOsjoPXERnAxrd3EiE8+AasYKOBby9dY7RaCZe5Qu7zEV4xV/BWkbX1b+Yk7FESEYoYb0BJKz2H+BwRlQu5ZTrfsB7s1Wg6RkIkPRgkrz1Ssz+pwGBJSzqThRuR
jstrain@jstrain-ubuntu",
      "user": "veeassh"
    }
  },
  "UBOOT-DEBUG": {
    "data_b64": "",
    "name": "UBOOT-DEBUG"
  },
  "UBOOT-MENU": {
    "data_b64": "",
    "name": "UBOOT-MENU"
  }
}
```


To install a license on the hub, run:

```
$ vhc hub licenses --add /path/to/license/file
Adding license file [/path/to/license/file] to C05BCB00C0A000001127:9000
(https://10.20.0.90:9000/licenses)...
 3.16 KB / 3.16 KB [================================================================================]
100.00%
Success
```

11. Porting from VeeaHub Toolkit Release 0.4

This section describes how to port YAML-based VeeaHub containers from VHT release 0.4 and earlier to the Dockerfile-based model starting with VHT release 0.5. The workflow method was changed to simplify the overall process and speed up development time for users. All containers created using VHT release 0.4 and earlier need to be ported using the steps described here.

Note: VHT release 0.4 (or earlier) is used as part of the porting process. Do not install the later release until instructed to do so. If you have already installed release 0.5, please remove it before proceeding.

This section uses the **IO Attach** application template as an example to illustrate the porting process and assumes that the user has an application ready for porting, with the UUID configured and set to run on unauthorized hosts.

First, change to the application directory:

```
$ cd vh_io_attach
```

Generate the *Dockerfile* for the target (VHC05 shown here, but substitute your target as necessary):

```
$ vhc build --gen --target vhc05
Generating Dockerfile for vhc05...
2019/09/13 10:27:04 Base image: arm32v7/busybox:latest, Arch: arm32
Dockerfile(s) generated!
```

The *Dockerfile* for the target should now be present:

```
$ ls
bin  config.yaml  Dockerfile.vhc05  README.md  src
```

The contents of *Dockerfile* should look as follows:

```
# This is an arm32 image. Please make sure all software is compatible.
FROM arm32v7/busybox:latest

RUN mkdir /app

# This copies your app to the docker container
COPY src/ /app/

WORKDIR /app
## Place your docker file steps here


# Docker lines

ARG VEEA_TARGET=vhc05

# Arguments

ARG TMM

# Secure container items
```

```
ARG PARTNER

LABEL com.veea.authentication.identifier="$PARTNER"

LABEL com.veea.authorisation.allowOnUnauthenticatedHost="true"

LABEL com.veea.authorisation.feature1="DEVELOPER"

ARG LICENSE_SERVER

LABEL com.veea.authentication.certificates.veeahub_license_server="$LICENSE_SERVER"

ARG LICENSE_AUTHORITY

LABEL com.veea.authentication.certificates.veeahub_license_authority="$LICENSE_AUTHORITY"


CMD ["./start.sh"] # Place your run command here.
```

At this point, please **remove VHT release 0.4 or earlier and install release 0.5**. See section 2 for more information.

Next, remove the extension from *Dockerfile*:

```
$ mv Dockerfile.vhc05 Dockerfile
$ ls
bin  config.yaml  Dockerfile  README.md  src
```

Now, using your favorite editor, make the following edits to *Dockerfile*:

1. Remove everything starting from **# Docker lines** to the line *before* **CMD ["./start.sh"] # Place your run command here**.
2. Though not required, consider condensing empty lines and removing the comments to keep things neat.
3. Remove everything after the closing **]** in the **CMD** line, that is, **# Place your run command here.**

Dockerfile should now look similar to this:

```
FROM arm32v7/busybox:latest

RUN mkdir /app
COPY src/ /app/
WORKDIR /app

CMD ["./start.sh"]
```

Edit *Dockerfile* to let VHC know which lines are specific to which Hub targets. VHC knows this via C/C++-like pragmas that must be added to the file. The pragmas should be added as follows:

```
#BEGIN <target>
<target-specific-lines>
#END
```

In this specific case, *Dockerfile* should now look like this:

```
#BEGIN vhc05
FROM arm32v7/alpine:3.9
```

```
#END

RUN mkdir /app
COPY src/ /app/
WORKDIR /app

CMD ["./start.sh"]
```

Add support for the VHX09-10 target using the same method from the previous step:

```
#BEGIN vhc05
FROM arm32v7/alpine:3.9
#END
#BEGIN vhx09-10
FROM arm64v8/alpine:3.9
#END

RUN mkdir /app
COPY src/ /app/
WORKDIR /app

CMD ["./start.sh"]
```

Note: the vhc06 target has been removed from VHC starting from release 0.5.

Edit *config.yaml* to remove all the items that are now contained in *Dockerfile*, that is, anything specified by the following keys:

```
docker.arg
docker.build32
docker.build64
docker.cmd
docker.environment
docker.image32
docker.image64
```

config.yaml now looks as follows:

```
app: vh io attach
app_info:
  deployment: single
  targets:
  - vhx09-10
  - vhc05
  type: app
  version: 1
  view: :9100
docker:
  secure:
    arg:
      partner_features: []
    label:
      devices: []
      features:
      - DEVELOPER
      unauth_host: true
      uuid: 9157561D-DC41-4255-8EFA-3185F9ACEDB7
```

```
      volumes:
        offers: []
        persists: []
        requests: []
hub:
  active: ""
  hubs: []
version: 1
```

Now, we must rebuild the image using VHT release 0.5 or later:

```
$ vhc build --target vhc05
Making vh_io_attach image for target vhc05...
Dockerfile.vhc05 does not exist.
Generating Dockerfile for Dockerfile.vhc05...
bin/vh_io_attach:vhc05.tar does not exist.
Building vh_io_attach image for vhc05...
Command Started
Sending build context to Docker daemon   16.9kB
Step 1/17 : FROM arm32v7/alpine:3.9
 ---> ea2ccc7da15e
Step 2/17 : RUN mkdir /app
 ---> Running in de27503bdc74
Removing intermediate container de27503bdc74
 ---> 58eeff0490ed
Step 3/17 : COPY src/ /app/
 ---> 75e6e65b577d
Step 4/17 : WORKDIR /app
 ---> Running in 87d624b7874c
Removing intermediate container 87d624b7874c
 ---> 1b6a10f02c04
Step 5/17 : LABEL com.veea.vhc.target="vhc05"
 ---> Running in 3bcf86d95b4f
Removing intermediate container 3bcf86d95b4f
 ---> 493dcb1755cb
Step 6/17 : LABEL com.veea.vhc.version="0.5.0"
 ---> Running in 2d478466022c
Removing intermediate container 2d478466022c
 ---> 3ddc06a79020
Step 7/17 : ARG PARTNER_LICENSE
 ---> Running in b3841792db24
Removing intermediate container b3841792db24
 ---> 3ef8088b04f2
Step 8/17 : LABEL com.veea.authentication.identifier="$PARTNER_LICENSE"
 ---> Running in b801b678fab2
Removing intermediate container b801b678fab2
 ---> f36fe6510ba6
Step 9/17 : ARG PARTNER_ID
 ---> Running in eddf42ba1a68
Removing intermediate container eddf42ba1a68
 ---> ad5cd0b5b0e8
Step 10/17 : LABEL com.veea.image.persistent_uuid="$PARTNER_ID-DC41-4255-8EFA-3185F9ACEDB7"
 ---> Running in 588d56b9d598
Removing intermediate container 588d56b9d598
 ---> 3443d2b9a9c0
Step 11/17 : LABEL com.veea.authorisation.allowOnUnauthenticatedHost="true"
 ---> Running in 8e293446cd6d
Removing intermediate container 8e293446cd6d
 ---> 8d09fb9b4798
Step 12/17 : LABEL com.veea.authorisation.feature1="DEVELOPER"
 ---> Running in a379799d5d81
Removing intermediate container a379799d5d81
 ---> 8465e60ff33f
Step 13/17 : ARG LICENSE_SERVER
```

```
 ---> Running in b12602da6714
Removing intermediate container b12602da6714
 ---> e817a7d80ba2
Step 14/17 : LABEL com.veea.authentication.certificates.veeahub_license_server="$LICENSE_SERVER"
 ---> Running in 214bfbcd3d35
Removing intermediate container 214bfbcd3d35
 ---> 240c7ab29490
Step 15/17 : ARG LICENSE_AUTHORITY
 ---> Running in 407c19315f79
Removing intermediate container 407c19315f79
 ---> f122bd866b49
Step 16/17 : LABEL
com.veea.authentication.certificates.veeahub_license_authority="$LICENSE_AUTHORITY"
 ---> Running in b59bf6327815
Removing intermediate container b59bf6327815
 ---> a92df8509f2d
Step 17/17 : CMD ["./start.sh"]
 ---> Running in 35f41d9676c0
Removing intermediate container 35f41d9676c0
 ---> 7806a38a1758
Successfully built 7806a38a1758
Successfully tagged vh_io_attach:vhc05


Saving vh_io_attach image for vhc05.
```

Note that the target *Dockerfile* and application image were built in a single step.

Check that the target *Dockerfile* was generated:

```
$ ls
bin  config.yaml  Dockerfile  Dockerfile.vhc05  README.md  src
```

Examine its contents:

```
$ more Dockerfile.vhc05

#BEGIN vhc05
FROM arm32v7/alpine:3.9
#END

RUN mkdir /app
COPY src/ /app/
WORKDIR /app

#BEGIN AUTO-GENERATED - DO NOT EDIT!!!
LABEL com.veea.vhc.target="vhc05"
LABEL com.veea.vhc.version="0.5.0"
ARG PARTNER_LICENSE
LABEL com.veea.authentication.identifier="$PARTNER_LICENSE"
ARG PARTNER_ID
LABEL com.veea.image.persistent_uuid="$PARTNER_ID-DC41-4255-8EFA-3185F9ACEDB7"
LABEL com.veea.authorisation.allowOnUnauthenticatedHost="true"
LABEL com.veea.authorisation.feature1="DEVELOPER"
ARG LICENSE_SERVER
LABEL com.veea.authentication.certificates.veeahub_license_server="$LICENSE_SERVER"
ARG LICENSE_AUTHORITY
LABEL com.veea.authentication.certificates.veeahub_license_authority="$LICENSE_AUTHORITY"
#END AUTO-GENERATED - DO NOT EDIT!!!

CMD ["./start.sh"]
```

Note the following:

- Only the lines within the pragma for the specified target are included, in this case **vhc05**.
- The Veea-specific additions to the *Dockerfile* are added just prior to the **CMD** (or **ENTRYPOINT**) statement and denoted via marker lines.
- Do not edit the target-specific *Dockerfile* as re-running the build command (**vhc build**) will overwrite the target-specific Dockerfile.
- The original source *Dockerfile* is not modified by VHC.

Check that the image was built:

Checking image vh_io_attach:vhc05...

```
$ ls bin
vh_io_attach:vhc05.tar
```

From this point forward, the process for loading an application on to the hub is the same as with previous versions of VHC, for example, via **vhc hub** commands.

12. Notes, FAQs and Tips

12.1. Why can't I ping the Sideload Server? (section 5.2)

The VHC ping times out if you do not have the DEV license installed.

The standard ping command works.

12.2. How do I know what templates are available (section 3.3)?

New templates are released when they are ready and are not tied to releases of the VeeaHub Toolkit.

You can update the index of templates, which is stored locally, using the **vhc update** command, then list the templates using the **vhc list** command. To start using any of the templates, use the **vhc new –template** command with the template name (section 4).

12.3. Where can I get advice on creating containers?

There is a separate Veea document on container development methodology [RD/3].

12.4. Debugging Tips

12.4.1. Mapping Source Code to the Development PC

- A convenient way to debug your container code is to map the source code on your build PC to a volume on the VeeaHub. This avoids having to repeat the build flow every time you make a change to the container during debugging.

  Suggested steps:

  a. Create a volume on the VeeaHub.
  b. Export the volume.
  c. Mount the volume via WebDAV on your build machine.
  d. Copy the code into the folder on the build machine.

You can now edit the code on the PC and see it on the VeeaHub. Note that if the VeeaHub is bootstrapped, you will lose your code.

12.4.2. Attaching to a Container

Use the command **vhc hub container --attach <container-id> /bin/sh** to help with debugging.

12.4.3. Debugging a Container

It is only possible to attach to a running container. If a container exits due to an error code, it is not possible to connect and can be difficult to see the error code. To debug a container that keeps exiting, the following procedure is suggested.

a. To keep the container alive, create a script that is a simple loop. For example:

```
#!/bin/sh
## startup script
## Hang around, as docker service will restart this if we complete
while true; do sleep 200d; done
```

b. Copy this file into your container and use as the entry point in *config.yaml*, for example:

```
  - COPY ./startup.sh  /home/.
  - RUN chmod +x /home/startup.sh
cmd: /home/startup.sh
```

c. Attach to the container when it is running on the VeeaHub.
d. Run your original code to debug.

13. Technical Support and Developer Community

Before contacting Technical Support, please consult the documentation, tutorials, and community topics available on the support web site www.veea.com/support/. Please sign up, if you don't already have an account, and sign in.

For unresolved queries, click on the Submit a request link located near the top of the screen.

Please complete the form with an appropriate subject, and as much detail as possible in the description field. Please include any relevant information such as Veea hardware serial numbers, logs, screenshots, etc.

An email will automatically be sent to your registered email address to confirm the request has been received.

There is a community for developers at developers.veea.com, where you may register to discuss all aspects of development of apps for the VeeaHub.

You can provide feedback on the development process through the support web site.