

OPNFV Build instructions for - Fuel deployment tool - OPNFV Arno SR1 release

Table of Contents

Abstract	1
License	1
Version history	1
Introduction	2
Requirements	2
Minimum Hardware Requirements	2
Minimum Software Requirements	2
Preparations	2
Setting up the Docker build container	2
Setting up OPNFV Gerrit in order to being able to clone the code	2
Clone the OPNFV code git repository	3
Building	3
Configure your build environment	3
Low level build method using make	3
Abstracted build method using build.sh	3
Artifacts	4
References	4

Abstract

This document describes how to build the Fuel deployment tool for the Arno SR1 release of OPNFV, the build system, dependencies and required system resources.

License

Arno SR1 release of OPNFV when using Fuel as a deployment tool DOCs (c) by Jonas Bjurel (Ericsson AB)

Arno SR1 release of OPNFV when using Fuel as a deployment tool DOCs (c) are licensed under a Creative Commons Attribution 4.0 International License. You should have received a copy of the license along with this. If not, see <<http://creativecommons.org/licenses/by/4.0/>>.

Version history

Date	Ver.	Author	Comment
2015-06-03	1.0.0	Jonas Bjurel (Ericsson AB)	Instructions for the Arno release
2015-09-24	1.1.0	Jonas Bjurel (Ericsson AB)	Instructions for the Arno SR1 release

Introduction

This document describes the build system used to build the Fuel deployment tool for the Arno SR1 release of OPNFV, required dependencies and minimum requirements on the host to be used for the buildsystem.

The Fuel build system is designed around Docker containers such that dependencies outside of the build system can be kept to a minimum. It also shields the host from any potential dangerous operations performed by the build system.

The audience of this document is assumed to have good knowledge in network and Unix/Linux administration.

Requirements

Minimum Hardware Requirements

- An x86_64 host (Bare-metal or VM) with Ubuntu 14.04 LTS installed
- ~30 GB available disc
- 4 GB RAM

Minimum Software Requirements

The build host should run Ubuntu 14.04 operating system.

On the host, the following packages must be installed:

- docker - see <https://docs.docker.com/installation/ubuntu/linux/> for installation notes for Ubuntu 14.04. Note: only use the Ubuntu stock distro of Docker (docker.io)
- git (simply available through `sudo apt-get install git`)
- make (simply available through `sudo apt-get install make`)
- curl (simply available through `sudo apt-get install curl`)

Preparations

Setting up the Docker build container

After having installed Docker, add yourself to the docker group:

```
<usermod -a -G docker [userid]>
```

Also make sure to define relevant DNS servers part of the global dns chain in in your `</etc/default/docker>` configuration file, eg.

```
<DOCKER_OPTS="--dns=8.8.8.8 --dns=8.8.8.4">
```

Then restart docker:

```
<sudo service docker.io restart>
```

Setting up OPNFV Gerrit in order to being able to clone the code

- Start setting up OPNFV gerrit by creating a SSH key (unless you don't already have one), create one with `ssh-keygen`
- Add your generated public key in OPNFV Gerrit [<https://gerrit.opnfv.org/>](https://gerrit.opnfv.org/) (this requires a linuxfoundation account, create one if you do not already have one)

- Select "SSH Public Keys" to the left and then "Add Key" and paste your public key in.

Clone the OPNFV code git repository

Now it is time to clone the code repository:

```
<git clone ssh://Linux foundation user'@gerrit.opnfv.org:29418/genesis>
```

Now you should have the OPNFV genesis repository with the Fuel directories stored locally on your build host.

Check out the Arno SR1 release: `<cd genesis> <git checkout arno.2015.2.0>`

Building

There are two methods available for building Fuel:

- A low level method using Make
- An abstracted method using build.sh

Configure your build environment

**** Configuring the build environment should not be performed if building standard Arno release ****

Select the versions of the components you want to build by editing the `fuel/build/config.mk` file.

Low level build method using make

The low level method is based on Make:

From the `<fuel/build>` directory, invoke `<make [target]>`

Following targets exist:

- none/all - this will:
 - If not already existing, initialize the docker build environment
 - If not already done, build OpenDaylight from upstream (as defined by `fuel-build config-spec`)
 - If not already done, build fuel from upstream (as defined by `fuel-build/config-spec`)
 - Build the defined additions to fuel (as defined by the structure of this framework)
 - Apply changes and patches to fuel (as defined by the structure of this framework)
 - Reconstruct a fuel .iso image
- clean - this will remove all artifacts from earlier builds.

If the build is successful, you will find the generated ISO file in the `<fuel/build/release>` subdirectory!

Abstracted build method using build.sh

The abstracted build method uses the `<fuel/ci/build.sh>` script which allows you to:

- Create and use a build cache - significantly speeding up the buildtime if upstream repositories have not changed.
- push/pull cache and artifacts to an arbitrary URI (`http(s):`, `file:`, `ftp:`)

For more info type `<fuel/ci/build.sh -h>`.

Artifacts

The artifacts produced are:

- <OPNFV_XXXX.iso> - Which represents the bootable Fuel image, XXXX is replaced with the build identity provided to the build system
- <OPNFV_XXXX.iso.txt> - Which holds version metadata.

References

-

Authors: Jonas Bjurel (Ericsson)

Version: 1.1.0

Documentation tracking

Revision: 563547b4a9f44090f32c0e17d040114854563760

Build date: Wed Sep 30 21:27:27 UTC 2015