Authors: Jonas Bjurel (Ericsson)
Version: 0.1

OPNFV Build instructions for - < Component denomination >

Abstract

This document describes how to build <Component>, build system dependencies and required system resources.

License

Contents

1 Version history

Date	Ver.	Author	Comment
2015-04-14	0.1.0	Jonas Bjurel	First draft
	0.1.1		
	1.0		

2 Introduction

<INTRODUCTION TO THE SCOPE AND INTENTION OF THIS DOCUMENT>

<EXAMPLE>:

This document describes build system used to build Fuel@OPNFV, required dependencies and minimum requirements on the host to be used for the buildsystem.

The Fuel build system is desigened 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.

3 Requirements

<PROVIDE A LIST OF MINIMUM HARDWARE AND SOFTWARE REQUIREMENTS FOR THE BUILD>

3.1 Minimum Hardware Requirements

<EXAMPLE>:

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

3.2 Minimum Software Requirements

<EXAMPLE>: 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/ubuntulinux/ 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)

3.3 Preparations

<EXAMPLE>:

3.3.1 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>

3.3.2 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/> (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.

3.3.3 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 Fuel@OPNFV stored locally on your build host.

4 Building

<DESCRIBE THE FULL PROCEDURES FOR THE BUILD OF THE OPNFV COMPONENT ARTIFACTS>

<EXAMPLE>:

There are two methods available for building Fuel@OPNFV:

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

4.1 Configure your build environment

Select the versions of the components you want to build by editing the fuel/build/config.mk file. Note if you want to build with OpenDaylight SDN controller you need to uncomment the lines starting with odl-main and java-main

4.2 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 allready 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!

4.3 Abstracted build method using build.sh

The abstracted build method useses 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>.

5 Artifacts

<DESCRIBE WHAT ARE THE PRODUCED ARTIFACTS AND WHERE THOSE CAN BE FOUND>

<EXAMPLES>:

The artifacts produced are:

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

6 References

<PROVIDE NEEDED/USEFUL REFERENCES>