OpenSwitch-The Hybrid Switch

Tuesday, October 4, 2016 8:26 PM

Currently supported plugins

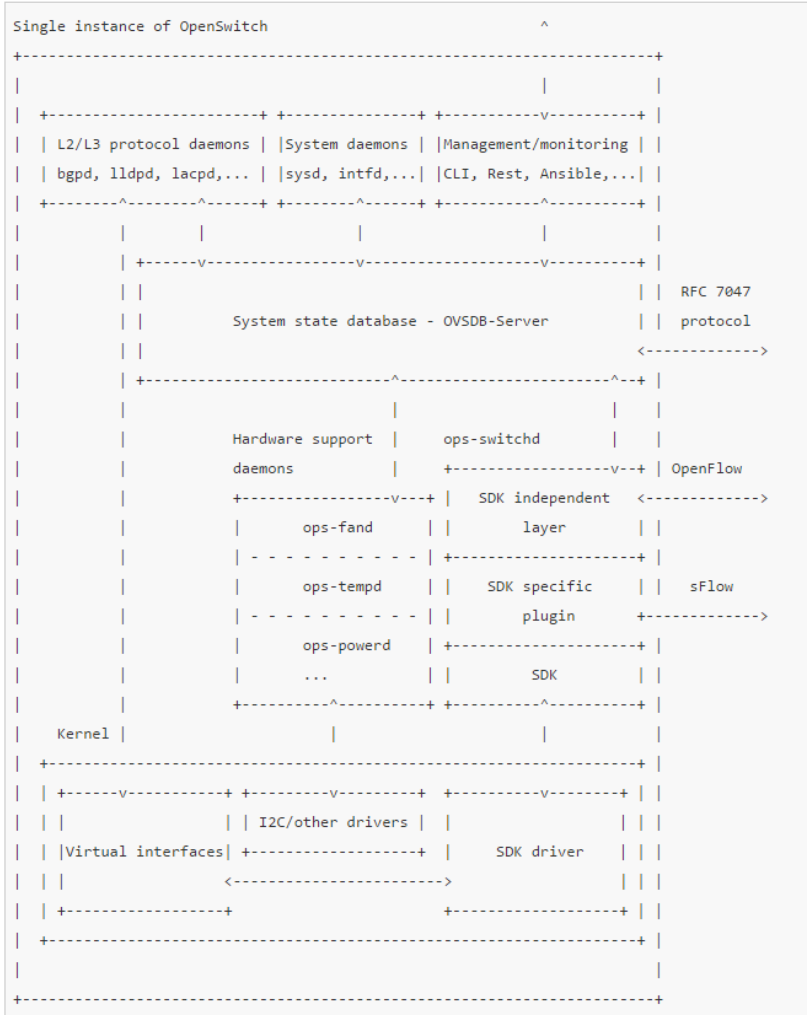
OpenSwitch currently supports two plugins:

- OpenNSL plugin - interfaces **OpenNSL SDK** and supports Broadcom Trident II ASIC.
- Docker container plugin - allows OpenSwitch to operate inside Docker container, where pure Open vSwitch is used as an ASIC emulation.

http://www.openswitch.net/documents/user/porting#porting-to-a-different-asic

https://github.com/open-switch/ops-switchd-opennsl-plugin/tree/master/src

Top level view of the system



Porting to a different ASIC

As further explained in **OpenSwitch Architecture**,
ops-switchd is conceptually constructed out of three layers -
SDK independent layer, SDK specific plugin and ASIC SDK itself.

SDK independent layer resides in openswitch/ops-openvswitch repository,
mainly in vswitch/ and ofproto/ directories.
It's responsible for communicating to OVSDB-server and Openflow controllers.

SDK specific plugins are essentially ASIC drivers of OpenSwitch system.
Each plugin resides in a separate repository called openswitch/ops-switchd-

[underlying SDK]-plugin.
Build process in the plugin repositories results in a dynamically linked library.
When started, ops-switchd searches for these libraries in a predefined path and loads the libraries.

The way that SDK itself is loaded and operated is specific to the plugin.

Feature Documentation

The following table lists the type of documents, the target locations, and the expected file names. For a description of each type of document, see the following table:

Doc Type	Repo	Directory	File Name	Purpose
Functionality Guide	openswitch/ops	/docs	[functionality]_guide.md	Details on the functionality and its corresponding CLIs
Feature Designs	openswitch/ops	/docs	[feature]_design.md	Details of how the various components come together to deliver feature functionality to end user
Feature Test Plans	openswitch/ops	/tests	[feature]_test.md	Document each test case from corresponding [feature]_test.py test script

Screen clipping taken: 10/10/2016 10:36 AM

Cloning OpenSwitch

The OpenSwitch source code is accessible in the **OpenSwitch Git Repository**, where the source code is organized into several projects. If OpenSwitch is being built only for the purpose of creating a software image, then the **openswitch/ops-build** project only needs to be cloned.

Refer to the **How to contribute to OpenSwitch** guide for instructions on how to contribute to the OpenSwitch code.

→ Browse to this location & lists all the repositories.

To clone the OpenSwitch repository, use the following `git clone` command and URL. The use of `<directory>` is optional (if omitted, `<directory>` defaults to `ops-build`).

Note: This command clones the OpenSwitch development environment and build system, which can be used to build an OpenSwitch image. The command does not automatically clone all of the OpenSwitch source code, although the Development Environment commands (`make devenv_add`) can be used to access the rest of the source code. See the **Changing OpenSwitch Code** documentation for more details.

} get the openswitch_build working

↓ use development Environment Commands (make devenv_add)

Use rest of the source code.

```
$ git clone https://git.openswitch.net/openswitch/ops-build [<directory>]
```

Screen clipping taken: 10/10/2016 12:08 PM

High level design of OF-DPA OpenFlow Hybrid Switch Feature

The OF-DPA OpenFlow Hybrid Switch Feature is contained within the ASIC plugins that implement it. One such ASIC plugin is the OpenNSL plugin.

Code in the ops-switchd repository contains an implementation of an OpenFlow agent. This implementation comes from the Open vSwitch code on which ops-switchd is based. Information about the OpenFlow agent from Open vSwitch can be found in the ovs repository (**openswitch/ovs**).

The OpenNSL plugin in the ops-switchd-opennsl-plugin repository is an example of an ASIC plugin that implements the OF-DPA OpenFlow Hybrid Switch. The details of the design are found in DESIGN.md document found in the ops-switchd-opennsl-plugin repository. This serves as a reference for the OpenNSL implementation or for adding support to other ASIC plugin implementations.

http://git.openswitch.net/cgit/openswitch/ops/tree/docs/layer3_design.md

```
# Basic Layer3 Design

# Introduction

OpenSwitch is designed to support layer3 features and protocols. To facilitate this, the following capabilities have been added.
```

Screen clipping taken: 10/10/2016 4:14 PM

http://www.openswitch.net/documents/dev/ops-switchd-opennsl-plugin/design

http://www.openswitch.net/documents/user/architecture

ops-switchd

The primary responsibility of ops-switchd is to translate the data model residing in the OVSDB into ASIC-specific SDK calls and vice versa. In fact, ops-switchd is the only daemon allowed to access the ASIC SDK.

Internal modularity of ops-switchd provides the ability to port OpenSwitch to different SDKs and the corresponding silicon. For more information on ASIC portability, refer to **"porting to a different ASIC"**.

Being based on ovs-vswitchd from Open vSwitch, ops-switchd also contains the Openflow agent and communicates directly to Openflow controllers. OpenFlow functionality is supported on certain platforms (the OpenNSL is one ASIC SDK that has support).

Screen clipping taken: 11/5/2016 9:06 AM

http://www.openswitch.net/use/usehome#downloads

https://github.com/open-switch/ops-docs/blob/master/deploy-to-physical-switch.md Installing OpenSwitch.

Machine generated alternative text: O www.openswitch.net/use/usehome#downloads rd University IELTSbuddy IELTS n n Interesting Links TUDarmstadt OnlineEdu LINUX FOUNDATION OpenSwitch Installing n n ThingstodoinSwitz Travel&Shopping n Programming Langue, n TKI Ass ignments n Entertainment n Finance KN2Project >Installing Use Farticjpate Develop YDownloads >IJser Guides References OpenSwitch supports hardware based on ONIE. Below you will find the installation guide. OpenSwitch also provides an appliance Virtual Machine that is useful for testing and demo purposes. >Quick Start Guide for Physical Switches >Quick Start Guide for Virtual Appliance Downloads OpenSwitch is still on development. but you can download pre-release images from our archive: Periodic images are built every 6 hours with the latest code from the master and Release images are tagged development check points. >OpenSwitch Download Archive User Guides >Access Control Lists(ACL) >Ansible >Authentication (AAA CLI) >BGP >DHCP/TFTP Server >Diagnostics >évent Log >Kernel Core Dump >Mirror >MSTP > NT p >Phv Interfaces >Sflow Show Core Show Tech >Zero Touch

https://www.hpe.com/h20195/v2/GetPDF.aspx/4AA6-3968ENW.pdf

http://openswitch.net/documents/user/mgmt_intf_cli --> CLI interface commands
http://www.openswitch.net/documents/dev/quick-start-physical

https://archive.openswitch.net/artifacts/periodic/master/ --> Download the latest image from here.

Open vSwitch module structure

Open vSwitch (OVS) module is built in OpenSwitch (OPS) by downloading its sources directly from **github.com/openvswitch/ovs**. OPS specific changes are applied as a
collection of patches on top of the OVS code.

The recipe of the module contains the list of patches that are applied to the OVS code after it is downloaded. Both, the recipe and patch files, reside on the **openswitch/ops-build**
repository. This repository contains all the information necessary to recreate any OVS
build.

Since reviewing the code directly in the patch files can be difficult as the changes are displayed without the full context of the file, the patch changes are kept and reviewed in the **openswitch/ovs** git repository. Note that this repository exists for the sole purpose of reviewing the code of the patches. It is neither part of the build process, tested, nor integrated with CIT.

The openswitch/ovs repository inherits the branches from the OVS project. It has
branches for each release. For example 2.5, the current release, is a branch named `feature/branch-2.5`. openswitch/ovs has special branches associated with each release that follow the name convention: `patches/branch-<OVS_VERSION>`. It contains all the patches that are applied to that particular OVS version in OPS.

Having all the patches applied one after the other on a branch, reduces the possibility of conflicts between the patch files. Adding new functionality is easy as it only requires to put a new commit on top of the corresponding patches

https://github.com/aweeraman/debian-ncc/blob/master/scripts/gengraph.py

How to Build and develop OpenNSL-plugin?!!
https://lists.openswitch.net/pipermail/ops-dev/2016-March/002742.html
https://mail.openvswitch.org/pipermail/ovs-dev/2016-September/323578.html

Only the first few lines(until cd -) is relevant for ops-switchd-opennsl-plugin changes.

```
make switch-platform as5712
make devenv_add ops-switchd-opennsl-plugin
cd src/ops-switchd-opennsl-plugin
# make changes, build & test on a switch
git format-patch /origin/master > /tmp/nsl.patch
cd -
# switch to container
make switch-platform genericx86-64
make devenv_rm ops-switchd-opennsl-plugin
make devenv_add ops-switchd-container-plugin
cd src/ops-switchd-container-plugin
# make changes, build & test in a container
git format-patch /origin/master > /tmp/container.patch
cd -
# switch back to as5712
make switch-platform as5712
make devenv_add ops-switchd-opennsl-plugin
make devenv_rm ops-switchd-container-plugin
cd src/ops-switchd-opennsl-plugin
git am < /tmp/nsl.patch
# make changes, build & test on a switch
# ... on & on
```

```
git clone  https://git.openswitch.net/openswitch/ops-build

  cd ops-build/

  git checkout -b 1.0.0

  git status

  git branch

  make configure as5712

  make devenv_init

  make devenv_add ops

  make devenv_add ops-openvswitch

  make devenv_add ops-switchd

  make devenv_add ops-switchd-opennsl-plugin

  make devenv_add ops-cli

  make
```

https://lists.openswitch.net/pipermail/ops-dev/2016-February/011386.html --> OpenSwitch a Monolithic OS. This is exactly why I did not attempt this.
https://wiki.yoctoproject.org/wiki/Building_your_own_recipes_from_first_principles to use Bitbake to add a new recipe.
http://www.yoctoproject.org/docs/current/dev-manual/dev-manual.html#usingpoky-extend-customimage-localconf

- ***Yocto Project Mailing Lists:*** To subscribe to the Yocto Project mailing lists, click on the following URLs and follow the instructions:
  - http://lists.yoctoproject.org/listinfo/yocto for a Yocto Project Discussions mailing list.
  - http://lists.yoctoproject.org/listinfo/poky for a Yocto Project Discussions mailing list about the OpenEmbedded build system (Poky).
  - http://lists.yoctoproject.org/listinfo/yocto-announce for a mailing list to receive official Yocto Project announcements as well as Yocto Project milestone
  - http://lists.yoctoproject.org/listinfo for a listing of all public mailing lists on `lists.yoctoproject.org`.
- ***Internet Relay Chat (IRC):*** Two IRC channels on freenode are available for Yocto Project and Poky discussions: `#yocto` and `#poky`, respectively.
- ***OpenEmbedded:*** The build system used by the Yocto Project. This project is the upstream, generic, embedded distribution from which the Yocto Project and to which it contributes.
- ***BitBake:*** The tool used by the OpenEmbedded build system to process project metadata.
- ***BitBake User Manual:*** A comprehensive guide to the BitBake tool. If you want information on BitBake, see this manual.
- ***Quick EMUlator (QEMU):*** An open-source machine emulator and virtualizer.

http://www.openswitch.net/documents/dev/changing-openswitch-code
http://www.openswitch.net/documents/dev/changing-openswitch-code#build-system-infrastructure
The build system is based in the Yocto Project. This document does not cover how Yocto works, but does include some important aspects related to OpenSwitch. The following directories and files are displayed after cloning OpenSwitch on the machine: build/ COP Y ING images/ Makefile REP.DME . md src/ tools/ yocto/ • build': This directory contains user configuration files and the output generated by Open Embedded: it is standard configuration. • COPYING: Licensing information. For more details. see the Apache License. • images/: Symbolic links to the images that have been built. The most important file is . tar . gz Makefile: The OpenSwitch Makefile. tools': This directory contains tools primarily used by the build system. This directory includes the Rules. make file.

which is the core of Makefile. yocto/: The core of the build system. Inside this directory: openswitch/ poky/ • poky': This directory is the core of the Yocto project. For more information. see YoctoProiect Documentation. • openswitch/: This directory is the core OpenSwitch directory. Its contents are shown in the following example:

http://www.openswitch.net/documents/dev/contribute-code#adding-a-new-component

Adding a recipe for the component For your component work properly, you must have a recipe. This recipe should be placed in the ops-build repo, inside yocto/openswitch/meta-distro-openswitch. There are different directories for the recipes: recipes-core recipes-kernel recipes-networking recipes-onie recipes-ops Choose the directory that applies and add the . bb file to it.