Basebox - Integrating Whitebox Switches into Linux

A Controller Implementation for OF-DPA Hardware

Daniel Fritzsche, Zsolt Magyari, Michael Schlosser and Tobias Jungel
BISDN GmbH
Berlin, Germany
daniel.fritzsche@bisdn.de

***Abstract*— In this paper we show an SDN controller implementation named Basebox that listens to Linux Netlink and translates commands into OpenFlow rules to control a domain of OF-DPA based switches in an OpenStack cluster.**

Keywords*— *whitebox switch; OpenStack; Netlink; OpenFlow; controller; SDN

I. INTRODUCTION

Recently, a number of open-source initiatives have commoditized the data center. While OpenStack provides the base for a relatively easy setup of data center software, the Open Compute Project certifies hardware that can be used to build an entire data center from commodity hardware. The integration of whitebox switches into OpenStack is however still dominated by commercial solutions.

Based on the principles of a layered SDN controller architecture and the OpenFlow protocol we developed Basebox, a modular software that listens to Linux Netlink commands [1] and translates them into OpenFlow rules for Broadcom switches (OF-DPA2.0, OpenFlow - Data Plane Abstraction) [2]. In comparison with other solutions, one instance of the software can control multiple hardware switches at once, which lets the user scale his switching capacity without any changes to the application that is used. Basebox can run any Linux networking application on top of it, e.g. IP-routers like Quagga. In a first use case, Basebox was utilized to integrate whitebox switches into OpenStack to target the needs of small-to-medium OpenStack installations. Consisting of an embedded Linux based operating system and two stacked SDN controllers (Baseboxd and CAWR) it integrates into OpenStack via the Modular Layer 2 (ML2) Neutron plug-in. This enables the creation of isolated tenant networks (network slicing), on-demand VLAN configuration for tenant networks as well as other features like multi-chassis link aggregation. Basebox is built on BISDN's 'Revised OpenFlow Library (rofl)' in C++ and can be deployed via ONIE [3]. Both, rofl and the standalone controller Baseboxd, are Open-source software and available on github [4].

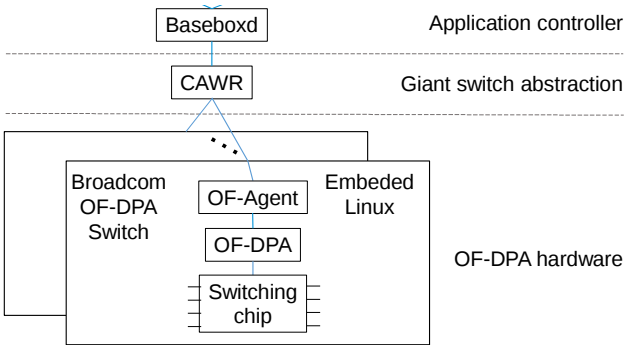

Fig. 1: Architecture and components of Basebox

II. CONTROLLER ARCHITECTURE AND COMPONENTS

In Fig. 1 the components of Basebox are shown. Baseboxd is the controller part that interacts with typical Linux network applications, whereas the CAWR controller provides a giant switch abstraction to Baseboxd, hiding details about the hardware domain like internal connections. Since the controllers use the OpenFlow protocol (requires IP-connectivity) they can be placed on the same host or on different hosts, in a virtual machine (VM) or in Linux containers, as long as the quality of the OpenFlow channel in terms of bandwidth and latency is sufficient for the applications being used.

Only the external ports of the CAWR domain are visible to the Baseboxd controller. From its view, the CAWR domain and the CAWR controller appear as a single giant network element which sends and receives FlowMods and other OpenFlow messages. The features of Baseboxd could have also been included in the CAWR controller. However, the layered approach followed here leads to better scaling and more innovation, as the Baseboxd developers do not need to care about the size of the switching domain. If no giant switch is needed, Baseboxd can be run standalone on a single switch as

well. In any case, the Linux application running on top of Baseboxd doesn't need to be changed, not matter if CAWR is used or not.

A. *The Baseboxd controller*

Baseboxd is a SDN controller released earlier this year, that translates Linux Netlink commands into OpenFlow switch rules and vice versa. The proposed solution can be easily managed and flawlessly integrated in any existing Linux environment. The application on top can be any Linux networking tool, e.g. software routing daemons like Quagga, BIRD, or even an OpenStack ML2 plugin that is demonstrated in this paper.

Baseboxd is a Linux application that has an OpenFlow (OF-DPA2.0) southbound interface that can be attached to a single OpenFlow northbound interface, either to a switch directly or to the CAWR controller (II. B.) to increase switching capacity. For each OpenFlow switch port that is attached to Baseboxd it creates a tap interface in Linux that can then be used like any other tap interface, e.g. be added to a Linux bridge or be bound to a VLAN. Baseboxd will translate any operation on the tap interface into OpenFlow rules and configure the switches properly. It is also able to handle packet_ins like ARP, interpret the message, update the network state in Linux and send the corresponding packet_out messages if required.

B. *The CAWR controller*

CAWR – Capability AWare Routing – is a supplemental shim OpenFlow controller that is layered in between the data path network elements and an application controller (e.g. Baseboxd) and that creates a giant switch abstraction from a set of OpenFlow switches. This giant switch smoothly integrates with Baseboxd and increases the effective switching capacity of the system.

The CAWR controller manages the ports of all network elements in the domain. It uses LLDP (Link Layer Discovery Protocol) protocol to discover its topology and OpenFlow to retrieve switch properties such as the number of ports, the number of tables and flow entries per table. Is a switch connected to another switch in the domain, then the two connected ports are considered as internal and only visible to the CAWR controller, otherwise they are external. External ports are exposed to the Baseboxd controller. If the multi-chassis link aggregation (MLAG) option is used, CAWR will manage a pair of switch ports that is connected to a single physical host but will export just a single port to Baseboxd. Both, Baseboxd and the Linux application running on top of it do not need to be changed in any way, regardless of how many switches are in use. Other tasks of CAWR are to translate FlowMods and other OpenFlow control messages; apply algorithms to calculate paths inside its domain; handle error cases like port and link failures, which result in re-routing internal traffic; or manage load balancing.

III. CLOUD INTEGRATION USE CASE

In a first use-case, we demonstrate the integration of whitebox switches into OpenStack. To do so we developed a Neutron ML2 plugin that can be combined with Baseboxd for the seamless integration with OpenStack. The ML2 plugin

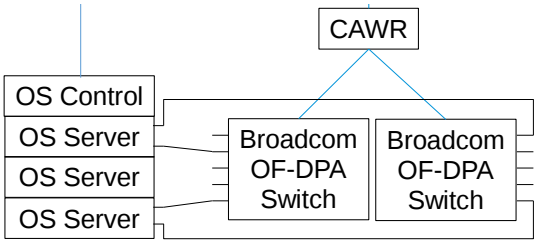

Fig. 2: OpenStack integration using Basebox

enables the system to adapt to the network state of OpenStack tenant networks automatically. Whenever a VM is created in an OpenStack tenant network, the OpenStack control node will notify Baseboxd / Netlink via the ML2 plugin which will then configure the switch ports via OpenFlow messages, i.e. enable the MAC of the VM for the corresponding VID and port.

If CAWR is also used, then it will translate the Openflow messages coming from Baseboxd depending on the network state, taking into account what ports or links are active and calculate paths in case of failover or load balancing. Having a domain of multiple switches means more available ports, but also the ability to do MLAG (connect two ports per OpenStack server to two different switches) and load balancing (reroute traffic internally in a way that links to and from servers can be fully utilized, but also make sure that internal switches and links are not congested).

This use case was tested in a medium sized OpenStack installation of a Berlin based data center operator. The realized setup uses a yocto-based embedded Linux running on Trident-II switches of Quanta and Accton/Edge-core, respectively. We used two switches to connect about 50 servers in an MLAG configuration. Each server was connected via a bond interface aggregating two physical ports, where each port was connected to a different switch. During the test, the switches were booted from scratch and our Linux distribution was installed along with the Basebox package. Then, virtual machines were created in a tenant network in OpenStack and Basebox configured the ports on the switches. We evaluated the flow tables of the hardware switched to check the results and the switch state was always updated accordingly. In another test, we simulated the case of a link or port failure on a server and the CAWR controller automatically added FlowMods to the switches in order to reroute traffic so that the server was still reachable from all ports of all other switches. In any case, no manual configuration of switch hardware was needed.

IV. Conclusion

Based on Broadcom's OF-DPA 2.0 specification, we developed Basebox, a layered SDN controller to integrate whitebox switches into Linux. The solution can be deployed via ONIE and can run a standalone Layer2 switch when Baseboxd runs locally. Alternatively, a remote OpenFlow controller named CAWR builds a giant switch abstraction across a number of whitebox switches and allows the user to scale their switching capacity. The same Baseboxd controller then manages an entire domain of switches via a single OpenFlow interface including features like automatic failover and multi-chassis link aggregation. Basebox can run any Linux networking application on top of it. In a first use case it was utilized to integrate OpenFlow switches into OpenStack via the ML2 plugin. By implementing multi-path routing and MLAG, we could combine a scalable data center switching solution with high availability.

Acknowledgment

This work was partially funded by the European Commission through the H2020 project 5GEx.

Special thanks goes to Tomasz Fratczak and Girmaye Delelegn Desta from BISDN for valuable support.

References

[1] https://www.linux.com/manpage/man7/netlink.7.html
[2] https://github.com/Broadcom-Switch/of-dpa
[3] http://onie.org/
[4] https://github.com/bisdn