The Latest Trends in SDN

This is a discussion of the latest trends in SDN and related technologies, as well as the new Stratosphere Inc. product, Omnisphere.

3.1 Recent Trends

SDN*1 is now a widely known buzzword, and market activity is thriving in associated fields. In Table 1, I have summarized what are in my view some of the recent topics of interest. These relate to the OpenFlow and VXLAN component technologies.

I have also included news about virtual OpenStack and CloudStack, which are a variety of software called orchestrators that collectively control hypervisors and associated components for managing virtual machines. When using migration functions you must construct the same network environment before and after the migration of virtual machines, but orchestrators use OpenFlow to coordinate this. The news that VMware had acquired Nicira, the de facto originator of OpenFlow, was particularly impactful.

In other noteworthy news, Google's SDN WAN announcement indicated they are putting OpenFlow-based network control to practical use on a global scale. Showing a different aspect to its use in orchestrators, the focus is on network traffic control. The integrated operation of networks controlled by conventional means such as BGP with networks controlled using OpenFlow was demonstrated. We also got a glimpse of Google's network operation, which they seldom discuss, leading to it being widely discussed in the industry.

Table 1: Recent SDN-Related News

Date	Topic
2012 February 7	Nicira emerged from stealth
February 29	Open vSwitch 1.4.0 Fedora packaged
March 10	Home Rack Workshop #2.5 in Akihabara held
March 21	Linux 3.3 including Open vSwitch kernel module released
April 16	CloudStack entered the Apache Incubator program
June	Google announced SDN WAN
June 25	OpenFlow 1.3 published
July 23	VMware acquired Nicira
September 6	OpenFlow 1.3.1 published
September 27	OpenStack Folsom released
September	Trema-edge forked
October 31	SSP 1.0 released
November 30	SSP 1.0.2 released
December 10	Linux 3.7 including VXLAN kernel module released
December 19	Omnisphere project launched
2013 February 5	JR East announced OpenFlow to be used in station facilities. Wireless LAN also supported.
February 13	SSP 1.0.3 released
March 20	Apache CloudStack became a top level project
March 29	SSP 1.1.1 released
April	OpenDaylight Project announced
April 3	Use of Trema-edge began
April 4	OpenStack Grizzly released
April 15	RDO (Red Hat) announced
April 17	Trema-openwrt announced
April 25	OpenFlow 1.3.2 published
May 28	Apache CloudStack 4.1 released
June 12	Interop Tokyo 2013 held - Omnisphere announced
June 25	Indigo Virtual Switch made open source (Big Switch Networks)
July 5	SSP 1.1.2 released
August 5	OpenFlow 1.4 published
September 17	SDN Japan 2013 held - Omnisphere 1.0 released
October 1	Apache CloudStack 4.2 released
October 17	OpenStack Havana released

: Topics involving Stratosphere

3.2 SDN in the Office Environment

Meanwhile, we were having problems with network operation in office environments. When laying network cables on office floors, we wire information outlets beneath the floor along with the power supply system, and implement centralized control via a switching hub on the floor rack. We allocate and operate different networks for each department, isolating networks using the VLAN switch function. For this reason, each time the office seating configuration changes due to staff reassignments, etc., the VLAN settings for floor information outlets need to be changed. This involved a heavy operational load, as well as significant burden associated with identifying the cause of and resolving issues such as storms originating from loops, etc. Problems are also starting to appear for wireless network operation. With the growing popularity of wireless devices, there have been issues with capacity exceeding initial specifications, such as a lack of addresses to assign using DHCP. We thought we might be able to resolve these issues using the SDN approach.

*1 https://www.opennetworking.org/ja/sdn-resources/sdn-definition

Until now, discussion of OpenFlow centered on orchestrators and SDN WAN. Orchestrators are mainly designed for data center environments in which clusters of machines are installed, and SDN WAN is for NOC environments that house backbone lines, etc., The OpenFlow system itself is not tailored to a given field, but due to continuity in the financial and research areas, large-scale networks were a thriving market for SDN. In contrast, Omnisphere is an effort to apply SDN to more familiar office environments.

3.3 Inside Omnisphere

Although it is possible that changes will be made in future implementations, here I will describe the inner workings of the current Omnisphere implementation. Figure 1 shows the structure of Omnisphere.

3.4 OpenFlow Switch

When implementing OpenFlow in an office environment, the first issue is whether you can obtain a reasonably-priced OpenFlow switch. Demonstrations using expensive OpenFlow switches are not compelling when this deviates significantly from the actual implementation you have in mind.

The OpenFlow project included a project called "Pantou for OpenWrt"*2 that uses the OpenFlow 1.0 reference implementation. OpenWrt*3 is a project that involves rewriting the firmware for commercially available routers so they can be used with generic Linux. It is still hard to get hold of dedicated OpenFlow switches now, but this seems to have been one of the OpenFlow switch solutions used since OpenFlow was first created.

It most likely became known to the Japanese community at "Home Rack Workshop #2.5 in Akihabara"*4. The ability to build a do-it-yourself OpenFlow hardware switch presented a challenge that appealed to hacker types. Home Rack Workshop #2.5 is an event that promotes the conversion of inexpensively obtainable Buffalo AirStations into OpenFlow-compatible switches to tinker around with them, and it was apparently quite popular. As an aside, one of the promoters of the group used this hardware to win an NEC award at the Interop Tokyo 2012 Special Open Router Competition*5. Amidst this, Stratosphere also began tests using the Buffalo AirStation. At the time, it was a natural transition to work on both wired and wireless LAN at the same time.

For the development of Omnisphere, we targeted OpenFlow 1.3 after comprehensively evaluating a number of factors. Specific examples include the fact that OpenFlow switches installed in office environments have less resources available to them than those envisioned for NOC installations, and the latest protocol that provided high performance with the smallest footprint at the time was preferred. Another factor was that the ONF (Open Networking Foundation) that develops OpenFlow had reached a consensus to promote the spread of OpenFlow 1.3. Additionally, no commercial OpenFlow switch supporting wireless LAN was available, so we would have to aim to support this almost from scratch. Trema had also just been forked to Trema-edge*6 with the goal of supporting OpenFlow 1.3, and development had progressed quite far.

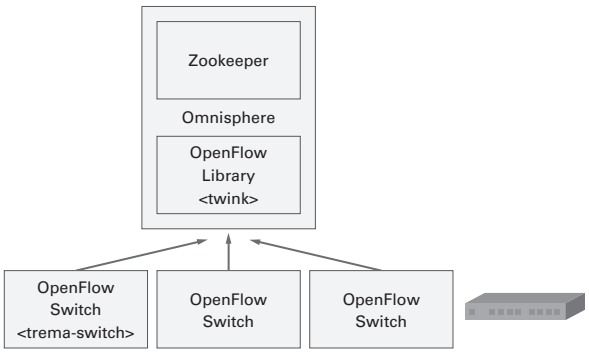

Figure 1: Omnisphere Framework

*2 http://archive.openflow.org/wk/index.php/Pantou_:_OpenFlow_1.0_for_OpenWRT
*3 https://openwrt.org/
*4 http://atnd.org/events/26147
*5 http://www.interop.jp/2012/orc/
*6 https://github.com/trema/trema-edge

Trema is known as a framework for OpenFlow controllers, and Trema-edge adds an OpenFlow 1.3 software switch written in C. Other switch implementations that support OpenFlow 1.3 include ofsoftswitch*7 and IVS*8. We decided to utilize the Trema-edge switch for development due to it using a comparatively simple implementation of OpenFlow 1.3. To run Trema-switch on OpenWrt, we created firmware using an OpenWrt feed. This feed is available on GitHub*9. We performed an actual demonstration using this switch implementation at Interop 2013.

3.5 OpenFlow Library

Because there was no framework that fulfilled our functional requirements for the OpenFlow protocol, we created our own. For OpenFlow application frameworks, it is crucial to be able to write flow rules easily and concisely. Having flow rules written concisely throughout a program contributes directly to the ease of maintaining it. In light of this, we made it possible to use the Open vSwitch ovs-ofctl description format.

It is common for OpenFlow libraries to integrate OpenFlow message analysis and setup with the TCP server function. When creating OpenFlow messages, it goes without saying that there is a high cost associated with learning to adhere to the library's unique conventions, and parts not supported by the library are out of reach. With the auxiliary connections introduced in OpenFlow 1.3 it is possible to have UDP sub-connections, but support for those with integrated server functions is difficult. In our created libraries, we have split off the I/O event loop and protocol analyzer parts to completely eliminate functions that translate the protocol binary format from the controller core.

During actual development and operation, we want to execute direct queries to the OpenFlow switch even if it is connected to the controller, so we included a controller that can relay queries such as these in the framework. We also made it possible to use connections from UNIX domain sockets as gateways for relays from external programs. Additionally, we created external outputs that monitor everything output from the OpenFlow switch.

Figure 2 shows a summary of the framework. When the controller communicates with multiple host applications, they are isolated using OpenFlow barrier messages. The flow rules written in the framework using Open vSwitch description take the ovs-ofctl argument, and are delivered to the OpenFlow switch via the controller. We based the placement of a monitoring port on Open vSwitch. It is also possible to chain multiple OpenFlow applications using relay ports.

Due to this configuration, we use the OpenFlow protocol binary format as-is for the core part of the library. A library that handles protocol binary formats is required when you want to directly create OpenFlow messages, so we created a separate one.

*7 http://cpqd.github.io/ofsoftswitch13/
*8 http://www.projectfloodlight.org/indigo-virtual-switch/
*9 https://github.com/iHiroakiKawai/trema-openwrt

3.6 ZooKeeper

SDN is an architecture that separates the network control part and data communications part to make the control part more programmable. A more detailed description can be found in Google's Onix announcement*10. Separating the control part means it is possible to use general-purpose servers, which continue to improve in performance. It also dramatically increases the freedom with which control part logic can be expanded compared to a unified solution. Use of general-purpose servers as computational resources implies that distributed computing is carried out. I believe the fact this kind of distributed computing environment is now available is one of the factors that brought about the establishment of SDN.

Omnisphere uses Apache ZooKeeper. ZooKeeper is a project derived from Hadoop, and is one of the core components of Hadoop 2.0. The popular Paxos agreement algorithm was used in Google's Onix, but ZooKeeper implements an equivalent algorithm called Zab.

ZooKeeper provides a tree structure maintained to allow use on distributed computers without inconsistencies. The tree structure can also be used directly, but it is not easy to use as-is, so ZooKeeper includes a class known as Recipe. The Recipe class uses this tree structure internally, and provides a convenient interface including queues and locks, as well as leader election algorithms. When used from an application, part of the tree structure is assigned to functions like this.

3.7 Summary

Omnisphere utilizes a freely usable implementation on the switch side, and is constructed using a modern software base on the controller side.

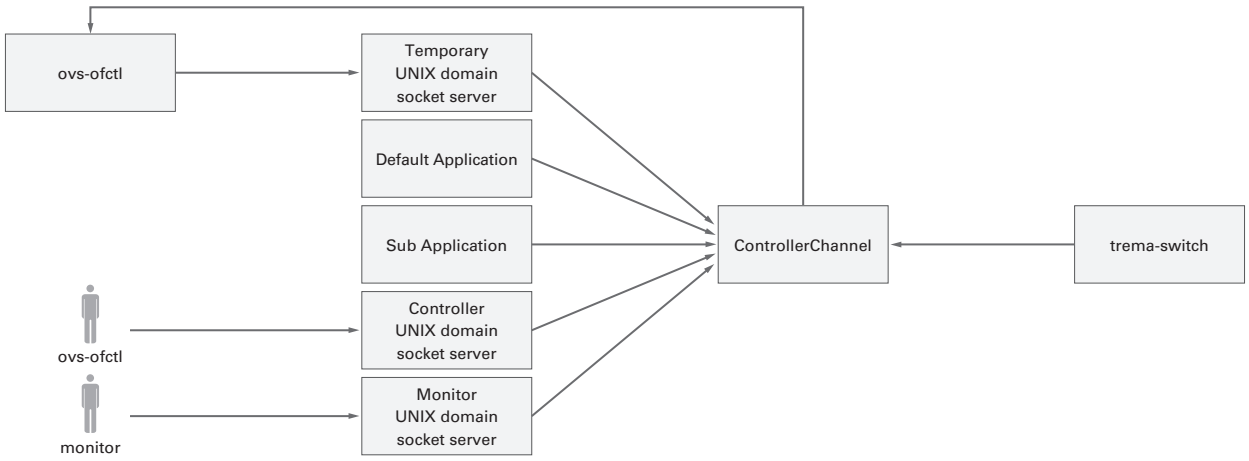

Figure 2: When OVS Description is Used in the Framework

Author:
Hiroaki Kawai
Stratosphere Inc. Mr. Kawai joined IIJ in 2003. He took part in the launch of i-revo in 2006. He lives in the Kansai area, and is an Apache fan who grew up on open source software.

*10 https://www.usenix.org/conference/osdi10/onix-distributed-control-platform-large-scale-production-networks