Overall Structure of Software-Defined Network with Controller Evalution : Opendaylight and Floodlight

Dohyun, Kwon
School of Computer Science and Engineering,
Chung-Ang University, Seoul, Republic of Korea
sugarcod@naver.com

Abstract — Software-defined network(SDN) is an emerging network architecture which is elastic, cost-effective, and adaptable. It has been receiving attention because of important role that manages continuously enlarging-scale and complex network structure. SDN can achieves this work by decoupling the control plane from data plane using centralized SDN controller. Currently, OpenFlow, the most receiving attention standard communication interface for SDN, is widely used to support the SDN architecture especially between SDN controller and SDN switches as southbound API. Thanks to flow table in OpenFlow switch, routers or switches just simply forward packets in data plane(or infrastructure layer) following the flow table rule. In this paper, I'll further elaborate the overall structure of SDN in detail and compare performance of SDN controllers(ODL and FL) regarding RTT, CPU utilization information, and average flow setup latency, and so forth with an equivalent network topology(tree structured with 27 hosts).

Keywords Software-defined networking · SDN · Opendaylight(ODL) · Floodlight(FL) · Performance evaluation

I. INTRODUCTION

In traditional network architecture, it widely used closed routers and switches. Although new technologies or methodologies that drastically improved network availability or stability were designed, experiments in large-scale verifying the system performance were impossibly conducted because of the equipments''closed' features. This is because routers and switches that consist the core of internet are totally closed, so testing new software on the equipments were sensationally blocked[1]. Also, vendor lock-in feature made obstacles so users should hesitate considering category of manufacturer before they update or install another function.

Today, the OpenFlow is the de facto standard of SDN, it was firstly proposed by a research team of Cleanslate of Stanford University. Originally, it was researched for controlling future internet infrastructure to deal with challenging issues on the internet. Thanks to widespread acceptance of OpenFlow by academia and industry, this SDN standard made SDN much more successful. In industry, many commercially-oriented SDN-enabled networks have been deployed, such as Google's backbone network[4]. In academia, the top international conference, SIGCOMM, has progressed a workshop named hotSDN, which calls for papers that report the latest achievements on SDN. In addition, many well-known universities, such as Princeton and Stanford, have also researched issues related to OpenFlow/SDN which involves routing decisions' optimization, programmable wireless networks, energy efficient datacenter network[4].

Recently, the demand of cloud service is highly increased and the advent of fog computing and improvement of IoT technology made cloud service provider to consider for better various network requirements such as QoS(Quality of Service), availability, bandwidth, and latency. Under these circumstances, SDN was emerged to satisfy these requirements[5]. This new model is simply novel, flexible, and revolutionary network architecture that can guarantee the scalability.

In general, SDN has three-layered structure : application layer, control layer, and infrastructure layer. The entire operation of SDN network architecture is as follows. User manipulate the ap-

plication which is in the application layer. Through the northbound API, which is between the application layer and the control layer, instructions from the user are delivered to SDN controller which is in the control layer. And then, the SDN controller control the virtualized OpenFlow switches in the infrastructure layer through the southbound API(or OpenFlow) which is between the control layer and the infrastructure layer obeying the OpenFlow protocols[5].

As briefly elaborated before, the data plane and control plane, which are relevant to SDN controller and virtualized SDN switches respectively, are distinguished in this architecture. So, we can directly program the SDN controller because it is decoupled from forwarding functions. Also, the network intelligence of this model is logically centralized in software-based SDN controllers that maintain a global view of the network, which appears to applications as a single, logical switch. In addition, SDN simplifies network design and operation because instructions are provided by SDN controllers instead of multiple, vendor-specific devices and protocols(vendor-neutral)[2].

Beside the research regarding the SDN and its standard interface, there is also a research direction that has not attracted much attention, which is the security of SDN[4]. Abusing the logically centralized controller, security challenges on the controller may occur and make serious problems and can affect the SDN switches in infrastructure layer. Also, the attacker can get a global view of the network and control it arbitrarily that can make the SDN system to be down. At the same time, other security challenges on each API and other layers in SDN architecture can occur whenever the attacker wants to invade the system. So, SDN system designer must consider countermeasures regarding these security issues. This issue will be covered in further work.

The rest of paper is organized as follows : section II presents overall architecture of SDN. And section III introduces experiment circumstances regarding performance evaluation of ODL and FL with various items. Next, section IV covers various results of performance evaluation about ODL and FL. Finally, section V concludes the paper by summing up previous issues and introduces future work.

II. Overall Architecture of SDN

II-1. Configuration of SDN

An SDN network is composed of infrastructure layer(SDN switches), control layer(SDN controller or Network OS), and application(Fig 1.) layer. We can say that the SDN controller is the core part of this network system because the entire flow rules are defined by this component and applied to the flow table of OpenFlow switch which is in infrastructure layer. So to speak, the flow in infrastructure layer generated by tremendously many user equipments such as cellular devices, PCs, and D2D communication devices can just flow by the rules in the table. In other words, the rules make this new network system more flexible and consistent than traditional system because the rules what the controller makes are not defined by hardware, but software. That is, the feature that controller is programmable is very powerful to manage and control the network[8]. The switches and routers interact with the controller all the time to handle various types of flow control in the network such as forwarding, routing, firewall operation, etc.

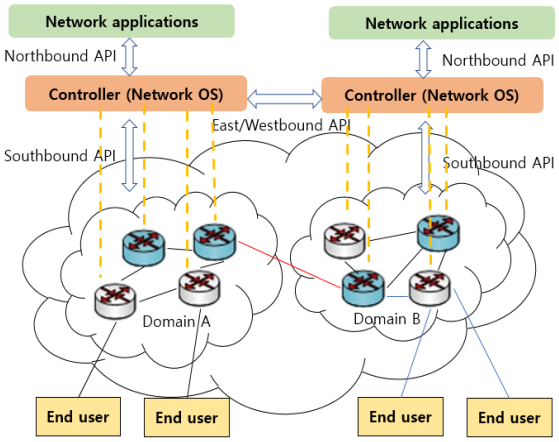


Fig 1. Overall structure of SDN network system

Above figure shows overall SDN structure from end users linked to the infrastructure layer to application layer. Blue switches are obeyed by BGP which are similar to gateway. Black-full lines between BGP switches and other normal switches in a domain stand for link information and red-full line between BGP switches is inter-domain link between different domains. Orange-dotted lines between controller and switches mean that the sw-

itches are controlled by the controller obeying the OpenFlow protocol which is in the flow table in it. If there are some needed additional information regarding forward frames or packets in network, administrator of the network can just programs the information to apply the rule to the system. In other words, he or she can adds, deletes, or modifies the forwarding rule in flow tables by programming the controller.

II-2. Structure of Flow Table

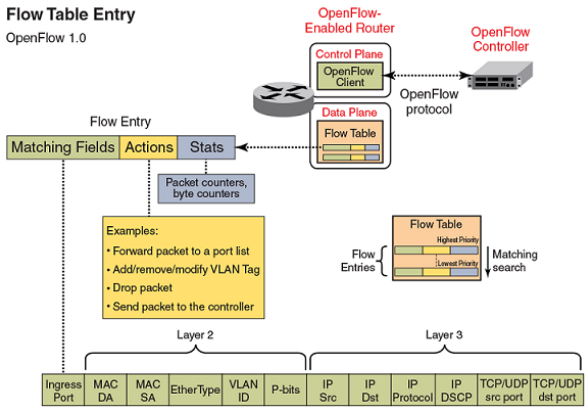


Fig 2. Configuration of flow table

The flow table in Fig 2.[7] shows whole configuration of it section by section. Each entry shows particular operation : rule, action, and stats. First of all, the rule entry manages the packet which arrives in SDN router by confirming each element of this entry. And then, appropriate action relevent to the rule is operated. This procedure is possible by interacting between control plane and data plane in OpenFlow-enabled router(SDN router). The control plane of the router uses OpenFlow client to interact with controller above its layer. And information regarding data control is applied to the flow table in data plane by the OpenFlow client. To sum up, frames or packets arrived in the switches or routers are handled by matching search among flow entries in the flow table respectively.

III. Experiment Circumstances

Up to now, it is impossible to overstate the importance of control layer in SDN system. It is not exaggerating the role of controller. So, It is worth showing each controller performance for choosing better SDN system. Now, I present experiment cir cumstances regarding ODL and FL on OFNet, which is a network simulator run on Ubuntu.

We can simulate various situations on any topology that we define by using OFNet[9]. In my case, I defined wired network with (3, 3, 3) tree topology : 3-level(without central switch), 3 siblings for each switch, 3-leaves for each parent of hosts(End users). And I made traffics between host and host under different switches. That is, this experiment is for layer-2 in OSI 7 layers, which is data link layer. Thus, in this topology, each host send frame, not packets. To sum up, there exists 13 switches and 27 hosts in this SDN obeying the OpenFlow protocol. The Ubuntu was operated on a single virtual machine(VirtualBox) with a single core and 1GB RAM. Entire experiment circumstance informations are presented in detail as following table :

OS	Ubuntu® 12.04 LTS
CPU(single)	Intel® Core™ i7-2670QM 2.2GHz
RAM	1GB
Tool	OFNet, DLUX
Controller	ODL, FL
Topology	Tree (3, 3, 3)
Bandwidth	1,000Mbps

Table 1. Circumstances of Topology

Based on this setting, various comparison between ODL and FL including :
1. Average flow setup latency
2. CPU utilization(relevant to power consumption)
3. Flow misses to controller
4. Average RTT(ms)

will be performed in section IV.

IV. Performance evaluation

Before I present the whole results of experiments, this section consists of topology generated by each SDN controller GUI tool such as DLUX[9] and four experiment results what I mentioned before in former section with detailed explanation. At first, figures regarding SDN topology generated by each GUI tool will be shown. And then, rest of this section will cover the results of experiments under the same network condition such as bandwidth of links and frame size, etc.

IV-1. Network Topology of Controllers

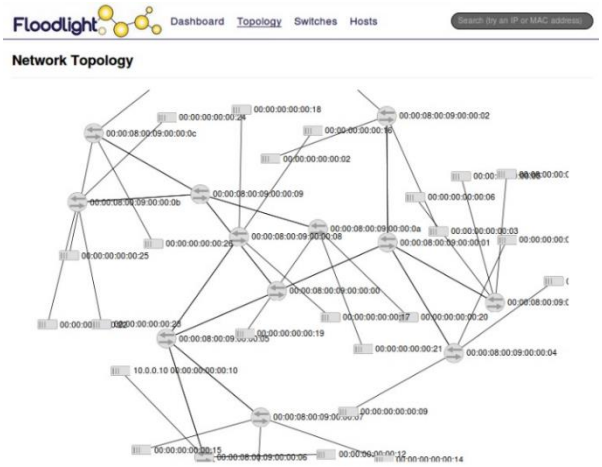


Fig 3-1. Topology shown by Floodlight GUI tool

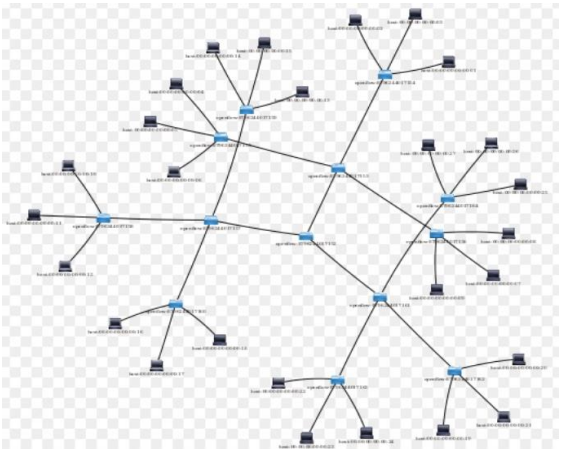


Fig 3-2. Topology shown by ODL GUI tool(DLUX)

Fig 3-1 and 3-2 show the tree topology (3, 3, 3). Each host has virtual MAC address from 10.0.0.1 to 10.0.0.27 and other switches have MAC address as well. In this structure, we can make situation such as host 1 sends a frame to host 24 using following command :

$ hsh gh1 ping -c 1 10.0.0.24

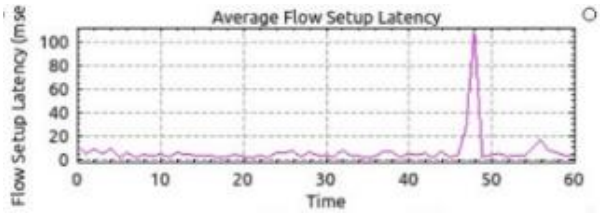
Or, each host can send other hosts by using a command like :

$ pingall

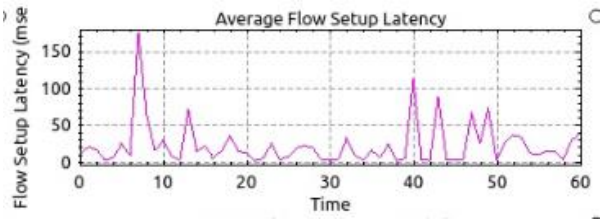
I entered a chain of instructions including above two commands with gh2 to gh14, gh1 to gh10, gh1 to gh10. And then, I entered following com mand to confirm various information and analyzed each graph as IV-2 to IV-5 :

$ trafficup

IV-2. Average Flow Setup Latency



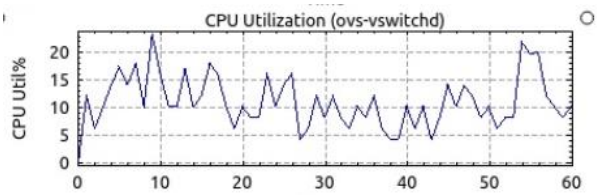
Graph 1-1. ODL average flow setup latency(ms)



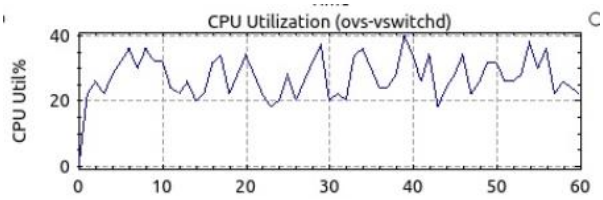
Graph 1-2. FL average flow setup latency(ms)

As shown in above Graph 1-1 and Graph 1-2, the average flow setup latency was measured during a minute. We can figure out the FL's one is more fluctuating than ODL's almost every time slot. Also, the value of the latency in Graph 1-2 is more higher than Graph 1-1's one in the greater part the time-axis. That is, the FL needs more time to set up flow than ODL.

IV-3. CPU Utilization



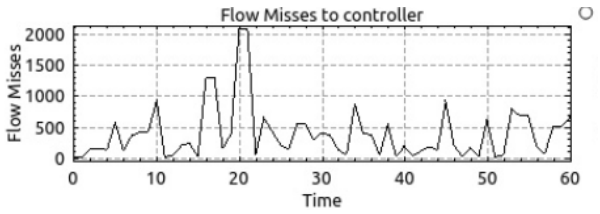
Graph 2-1. ODL CPU Utilization(%)



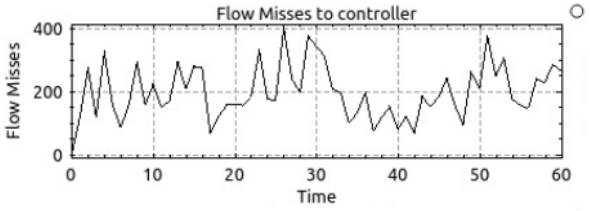
Graph 2-2. FL CPU Utilization(%)

The CPU utilization is relevant to power consumption which is a key factor for wireless sensor network to evaluate whether the configuration of the network is appropriate or not. So, suppose that if this system is applied to such network, it is impossible to overstate the importance of this evalution information. As same as IV-2, CPU utilization was measured during a minute using ODL and FL. The average CPU utilization of FL was approximately 30%. On the other hand, the average CPU utilization of ODL was 13%. Considering we simulated the same circumstances for each controller, we can say that ODL is more energy-efficient controller.

IV-4. Flow Misses to Controller



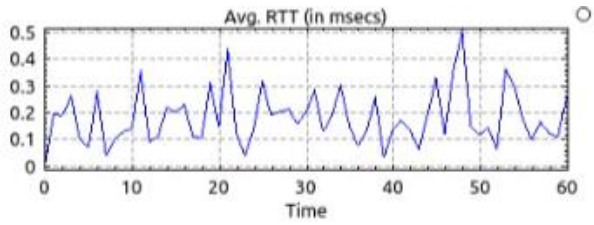
Graph 3-1. ODL flow misses to controller



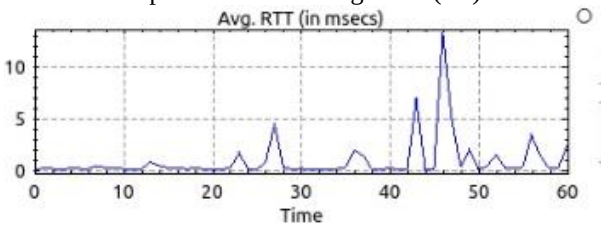
Graph 3-2. FL flow misses to controller

Above graphs show flow misses to each controller during a minute. In this experiment, FL's rate of flow miss to controller was lower than ODL's one. That is, OF messages from the switches to controller was more well-delivered to FL than ODL. Flow misses which occurred from switches to controller may cause fatal problems because flow entries in flow table which is in each OF switch probably need to be modified or added to forward frames properly. So, these flow transfers from switches to controller stands for reliability of performance of switch in terms of forwarding frames appropriately. According to graph 3-1 and 3-2, the average flow misses of ODL SDN system was almost triple compared to FL's one.

IV-5. Average RTT



Graph 4-1. ODL average RTT(ms)



Graph 4-2. FL average RTT(ms)

Finally, comparing the average RTT of ODL and FL, almost every value of average RTT was similar except some time slots. So to speak, except these time slot, the two controller showed alike performance in terms of time consumption with same topology and a chain of instructions.

V. Conclusion and Future Work

Enfin, I introduced overall structure of SDN and elements organizing each of three layers. In addition, I conducted four experiments to evaluate performance of two famous SDN controllers(ODL, FL) which are core of the SDN system. In terms of setup latency and CPU utilization relative to power consumption or energy-efficient operation, the ODL showed superior performance compared to FL. However, flow to controller, which includes request for adding or updating the flow entries in flow table, were missed much more in ODL than FL. This may cause serious problems to SDN system because the OpenFlow switches need to know exact information about forwarding in real time. So, we can say that overfull flow misses to controller in ODL compared to FL are more serious flaw despite of time average shorter setup latency and time average less CPU consumption.

In this paper, I conducted experiments regarding two SDN controllers performance within data link layer with 13 switches and 27 hosts in a single virtual machine. In future, I will carry out a research

regarding flow handling in east/westbound API between controllers. Also, communication between switches obeying BGP which are involved in separate domain and both security issues may occur in SDN system and their countermeasures will be researched as well.

REFERENCE

[1] Jarschel, Michael, et al. "Interfaces, attributes, and use cases: A compass for SDN." *IEEE Communications Magazine* 52.6 (2014): 210-217.

[2] Software-efined Networking (SDN) Definition. [Online]. Available : https://www.opennetworking.org/sdn-resources/sdn-definition

[3] T.S Choi,et al. “Trends and Forecast of SDN Technology rossing Peak of Hype Cycle.” Electronics ad Telecommunications Trends (2014), ETRI

[4] Shu, Zhaogang, et al. "Security in software-defined networking: Threats and countermeasures." *Mobile Networks and Applications* 21.5 (2016): 764-776.

[5] Hu, Fei, Qi Hao, and Ke Bao. "A survey on software-defined network and openflow: From concept to implementation." *IEEE Communications Surveys & Tutorials* 16.4 (2014): 2181-2206.

[6] Kim, Hyojoon, and Nick Feamster. "Improving network management with software defined networking." *IEEE Communications Magzine* 51.2 (2013): 114-119.

[7] Overview of OpenFlow. [Online]. Avaliable : http://www.brocade.com/content/html/en/configuration-guide/FI_08030_SDN/GUID-F1BFA637- EB68447887A3269CF5E1583F.html

[8] Feamster, Nick, Jennifer Rexford, and Ellen Zegura. "The road to SDN: an intellectual history of programmable networks." *ACM SIGCOMM Computer Communication Review* 44.2 (2014): 87-98.

[9] OFNet SDN network emulator. [Online]. Available : http://www.brianlinkletter.com/ofnet-a-new-sdn-network-emulator/