Improvement of Data Transmission Speed and Fault Tolerance over Software Defined Networking

SM Shamim
Dept. of Information and Communication and Technology
Mawlana Bhashani Science and Technology University
Dhaka, Bangladesh

Mohammad Badrul Alam Miah
Dept. of Information and Communication and Technology
Mawlana Bhashani Science and Technology University
Dhaka, Bangladesh

Nazrul Islam
Dept. of Information and Communication and Technology
Mawlana Bhashani Science and Technology University
Dhaka, Bangladesh

***Abstract*—Software Defined Networking (SDN) is a new networking paradigm where control plane is decoupled from the forwarding plane. Nowadays, for the development of information technology large number of data traffic has been added in the global network each day. Due to proliferation of the Internet, e-commerce, video content and personalized cloud-based services higher channel bandwidth required to deliver larger data from one center to others. Lower data communication speed and fault tolerance are major factors for SDN which degrades network performance. This paper presents enhancement of data communication speed and fault tolerance over SDN using Link Aggregation Control Protocol (LACP. The result of this paper shows network performance has been improved by increasing approximately 31% data transmission speed over SDN using LACP. Moreover, this paper shows fault tolerance have been improved by LACP which prevents failure of any single component link from leading to breakdown of the entire communications.**

Keywords—Fault Tolerance; Link Aggregation Control Protocol (LACP); OpenFlow; Mininet Emulator; Software Defined Networking (SDN)

I. Introduction

Software Defined Networking (SDN) [1, 2, 3] is a new approach for managing, building and designing computer networks which decouple the network's control plane from the forwarding planes. It has emerged as new paradigm in networking which has the possibility to enable ongoing network innovation and enable the network as a programmable, pluggable component of the larger cloud architecture [4]. SDN is being strongly considered as the next promising networking platform. In recent years, SDN has been developing tremendously in different organizations [5]. In order to reduce operational costs and strengthen network architecture different companies are planning or deploying SDN in their network [6]. In the next five years, SDN will be considered one of the most advanced information technologies over the world [7, 8]. About US $2 billion has been estimated to invest in SDN for knowledge discovery [9].

In order to handle the larger data high configuration router and switch are needed. Server and storage resources are interconnected via switches and routers [10]. Adding more switch and router will increase operational cost which reduces the network performance. In addition, network path failure one of the major problem which reduce network efficiency. An efficient routing, sever load balancing, access control and traffic monitoring system has to be designed to overcome these limitation. One of the possible solutions of these problems is Link Aggregation control protocol where two or more ports in an Ethernet switch are combined together to operate as a single virtual port. It increases available bandwidth by aggregating two or more links between network devices.

Due to programmability of Software Defined Networking, standard mechanisms needed for achieving higher data transmission speed and fault tolerance. Different researchers propose few techniques [11, 12, 13, 14 and 15] to improve data transmission speed and fault tolerance over Software Defined Networking. In this consequence, the paper [11] analyzed Bandwidth and latency aware routing using OpenFlow over SDN which improve network performance. Paper [12] proposed a novel architecture BRAS (Broadband Remote Access Server) which could enhance data transmission speed according to users' preference in specific applications.

In [13] authors presents Software Defined Networking based on OpenFlow can be used to build efficient solutions in order to handle fault-tolerant multicast in substation environments. Their implementation handles single link failure and also indicates how their approach can be expanded to handle multiple link or node faults. Fault tolerance issue with systematically review the existing methods has been proposed in [14] which are useful in failure recovery. In [15] proposes new architecture to strengthen the reliability and fault-tolerance over SDN in terms of network operations and management. After studying related works realized that several researchers improve data transmission speed and fault tolerance over SDN separately with different technique.

However, there needed further studies to improve data transmission speed and fault tolerance over SDN in order to increase network performance. Yet, there has been lack of studies, which can enhance both of these two major facts at same time. Though Link Aggregation control protocol is known for traditional network architecture, no implementation has been done over Software Defined Networking. Link Aggregation [16] is a technology defined in IEEE802.1AX-2008, which is a method of combining multiple physical lines to be used as a logical link. It increases capacity and availability of the network between specific devices (both switches and end stations) with the help existing Fast Ethernet and Gigabit Ethernet technology. It has higher potential transmission speed and higher accessibility in contrast to conventional connections using an individual cable. The purpose of this paper is to enhance data communication speed and fault tolerance over SDN at same time using Link Aggregation control protocol (LACP).

The rest of the paper is organized as follows. Section II detail describes Link Aggregation with the types in background. Section III describes the research methodology. Section IV presents the experiment setup in details. Experimental results and discussion are evaluated in Section V. Section VI concludes the paper and deliberates the future perspectives.

II. Background Study

In Link Aggregation, multiple parallel physical links are combined together between two devices in order to form single logical link. This function also provides load balancing where the processing and communications activity distributed over multiple links to avoid single link overwhelmed. The architecture diagram of Link Aggregation function is shown on Fig-1.

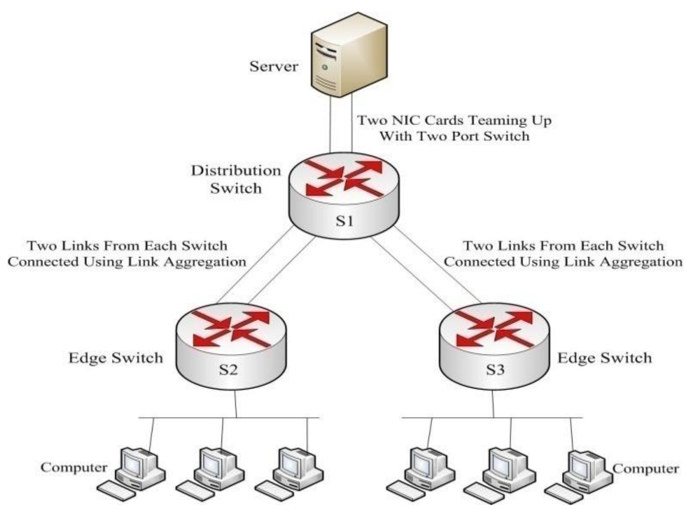

Fig. 1. Link Aggregation Architecture Diagram [17]

Two Edges switch are connected to distribution switch and distribution switch connected to server using link aggregation. In order to form a higher data transmission capacity several link from server may connect with different switch ports. Link Aggregation provides higher link availability, increase link capacity and arrogates replace upgrading over conventional network [17].

III. Research Methodology

A literature review has been performed in Software Defined Networking research scope and challenges. After the review, fault tolerance and data communication speed option arrived over SDN. A handy simulation tool was needed to analyze the SDN. Different experimental studies have been performed among OMNET++ [18], EstiNet [19], OFNet [20], Maxinet [21], NS-3 [22] and Mininet [23, 24]. The studies appear that open-source network simulator Mininet has good potential for simulating Software Defined Networking. In order to evaluate designed network topology with OpenFlow virtual switch Mininet has been installed over Ubuntu 14.04. Its installation and configuration is easy and straightforward than other simulators. Virtual Software Defined Networking can be designed using Mininet which consists of OpenFlow [25] controller, OpenFlow-enabled Ethernet switches and multiple hosts connected to those switches. OpenFlow is one of the most widely deployed SDN communications standards protocols. This protocol used in order to communicate between controller and other networking devices i.e. switch, router etc. A component based Software Defined Networking framework Ryu has been used as OpenFlow controller. Ryu Controller managed and maintained by open Ryu which is written in Python [26]. After careful study of the experiment different network analysis graph has been plotted which shows the expected results.

IV. Experiment Setup

Custom network topology has been designed using Mininet API which is shown on Fig-2. Designed network consist of one OpenFlow switches, an OpenFlow Ryu controller and three hosts. All the host h1, h2 and h3 are connected with the OpenFlow switch s1. Link Aggregation function has been implemented between OpenFlow switches s1 and host h1.

All of the hosts have assigned unique IP address and MAC address. The IP address and MAC address for host h1 are '10.0.0.1/24' and '00:00:00:00:00:01'. For all the other host corresponding IP and MAC address is also assigned i.e. host h2 ('IP=10.0.0.2/24' and MAC='00:00:00:00:00:02') , host h3 (IP='10.0.0.3/24' and MAC='00:00:00:00:00:03') , and host h4 (IP='10.0.0.4/24' and MAC='00:00:00:00:00:04').

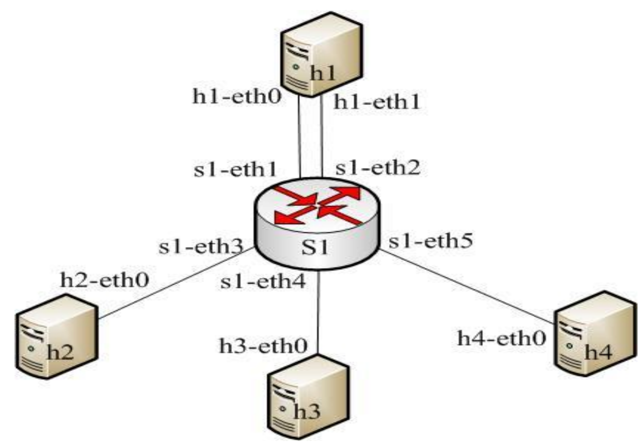

Fig. 2. Designed Network Topology

The Linux bonding driver [27] provides a method for combining more than one network interface controllers (NICs) into single logical bonded interface. Initially, bonding driver module has been loaded in host h1 to perform link aggregation. There are two interfaces in host h1 which are h1-eth0 and h1-eth1. These two interfaces are bond together in order to form one logical interface, i.e. bond0One of the commonly used network analysis tool iperf has been used to measure the performance. Network analysis tool Wireshark formerly known as Ethereal has been used which captures packets in real time and display in human-readable format.

Two scenarios have been executed to evaluate the performance where deigned network topology has been executed without LACP implementation and secondly topology executed with LACP implementation. The corresponding result of each execution has been captured by Wireshark. For each corresponding result three performance analysis graphs throughput graph, time sequence graph and round trip time graph has been drawn. Details comparison among these graphs has been shown in result section.

V. Experiment Result

A. *Throughput Graph*

In data communication network throughput refers to average rate of successful message delivery over a transmission channel which measured in bits per second or in data packets per second or data packets per time slot. Data may be delivered over a physical or logical link, or pass through a certain network node. Figure-3 shows throughput graph with implementation of LACP over SDN and Figure-4 shows another throughput graph for without implementation of LACP over SDN. Throughput graph is valuable in understanding end-to-end performance.

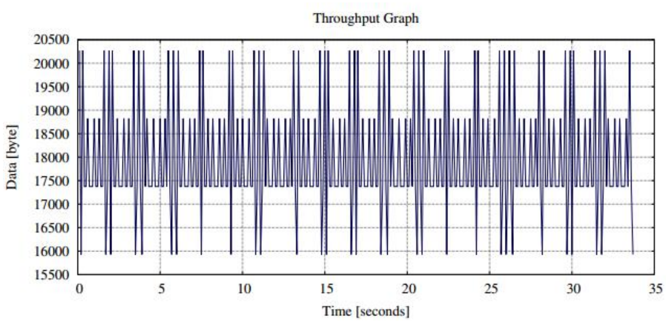

Fig. 3. With LACP SDN Throughput Graph

From the With LACP SDN throughput graph Fig-3, highest throughput in bytes approximately 20250 bytes and lowest is 16000 bytes. There are about 4202 packets has been transmitted between sender and receiver. The size of average packet and average Bytes per second are 108.765 and 125004.869.

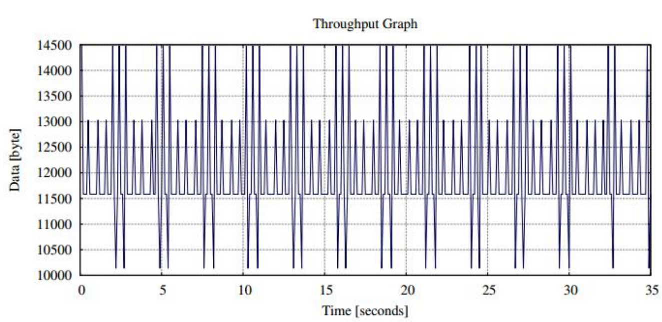

Fig. 4. Without LACP SDN Throughput Graph

From the Without LACP SDN throughput graph Fig-4, there are about 2896 packets has been exchanged between sender and receiver. The size of average packet and average Bytes in per second are approximately 82.699 and 82701.

Without LACP SDN throughput begins from the lower value approximately 10,100 after that it keeps on increasing and decreasing within some specific range. At the same time, the LACP SDN throughput graph begins with large value approximately 16,000. After some time it decreases sharply and again increase sharply within some specific range. A details comparison between two throughput graphs has been shown on table-1.

TABLE I. Comparison of Throughput Result between LACP SDN and without LACP SDN

Features	LACP SDN	Without LACP
Total Packets	4202	2896
Time Duration Between First	54.864	35.019
Avg. Packets/Sec (bytes)	108.765	82.699
Avg Packet Size	1506.591	1511.572
Bytes	6330694	4377512
Avg Bytes/Sec	163864.50	125004.87
Avg Mbit/Sec	1.311	1.00

From the Table-1, average megabyte per second for the without LACP SDN is 1.00MB and for the LACP SDN is 1.311 MB which is approximately 31% higher. The simulation graph and data table-1 are shown that LACP SDN has higher rates of throughputs than without LACP SDN. Data communication speed has been improved approximately 31% for LACP SDN compare to without LACP SDN.

B. *TCP Time Sequence Graph*

Time-Sequence graphs visualize TCP-based traffic. In an ideal situation, the graph plots from the lower left corner to the upper right corner in a smooth diagonal line.

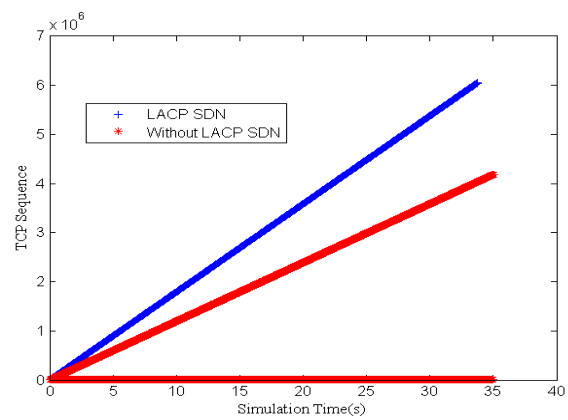

Fig. 5. TCP Time Sequence Graph

Fig-5 shows comparison of two time sequence graph where LACP SDN time sequence graph identified by blue lines and without LACP SDN time sequence graph indicated by red lines. The Y axis defines the TCP sequence numbers and X axis defines the simulation time. The slope of the line would be the theoretical bandwidth of the pipe. The steeper the line, the higher the throughput. From the Fig-5 more packets has been transmitted for LACP SDN compare to without LACP SDN.

C. Round Trip Time Graph

Round-trip time (RTT) is the length of time takes a data packet to be sent plus length of time takes for acknowledgment to be received of that packet to be received. RTT Graph depicts round trip time from a data packet to corresponding ACK packet. The Y axis is created based on the highest round trip latency time. Latency times are calculated as the time between a TCP data packet and the related acknowledgment.

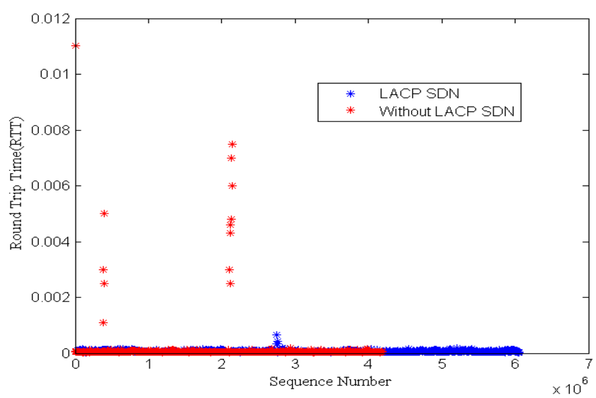

Fig. 6. TCP Round Trip Time Graph

The Y axis defines the Round Trip Time (RTT) in seconds and X axis defines TCP Sequence Numbers. Fig-6 shows latency times are very high at few points in the trace file and there are specific moments when the traffic is bursty in nature. For without LACP SDN RTT graph, some packet has higher latency time compare to average latency time. There are also some packets for LACP SDN RTT graph has higher latency time. Lower round trip time for the corresponding sequence number always expected. After comparing two graphs, LACP SDN RTT graph shows better output which has less round trip time than without LACP SDN RTT graph.

D. Improvement of Fault Tolerance

Fault tolerance is the ability of the system to perform its function even in the presence of one of multiple link failures. Fault tolerance setup or configuration prevent computer network device from failing in the event of an unexpected problem or error among connected devices. It is one of the major problems for networking application which can be improved by using Link Aggregation function. Designed network topology in Fig-2, two aggregated link are available between host h1 and switch s1 where each side link has separated port numbers (h1-eth0, h1-eth1, s1-eth0, and s1-eth1). All of the host (h2, h3 ad h4) can communicate with the host h1 using either port s1-eth1 or port h1-eth0. Each of the host can also communicate by using port s1-eth2 in switch and port h1-eth1 in host h1. Now disable one communication channel from aggregated group where disabled channel port number h1-eth0 which is opposite interface of s1-eth1.

After separating one channel, test the network connectivity by using ping command which sends ICMP echo request message and wait for corresponding reply between defined nodes. Now check connectivity between host h2 and host h1 and captured the corresponding result using Wireshark. The result shows in Fig-7, host h2 still communicate with host h1 using port s1-eth2 and port h1-eth1.

No.	Time	Source	Destination	Protocol	Length	Info
19	3.632590	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) request
20	3.632635	10.0.0.1	10.0.0.2	ICMP	98	Echo (ping) reply
26	4.631424	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) request
27	4.631459	10.0.0.1	10.0.0.2	ICMP	98	Echo (ping) reply
30	5.631364	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) request
31	5.631395	10.0.0.1	10.0.0.2	ICMP	98	Echo (ping) reply
32	6.631437	10.0.0.2	10.0.0.1	ICMP	98	Echo (ping) request
33	6.631469	10.0.0.1	10.0.0.2	ICMP	98	Echo (ping) reply

Fig. 7. Ping Result from host h2 to host h1

If we remove port h1-eth1 from the aggregation, h2 will still communicate to the host h1 using port s1-eth1 in the switch and port h1-eth0 in the host h1. Instead of failure occurs one links, Link Aggregation function able to check and automatically recover the communication using other links.

VI. Conclusion

This paper presents improvement of data transmission speed and fault tolerance over Software Defined Networking. The result obtained after extensive simulation study which has evaluated by TCP time sequence graph, throughput graph and round trip time graph. Throughput graph shows data communication speed has improved approximately 31% over SDN by using Link Aggregation control protocol. Simulation result in TCP time sequence graph and RTT graph shows network performance also improved for LACP SDN. LACP ensure failure safety systems which are crucial for every network administrator. The automatic configuration protocol LACP provides redundancy with dynamic switching to the standby link in case the active link fails. Moreover, it can be implemented in SDN using existing hardware which decreases the operational cost for upgrading the performance and resiliency of a system. Our future works involves improvement of fault tolerance and data transmission speed over Software Defined Wireless Networking (SDWN).

References

[1] B. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, T. Turletti, et al., “A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks,” *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 3, pp. 1617–1634, 2014.

[2] Y. Jarraya, T. Madi, and M. Debbabi, “A Survey and A Layered Taxonomy of Software Defined Networking,” *Communications Surveys & Tutorials, IEEE*, vol. 16, no. 4, pp. 1955–1980, 2014.

[3] D. Kreutz, F. M. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-Defined Networking: A Comprehensive Survey,” *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[4] M. Kobayashi, S. Seetharaman, G. Parulkar, G. Appenzeller, J. Little, J. Van Reijendam, P. Weissmann, and N. McKeown, “Maturing Of OpenFlow and Software-Defined Networking through Deployments,” *Computer Networks*, vol. 61, pp. 151–175, 2014.

[5] B. Leng, L. Huang, X. Wang, H. Xu, and Y. Zhang, “A Mechanism For Reducing Flow Tables in Software Defined Network,” *IEEE International Conference on Communications (ICC)*, pp. 5302–5307, IEEE, 2015.

[6] C. Yoon, T. Park, S. Lee, H. Kang, S. Shin, and Z. Zhang, "Enabling Security Functions With SDN: A Feasibility Study," *Computer Networks*, 2015.

[7] "Network world, gartner: 10 critical it trends for the next five years." Online Available: http://www.networkworld-.com/news/2012/102212gartner-trends-263594.html. Accessed: 2015-11-20.

[8] M.t. review, 10 emerging technologies: Tr10: software-defined networking." Online Available: http://www2.technologyreview.com/article/412194/tr10software-defined-networking. Accessed: 2015-11-20.

[9] "Enterprise networking, IDC: SDN a 2 billion market by 2016," Online Available: http://www.enterprisen-etworkingplanet.com/datacenter/-idcsdn-a-2-billion-market-by-2016.html. Accessed: 2016-10-25.

[10] TBitar, N., Gringeri, S., & Xia, T. J. "Technologies and protocols for data center and cloud networking". *IEEE Communications Magazine*, 51(9), 24-31. (2013).

[11] Pongsakorn, U., Kohei, I., Putchong, U., Susumu, D., & Hirotake, A. (2014). "Designing of SDN-Assisted Bandwidth and Latency Aware Route Allocation". *Journal of Information Processing Society of Japan*, 2014(2), 1-7.

[12] Chicago Li, K., Guo, W., Zhang, W., Wen, Y., Li, C., & Hu, W. (2014, May). "QoE-based bandwidth allocation with SDN in FTTH networks". *In 2014 IEEE Network Operations and Management Symposium (NOMS)* (pp. 1-8). IEEE.

[13] Pfeiffenberger, T., Du, J. L., Arruda, P. B., & Anzaloni, A. (2015, July). "Reliable and flexible communications for power systems: fault-tolerant multicast with SDN/OpenFlow". *7th International Conference on New Technologies, Mobility and Security (NTMS) in 2015* (pp. 1-6). IEEE.

[14] Chen, J., Chen, J., Xu, F., Yin, M., & Zhang, W. (2015, November). "When Software Defined Networks Meet Fault Tolerance: A Survey". *In International Conference on Algorithms and Architectures for Parallel Processing (pp. 351-368). Springer International Publishing*.

[15] Kim, D., & Gil, J. M. (2015). "Reliable and Fault-Tolerant Software-Defined Network Operations Scheme for Remote 3D Printing". *Journal of Electronic Materials*, 44(3), 804-814.

[16] "Link Aggregation Control Protocol (LACP) (802.3ad) for Gigabit Interfaces." Online Available: http://www.cisco.com/c/en/us/td/docs/ios/12_2sb/feature/guide/gigeth.html. Accessed: 2016 − 08 − 27.

[17] "link aggregation according to IEEE standard 802.3ad online." Online Available:https://www.google.com.-bd/url?sa=trct=jq=esrc=ssource=webcd=3cad=rjauact=8ved=0ahUKEwjEk5GQ7aJ AhV XCI4KHR7pAjUQF gg3MAIurl = http, Accessed: 2015 − 11 − 20.

[18] "OMNeT++ Discrete Event Simulator" Online Available: https://omnetpp.org/, Accessed: 2016-09-25

[19] "EstiNet 9.0 | Simulator", Online Available: http://www.estinet.com/ns/?page_id=21140, Accessed: 2016-09-30

[20] "OFNet- Quick User Guide "Online Available: http://sdninsights.org/, Accessed: 2016-10-19

[21] "MaxiNet: Distributed Network Emulation", Online Available: https://maxinet.github.io/, Accessed: 2016-11-08

[22] " NS-3", Online Available: https://www.nsnam.org/, Accessed: 2016-11-08

[23] "Mininet: An Instant Virtual Network on your Laptop (or other PC) - Mininet" Online Available: http://mininet.org/. Accessed: 2016-06-15.

[24] F. Keti and S. Askar, "Emulation of Software Defined Networks Using Mininet in Different Simulation Environments," *6th International Conference on in Intelligent Systems, Modeling and Simulation (ISMS)*, 2015, pp. 205–210, IEEE, 2015.

[25] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[26] "osrg/ryu . github." Online Available: https://github-.com/osrg/ryu. Accessed: 2015-11-01

[27] Bonding, Online Available: http://www.linuxfoundation-.org/collaborate/workgroups/-networking/bonding, Accessed: 2016-06-15.