UNIVERSITI PUTRA MALAYSIA

OFDPv2: AN EFFICIENT PROTOCOL FOR TOPOLOGY DISCOVERY IN OPENFLOW-BASED SOFTWARE DEFINED NETWORKING

ABUBAKAR MUSA ALKALI TANKO

FSKTM 2017 9

OFDPv2: AN EFFICIENT PROTOCOL FOR TOPOLOGY DISCOVERY IN OPENFLOW-BASED SOFTWARE DEFINED NETWORKING

By

ABUBAKAR MUSA ALKALI TANKO

Thesis Submitted to the School of Graduate Studies, Universiti Putra Malaysia, in Fulfillments of the Requirements for the Degree of Master of Computer Science

July 2017

COPYRIGHT

DEDICATIONS

This project is dedicated to my late parents, family for their love and endless patience and support throughout my life.

ABSTRACT

OFDPv2: AN EFFICIENT PROTOCOL FOR TOPOLOGY DISCOVERY IN OPENFLOW-BASED SOFTWARE DEFINED NETWORKING

By

ABUBAKAR MUSA ALKALI TANKO

July 2017

Supervisor : Azizol Bin Hj Abdullah, PhD

Faculty : Computer Science and Information Technology

Software Defined Networking (SDN) is a new networking pattern, with great possibilities to increase network efficiency, ease the complexity of network control and management, and speed up the degree of technological novelty. Earliest, one of the core concepts of SDN is the separation of the network's control and data plane. The intellect and the control of the network operation and management, such as routing, are removed from the forwarding devices and are concentrated in a logically centralized component, called the SDN controller. In order for the controller to configure and manage the network, it needs to have up-to-date information about the network state, more especially its topology. Thus, topology discovery is a thoughtful section of any Software Defined Network Architecture. In this project, we evaluate the efficiency of OFDP, the current de facto standard approach implemented by the major SDN controller frameworks, and proposes an improved version with simple and practical modifications, which achieve a significantly improved efficiency and reduced control overhead. We have implemented our new topology discovery approach on the widely used POX controller platform, and have evaluated it for a range of network topologies

through experiments using the Mininet network emulator. Our results show that our proposed modifications achieve a significantly reduced the number of control messages and increase efficiency to a range of an up to minimum 67-80% maximum of the SDN controllers and switches whereas the number of LLDP *Packet-In* messages, the other type of topology discovery control messages, is unchanged. And also achieves bandwidth availability to a range of 50-80% over the state-of–art (OFDP) by remarkably reduced control traffic overhead on the SDN controller and also our proposed protocol is completely compatible with the OpenFlow standard while delivering identical discovery functionality.

ACKNOWLEDGEMENT

To Almighty Allah (SWT), I am thankful for the blessings and virtues, and for reconciling, strength, patience, courage, and determination he gave me to complete this work to the fullest, Alhamdulillah.

I would like to express my sincere gratitude and appreciation to my supervisor, Dr. Azizol Abdullah for his continuous support, advice, and interest. His guidance has helped me throughout my project and writing of this thesis. I would also like to thank my assessor, Dr. Abdullah Mohammed for his encouragement and interest the course of this study and all my life.

Finally, I must extend my sincere thanks to the TETFUND, especially Waziri Umaru Federal Polytechnic Birnin Kebbi, Kebbi State Nigeria for their support by sponsoring me in my study. Nevertheless, my gratitude to the Malaysian people in general for their perfect hospitability in their green land during my study period.

DECLARATION

I declare that the thesis is my original work except for quotation and citations which have been duly acknowledged. I also declare that it has not been previously, and is not concurrently, submitted for any other degree at Universiti Putra Malaysia or other institution.

Signature: _____ Date: _____

Name and Matric No: Abubakar Musa Alkali Tanko GS45922

APPROVAL SHEET

This thesis submitted to the faculty of Computer Science and Information Technology of University Putra Malaysia and has been accepted as partial fulfillment of the requirement for the degree of Master of Computer Science.

The member of the Supervisory Committee were as follows:

Supervisor: Azizol Bin Hj Abdullah, PhD.

Department of Communication Technology and Network

Faculty of Computer Science and Information Technology

University Putra Malaysia

Date and Signature: _____

Assessor: Abdullah Mohammed, Ph.D.

Department of Communication Technology and Network

Faculty of Computer Science and Information Technology

University Putra Malaysia

Date and Sinagutre:_____

TABLE OF CONTENTS

CHAPTER ONE

INTRODUCTION

1.1 BACKGROUND

OpenFlow is a communications protocol that gives access to the forwarding plane of a network such as a switch or a router over the network. OpenFlow is a software-defined networking protocol that can be used to centrally control switches and traffic flows in a network. In a traditional IP network, switches handle both high-level routing (the control plane) and packet forwarding (the data plane).

Software Defined Networking (SDN): originated from OpenFlow, it is a centralized controller easy to program, change routing policies on the fly. The main goal of SDN is the networks to be open and programmable. If the organization require a specific type of network behavior, it can develop and start the application to do what they need. These applications may be a couple of network application functions. For example, traffic engineering, security, quality of service (QoS), routing, switching, virtualization, monitoring, load balancing, new innovations, etc.

SDN is not really a technological development in a classical sense but it is merely a way of organization network functionality. Thus, SDN is not a mechanism, it is a framework to solve a set of problems with many solutions. In general, SDN is not OpenFlow, since SDN is a concept of the physical separation of the network control plane from the forwarding plane, and where a control plane controls several devices (Raj, J. 2013).

1.2 Motivation

The Key Ideas of OpenFlow is Separation of control and data planes, Centralization of control, Flow-based control and takes advantage routing tables in Ethernet switches and routers. This can be resulting in achieving Control logic

is moved to a controller, Switches only have forwarding elements, One expensive controller with a lot of cheap switches and finally, OpenFlow is the protocol to send/receive forwarding rules from the controller to switches.

The SDN model comprises of Network Operating System which serves as a middleware and is commonly called SDN controller. It typically has core services as its job and provides an interface to the network node. Secondly is the forwarding devices which received the packet and take actions to this packet and update counters. The types of actions include dropping these packets, modifying packet header and sending the packet out to the single port or a multi-ports. In short, it handles how the packet originates with the SDN controller. Lastly, the Network Application which consists of many purposes, of course for the SDN they are on network focus.

In order for an SDN controller to be able to manage the network and to provide services such as routing, it needs to have up to date information about the network state, in particular, its topology. Thus, a reliable and efficient topology discovery mechanism is essential for any Software Defined Network. Earliest, the SDN controller does not need to discover the network switches since it is assumed that they will initiate a connection to the controller, and thereby announce their existence. As a result, it is worthwhile studying from OpenFlow-based SDN.

1.3 Problem Statement

The NOX controller puts a lot of overhead on the SDN controller due to the number of LLDP which is sent out by the controller. Sending an LLDP *Packet-Out* message for number ports on each switch seems inefficient. A better alternative would be to only send a single *Packet-Out* message to each switch regardless of how many numbers of ports available. And also ask

it to flood the corresponding LLDP packet out through all its ports. This functionality is supported within OpenFlow of the SDN.

The problem is that each LLDP packet needs to have the *Port* ID TLV initialized to the corresponding switch egress port. This is required so that the controller, upon receiving the LLDP packet through a *Packet-In* message from the receiving switch, can determine the source port of the discovered link. The current way to achieve this in OFDP is for the controller to prepare and send a dedicated LLDP packet through a separate *Packet-Out* message for each port of every switch.

Meanwhile, the initial switch-controller handshake informs the controller about the existence of the switches in the network, but what is missing for is, a complete topology discovery view that is, the information about the available links between switches in the network.

1.4 Project Objective

The objective is to improve a state-of-art (OFDP) discovery protocol by introducing a new topology discovery (OFDPv2) algorithm to efficiently discover the network topology. The goal of OFDPv2 is to reduce the overhead of the topology discovery mechanism by reducing the number of control messages that need to be sent by the controller.

The basic idea is simple. Instead of creating a unique LLDP packet for each port of each switch, and sending each such packet to the corresponding switch through a separate OpenFlow *Packet-In* message, as is the case in OFDP, we create and send only a single LLDP packet to each switch. We further provide instructions to the switch to forward the LLDP packet through each of its ports, after adding a unique port identifier which allows the receiving switch to identify the source port.

We propose a new version of the current SDN topology discovery protocol and call it OFDPv2. Later, we provide further technical details and discuss the implementation of two new protocol of this basic idea, which we refer to as *OFDPv2-A* and *OFDPv2-B*.

The final result of the proposed protocol is expected to meet the following outcomes

1. Reducing the number of *Packet_Out* message sent by the Controller while minimizing the control traffic overhead on SDN controller.
2. Reducing control traffic overhead while increasing efficiency of having available bandwidth usage for each switch on the network.

1.5 Project Scope

The scope of this project is to propose a reliable and efficient topology discovery protocol in OpenFlow-based Software Defined Network. The proposed topology discovery protocol is an improved version of OpenFlow Discovery Protocol (OFDP) the current de facto standard. The proposed new version of the current SDN topology discovery protocol, and call it OFDPv2 can be divided into two different phases, which refer to as OFDPv2-A and OFDPv2-B.

1.6 Project Contribution

The main contribution of this project is introducing new algorithms that will work with the improved version of topology discovery protocol both the two main categories. The contribution of the project includes an investigation of the overhead of the current de facto standard for SDN topology discovery.

1.7 Organization of the Project

The project is written based on the standard structure of University Putra Malaysia to cover how the project research is accomplished and the remainder of the project is organized as follows:

In **Chapter 2,** a literature review of topology discovery in OpenFlow-based Software Defined Networking has been presented. However, journals, conference proceedings, seminars, thesis, and books and online resources have been used to enrich this chapter as the main references.

In **Chapter 3,** the design method of the topology discovery has been introduced. The methodology follows the standard procedure for analysis relative reduction in the number of control messages and investigating the control traffic overhead.

In **Chapter 4,** OpenFlow Topology Discovery Protocol (OFDP), and the Implementation of the proposed new versions has been discussed. And OFDPv2 has been evaluated and the generated results are discussed and presented.

In **Chapter 5,** the conclusion of the overall project, and the future work of the proposed topology discovery protocol has been presented.

1.8 Summary

In this chapter, we introduced OpenFlow Discovery Protocol version 2 (OFDPv2) as a new current SDN topology discovery protocol and compared with the OpenFlow Discovery Protocol (OFDP), the current de facto standard. In this chapter, we clarified the need for reducing the overhead of the topology discovery protocol by reducing the number of control messages that need to be sent by the controller based on defined scope and contribution.

alternative protocol that will secure the Topology Discovery in OpenFlow-based SDN. Furthermore, our improved version (OFDPv2) does not reduce the number of *Packet-In* messages that the controller periodically receives from switches.

REFERENCES

Akyildiz, I. F., Lee, A., Wang, P., Luo, M. and Chou, W. (2016). "Research challenges for traffic engineering in software defined networks", *IEEE Network,* vol. 30, pp. 52- 58.

Alexander, D. *et al.*, (1998). "The SwitchWare active network architecture", *IEEE Netw.*, vol. 12, no. 3, pp. 29–36.

Alharbi, Talal, Marius Portmann, and Farzaneh Pakzad (2015). "The (In) Security of Topology Discovery in Software Defined Networks", Local Computer Networks (LCN), 2015 IEEE 40th Conference on IEEE.

Aslan, M. and Matrawy, A. (2016) "On the Impact of Network State Collection on the Performance of SDN Applications".

Azzouni, and Abdelhadi, (2017). "Limitations of OpenFlow Topology Discovery Protocol", arXiv preprint arXiv: 1705.00706.

Barry Levine (August 26, 2013). "Linux' 22th [*sic*] Birthday Is Commemorated - Subtly - by Creator". Simpler Media Group, Inc. Retrieved May 10, 2015. Originally developed for Intel x86-based PCs, Torvalds' "hobby" has now been released for more hardware platforms than any other OS in history.

Casado M. *et al.*, (2006). "SANE: A protection architecture for enterprise networks", in *Proc. 15th Conf. USENIX-SS*, Berkeley, CA, USA, vol. 15, p. 10.

Casado M. *et al.*,(2007). "Ethane: Taking control of the enterprise", in *Proc. Conf. SIGCOMM Appl., Technol., Archit., Protocols Comput. Commun.*, pp. 1–12.

Casado M. *et al.*, (2009). "Rethinking enterprise network control", *IEEE/ACM Trans. Netw.*, vol. 17, no. 4, pp. 1270–1283.

Doria A. *et al.*, (2010). "Forwarding and Control Element Separation (ForCES) Protocol Specification", RFC 5810. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf.

Eckert, Jason W. (2012). *Linux+ Guide to Linux Certification* (Third ed.). Boston, Massachusetts: Cengage Learning. p. 33. ISBN 978-1111541538. Retrieved April 14, 2013. The shared commonality of the kernel is what defines Linux; the differing OSS applications that can interact with the common kernel are what differentiate Linux distributions.

Erickson, D. (2013). "The Beacon OpenFlow controller", in Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, ACM, pp. 13– 18, doi:10.1145/2491185.2491189.

Farrel, A. Vasseur, J.-P. and Ash, J. (2006). "A Path Computation Element (PCE)-Based Architecture, RFC 4655. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf

Feamster, N. Rexford, J. Zegura, E. (2013). "The road to SDN", Queue 11 (12) 20, doi:10.1145/2559899.2560327.

Feamster, N. Balakrishnan, H. Rexford, J. Shaikh, A. and van der Merwe, J. (2004). "The case for separating routing from routers", in *Proc. ACM SIGCOMM Workshop FDNA*, pp. 5–12.

Ferro, G. (2012). "*OpenFlow and Software-Defined Networking*", http://etherealmind.com/software-defined-networking-openflow-so-farand-so-future/.

Floodlight SDN Controller URL (2017). Retrieved online at http://www.projectfloodlight.org/floodlight/

Gani, A., Nayeem, G. M., Shiraz, M., Sookhak, M., Whaiduzzaman, M. and Khan, S. (2014). "A review on interworking and mobility techniques for seamless connectivity in mobile cloud computing", *Journal of Network and Computer Applications,* vol. 43, pp. 84- 102.

GENI Wiki URL (2017). Retrieved online at http://groups.geni.net/geni/wiki/OpenFlowDiscoveryProtocol.

Greenberg A. *et al.*, (2005). "A clean slate 4D approach to network control and management", SIGCOMM *Comput. Commun. Rev.*, vol. 35, no. 5, pp. 41–54.

Gude, N. Koponen, T. Pettit, J. Pfaff, B. Casado, M. McKeown, N. Shenker, S. (2008). "Nox: Towards an operating system for networks", SIGCOMM Comput. Commun. Rev. 38 (3) 105–110, doi:10.1145/1384609.1384625.

Handigol, N. Heller, B. Jeyakumar, V. Lantz, B. McKeown, N. (2012). "Reproducible network experiments using container-based emulation", in Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies, CoNEXT '12, ACM, New York, NY, USA, pp. 253–264, doi:10.1145/2413176.2413206.

Handley, M. Hodson, O. and Kohler, E. (2003). "XORP: An open platform for network research", *ACM SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 1, pp. 53–57.

Hong, S., Xu, L., Wang, H. and Gu, G. (2015). "Poisoning Network Visibility in Software-Defined Networks: New Attacks and Countermeasures", in *NDSS*.

Hu, F. Hao, Q. Bao, K. (2014). "A survey on software-defined network and OpenFlow: from concept to implementation", IEEE Commun. Surv. Tutor. 16 (4) 2181–2206, doi:10.1109/COMST.2014.2326417.

IBM, (2012). "*Software-Defined Networking: A New Paradigm for Virtual, Dynamic, Flexible Networking".*

IEEE Standard for Local and Metropolitan Area Networks – Station and Media Access Control Connectivity Discovery, IEEE Std 802.1AB-2009 (Revision of IEEE Std 802.1AB- 2005), 2009, 1–204. doi:10.1109/IEEESTD.2009.5251812.

Jain, S. Kumar, A. Mandal, S. Ong, J. Poutievski, L. Singh, A. Venkata, S. Wanderer, J. Zhou, J. Zhu, M. Zolla, J. Hölzle, U. Stuart, S. Vahdat, A. (2013). "B4: experience with a globally-deployed software defined WAN", SIGCOMM Comput. Commun. Rev. 43 (4) 3–14, doi:10.1145/2534169.2486019.

Khan, S., Shiraz, M., Abdul Wahab, A. W., Gani, A., Han, Q. and Abdul Rahman Bin Z. (2014). "A comprehensive review on the adaptability of network forensics frameworks for mobile cloud computing", *The Scientific World Journal*, vol. 2014, 2014.

Kim, H. Feamster, N. (2013). "Improving network management with software defined networking", IEEE Commun. Mag. 51 (2) 114–119, doi:10.1109/MCOM.2013.6461195.

Kloti, R., Kotronis, V. and Smith, P. (2013). "OpenFlow: A security analysis", in *Network Protocols (ICNP), 2013 21st IEEE International Conference on*, pp. 1-6.

Kohler, E. Morris, R. Chen, B. Jannotti, J. and Kaashoek, M. (2000). "The click modular router," *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297.

Koponen, T. Casado, M. Gude, N. Stribling, J. Poutievski, L. Zhu, M. Ramanathan, R. Iwata, Y. Inoue, H. and Hama, T. (2010). "Onix: a distributed control platform for large-scale production networks", in Symposium on Operating Systems Design and Implementation, OSDI, vol. 10, pp.1–6. http://static.usenix.org/events/osdi10/tech/full_papers/Koponen.pdf.

Kotronis, V., Dimitropoulos, X., and Ager, B. (2012) "Outsourcing the routing control logic: Better Internet routing based on SDN principles", in *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, pp. 55-60.

Kreutz, D. Ramos, F.M.V. Veríssimo, P. Rothenberg, C.E. Azodolmolky, S. Uhlig, S. (2015). "Software-defined networking: a comprehensive survey", Proc. IEEE 103 (1) 14–76, doi:10.1109/JPROC.2014.2371999.

Lakshman, T. Nandagopal, T. Ramjee, R. Sabnani, K. and Woo, T. (2004). "The softer outer architecture", in *Proc. ACM SIGCOMM Workshop Hot Topics Netw.*, pp. 1–6.

Lantz, B. Heller, B. McKeown, N. (2010). "A network in a laptop: rapid prototyping for software-defined networks", in Proceedings of the 9th ACMSIGCOMM Workshop on Hot Topics in Networks, ACM, p. 19, doi:10.1145/1868447.1868466.

Luo, J. Pettit, J. Casado, M. Lockwood, J. and McKeown, N. (2007). "Prototyping Fast, Simple, Secure Switches for Etha," in *Proc. 15th Annu. IEEE Symp. HOTI*, pp. 73–82.

Lyons, Daniel (March 15, 2005). "Linux rules supercomputers". Forbes. Retrieved February 22, 2007.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J. (2008). "OpenFlow: enabling innovation in campus networks", *ACM SIGCOMM Computer Communication Review*, vol. 38, pp. 69-74.

McKeown, N. (2009). "Software-defined networking", INFOCOM Keynote Talk, URL http://yuba.stanford.edu/nickm/talks/infocom_brazil_2009_v1-1.pdf.

Mogul, J. C., AuYoung, A., Banerjee, S., Popa, L., Lee, J., Mudigonda, J. (2013). "Corybantic: towards the modular composition of SDN control programs", in *Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks*, p. 1.

OpenDaylight URL (2017). Retrieved online at http://www.opendaylight.org/project/technical- overview.

Open Flow Standard version 1.4 URL (2017). Retrieved online at https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf specifications/openflow/openflow-spec-v1.4.0.pdf.

OpenFlow Switch Consortium (2012). "*OpenFlow Spec, v1.3.0* ",https://www.opennetworking.org/images/stories/downloads/sdnresources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf.

Open Networking Foundation (2017). Retrieved online at URL https://www.opennetworking.org.

Open vSwitch URL (2017) retrieved online at http://openvswitch.org.

Pakzad, F., Portmann, M., Tan, W. L. and Indulska, J. (2015). "Efficient topology discovery in OpenFlow-based Software Defined Networks", *Computer Communications.*

POX SDN Controller URL (2017). Retrieved online at http://www.noxrepo.org/pox/about-pox/

Quagga Routing Software Suite. [Online]. Available: http://www.nongnu.org/quagga/

Raj, J. (2013). " Lecture note on Introduction to OpenFlow", retrieved online at http://www.cse.wustl.edu/~jain/cse570-15/m_15ofl.htm.

Rexford J. *et al.*, (2004). "Network-wide decision making: Toward a wafer-thin control plane", in *Proc. HotNets*, pp. 59–64.

Ryu SDN Controller URL (2017). Retrieved online at http://osrg.github.io/ryu/

Saha, A. K., Sambyo, K., and Bhunia, C. (2016). "Topology Discovery, Loop Finding and Alternative Path Solution in POX Controller", in *Proceedings of the International MultiConference of Engineers and Computer Scientists*.

Scott-Hayward, S., Natarajan, S. and Sezer, S. (2016). "A Survey of Security in Software Defined Networks", *IEEE Communications Surveys & Tutorials*, vol. 18, pp. 623-654.

Shenker, S., Casado, M., Koponen, T. and McKeown, N. (2011). "The future of networking, and the past of protocols", *Open Networking Summit*, vol. 20.

Shin, M.K., Ki-Hyuk Nam, K.H., Kim, H.J., (2012). "Software-Defined Networking (SDN): A Reference Architecture and Open APIs," *Proceedings, International Conference on ICT Convergence (ICTC)*, pp.360–361.

Smith, J. *et al.*, (1996). "Switchware: Accelerating network evolution", CIS Dept., Univ. Pennsylvania, Philadelphia, PA, USA, White Paper.

Software-defined networking (2012). "The new norm for networks" Palo Alto, CA, USA, White Paper, Apr. 2012. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/white-papers/wp-sdnnewnorm.pdf

Staff, C. (2016). "A purpose-built global network: Google's move to SDN", *Communications of the ACM*, vol. 59, pp. 46-54.

Suleman, K. Abdullah, G. Ainuddin, W. Abdul, W. and Muhammad, K.K. (2016). "Topology Discovery in Software Defined Networks: Threats, Taxonomy, and State-of-the-art", IEEE communication surveys & tutorials.

Tarnaras, George, Evangelos Haleplidis, and Spyros Denazis (2015). "SDN and ForCES based optimal network topology discovery", Network Softwarization (NetSoft), 1st IEEE Conference on. IEEE.

Thanh, N.H.,Nam, P.N.,Truong, T.-H.,Hung, N.T.,Doanh, L.K., Pries, R., (2012). "Enabling Experiments for Energy-Efficient Data-Center Networks on an OpenFlow-Based

Platform", *Proceedings, Fourth International Conference on Communications and Electronics (ICCE),* pp.239–244.

The BIRD Internet Routing Daemon. [Online]. Available: http://bird.network.cz/

Thibodeau, Patrick (2009). "IBM's newest mainframe is all Linux". Computerworld (published December 9, 2009). Archived from the original on November 11, 2016. Retrieved February 22, 2009.

Thomas, B. A. Idris, N. Al-Hnaiyyan, A. Binti Mahmud, R. Abdelaziz, A., Khan, S. (2016). "Towards Knowledge Modeling and Manipulation Technologies: A Survey", *International Journal of Information Management*.

Tootoonchian, A. Gorbunov, S. Ganjali, Y. Casado, M. Sherwood, R. (2012). "On controller performance in software defined networks", in Proceedings of USENIX Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services, Hot-ICE,vol.54.

Vishnoi, A. Poddar, R. Mann, V. Bhattacharya, S. (2014). "Effective switch memory management in OpenFlow networks", in Proceedings of the 8th ACM International Conference on Distributed Event-Based Systems, pp. 177–188, doi:10.1145/2611286.2611301.

Vu, T.H.,Nam, P.N.,Thanh, T.,Hung, L.T.,Van, L.A.,Linh, N.D.,Thien, T.D.,Thanh, N.H., (2012). "Power-Aware OpenFlow Switch Extension for Energy Saving in Data Centers ", *Proceedings, International Conference on Advanced Technologies for Communications (ATC),* pp.309–313.

Wang W. *et al.*, (2007). "Design and implementation of an open programmable router compliant to IETF ForCES specifications," in *Proc. 6th ICN*, pp. 1–6.

Wireshark URL (2017). Retrieved online at https://www.wireshark.org/.

Yan H. *et al.*, (2007). "Tesseract: A 4D network control plane," in *Proc. 4th USENIX Conf. NSDI,* p. 27.

Yang, L. Dantu, R. Anderson, T. and Gopal, R. (2004). "Forwarding and Control Element Separation (ForCES) Framework", RFC 3746. [Online]. Available: http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf