Advances in Computer Science and Information Technology (ACSIT)
Print ISSN: 2393-9907; Online ISSN: 2393-9915; Volume 2, Number 1; January-March, 2015 pp. 81-85
© Krishi Sanskriti Publications
http://www.krishisanskriti.org/acsit.html

Scalability Issues in Software Defined Network (SDN): A Survey

Smriti Bhandarkar[1], Gyanamudra Behera[2] and Kotla Amjath Khan[3]

[1]M. Tech. Student School Of Computer Engineering KIIT University, Bhubaneswar, Odisha
[2]M. Tech. Student School Of Computer Engineering KIIT University, Bhubaneswar, Odisha
[3]School Of Computer Engineering KIIT University, Bhubaneswar, Odisha
E-mail: [1]smriti.bhandarkar@gmail.com, [2]gyanamudrabehera@gmail.com, [3]kotla.amjath@gmail.com

Abstract—*Software defined network deals with splitting of infrastructure layer from control layer which enhances the programming capability, flexibility, malleability and manageability of the network. This survey concerns about scalability issues in SDN which includes modification of hardware and software of networking devices and basic centralized architecture of SDN. We also give some light on scalability evaluation of different models of SDN. So, we conclude that by distributing control/intelligence over multiple controllers scalability of network can be increased.*

Keywords : *scalability, data plane, control plane, hierarchy of SDN*

1. INTRODUCTION

Software defined network[1-3] is a new emerging technology in the field of networking in which programs written in high-level languages like C, java, ruby, Perl etc for control plane by the network administrator are used to control the behavior of whole network.

Even though the traditional network is fully developed, it is unable to fulfill today's network requirements. There is no such a big change in this network, since 1970. The reasons for the need of this technology is as follows[4-5]

1. Change in traffic patterns
2. Complex network operations and management
3. Intelligent system which control behavior of lots of network devices
4. Increase in amount of data i.e. big data

In spite of having lots of benefits over traditional network like flexible, adaptable, manageable, cost effective(in terms of time), scalability of SDN is a big issue i.e. centralized nature of control plane is not friendly with the growing organizational network.

There are various factors which affects the scalability of SDN. They are

1. Processing power of controllers and forwarding devices
2. Capacity of memory/buffer
3. Placement of controllers in the network
4. Latency/delay between controllers and network devices to transfer packets
5. Traffic in the link

In this paper, we focus on challenges in SDN to scale-up the network and their proposed solutions. For this, we get that scalability of SDN can be enhanced by resolving the scalability issues of data plane and control plane. The rest paper is organized as follows. Section 2 discusses History of SDN. Section 3 focuses on Scalability issues of data plane. Section 4 focuses on hierarchy of SDN models and their scalability issues and finally section 5 has conclusion of paper with future work.

2. HISTORY OF SDN

The SDN has a incredibly long history but it is a recent concept in the field of networking and research. It depends on the concept of three existing network technologies i.e. programmable/active networks, centralized networks and network virtualization.

In active networks[6], each and every node has the capability to do computations and modifications on the content of the packets. It supports packet downloading and some tiny programs are appended in that packet then both are encapsulated in frames which are transmitted and processed at every node of the network along with their route.

The centralized network[7] has the concept that the server on the middle of the network controls and monitors other devices present in the network in the same way controller control and monitor the forwarding plane by providing services from the management plane.

The main idea of virtual networks[8] is to represent one or more logical network topologies on the single network infrastructure. It creates logical/virtual networks that are decoupled from the underlying network hardware to ensure the network can better integrate and support increasing virtual environments. In virtual infrastructure, NV(network virtualization) can be used to create virtual network and this enables us to support the complex requirements in multi-tenancy environment. For example, it isolates network traffic in zones/containers. Tempest[9] is the first network virtualization project in which several switchlets runs on the top of the single ATM switch and share the same physical resources.

The concept of splitting infrastructure layer from the control layer[10] e.g. RCP[11] makes it different and more flexible from the traditional network ForCES[12] and OpenFlow[13] are the communicating protocols between control plane and forwarding plane. By using these protocols the controllers dynamically modifies the flow table entries in the flow table of the forwarding plane by adding , deleting and updating the flow entries.

The ONF[14] is the open source consortium which promotes the adoption and concepts of SDN by developing the OpenFlow protocol as standard protocol to communicate between the control plane and data plane(currently released version 1.4.0).

3. SCALABILITY ISSUES OF DATA PLANE

In the infrastructure layer, scalability is enhanced by modifying the hardware and software of the network or forwarding device means it depends on memory size and processing speed of CPU.

Kannan et al.[15] works on TCAM flow table size and power dissipation. They proposed the compact TCAM in which flow entries in the TCAM are replaced by shorter flow-ID and a field flow-ID is added/modified in the packet header. The controller contains a new special table known as flow-ID table to store the ID of various flows in the network.

Narayanan et al.[16] modifies the hardware of the switch by using multi-core programmable ASIC's and its architecture by using a high-speed bus and splitting the forwarding path into TCAM-based and software based path. This enhances the bus speed between on chip CPU and pipeline.

Lu et al.[17] uses the well equipped switches having powerful CPUs and DRAM (in GB) to setup an internal link having high bandwidth between ASIC and CPU.

Tanyingyong et al.[18] proposed a lookup procedure (software based) which used standard NIC and having cache flow table entries on that network interface card(NIC).

Luo et al.[19] proposed an OpenFlow switch algorithm which enhances the lookup procedures in the switch and CPU power.

Kang et al.[20] prepared an algorithm for the rule replacement in the controller which distributes the forwarding polices throughout the network and also update the flow table dynamically with the new flow entries.

4. HIERARCHY OF SDN AND THEIR SCALABILITY ISSUES

The SDN has centralized and decentralized architectures.

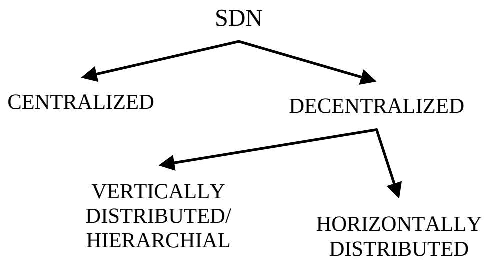

Fig. 1: Hierarchy of SDN.

4.1. Centralized Model

In this architecture, the network intelligence i.e. control plane is centralized which means the single controller has the global view of network and it handle whole network as shown in fig. 2.

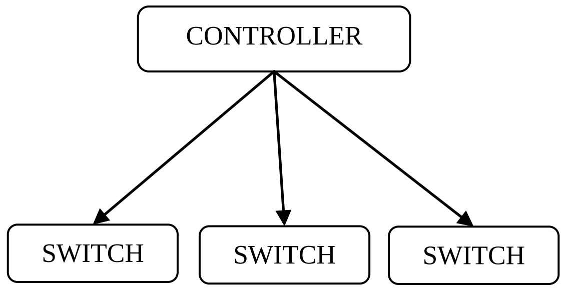

Fig. 2: Centralized control plane.

Devoflow[21] has the centralized architecture, it reduces the traffic i.e. flow requests and statistics between control layer and infrastructure layer by cloning rules, triggers and sampling. Shorter-flows are handled by data plane and important-flows are transferred to control layer, it reduces the traffic going towards the controller and enhance scalability.

Zuo Qingyun et al.[22] works on centralized architecture to increase the scalability of the network by reducing network traffic, for this they provide intelligence in the switch to handle the redundant packet-in messages and collects the flow statistics from the data plane by using statistics server and sends a summarize report to the controller.

FlowVisor[23-24] is a special OpenFlow controller behaves like a transparent proxy running between switches and multiple controllers. It manages many controllers by slicing the network resources and tell one controller to control and monitor its own slice.

4.2. Decentralized Model

control plane splits between a number of controllers. The network is handled by two architectures

1. Horizontally distributed control plane
2. Vertically distributed control plane

4.2.1. Horizontally Distributed Control Plane. Multiple controllers with comparatively equal capabilities control the whole network as shown in fig. 3.

In this architecture, single controller has the topology information of switch connected to the controller and the neighbor controller (local view) or the image of the whole network (global view). The controllers communicate with each other by using east-west bound interface.

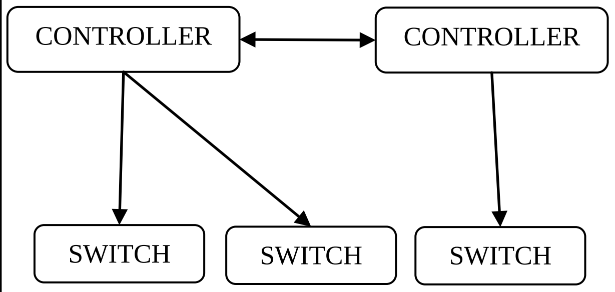

Fig. 3: Horizontally distributed control plane.

Onix[25] has distributed control over the network. There exists some general API to access the network state distributed over the onix instances.

Hyperflow[26] takes the advantages of centralized control plane and gives the authority to all the controllers to share the same network wide view/global view and locally process the requests without disturbing any other remote node.

4.2.2. Vertically Distributed Control Plane. In this architecture, controllers/switches are arrange in a hierarchical level in which lower level sends network information to upper or higher level.

kandoo[27] has the hierarchical structure of two level as shown in fig. 4(a) in which the root controller manages those applications which require global view of the network like load balancing, routing etc and acts as a mediator between local controllers for co-ordination.

DIFANE[28] has the hierarchical structure as shown in fig. 4(b). Switches in the first level has the intelligence to manage some of the messages which makes it more scalable than architecture shown in fig. 4(a). The network administrator has the power to forward, modify, drop and measure the traffic on the switch. It works on two basic ideas

1. Distribute the rules over the authority switches.
2. The controller partition the rules according to the partition algorithm.

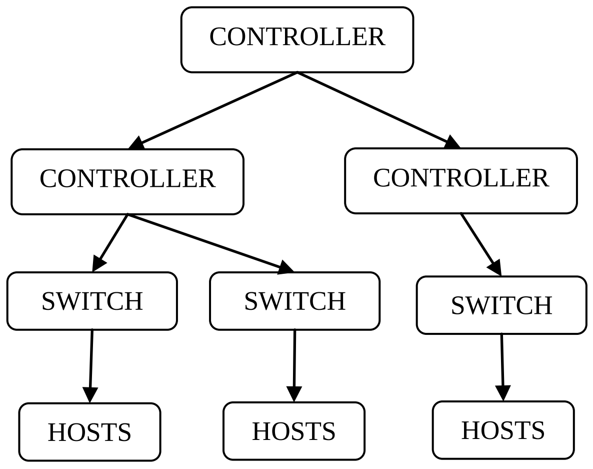

Fig. 4(a): Vertically distributed control plane.

PALETTE[29-30] follows the ideas of DIFANE and uses pivot bit decomposition method and cut based decomposition graphical method to partition the flow table. Then, the minimized flow tables are distributed over the network to the different authority switches. To check the minimization of flow table they use Rainbow path coloring problem.

R-SDN[31] has a vertically distributed control plane. Number of network/forwarding devices on each layer increases according to the Fibonacci series as the idea keep in mind that series increase like branches of a tree (spanning tree) with no loop. They manages the network by using Fibonacci heap ordered tree for load balancing and routing. The algorithm is solvable in polynomial time and gives less response time as compared to the traditional network.

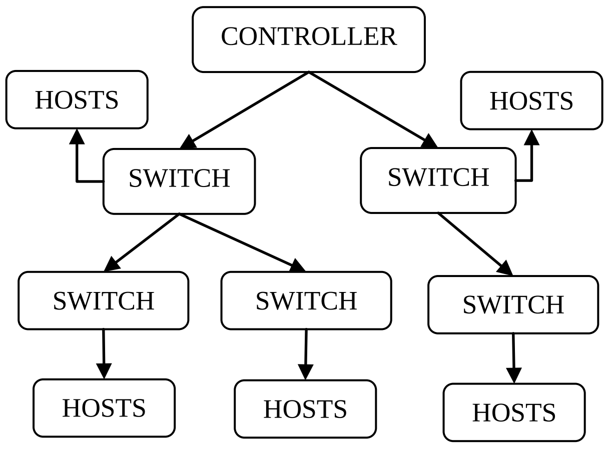

Fig. 4(b): Vertically distributed control plane.

5. SCALABILITY EVALUATION OF DIFFERENT MODELS OF SDN

Jie Hu et al.[32] proposes a metric which evaluates scalability of different models of SDN i.e. centralized control plane, vertically distributed/hierarchical control plane and horizontally distributed control plane.

They define scalability of controller contains number of nodes varies from N_1 to N_2 as follows

$$\varphi(N_1, N_2) = \frac{F(N_1)}{F(N_2)} \quad (1)$$

where,

$F(N_1)$ and $F(N_2)$ ≡ productivity of control plane having nodes N_1 and N_2 respectively

Productivity[33] of control plane is defined as

$$F(N) = \varphi(N) \times \frac{T(N)}{C(N)} \quad (2)$$

where,
$\varphi(N)$ ≡Throughput of control plane for processing network requests
$T(N)$ ≡average response time for each request

$C(N)$ ≡ cost of deployment of control plane

N ≡ Number of nodes in the network

They proposed three theorems in which they find the scalability of different models of SDN and proves that vertically distributed control plane is more scalable than the other models but they also proves that as the distance between network devices increase the scalability of every model or architecture drops due to delay in packet transfer in the network which is explained in the controller placement problem[34].

Syed A. S. et al.[35] gives the guidelines to modify the design of controller to scale up the network. They do performance evaluation on four controllers floodlight, maestro, nox and becon based on latency and scalability of thread by continuously sending packets to the controller and calculating response time per second for different number of threads and switches.

6. CONCLUSION

By this study, we conclude that SDN is easy to program, manageable and less complex as compared to the conventional network but the basic centralized nature of SDN is bottleneck for scalability of this network and it can be enhanced by decentralizing the control plane and modifying hardware and software of forwarding device and controller for load balancing, routing, traffic engineering etc. We further works on placement of controllers in the network so as to get maximum throughput and efficiency with less cost.

7. ACKNOWLEDGEMENTS

This work was supported by KIIT University, Bhubaneswar, Odisha, India.

REFERENCES

[1] Yosr Jarraya, Taous Madi, and Mourad Debbabi, "A Survey and a Layered Taxonomy of Software-Defined Networking", published in Communications Surveys and Tutorials, IEEE Issue: 99 , 2014.
[2] B. Nunes, M. Mendonca, X. Nguyen, K. Obraczka, and T. Turletti,“A survey of software defined networking: Past, present, and future of programmable networks,”Communications Surveys Tutorials, IEEE,no. 99, pp. 1–18, 2014.
[3] F. Hu, Q. Hao, and K. Bao, “A survey on software defined networking(SDN) and openflow: From concept to implementation,”IEEE Commun. Surveys Tuts., no. 99, pp. 1–1, 2014.
[4] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks," ONF White Paper, April 12, 2012.
[5] Myung-Ki Shin, Ki-Hyuk Nam,Hyoung-Jun Kim, “Software-Defined Networking (SDN): A Reference Architecture and Open APIs”, International Conference on ICT Convergence (ICTC), 2012.
[6] J. M. Smith and S. M. Nettles, “Active networking: one view of the past, present, and future,” IEEE Trans. Syst. Man Cybern, C, Appl. Rev., vol. 34, no. 1, pp. 4 –18, February 2004.
[7] Yacoby, Amnon, and Eden Shochat. "Centralized network control." U.S. Patent 7,788,366, issued August 31, 2010.
[8] Chowdhury, N. M., and Raouf Boutaba. "A survey of network virtualization." Computer Networks 54, no. 5 (2010): 862-876.
[9] Van der Merwe, J.E., Rooney, S., Leslie, I.M. and Crosby, S.A., "The Tempest - A Practical Framework for Network Programmability", IEEE Network, November 1997

[10] SDN[EB/OL].[2013-9-24]. https://www.opennetworking.org/sdnresources/sdn-library /whitepapers. 2013.

[11] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. van der Merwe, “Design and Implementation of a Routing Control Platform,” in ACM/USENIX NSDI, 2005.

[12] A. Doria, J. H. Salim, R. Haas, H. Khosravi, W. Wang, L. Dong, R. Gopal, and J. Halpern, Forwarding and Control Element Separation (ForCES) Protocol Specification, http://tools.ietf.org/html/rfc5810/,request for Comments (RFC) 5810. ISSN: 2070-1721. March 2010.

[13] MCKEO WN N, ANDERSON T, BALAKRISHNAN H, et al. OpenFlow: Enabling Innovation in Campus Networks[J]. ACM SIGCOMM Computer

[14] Open Networking Foundation, OpenFlow Switch Specification, October 2013, version 1.4.0

[15] K. Kannan and S. Banerjee, “Compact TCAM: Flow Entry Compaction in TCAM for Power Aware SDN,” in Distributed,Computing and Networking, ser. Lecture Notes in Computer Science, D. Frey, M. Raynal, S. Sarkar, R. Shyamasundar, and P. Sinha, Eds. Springer Berlin Heidelberg, 2013, vol. 7730, pp. 439–444.

[16] R. Narayanan, S. Kotha, G. Lin, A. Khan, S. Rizvi, W. Javed, H. Khan, and S. Khayam, “Macroflows and Microflows: Enabling Rapid Network Innovation through a Split SDN Data Plane,” in European Workshop on Software Defined Net-working (EWSDN), October 25th-26th, Darmstadt, Germany. Washington, DC, USA: IEEE Computer Society, 2012, pp. 79–84.

[17] G. Lu, R. Miao, Y. Xiong, and C. Guo, “Using CPU as a Traffic Co-Processing Unit in Commodity Switches,” in the Proceedings of the first workshop on Hot Topics in Software Defined Networks, ser. HotSDN ’12. New York, NY, USA: ACM, 2012, pp. 31–36.

[18] V. Tanyingyong, M. Hidell, and P. Sjodin, “Using Hardware Classification to Improve PC-based OpenFlow Switching,” in IEEE 12th International Conference on High Performance Switching and Routing (HPSR), 2011, pp. 215–221.

[19] Y. Luo, P. Cascon, E. Murray, and J. Ortega, “Accelerating OpenFlow Switching with Network Processors,” in 5th ACM/IEEE Symposium on Architectures for Networking and Communications Systems, ser. ANCS ’09. New York, NY, USA: ACM, 2009, pp. 70–71.

[20] N. Kang, Z. Liu, J. Rexford, and D. Walker, “Optimizing the “One Big Switch” Abstraction in Software-Defined Networks,” in Proceedings of the 9th conference on Emerging networking experiments and technologies, ser. CoNEXT ’13. New York, NY, USA: ACM, 2013.

[21] A. R. Curtis et al., “DevoFlow: Scaling Flow Management for High-Performance Networks,” Proc. ACM SIGCOMM ’11, 2011, pp. 254–65.

[22] Zuo, Qingyun, Ming Chen, Ke Ding, and Bo Xu. "On generality of the data plane and scalability of the control plane in software-defined networking." Communications, China 11, no. 2 (2014): 55-64.

[23] B. Sonkoly, A. Gulyas, F. Nemeth, J. Czentye, K. Kurucz, B. Novak, and G. Vaszkun, “OpenFlow Virtualization Framework with Advanced Capabilities,” in 2012 European Workshop on Software Defined Networking, October 25th-26th, Darmstadt, Germany., ser. EWSDN ’12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 18–23.

[24] R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T.-Y. Huang, P. Kazemian, M. Kobayashi, J. Naous, S. Seetharaman, D. Underhill, T. Yabe, K.-K. Yap, Y. Yiakoumis, H. Zeng, G. Appenzeller, R. Johari, N. McKeown, and G. Parulkar, “Carving Research Slices out of your Production Networks with OpenFlow,” SIGCOMM Comput. Commun. Rev., vol. 40, no. 1, pp. 129–130, Jan. 2010.

[25] T. Koponen et al., “Onix: A Distributed Control Platform for Large-Scale Production Networks,” Proc. 9th USENIX OSDI Conf., 2010, pp. 1–6.

[26] A. Tootoonchian and Y. Ganjali, “Hyperflow: A Distributed Control Plane for OpenFlow,” Proc. 2010 INM Conf., 2010, pp. 3–3.

[27] S. Hassas Yeganeh and Y. Ganjali, “Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications,” Proc. HotSDN ’12 Wksp., 2012, pp. 19–24.

[28] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, “Scalable flow-based networking with difane,” ACM Comput. Commun. Rev., vol. 41, no. 4,2010.

[29] Y. Kanizo, D. Hay, and I. Keslassy, “Palette: Distributing tables in software-defined networks,” Technion, Tech. Rep. TR12-05, 2012.

[30] Yossi Kanizo, David Hay, and Isaac Keslassy. Palette: Distributing tables in software-defined networks. In INFOCOM, pages 545–549,2013

[31] DAI, Wei, Guochu SHOU, Yihong HU, and Zhigang GUO. "R-SDN: A RECUSIVE APPROACH FOR SCALING SDN."

[32] Hu, Jie, Chuang Lin, Xiangyang Li, and Jiwei Huang. "Scalability of Control Planes for Software Defined Networks: Modeling and Evaluation."

[33] P. Jogalekar and M. Woodside, “Evaluating the scalability of distributed systems,” Parallel and Distributed Systems, IEEE Transactions on, vol. 11, no. 6, pp. 589–603, 2000.

[34] Jimenez, Yury, Cristina Cervello-Pastor, and Aurelio J. Garcia. "On the controller placement for designing a distributed SDN control layer." In Networking Conference, 2014 IFIP, pp. 1-9. IEEE, 2014.

[35] Shah, Syed Abdullah, Jannet Faiz, Maham Farooq, Aamir Shafi, and Syed Akbar Mehdi. "An architectural evaluation of SDN controllers." In Communications (ICC), 2013 IEEE International Conference on, pp. 3504-3508. IEEE, 2013.