International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 18 (2017) pp. 7638-7641

Centralized Controllers of SDN and its problem spaces

Smitha Vinod
Research Scholar, Department of Computer Applications,
Karpagam Academy of Higher Education, Eachanari, Coimbatore– 641 021, India.
Orcid Id: 0000-0002-5547-5981

X. Agnise Kala Rani
Professor, Department of Computer Applications,
Karpagam Academy of Higher Education, Eachanari, Coimbatore– 641 021, India.
Orcid Id: 0000-0002-1465-0595

Abstract

As promised as an Operating system for a network, the frequently heard name is Software Defined Networks (SDN). Network architecture has always been dependent on proprietary hardware devices, rather than programmable hardware components. This used to make the overall network management cost high. Some of the best promises SDN gives are low cost, high speed, good quality, programmable networking solutions to the propitiatory solutions world. As promised by SDN there is a considerable change in the scenario after the implementation of programmable and controllable network equipment's into the networking sector.

This paper discusses the various issues faced by the centralized controllers and its some of the problem spaces.

Keywords: SDN; Controllers; Flow-based forwarding.

INTRODUCTION

The society was always depended on [1] proprietary hardware devices and this used to be the major reason for the high overall network management cost. One of the expectations was, there will be a considerable change in this scenario after the successful implementation of the SDN technology because the trend will be changed into programmable hardware components instead of proprietary components. So SDN would enable the network to be programmable, thereby reduced the operational costs, enabling a more intelligent view of the network, and allowing applications to interact directly with the network. So, in totality SDN was expected to reduce the operational costs and equipment costs and it will indirectly help to reuse the old hardware components with new software solutions.

The freedom to control proprietary hardware would be very less in comparison with the programmable one. This programmable hardware was expected to allow enterprises to control the entire network through a single centralized [13] controller which gives more power and freedom to update, and modify the code inside it. The thought was that this change in technology will definitely bring in a huge change in the networking work style.

The SDN controller mostly works by taking the control plane from the network hardware and making it more like a single point software access and control. This controller facilitates the network management from manual to automated mode, and makes it easier to install, integrate, and manage the applications with more ease and low operational costs. So, in a nutshell the controllers of the SDNs were supposed to remove the intelligence from distributed control planes, replacing them with the centralized calculation of network paths. This was possible because an individual router or a switch has only a localized view of network conditions wherein a centralized controller easily gets a better overall view. This single view would allow the controller to proficiently manage and control traffic through the network. This would definitely help to improve the overall efficiency of the network by improving the performance of individual applications passing through the network. But it so happened that the SDN controllers which were considered as Network Operating Systems often started to get viewed as a single point of failure [7] too.

The SDNs were supposed to provide a much more modular level of control from the top to bottom. So, even the flow level control would be possible for the controller. This would help to make and implement the policy controls in each level of implementations. Which starts from discovery, packet path identification, flow directions, calculating operational costs, maintaining quality of service, etc. All these calculations need to be done for each individual user's and each individual application execution.

Over the year's definition, SDN has been viewed differently, now it is about any network technology that allows programmatic access to information that can have the control over the network.

NETWORK PATH CALCULATION

Centralized Network path calculations

In a policy-driven path selection [6] which means choosing a path through the network is less optimum than the shortest path from a metrics perspective, but more optimal from a network usage or quality-of-service perspective, the processing power and memory requirements of individual routers speeds up. The optimal path calculation becomes very complex day by day.

If distributed control planes could be eliminated and replaced by a controller or set of controllers, the complexity of each forwarding device could be reduced drastically, because the jobs of discovering the local topology and calculating the best path per destination would be offloaded from the individual boxes, and pushed onto the controller. By removing this processing from the routers, small, cheap, lightweight forwarding only devices could be used instead of the traditional router.

Centralized control, its complexity and its forwarding issues.

An important task in SDN architectures is that of controller placement. Other issues range from latency constraints to failure tolerance and load balancing. In most scenarios, atleast some of these objectives are competing, thus no single best placement is available and decision makers need to find a balanced trade-off [10].

Avoiding complexity is not at all so easy. Allowing the centralized controller to scale big and work with a smaller network both are not very easy to attain. In some situations, one, two or three controllers providing the forwarding information to huge number of switches can also create issues. Few examples for such situations are the relationship between centralized computation and reactive control planes and then remote reactions to local topology and reachability changes.

In distributed control planes before the first packet is transmitted across the network, the routing protocol [13] must discover a set of loop-free paths that can reach every destination in the network. Since this discovery and calculation process typically involves flooding, processing, and managing a lot of information, distributed control planes often rely on information hiding through aggregation to manage the amount and speed of state being carried in the protocol. When the calculation of routes is centralized, there must still be some form of information hiding to scale the control plane. Instead of aggregation at specific topological points in the network, SDN control planes most often opt for moving to a reactive control plane, meaning the forwarding devices discover reachability information only when they receive the first packet in a flow. While this does reduce the amount of forwarding state in any particular device, it also has many drawbacks. In reactive control planes frequent disconnects, sudden change of state in the network, local cache timeouts, cache failures all are part of this system which is considered as a worst situation in proactive control planes.

Centralized control planes always take a complex method to do the recalculation of the best path. This is because if there is a failure in a local node, that information is to be updated to the nearest controller and then it should recalculate the path and give back the information to the affected switches. So, there is not many short cuts, or quick solutions to these problems. This is because of the centralized decision making of the controllers, if this was done by a switch itself these situations might not have arised. Even if it is not a disadvantage of the centralized controllers sometimes it is considered so.

Communication and issues between the controllers.

In the earlier distributed control planes the frequent effort required was very high; in-terms of configuration, setup, final deployment, maintenance, troubleshooting, daily monitoring [14], failure chances, etc. But when it is started comparing it with a centralized controller all these were looking very simple and easy. But it's not so, in a larger network with thousands of switches it is not practically affordable to have a single controller and when there is a big unfortunate failure chances. So always expected is a geographically faraway controller which always will have a complete knowledge about the functioning of the other. In reality it is not that simple to handle both the controllers in a similar manner. For the communication between these two controllers there should be made available a distributed control plane [4].

Fingerprinting in SDN controllers.

According to [7] there is always a possibility of fingerprinting in SDN controllers by detecting which controller is managing the network.

Authors of [11] discusses the importance of machine-learning based anomaly detection and analytics to identify SDN software faults and help guide real-time network responses. All this is because the programmability of SDN provides tremendous flexibility and adaptability to change network conditions and demands, it also exposes networks to significant vulnerabilities through software faults in network applications, as well as in the control and data planes.

Zang et al. [8] [9] demonstrated the separation of planes in SDN, however, introduces new vulnerabilities in SDN networks, since the difference in processing packets at each plane allows an adversary to fingerprint, the network's packet-forwarding logic. By the information leveraged from the RTT and packet-pair dispersion of the exchanged packets, fingerprinting attacks on SDN networks succeed with overwhelming probability. These attacks are not restricted to

active adversaries, but can also be mounted by passive adversaries that only monitor the traffic exchanged with the SDN network.

FORWARDING DECISIONS

One of the major purposes of the SDN was replacement of few of the large existing distributed protocols. In order to replace the existing protocols, it was not so easy as expected. When the tasks became tedious, the number of lines of code required also became huge. So finally, there came a point when people started to think that all the efforts are worth doing. The existing routing protocol are capable of finding the network path and packed forwarding to the destination address. But they were not very helpful in flow based forwarding control. Where one of the expectations were independent path identification based on different applications. But here there are many issues came into the picture. In order to forward each different applications communication flow through different available paths became very complex. This is because of the number of nodes, and the number of applications run on those nodes, and the number of times it requires the connection and termination, etc. One of the main issues over here is the number of flows happening at the single point of time and the other is the connection setting complexity part of it. When we consider the network to be large, the communication happening at a particular point can be countless. Here the setup cost and operational cost also goes very high which was not expected at the time of design of SDN. The volume of flow states expected at the design time of SDN may cross the capacity of the hardware too. All these are against the development purposes of SDN. In flow-based forwarding control, in each hop in the network each packet header is supposed to be analyzed for the path identification. This incur a huge operational expenditure in terms of hardware capacity to maintain the per-flow tables for each flow of information and operational expenditure in terms of maintenance cost and power utilization costs.

CONCLUSION

Despite all the promising opportunities it is expected from SDN, there lies a lot of responsibilities which are equally complicated and challenging to fulfill.

IT organizations and network enterprises should be aware of these challenges and explore the functionality of the SDN architecture to counter these criticisms[12].

REFERENCES

[1] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievs-ki, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Hölzle Stephen Stuart, Amin Vahdat, "B4: Experience with a globally-deployed software defined wan", Special Interest Group on Data Communication (SIGCOMM) 2013 ACM International Conference on, pp. 3-14, August 2013.

[2] D. Kreutz, F.M.V. Ramos, P. Esteves Verissimo, C. Esteve Rothenberg, S. Azodolmolky, S. Uhlig, "Software-defined networking: A comprehensive survey", Proceedings of the IEEE, vol. 103, no. 1, pp. 14-76, January 2015.

[3] Nick Mckeown, Scott Shenker, Tom Anderson, Larry Peterson, Jonathan Turner, Hari Balakrishnan, Jennifer Rexford, "Openflow: Enabling innovation in campus networks", ACM Computer Communication Review, April 2008.

[4] Liron Schiff, Stefan Schmid, and Petr Kuznetsov, "In-Band Synchronization for Distributed SDN Control Planes," ACM SIGCOMM Computer Communication Review, Volume 46, Number 1, January 2016, http://www.sigcomm.org/sites/default/files/ccr/papers/2016/January/0000000-0000004.pdf

[5] Russ White and Jeff Tantsura, "Navigating Network Complexity: Next-Generation Routing with SDN, Service Virtualization, and Service Chaining", Addison-Wesley Professional, 2015, ISBN-13: 978-0133989359.

[6] Russ White and Shawn Zandi, "Cloudy-Eyed: Complexity and Reality with Software-Defined Networks", The Internet Protocol Journal, Volume 19, Number 3, November 2016.

[7] A. Azzouni, O. Braham, T. M. T. Nguyen, G. Pujolle and R. Boutaba, "Fingerprinting OpenFlow Controllers: The First Step to Attack an SDN Control Plane," *2016 IEEE Global Communications Conference (GLOBECOM)*, Washington, DC, 2016, pp. 1-6.

[8] H. Cui, G. O. Karame, F. Klaedtke and R. Bifulco, "On the Fingerprinting of Software-Defined Networks," in *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 10, pp. 2160-2173, Oct. 2016.

[9] H. Cui, G. O. Karame, F. Klaedtke and R. Bifulco, "Fingerprinting Software-Defined Networks," *2015 IEEE 23rd International Conference on Network Protocols (ICNP)*, San Francisco, CA, 2015, pp. 453-459.

[10] S. Lange et al., "Heuristic Approaches to the Controller Placement Problem in Large Scale SDN Networks," in IEEE Transactions on Network and Service Management, vol. 12, no. 1, pp. 4-17, March 2015.

[11] L. J. Jagadeesan and V. Mendiratta, "Programming the Network: Application Software Faults in Software-Defined Networks," *2016 IEEE International*

Symposium on Software Reliability Engineering Workshops (ISSREW), Ottawa, ON, 2016, pp. 125-131.

[12] Manar Jammal, Taranpreet Singh, Abdallah Shami, Rasool Asal, Yiming Li, "Software defined networking: State of the art and research challenges", Computer Networks, Volume 72, 29 October 2014, Pages 74-98, ISSN 1389-1286.

[13] Xiao, Peng, Zhiyang Li, Song Guo, Heng Qi, Wenyu Qu and Haisheng Yu. "A K self-adaptive SDN controller placement for wide area networks." Frontiers of IT & EE 17 (2016): 620-633.

[14] C. Caba and J. Soler, "Mitigating SDN Controller Performance Bottlenecks," *2015 24th International Conference on Computer Communication and Networks (ICCCN)*, Las Vegas, NV, 2015, pp. 1-6.