Journal of Engineering and Applied Sciences 13 (Special Issue 13): 10790-10796, 2018
ISSN: 1816-949X
© Medwell Journals, 2018

Software Defined Networking: An Overview

Farah Hussein Mohammed Jawad
University of Babylon, Hillah, Iraq

Abstract: SDN has major potential in simplifying the implementation and operation of networks and for innovative management of such networks with the application of network programmability. This study describes SDN technology and evaluates its benefits. These are followed by a brief discussion on the fundamentals of SDN architecture. The research study also reviews state of the art applications of SDN and its future prospects from available literature. Along with this it critically analyzes the various problems associated with the deployment and realization of SDN technology and the direction of relevant research.

Key words: Software defined networking, oneflow, data plane, control plane, technology, realization

INTRODUCTION

Conventional networks generally have the control and data planes merged in the same network node. The control plane deals with node configuration and path programs utilized by data. After the data paths are properly identified and assigned, they are sent to the data plane. The way data is forwarded in the hardware is then based on such control information. However, there is a serious disadvantage to this established approach. Once, the flow of data (forwarding policy) is determined and established, the system is very inflexible. The only way to make adjustments then is to change the configuration of the network devices. This is not conducive to the efforts of network operators who want to increase the scale of their networks based on changes in network traffic, the ever increasing use of mobile and cellular devices and the handling of “big data” (Sezer *et al*., 2013).

Software Defined Networking (SDN) is a novel networking approach which addresses the issues discussed above by decoupling the forwarding network hardware from control decisions. This new paradigm in networking holds the promise of radically simplify network management and boosting innovation and network evolution. The key idea is enabling software engineers to depend upon network resources just as they do on data storage and other computer related resources. SDN is unique in that the network intelligence that runs the network is very logically located in the control panel, specifically in the software-based controllers. Thus, the network devices in SDN are nothing but simple data packet forwarding instruments (the data plane) and are amenable to programs through an open interface (e.g., ForCES, OpenFlow) (Handigol *et al*., 2012).

SDN traces its origins in the effort to design a better networking solution for facilitating innovation and enabling simple programmatic control of the flow and path of network data. As illustrated in Fig. 1, the decoupling of the forwarding hardware from the control logic facilitates the easy implementation of novel protocols and network applications, the straightforward network visualization and data management and the synergy of various middleboxes into the centralized software control. Enforcement of policies and execution of protocols are no longer needed to be implemented via distributed devices. Rather, the network is simplified into a “simple” forwarding hardware having decision-making network controller(s). The forwarding hardware consists of a flow table containing an entry and an action to take on active flows and an abstraction layer that communicates securely with a controller regarding new entries not present in the flow table.

SDN has certain important advantages over internet protocol. These include: user friendless, cost efficiency, simplicity, adaptability and scalability in any network environment. SDN enables the network administrators to have central programmable control over network traffic without the requirement for direct access to the hardware Costanzo. SDN also grants enhanced flexibility in addressing issues pertaining to policies related to intrusion detection, firewalls and load balancing. Such ease in addressing these issues makes the management of SDN networks more flexible. Additionally, SDN helps to realize the separation between the control and data planes. This enables the network administrators to easily make decisions regarding the path of network data (Koldehofe *et al*., 2012). Many researchers and practitioners have claimed that this separation of the two planes makes networking more simplified, faster, less susceptible to data overload and much more user friendly (Koldehofe *et al*., 2012).

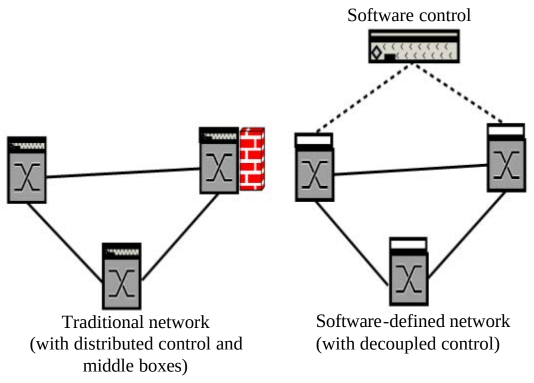

Fig. 1: Schematics of how the SDN architecture separates the control logic from the forwarding hardware and facilitates the integration of middleboxes, simpler policy management and novel functionalities (Nunes *et al*., 2014)

Indeed, these benefits of SDN have paved the way for rapid growth of this novel paradigm. However, some research challenges are still to be met (Nunes *et al*., 2014). This study, therefore, engages in an in-depth review of relevant literatures dealing with issues and challenges in SDN implementation, current and future applications and direction of previous research. This critical review will therefore enable more useful insight into the different techniques which should be used and the future orientation of SDN research.

MATERIALS AND METHODS

SDN basic architecture: The SDN architecture as implemented by OpenFlow protocol is described as:

Switches: Switches in SDN are usually considered as basic forwarding hardware which is accessible through an open interface. This is because the control logic and algorithms are off-loaded to a controller.

The communication between a controller and a switch is established via. the OpenFlow protocol. In this protocol, a group of defined messages is exchanged between the two entities through an encrypted communication channel. The OpenFlow protocol, therefore, enables remote access to add, update or delete flow entries from a particular switch's flow tables. This can be achieved both reactively (in response to a data packet's arrival) or proactively (Nunes *et al*., 2014).

SDN controller: SDN controller interfaces between the application layer and the network devices. These controllers can be programmed for taking decisions regarding data routing. The SDN controllers can then relay the decisions to every network device through OpenFlow communication portal. OpenFlow helps to update the flow tables associated with the relevant switches which are then used by the network devices to properly channel data packet's flows (Koldehofe *et al*., 2012).

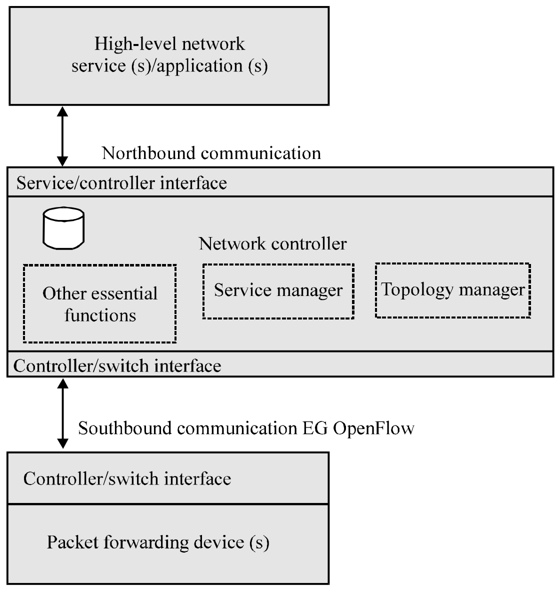
High-level network service (s)/application (s)
Northbound communication
Service/controller interface
Network controller
Other essential functions
Service manager
Topology manager
Controller/switch interface
Southbound communication EG OpenFlow
Controller/switch interface
Packet forwarding device (s)

Fig. 2: A controller with a "Northbound" and a "Southbound" interface

Figure 2 demonstrates graphically that a controller which acts as a network operating system has to use a minimum of 2 interfaces: a "Southbound" interface which will facilitate switches to communicate with the controller and a "Northbound" interface that presents a programmable API to network control and high-level policy applications/services.

Communication protocol (OpenFlow): OpenFlow helps remote access of flow tables which instruct switches and routers to properly route network traffic though software-based applications. These flow tables give the network administrators the ability to rapidly change the network layout and data traffic flow. The OpenFlow specification is controlled and published by a nonprofit organization known as Open Network Foundation (ONF). This foundation is governed by a board which comprises representatives from seven companies that own and operate the largest and most extensive networks in the world (Deutsche Telekom, Facebook, Google, Microsoft, Verizon, Yahoo and NTT). The ONF grants license of the trademark "OpenFlow Switching" to network providers who follow this specified standard (Nunes *et al*., 2014).

Table 1: SDN applications

Applications	Example	Benefits	Researchers
Enterprise and campus environments	Universities	Helps to implement and regulate network policies as well as to observe network activities and adjust network performance Simplifies the network by eliminating middleboxes and consolidating their functionality within the network controller. Example: NAT, firewalls, load balancers and network access control Passarella (2012) Enables integrated control and management of data for complex middleboxes whose functionalities cannot be implemented directly without adversely affecting network performance (e.g., deep packet inspection)	Nunes *et al.* (2014) Ghorbani and Caesar (2012), Yeganeh and Ganjali (2012) Yap *et al.* (2010)
Data centers environments	Service providers and clouds vendors such as: Amazon, Google, MSN, Yahoo, Badu, IBM, AT&T and Verizon	An SDN can route data traffic through a wireless network from a source point to a destination point on the basis of real-time status of all network elements and the policies defined for each of the points A single high-level program (API) can be implemented to decouple and control the data and the control planes through a network controller device depending upon the requirements of the particular business The SDN enables exceptionally large scale growth in the use and scalability of their data center network equipment An SDN is far more flexible than current network architectures and can be used to address the changing needs of a new business In very large and dynamic enterprises, the capability to handle data solely on the basis of internal service policies helps facilitate the attainment of competitive business advantages over networks which solely rely on IP protocols The cloud network providers incur large operating costs related to data transport, storage, and network administration. These can be lowered by the implementation of SDN. Then virtual machines can replace manual device-by-device configurations which are only achieved by employing a large group of network administrators The capability to channel and manage data over designated network channels and slot information can help the business to lower bandwidth consumption in order to reduce the number and expenses of dedicated circuits from global network providers OpenFlow controllers help network administrators to implement policies to drop data packets that threaten network security and vulnerabilities by using distributed denial of service attacks A single high-level program (API) can be utilized to decouple and control both data and control planes with the aid of network controller devices as per the demands of a given business	Nunes *et al.* (2014) Takacs *et al.* (2013) Dely *et al.* (2011) Heller *et al.* (2012) Suresh *et al.* (2012)
Home and small business environments		SDNs helps to improve the security of enterprises with high amounts of data traffic, yet, low latency requirements by the use of domain isolation within a single data center SDNs increase the firewall and security capabilities of networks by enabling centralized control from a single control plane of many end-devices The SDN controller can also be setup to work as a proxy for other network applications in order to monitor and control network device accesses SDNs can also be utilized for inserting network services into network traffic as it flows among network devices through the use of "service chaining" "Service chaining" can also be used to implement Virtual LAN "VLAN" instructions in SDN. This can provide Access Control Lists (ACLs) for enforcement of network security	Nunes *et al.* (2014) Takacs *et al.* (2013) Dely *et al.* (2011) Heller *et al.* (2012) Suresh *et al.* (2012)
Infrastructure-based wireless access networks such as cellular, WiFi	open roads project	Dely *et al.* (2011) discussed that SDN-based wireless Architecture facilitates the users to freely navigate across various wireless infrastructures which are owned and operated by different service providers. They employed a testbed using OpenFlow-enabled wireless resources like WiFi, APs and WiMAX base stations controlled by NOX and flowvisor controllers and demonstrated that there was improvement in the performance of these handover events	Dely *et al.* (2011)

SDN applications: SDN can be applied in several different types of networked environments. The separation of the control and data planes means that sprogrammable networks can facilitate customized controls, the ability to abolish middleboxes as well as simplified development and implementation of innovative network services and protocols. Table 1 lists different network environments in which SDN solutions have either been proposed or deployed.

RESULTS AND DISCUSSION

SDN challenges and issues: SDN has great potential for the simplification of network implementation and management. Additionally, it can help to lower the total cost of managing enterprise and carrier networks by enabling programmable network services. However, there are some persistent problems. This study focuses on four specific issues related to the implementation of SDN.

Performance vs. flexibility: Performance refers to the speed of processing in a network node in terms of both throughput and data latency. Programmability is defined as the ability of a system to alter and/or follow a new body of instructions in order to change its function (Sezer *et al*., 2013). Flexibility, on the other hand is the capacity of a system to support novel and unforeseen features (e.g., applications, protocols and security measures, etc).

An exhaustive review of this programmability/ performance trade-off in data processing points to a hybrid approach as the most effective way of implementing SDN. The principal SDN node functions can be broken down into groups of subfunctions in such a way that feature-specific technologies (within or across nodes) can be utilized for the best performance vs. programmability trade-off with regards to power dissipation, cost and scalability (Nunes *et al*., 2014).

For example, the development of a network platform on the basis of custom-built devices (e.g., PLDs/ASSPs) integrated with NPUs/NFPs and a CPU/GPP is a viable hybrid programmable architecture (Sezer *et al*., 2013). This type of platform can handle fast forwarding of data on existent flows in the network along with programmability and controlled processing for encapsulated traffic and new flows.

Scalability: This can be divided broadly into controller and network node scalability. This study focuses on controller scalability where three specified problems have been determined. The first is the issue related to data latency which results from the exchange of network information among many nodes and a single controller. The second is the way SDN controllers communicate with each other by utilizing east and westbound APIs. The third and final challenge is the scale and functioning of the back-end database of a controller (Yeganeh and Ganjali, 2012).

The first challenge can be met by using a distributed or peer-to-peer infrastructure for the controller. This would help share the communication load of the controller. However, this technique does not address the second problem discussed above. To solve this second problem which is related to inter-controller communication, an overall network view is a must. In a pure SDN environment, a single controller or a group of controllers provide the control plane which service and control a large number of data forwarding nodes. This enables a systemwide view of network resources. Other methods which fit the aims of SDN with the existent routing protocols consist of an extra orchestration layer containing an API. This can be used by network applications to obtain the desired performance from the transport layer. Many organizations have proposed the extension of the Application Layer Traffic Optimization (ALTO) Model. In this model, the ALTO server contains aggregated information which is then accessible to individual controllers by a designated link. The express aim of the ALTO is to divert applications to one of many hosts which are able to provide the required resources. proposed a method of supporting the global network view by using a vertical network architecture with bidirectional information flow between each SDN controller and the ALTO server (Tootoonchian and Ganjali, 2010). Thus, ALTO with SDN is an effective way to enhance network application performance.

For realizing full scalability in SDN, an evolutionary approach to network programmability is essential. For instance such a hybrid architecture can resolve a large volume of queries in the node CPU. This would otherwise be diverted to the network controller for processing purposes. This can, therefore, reduce the size of the database at the controller and at the same time lower the communication between the controller and its nodes.

Security: Potential weaknesses in network security exist across the entire SDN platform. At the level of controller-application issues exist related to the authentication and the authorization mechanisms. These mechanisms facilitate several organizations to access network resources at the same time ensuring effective protection of these vital resources (Nunes *et al*., 2014). Additionally, the lack of a strong, secure controller platform can encourage network intruders to disguise themselves as controllers and damage the network. Furthermore, introduction of open interfaces in SDN and known protocols to simplify network programming has increased the susceptibility of these networks to attackers. Complete knowledge on the way to control the network with access to the controller, the operation of the network can rapidly and easily be damaged by the attacker.

The benefit of this SDN architecture is that it facilitates very reactive security monitoring, analysis and response system. From the security perspective SDN can support (Sezer *et al*., 2013):

Network forensics: Enable fast and straightforward, adaptive threat determination and management of data through a cyclic intelligence from the network, analyzing it, updating policy and then reprogramming to optimize network performance.

Alteration of policies for network security: Allows the user to develop a security policy and disseminate it to all network infrastructure components. This reduces the rate of configuration mismatch and conflict of policies in the entire network infrastructure.

Insertion of security services: Enable the implementation of security service where the use of firewalls and Intrusion Detection Systems (IDSs) can be applied to specific traffic in the policies of a concerned organization.

Interoperability: It is more effective to deploy a straightforward and completely new infrastructure on the basis of SDN technology. To accomplish this implementation, all components and resources of the network must be compatible with SDN. The change to SDN demands simultaneous support of SDN and the legacy equipment. The IETF Path Computation Element (PCE) (Nunes *et al*., 2014) can be used in this respect for gradual or partial migration to SDN. PCE path computation component of the network is shifted from the networking node to a centralized role while the conventional network nodes which do not use PCE will keep on using the existent path computation functions. Additionally, the development is necessary to attain a hybrid SDN infrastructure where the conventional, SDN-enabled and hybrid network nodes can work together without conflict. Such interoperability needs the support of an effective protocol which introduces the requirements for SDN communication interfaces and at the same time enables backward compatibility with the already existent IP routing and Multiprotocol Label Switching (MPLS), control plane technologies. Introducing a new network protocol dictates that standardization considerations are taken into account for the best performance. The research of the IETF, ETSI, ONF and other industry working groups must be synchronized, so that, maximum advantage of the established standards can be harnessed while developing the most efficient standards to aid in the transition from conventional to SDN.

SDN current research direction: This study highlights the previous research focusing on addressing SDN from different perspectives.

Many researchers have investigated the scalability, performance and security issues concerning the network controller and the design of the network switch. Curtis *et al.* (2011) proposed the management of "short-lived" flows in switches and "long-lived" flows in the controller. He proposed this in the transition of the flow setup delay and controller overhead. Mogul and Congdon (2012), advocated changing the counters on ASIC by a stream of rule-matching records and processing them in the CPU. This should be done to enable efficient access to the counters. Flare is a new network node model and deals with "deeply programmable networks" that help to realize programmability in the data and control planes as well as the proper interfacing between the two. Handigol *et al.* (2012) discussed the salient features in the design of network controllers including hierarchical types.

Most current research dealt with the use of SDN with respect to single administrative domain. This is very compatible with the SDN's logically centralized control model. However, decentralized network environments such as the internet, require a logically distributed control plane. This will allow participating Autonomous Systems (ASes) to be controlled independently by their own (logically centralized and yet, physically separated) controllers. Until now, a few studies have investigated this idea of a software-defined internet. For instance, Rais *et al.* (2011) proposed a software-defined internet architecture which uses MPLS and the difference between network edge and core. This is done to efficiently separate the task between inter-domain and intra-domain elements. Only the routers in the network boundary and their corresponding controllers in an individual domain are concerned in an inter-domain task. Therefore, changes in inter-domain service models are restricted to the modifications in the software at the inter-domain controllers rather than in the entire infrastructure.

Some proposals, e.g., Foster *et al.* (2011) Voellmy *et al.* (2012) recommended the deployment of a specific network configuration language in order to express the network policies. For instance, Voellmy *et al.* (2012) advocated a network policy layer superimposed on the established controllers in order to interface with configuration files, GUIs and external sensors. This network policy layer will be concerned with the conversion of high-level policies to data flow constraints for a particular controller. By Kim and Feamster (2013), network configuration and management techniques are discussed which deal with making changes to the condition and state of the wireless network, supporting the configuration of the network and definitions of the policies as well as ensuring visibility and control on the jobs related to network diagnostics and problem solving. They also discussed the use of a Northbound interface through a policy layer and the use of a high level programming language like Procera.

CONCLUSION

In this study, we provided an overview of programmable networks and critically examined the emergent field of Software-Defined Networking (SDN). We discussed the history of programmable networks, from the inception of the first concepts to the most current developments. In particular, the study talked about the SDN architecture in depth along with the established standards for OpenFlow (Nunes *et al*., 2014). We presented, present SDN implementations and the testing platforms and examined the network services which have their origin on the basis of the SDN paradigm. Finally, the study concluded with an in-depth discussion on the future orientation of SDN, from the support for heterogeneous networks to Information Centric Networking (ICN).

REFERENCES

Curtis, A.R., J.C. Mogul, J. Tourrilhes, P. Yalagandula and P. Sharma *et al*., 2011. DevoFlow: Scaling flow management for high-performance networks. Proceedings of the 2011 International Conference on ACM SIGCOMM (SIGCOMM '11), August 15-19, 2011, ACM, New York, USA., ISBN:978-1-4503-0797-0, pp: 254-265.

Dely, P., A. Kassler and N. Bayer, 2011. Openflow for wireless mesh networks. Proceedings of the 2011 20th International Conference on Computer Communications and Networks (ICCCN), July 31-August 2, 2011, IEEE, Maui, Hawaii, ISBN:978-1-4577-0637-0, pp: 1-6.

Foster, N., R. Harrison, M.J. Freedman, C. Monsanto and J. Rexford *et al*., 2011. Frenetic: A network programming language. ACM. Sigplan Not., 46: 279-291.

Ghorbani, S. and M. Caesar, 2012. Walk the line: Consistent network updates with bandwidth guarantees. Proceedings of the 1st International Workshop on Hot Topics in Software Defined Networks, August 13, 2012, ACM, New York, USA., ISBN:978-1-4503-1477-0, pp: 67-72.

Handigol, N., B. Heller, V. Jeyakumar, D. Mazieres and N. McKeown, 2012. Where is the debugger for my software-defined network?. Proceedings of the 1st International Workshop on Hot Topics in Software Defined Networks, August 13, 2012, ACM, New York, USA., ISBN:978-1-4503-1477-0, pp: 55-60.

Heller, B., R. Sherwood and N. McKeown, 2012. The controller placement problem. Proceedings of the 1st International Workshop on Hot Topics in Software Defined Networks (HotSDN '12), August 13, 2012, ACM, New York, USA., ISBN:978-1-4503-1477-0, pp: 7-12.

Kim, H. and N. Feamster, 2013. Improving network management with software defined networking. IEEE. Commun. Mag., 51: 114-119.

Koldehofe, B., F. Durr, M.A. Tariq and K. Rothermel, 2012. The power of software-defined networking: Line-rate content-based routing using OpenFlow. Proceedings of the 7th International Workshop on Middleware for Next Generation Internet Computing, December 3-7, 2012, ACM, New York, USA., ISBN:978-1-4503-1607-1, pp: 1-3.

Mogul, J.C. and P. Congdon, 2012. Hey, you darned counters!: Get off my ASIC!. Proceedings of the 1st International Workshop on Hot Topics in Software Defined Networks, August 13, 2012, ACM, New York, USA., ISBN:978-1-4503-1477-0, pp: 25-30.

Nunes, B.A.A., M. Mendonca, X.N. Nguyen, K. Obraczka and T. Turletti, 2014. A survey of software-defined networking: Past, present and future of programmable networks. IEEE Commun. Surveys Tutorials, 16: 1617-1634.

Passarella, A., 2012. A survey on content-centric technologies for the current Internet: CDN and P2P solutions. Comput. Commun., 35: 1-32.

Rais, R.N.B., M. Mendonca, T. Turletti and K. Obraczka, 2011. Towards truly heterogeneous internets: Bridging infrastructure-based and infrastructure-less networks. Proceedings of the 2011 3rd International Conference on Communication Systems and Networks (COMSNETS 2011), January 4-8, 2011, IEEE, Bangalore, India, ISBN:978-1-4244-8952-7, pp: 1-10.

Sezer, S., S. Scott-Hayward, P.K. Chouhan, B. Fraser and D. Lake *et al*., 2013. Are we ready for SDN? Implementation challenges for software-defined networks. IEEE. Commun. Mag., 51: 36-43.

Suresh, L., J. Schulz-Zander, R. Merz, A. Feldmann and T. Vazao, 2012. Towards programmable enterprise WLANS with Odin. Proceedings of the 1st International Workshop on Hot Topics in Software Defined Networks (HotSDN '12), August 13, 2012, ACM, New York, USA., ISBN:978-1-4503-1477-0, pp: 115-120.

Takacs, A., E. Bellagamba and J. Wilke, 2013. Software-defined networking: The service provider perspective. Ericsson Rev., 2: 2-8.

Tootoonchian, A. and Y. Ganjali, 2010. Hyperflow: A distributed control plane for openflow. Proceedings of the 2010 Internet Network Management Conference on Research on Enterprise Networking (INM/WREN'10), April 27, 2010, USENIX Association, Berkeley, California, USA., pp: 3-3.

Voellmy, A., H. Kim and N. Feamster, 2012. Procera: A language for high-level reactive network control. Proceedings of the 1st International Workshop on Hot Topics in Software Defined Networks, August 13, 2012, ACM, New York, USA., ISBN:978-1-4503-1477-0, pp: 43-48.

Yap, K.K., R. Sherwood, M. Kobayashi, T.Y. Huang and M. Chan *et al*., 2010. Blueprint for introducing innovation into wireless mobile networks. Proceedings of the 2nd ACM SIGCOMM Workshop on Virtualized Infrastructure Systems and Architectures (VISA '10), September 3, 2010, ACM, New York, USA., ISBN:978-1-4503-0199-2, pp: 25-32.

Yeganeh, S.H. and Y. Ganjali, 2012. Kandoo: A framework for efficient and scalable offloading of control applications. Proceedings of the 1st International Workshop on Hot Topics in Software Defined Networks, August 13, 2012, ACM, New York, USA., ISBN:978-1-4503-1477-0, pp: 19-24.