International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Software Defined Networks: Challenges, Opportunities and Trends

Hrishikesh Arun Deshpande

[1]Member of Technical Staff R&D, NetApp India Pvt. Ltd, Bangalore, India

Abstract: ***Traditional networks often consist of a static arrangement of switches, routers and other components. Such a configuration is quite inflexible and inappropriate to meet the dynamic traffic and programmable bandwidth requirements of today's applications. Moreover, with the advent of cloud computing and big data, we need a network that is agile and dynamically scalable, something that a traditional network cannot guarantee. A software defined network abstracts the lower level hardware configuration from the higher level applications by decoupling the decision making control plane from the packet forwarding data plane. Thus software defined networks are better suited to meet the dynamic requirements of today's applications. This paper gives insights into the requirements that are driving the networking community towards software defined networks, challenges involved in implementing a programmable network, opportunities and advantages offered by software defined networks vis-à-vis traditional networks and emerging trends in research and standardization of software defined networks.***

Keywords: software defined networks, flow table, OpenFlow, Network Function Virtualization, programmable bandwidth, scalability

1. Introduction

Traditional computer networks often comprise of myriad types of networking devices such as routers, network switches, firewalls, authentication gateways, network address translators (NAT) and so on. These devices are tightly coupled within the network and are guided by complex protocols and policies to handle host connectivity, network speeds and topologies [1]-[2]. But such a static arrangement in traditional networking architectures is inflexible and unsuited to meet the vastly dynamic and scalability requirements of today's users, carriers and enterprises. With the emergence of cloud services, server virtualization, mobile devices and big data, it has become imperative that the the underlying network infrastructure is programmable, agile and adjust dynamically to unpredictable traffic patterns and bandwidth requirements [2]. Software defined networks (SDN) have been touted as an enabling technology to help network administrators manage network controls by abstracting the lower level networking hardware from network services and applications [3]. This basically results in networks that are programmable and satisfy the demands of high bandwidth, scalability and dynamic computing needs of today's enterprise data centers and carriers. This makes SDN's a viable alternative to traditional networks in meeting the dynamic traffic and bandwidth requirements of today's applications [1]-[3].

2. Software Defined Networks

The static and complex nature of traditional computer networks makes it difficult for them to meet the current market requirements of mobility, scalability and high bandwidth [1]. Thus efficient network management and network utilization is challenging, forcing network administrators to resort to tedious manual configuration processes and device level management tools [2]. These techniques are quite inefficient and fail to meet the challenging requirements of current markets and have prompted the networking community to reexamine traditional network architectures and consider networks that are dynamically programmable and agile [4]. This has generated significant interest towards SDN in both the academia and industry [5].

2.1 Need for Software Defined Networks

Since traditional networks are unable to handle effectively dynamically changing traffic patterns and high bandwidth requirements, the academic community formed the Open Flow Network Center with emphasis on SDN research. Similarly, network operators and service providers formed the Open Networking Foundation, an industry driven initiative, to further promote SDN research [1]. Several factors prompted the academia and industry to consider SDN's as an alternative to traditional networks. Table 1 below summarizes the drivers that led the networking community towards SDN research [1]-[5].

Table 1: Drivers for SDN research

Sl. No	Driver	Reason
1.	High Bandwidth	Traditional Networks with static configuration of network devices are unable to meet the ever increasing bandwidth requirements of customers. SDN's which can adapt dynamically to high bandwidth requests are better suited in such cases.
2.	Dynamic traffic patterns	Users are beginning to access different databases, servers and corporate networks from different devices from anywhere and at any point of time. Thus it's not possible to assume a certain constant traffic pattern over a network and traffic often tends to be unpredictable.
3.	Scalability	As demands for bandwidth explode, it's difficult for traditional networks with fixed configurations to scale dynamically. Scaling a network results in addition of several thousands of

Licensed Under Creative Commons Attribution CC BY

		network devices greatly complicating their maintenance and increasing costs.
4.	Cloud Services	With wide acceptance of cloud based services, there's been a spurt in demand for agile applications and dynamic provisioning of resources.
5.	Network Management	Constant scaling of traditional networks results in addition of several thousands of network devices that need to configured and maintained thereby making network management quite difficult.
6.	Network Complexity	Today's networks have a myriad type of network devices and topologies that are guided by a diverse set of protocols. Each protocol serves different functions making the overall network complex and difficult to manage.
7.	Vendor independence	Since each vendor has different hardware refresh cycles, enterprises are forced to configure each vendor's equipment separately. This together with the absence of an open interface restricts enterprises from adapting their network dynamically.

2.2 Architecture of Software Defined Networks

SDN's have been defined in several ways. But in simplest terms, a software defined network can be defined as a programmable network where lower level hardware functionality is abstracted from higher level network services and applications [4]. This is accomplished by decoupling the control plane from the data plane. Figure 1 illustrates the architecture of a typical SDN network.

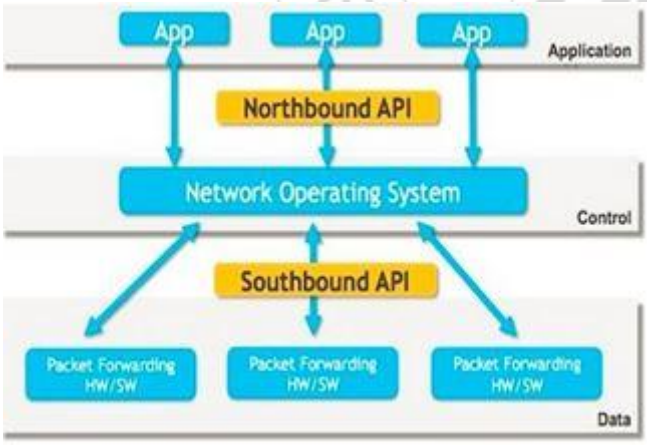

Figure 1: SDN Architecture

Control plane is the layer where decisions of the destination to which the traffic is to be sent are made while data plane is the layer that forwards the traffic to the selected destination. The data plane consists of networking hardware such as switches routers and gateways that are connected together to form the underlying networking infrastructure [6]. The data plane communicates via a southbound interface with the control plane that consists of a network operating system (OS) [7]. The network OS interacts with the underlying hardware and abstracts the network topology from the application layer. In most cases, the OS forms a part of a separate SDN controller that makes routing decisions. The application layer typically consists of network access and management applications such as access control, mobility control, traffic scheduling and energy-efficient networking. The application layer interacts with the Network OS via a northbound interface [6]. Thus the SDN controller in the control plane together with the high level applications manage traffic control and network management thereby providing multiple layers of abstraction to the outside world [7]. Both northbound and southbound API's are implemented as open interfaces making the network programmable to suit specific needs [4]. Also, since the lower level network infrastructure is abstracted from the application layer, the applications can make service requests without having to worry about network specific configurations [4], [6], [7].

A typical SDN operates by dividing the incoming network traffic into flows. In simplest terms, a flow can be a stream of packets with the same MAC address or IP address, a TCP connection, packets arriving from the same switch port or a Virtual Local Area Network (VLAN) tag [7]. The entries of individual flows are maintained in a flow table that forms the part of every network switch. The SDN controller coordinates the overall activities of network switches such as addition or removal of flow entries from switch flow tables and switch interactions [7]. Whenever a newly arriving packet doesn't match an existing flow entry, the SDN controller is responsible for deciding if it forms a part of an authentic flow. If yes, then the SDN sends the flow entry details and forwarding information to the corresponding switch which updates the same in its flow table [7]. Thus the SDN controller, that facilitates coordination between switches and takes routing decisions, is the most important component in a typical SDN [5]-[6].

3. Challenges in Implementing Software Defined Networks

Although SDN's appear like a panacea for various network scheduling and management problems such as dynamic traffic, adaptable scaling, high bandwidth etc. implementing a true SDN has several challenges and pitfalls that need to be overcome [8]. Some of the key challenges faced while implementing SDN's have been described below.

3.1 Interoperability Issues

The key challenge in successfully deploying SDN's is to ensure interoperability with existing networks. Currently there exists a vast install base of traditional networks that supports critical business operations [8]. Thus bumping off an old network and replacing it with entirely with SDN cannot happen without inducing massive network outages resulting in denial of service to customers. Hence, it's essential that SDN deployment happens gradually with both traditional networks and SDN's working together in close coordination [8].

A possible way to ensure interoperability would be to define a new protocol that not only specifies the requirement for SDN communication interfaces but also ensures backward

Licensed Under Creative Commons Attribution CC BY

compatibility with existing IP routing and MPLS control plane technologies [8]. The specifications of such a protocol need to be standardized to ensure uniform acceptance across the industry. The Internet Engineering Task Force (IETF) is currently working on developing standards for protocols, interfaces and mechanisms in order to ensure smooth interoperability of SDN's with existing networks [9].

3.2 Security Issues

Although much attention has been paid as to how SDN's can be successfully deployed and interoperate with existing networks, there has been a limited push towards research in the security aspects of SDN's from both academia and industry [10]. It's not possible to successfully deploy SDN's in the absence of proper security and intrusion detection mechanisms. Table 2 summarizes the key security threats that can affect the functioning of SDN's.

Table 2: Security Issues in SDN

Sl. No	Security Threat	Description
1.	Denial of Service Attack	If a controller is centralized, it becomes a single point of failure. Thus, an attacker can overload the SDN controller and switch memory by introducing several new and unknown flows that the controller is not designed to handle causing outages and denial of service.
2.	Switch Vulnerabilities	A malicious switch can potentially flood the controller with malicious packets, slow down network traffic or even ignore new routing requests.
3.	Controller Vulnerabilities	Since an SDN controller and Network OS form the backbone of a SDN network, vulnerabilities in the controller can be exploited to gain full control of the network.
4.	Man-in-the-middle Attacks	In the absence of Transport Layer Security(TLS), vulnerabilities in the control plane communications can be exploited to introduce man-in-the-middle attacks and severely compromise SDN security
5.	Authentication Issues	With multiple nodes accessing a single controller or a single node connected to multiple controllers, the potential for unauthorized access and illegal configuration manipulation increases.
6.	Open Interfaces	Since SDN's support open interfaces and known protocols to simplify networking, these interfaces can be exploited by an attacker to gain full control of the network.

These threats can be mitigated by implementing transport layer security (TLS) that ensures mutual authentication between SDN controllers and underlying switches [8]. Further, proper forensics need to be put in place to ensure detection of attacks and strong security policy mechanisms need to be put in place to ensure integrity and authenticity of data. Also, the usage of authentication gateways, vulnerability scanners and honeypots can be considered to deal with these threats [10]-[11].

3.3 Availability of Service

In traditional networks, if one or more of the networking devices fail, traffic can be routed through alternate paths and processed by other devices in the network to ensure continuous availability of service. But SDN's introduce an additional issue of single point of failure [12]. In a centralized SDN architecture with a single controller, if the SDN controller is compromised or rendered unusable, then this will disrupt the entire network resulting in non-availability of service to legitimate users [12]. Thus it's essential to provide a cluster of active stand-by controllers as backups to the existing controller to ensure a non-disruptive service to users [13]. A distributed SDN controller architecture can also be considered to implement a load balancing based SDN network where the traffic load can be shared by multiple controllers in the configuration. Further, path-link failures can hamper the availability of service to intended users. The SDN architecture should thus support a multipath configuration so that the controller can redirect traffic from dead links to active routes [12].

3.4 Scalability Issues

Traditional networks have a static configuration of routers, switches and other networking components that are governed by specific policies and protocols [14]. Thus scaling the network dynamically becomes quite challenging, since it requires a reconfiguration of existing networking elements, addition of new elements and make them conform to some common policy settings. This process is quite expensive and laborious. But in a SDN, control plane is decoupled from the data plane. Hence any addition of new networking elements required for scaling doesn't change the network control functions resulting in seamless scaling [14].

Although addition of new switches and routers is seamless, SDN introduces other scalability issues. Firstly, an increase in the number of switches and flows results in a proliferation of traffic that might overwhelm the SDN controller with several thousand requests [8]. This can actually result in queuing at the controller thereby slowing down the network. Further, addition of new flows can also cause scalability issues. Addition of a new flow requires setting up new entries in a flow table that maintains the records of various flows. It also involves switch updates and multiple interactions between the controller and the switches that can potentially slowdown the network. Addition of new networking components will increase the corresponding flow table updates thereby overloading the controller further compounding scalability problems [14]. Several solutions have been proposed to solve scalability issues, the prominent one being a proposal to have a multicore controller executed on a shared memory infrastructure where flow processing can be shared by multiple cores [15]. Another approach is to process a certain portion of queries at the intervening nodes itself so that a single controller is not overwhelmed with several requests [14].

Licensed Under Creative Commons Attribution CC BY

3.5 Performance Problems

As already mentioned, centralized controller architecture will overwhelm the SDN controller thereby greatly slowing down the network, resulting in serious performance degradation [16]. Possible solutions to this issue include setting up of hybrid SDN architecture with sharing of workloads by multiple controllers. Such a hybrid setup can ensure fast forwarding of existing flows [17]. But addition of a new flow can result in performance bottlenecks since the flow setup process involves multiple flow table updates, switch updates and interactions between various components [12]. Thus, flow setup needs to be optimized based on the processing time and I/O performance of the controller. Such an optimization can be used to determine the ideal time for flow setup and thus reduce the flow setup latency [17]. Hence a hybrid distributed architecture with multiple SDN controllers and an optimized algorithm for flow setup time can together reduce possible bottlenecks and improve performance [12], [16], [17].

4. Opportunities and Advantages

Despite the initial installation and implementation problems, SDN has been touted as a network of the future mainly due to its flexibility in adding new interfaces, dynamic addition of network switches and its programmable nature with respect to handling dynamic traffic flows. These advantages make SDN's an attractive alternative to traditional networks [1]-[3]. Some of the compelling opportunities provided by SDN's in different fields have been described below.

4.1 Network as a Service (NaaS)

With SDN, control plane is decoupled from the underlying network infrastructure. This abstracts the lower level network from incoming traffic flows [3]. Such an abstraction is possible by defining application layer-control plane interfaces (API's) and control plane-data plane interfaces. These interfaces facilitate communication between various layers [4]. Such a virtualization allows applications to focus on serving external traffic and effectively manage the network without worrying about the lower level switch infrastructure. Thus the underlying network itself can be presented as a service to the application layer wherein the applications process incoming traffic requests and request a service from lower layers [18].

4.2 Programmable Bandwidth

Since SDN's abstract the lower level hardware implementation from the control functions, it's possible to dynamically add new transport paths [6]. This is achieved through the addition or removal of flow paths whose information is maintained in a flow table. The flow table is then used by the SDN controller to make intelligent routing decisions based on the available active links [7]. Such a dynamic alteration of flows allows effective workload management and job distribution [19]. This automated provisioning of network bandwidth facilitates effective handling of unpredictable traffic flows and better congestion management.

4.3 Network Customization

The programmable nature of SDN's allows them to be tailored and customized to meet the requirements of specific vendors or individual companies [20]. With the decoupling of control and data planes [6], SDN's can also be customized at switch, controller and network management levels. This is achieved by dynamically altering flow paths and routing mechanisms based on specific requirements [7]. Also, SDN interfaces are being implemented as open standards. Thus, networks can be customized to allow specific policies w.r.t traffic control, Quality of Service (QoS), packet forwarding and device control mechanisms [20]. Also these policies can be further tailored for specific needs in fields such as education, healthcare, energy, banking systems among others.

4.4 Vendor Neutrality

In traditional networks, network design and operation is guided by vendor specific polices and protocols [1]. Further, the absence of any common standards and open interfaces resulted in a static, vendor dependent configuration, making it difficult for network operators to tailor the networks to meet their specific requirements [2]. But in case of SDN, communication between control, data and application layers happens via standardized, open interfaces [7]. The development of open standards for SDN has made it vendor neutral thus making it possible for operators, carries and enterprises to have common configuration and policy settings irrespective of the vendors involved [21].

4.5 Cloud Services

With a spurt in cloud based deployments, enterprises require dynamic access to applications, infrastructure and other services. SDN, with its programmable nature, allows dynamic provisioning of storage, network and computing resources [2]. Also, SDN controllers can provide for effective pooling of existing network resources ensuring better resource utilization [8]. A SDN framework called "Meridian" has been developed for cloud computing networks. This framework enables users to create and maintain a suitable topology for cloud workloads and handle dynamic traffic through virtual implementations on underlying network hardware [22].

4.6 Big Data

Today, big data involves massive parallel processing of large data sets on thousands of servers. This can place a huge bandwidth demand on networks, requiring the networks to scale to a previously unimaginable size. Traditional networks with static configurations are unable to respond effectively to such massive bandwidth requirements resulting in outages [23]. If SDN controllers are connected in a distributed configuration and have a programmable bandwidth capability through dynamic management of flows, network admins will gain the required flexibility to program their networks to suit the big data paradigm [6]-[7]. This enables network operators to effectively handle the processing of big data sets

Licensed Under Creative Commons Attribution CC BY

and the demanding bandwidth and scalability requirements that these data sets place on the network [23].

5. Recent Trends in SDN research

Ever since the idea of SDN was conceptualized, it has evoked significant interest from both academia and industry. Many path breaking innovations and research activities have been witnessed to simplify SDN implementation and justify SDN as a viable alternative to traditional networks. Further, several attempts have been made to standardize SDN architecture and communication interfaces so as to ensure a common vendor independent framework. The forthcoming sections discuss the recent trends in SDN standardization and research [5].

5.1 SDN standardization

A conglomerate of vendors, network operators and service providers established the Open Networking Foundation (ONF) which is developing an open standard for SDN called the OpenFlow Standard [24]. Implementing open interfaces for SDN makes it vendor neutral and thus facilitates better standardization. ONF has several working groups (WG) to deal with various aspects of SDN. For example, the Architecture and Framework WG is focused on defining ONF's idea of an SDN together with its interfaces. Similarly, the Configuration & Management WG has defined an OpenFlow Configuration Protocol to deal with configuration of SDN switches [24].

Similarly, the Internet Research Task Force (IRTF) which is an open international community of vendors, network designers and researchers, has established a SDN research group (SDNRG) to deal with various issues encountered in the SDN space such as cloud management and routing control [26]. SDNRG has formed the Interface to Routing System (I2RS) WG to implement interfaces for network controller, user and management applications and handle specific service requests [27].

European Telecommunications Standards Institute (ETSI) has formed an Industry Specification Group (ISG) to develop standards for Network Functions Virtualization (NFV) [28]. The group aims to virtualize network functions in order to build new networking environments. NFV, which can be seen as a complementary technology to SDN, intends to virtualize entire classes of network code functions to create Virtual Network Function (VNF) interfaces that can be orchestrated and deployed on a framework to provide communication services. NFV is a much broader technology and network virtualization can be accomplished using SDN architecture [28]. Thus, NFV can be considered as one of the use cases of SDN. Realizing the correlation between SDN and NFV, ETSI and ONF recently entered into a strategic collaboration to implement NFV using SDN architecture [29].

International Telecommunications Union (ITU) has a division called ITU Standardization Sector (ITU-T) that develops standards for telecommunications industry [30]. ITU-T has formed a Joint Coordination Committee (JCA) for SDN research called JCA-SDN. The committee is working on developing standards for SDN. JCA-SDN also keeps a track of recent activities in SDN research and their possible applications [30].

5.2 Recent Work

Agarwal deals with the issue of traffic engineering in SDN's by explaining how a centralized controller can be optimized for traffic engineering using Fully Polynomial Time Approximation Schemes (FPTAS) in order to achieve better network utilization and reduce packet loss and delays [31]. Pakzad talks about a new scheme to efficiently discover topologies in SDN's by proposing simple and practical modifications to existing topology discovery mechanisms so as to reduce control overhead and improve efficiency [32]. Adrichem talks about an efficient recovery mechanism for SDN's [33]. Existing recovery mechanisms involve detection of a failed link, sending failure information to the controller and then choosing a new active link for blocked traffic. Such a technique is inefficient and has significant downtime. The new technique by Adrichem proposes to reduce the recovery time by using a failover scheme with per-link- bi-directional packet forwarding detection using preconfigured primary and backup paths [33].

Karakus explains how scalability is a prominent challenge in implementing SDN's and proposes an approach based on levels to solve this issue. In this solution, multiple controllers together with their networks reside in various levels and a main controller co-ordinates their activities like a broker [34]. Gurbani talks about abstracting network topology to the higher level applications using Application Layer Traffic Optimization (ALTO) protocol. ALTO provides applications with an abstract view of the underlying network and thus allows them to leverage the network without worrying about the network's internal policies or configuration [35].

Further, Blendin deals with software defined network service chaining [36]. Service chaining allows forming services by combining multiple service functions. Blendin explains how service chaining can be implemented in telecommunication networks using SDN OpenFlow architecture [36]. Singh explains how a Denial of Service (DoS) attack can affect the working of a SDN by exploiting the vulnerabilities in the flow table and proposes a solution to mitigate such an attack in its initial stage itself before it harms the network [37]. Cong proposes a software defined on-chip network (SDNoC) which is an on-chip networking technology using SDN. SDNoC decouples the control logic from the underlying network hardware so that applications can configure their network as per requirements [38].

Lombardo proposes a performance evaluation of SDN's using an analytical tool that takes into account the functions requested by traffic flows and the Quality of Service (QoS) that the network is able to provide them [39]. This approach accounts for processing capabilities of network nodes, routing and statistical characterization of flows requesting each network virtualization function (NFV). Further, Rad talks about a low latency SDN that minimizes virtualization

Licensed Under Creative Commons Attribution CC BY

overhead and provides high performance for multi-tenant cloud infrastructures [40].

6. Conclusion

Thus we can conclude by saying that traditional networks with their static configuration of routers and switches are inflexible and cannot dynamically adapt to the unpredictable traffic patterns and programmable bandwidth requirements of today's applications. Moreover, the advent of big data and cloud services has triggered a demand for agile access to network applications, dynamic scalability and self-provisioning something that traditional networks are unable to deliver. SDN, which is an emerging technology, decouples the decision making control plane from the packet forwarding data plane by abstracting the underlying hardware from the application layer. Such an additional level of abstraction allows applications to request services from SDN without having to worry about internal configuration details and policies. This flexibility allows enterprises to handle dynamically changing traffic patterns and programmable bandwidth requirements of today's applications. Also, the implementation of open interfaces promotes vendor neutrality.

However, implementing SDN's has some challenges such as interoperability with existing traditional networks, security issues arising out of open interfaces and centralized controllers, scalability concerns, performance problems and ensuring continuous availability of service if the SDN controller is compromised. Despite these challenges, SDN's provide tremendous opportunity in realizing the paradigm of network as a service (NaaS), programmable bandwidth, vendor neutrality, network customization and is touted to be a future network for cloud services and big data applications. These opportunities have generated significant interest in SDN's from both academia and industry. Various organizations such as ONF, ITU-T, ETSI and IRTF are currently involved in furthering SDN research and standardization.

References

[1] B. Nunes, M. Mendonca, X. Nguyen, K. Obraczka, T. Turletti, "A survey of software-defined networking: Past, present, and future of programmable networks," In IEEE Communications Surveys & Tutorials, vol. 16, no. 3 pp. 1617-1634, 2014.

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

[3] J. Tourrilhes, P. Sharma, S. Banerjee and J. Pettit, "The Evolution of SDN and OpenFlow: A Standards Perspective," IEEE Computer Society, vol. 47, no 11, pp. 22-29, 2014.

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

[5] M. Jammal, T. Singh, A. Shami, R. Asal, Y. Li, "Software defined networking: State of the art and research challenges," Computer Networks, Elsevier, vol. 72, pp. 74-98, July 2014.

[6] C. Rothenberg, R. Cunha, J. Bailey, M. Winter, C. Correa, S. Lucena, M. Salvador, T. Nadeau, "When open source meets network control planes," IEEE Computer, vol. 47, no. 11, pp. 46-54, 2014.

[7] OpenFlow Switch Consortium, OpenFlow Spec. v1.3.0, https://www.opennetworking.org/images/stories/downloads/sdnresources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf, 2012. [Accessed: Aug. 31, 2015]

[8] S. Sezer, S. Scott-Hayward, P. Chouhan, B. Fraser, D. Lake, J. Finnegan, N. Viljoen, M. Miller, N. Rao, "Are we ready for SDN? Implementation challenges for software-defined networks", IEEE Communications Magazine, vol. 51, no. 7, pp. 36–43, 2013.

[9] H. Kim, N. Feamster, "Improving network management with software defined networking," IEEE Communications Magazine, vol. 51, no. 2, pp. 114-119, 2013.

[10] S. Scott-Hayward, G. O'Callaghan, S. Sezer, "Sdn security: A survey," In Proceedings of SDN Future Networks and Services (SDN4FNS), IEEE, pp. 1-7, 2013.

[11] D. Kreutz, F. M. Ramos, and P. Verissimo, "Towards secure and dependable software-defined networks," In Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking, ACM, pp. 55–60, 2013.

[12] Ashton, Metzler, and Associates, "Ten Things to Look for in an SDN Controller", Technical Report, 2013.

[13] A. Tootoonchian, Y. Ganjali, "HyperFlow: A Distributed Control Plane for OpenFlow," In Proceedings of the 2010 Internet Network Management Conference on Research on Enterprise Networking, pp. 3-3, 2010.

[14] S. Yeganeh, A. Tootoonchian, Y. Ganjali, "On scalability of software-defined networking," IEEE Communications Magazine, vol. 51, no. 2, pp. 136–141, 2013.

[15] A. Voellmy, J.C. Wang, "Scalable Software-Defined Network Controllers," In Proceedings of ACM SIGCOMM Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, pp. 289–290, 2012.

[16] A. Tootoonchian, S. Gorbunov, "On Controller Performance in Software-Defined Networks," In Proceedings of 2nd USENIX Conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services, vol. 54, pp. 10, 2012.

[17] J. Mogul, L. Tourrilhes, P. Yalagandula, P. Sharma, A. Curtis, S. Banerjee, "DevoFlow: Cost-Effective Flow Management for High-Performance Enterprise Networks," In Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks(HotnetsIX), pp. 1, 2010.

[18] P. Costa, M. Migliavacca, P. Pietzuch, A. Wolf, "NaaS: Network-as-a-Service in the Cloud," In Proceedings of the 2nd USENIX conference on Hot Topics in

Licensed Under Creative Commons Attribution CC BY

Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE), vol. 12, 2012.
[19] M. Altufaili, "Intelligent Network Bandwidth Allocation using SDN (Software Defined Networking)," International Journal of Engineering Research & Technology (IJERT) www.ijert.org, vol. 4, no 7, pp. 493-496, 2015.
[20] S. Liu, "Software-Defined Networking: Opportunities And Challenges", forbes.com, para 7, Feb. 25, 2015. [Online]. Available: http://www.forbes.com/sites /huawei/2015/02/24/software-defined-networking-opportunities-and-challenges/ [Accessed: Aug. 30 2015].
[21] "Software-Defined Networking (SDN) Definition", opennetworking.com, para 6. [Online]. Available: https://www.opennetworking.org/sdn-resources/sdn-definition. [Accessed: Aug. 29, 2015].
[22] M. Banikazemi, D. Olshefski, A. Shaikh, J. Tracey, H. Wang, "Meridian: an SDN Platform for Cloud Network Services," IEEE Communications Magazine, vol.51, no.2, pp.120–127, 2013.
[23] I. Monga, E. Pouyoul, C. Guok, "Software defined networking for big-data science," In High Performance Computing, Networking, Storage and Analysis (SCC), IEEE, pp. 1629-1635, 2012.
[24] ”ONF Overview”, opennetworking.org, para 3. [Online]. Available: https://www.opennetworking.org/about/onf-overview. [Accessed: Aug. 28, 2015].
[25] “SDNRG Background”, irtf.org, para 3, Oct. 1 2013. [Online]. Available: https://irtf.org/sdnrg [Accessed: Aug. 28, 2015].
[26] D. Meyer, "The Software-Defined-Networking Research Group," Internet Computing, IEEE, vol.17, no.6, pp.84-87, 2013.
[27] S. Hares and R. White, "Software-Defined Networks and the Interface to the Routing System (I2RS)," IEEE Internet Computing, vol. 17, no. 4, pp. 84-88, 2013.
[28] “Network Functions Virtualization”, portal.etsi.org, para 2, [Online]. Available: http://portal.etsi.org/portal/server.pt/community/NFV/367. [Accessed: Aug. 28, 2015].
[29] “Open Networking Foundation and ETSI announce strategic collaboration for SDN support of NFV”, etsi.org, para 3, Mar. 14, 2014. [Online]. Available: http://www.etsi.org/news-events/news/764-2014-03-onf-and-etsi-announce-strategic-collaboration-for-sdn-support-of-nfv. [Accessed. Aug. 28, 2015].
[30] “Joint Coordination Activity on Software-Defined Networking”, itu.int, para 1, Nov. 18 2012. [Online]. Available: http://www.itu.int/en/ITU-T/sdn/Pages/default.aspx. [Accessed. Aug 28. 2015].
[31] S. Agarwal, M. Kodialam, and T. V. Lakshman. "Traffic engineering in software defined networks." In IEEE INFOCOM Proceedings, pp. 2211-2219, 2013.
[32] F. Pakzad, M. Portmann, W. Tan, J. Indulska, "Efficient topology discovery in software defined networks," In Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on, pp. 1-8. IEEE, 2014.
[33] V. Adrichem, LM Niels, J Benjamin, V. Asten, F. Kuipers, "Fast recovery in software-defined networks," In Third European Workshop on Software Defined Networks (EWSDN), pp. 61-66, 2014.
[34] M. Karakus, A. Durresi, "A Scalable Inter-AS QoS Routing Architecture in Software Defined Network (SDN)," In 29th International Conference on Advanced Information Networking and Applications (AINA), IEEE, pp. 148-154, 2015.
[35] V. Gurbani, M. Scharf, T. Lakshman, V. Hilt, "Abstracting network state in Software Defined Networks (SDN) for rendezvous services", Workshop on Software Defined Networks, co-located with IEEE ICC, June 2012.
[36] J. Blendin, J. Ruckert, N. Leymann, G. Schyguda, D. Hausheer, "Position paper: Software-defined network service chaining," In Third European Workshop on Software Defined Networks (EWSDN), IEEE, pp. 109-114, 2014.
[37] S. Singh, R.A. Khan, A. Agarwal, "Prevention mechanism for infrastructure based Denial-of-Service attack over software defined network," In International Conference on Computing, Communication & Automation (ICCCA), IEEE, pp. 348-353, 2015.
[38] L. Cong, W. Wen, W. Zhiying, "A configurable, programmable and software-defined network on chip," In IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA), pp. 813-816, 2014.
[39] A. Lombardo, A. Manzalini, V. Riccobene, G. Schembra, "An analytical tool for performance evaluation of software defined networking services," In Network Operations and Management Symposium (NOMS), IEEE, pp. 1-7, 2014.
[40] P. Rad, R. Boppana, P. Lama, G. Berman, M. Jamshidi, "Low-latency software defined network for high performance clouds," In 10th System of Systems Engineering Conference (SoSE), IEEE, pp. 486-491, 2015.

Licensed Under Creative Commons Attribution CC BY