Volume 6, Issue 6, June 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Augmentation of Network Administration through Software Define Networks (SDN)

Santosh Kumar Srivastava*
M.Tech (CSE) Student
BRCM College of Engineering & Technology,
Behal, Haryana, India

Basant Sah
Asst Prof (Dept. of CSE)
BRCM College of Engineering & Technology,
Behal, Haryana, India

Abstract: ***To Organize the Network infrastructure it is really very typical and challenging issue. Managing a Secure Communication Network by Network managers should have ability to hold with law-level vendor-specific configuration to apply complex high-level policies. In earlier if we want to control & manage services over the existing network Architecture where the Appliances & peripherals are from the various specific vendors this was a complex and challenging issue for the same. So the Network Administrator have only options to do the same either replace the Appliance vendor and make same Appliance in his Network or fight to control. But now a day a new approach in networking is software defined networking (SDN), it proposes. Isolate the data plane & the control plane. The SDN provide the mechanism to make network devices (Switches, Routers) in the data plane simple packet forwarding device & act as a logically centralized software program to manage the action of the entire existing network. It proposes a new outlook for network configuration & management methodology. In this paper will distinguish difficulty regarding the current network Architecture configuration management mechanism & the SDN management mechanism.***

Keywords: ***Network, Rooter, Firewall, Switches, Configuration, Diagnosis, Infrastructure.***

I. INTRODUCTION

In the field of Computer Networks the Role of network administrator are very dynamic and complex, that's why to configure, manage, control monitor of the network infrastructure is always becomes a challenging issue. In the existing network there have several tasks doing like switches, routers, firewalls configuration and a number of services related task on the time based or reputedly. And sometimes most of the events accord in the same time period and different locations. The major role of the network administrator is to configure appliances, maintain all services, bandwidth allocation, monitoring of the allotted bandwidth and the troubleshooting of the same, and all the same are not one time tasks, because serviced always varies in the regular routine therefore the load on the network peripherals are also varies. The mechanisms which we have to solve problems are not sufficient and dynamic so the solution becomes very complex. The present mechanism not provide automatic load balance then on a single device have more load as desire and the another one is less utilize as desire.

Software Defined Networking (SDN) is facilitating organizations to speed up application redeployment and serving, dramatically reduction of information technology costs through centralization of workflow computerization. SDN technology provides centralized architectures by delivering a computerized platform as GUI based application. SDN facilitates as centralization of the network, resource enhancement, flexibility, monitoring IT infrastructure, costs and overhead.

In this paper we describe that how can SDN enhance the better mechanisms for the existing network infrastructure where the network devices are from various vendors. While many articles and papers have research about the merits of the implementation of SDN in the network.

II. LIMITATIONS OF CURRENT NETWORKING TECHNOLOGIES

Meeting current market requirements is virtually impossible with traditional network architectures. Faced with flat or reduced budgets, enterprise IT departments are trying to squeeze the most from their networks using device-level management tools and manual processes. Carriers face similar challenges as demand for mobility and bandwidth explodes; profits are being eroded by escalating capital equipment costs and flat or declining revenue. Existing network architectures were not designed to meet the requirements of today's users, enterprises, and carriers; rather network designers are constrained by the limitations of current networks, which include:

Complexity that leads to stasis:

Networking technology to date has consisted largely of discrete sets of protocols designed to connect hosts reliably over arbitrary distances, link speeds, and topologies. To meet business and technical needs over the last few decades, the industry has evolved networking protocols to deliver higher performance and reliability, broader connectivity, and more

stringent security. Protocols tend to be defined in isolation, however, with each solving a specific problem and without the benefit of any fundamental abstractions. This has resulted in one of the primary limitations of today's networks: complexity. For example, to add or move any device, IT must touch multiple switches, routers, firewalls, Web authentication portals, etc. and update ACLs, VLANs, quality of services (QoS), and other protocol-based mechanisms using device-level management tools. In addition, network topology, vendor switch model, and software version all must be taken into account. Due to this complexity, today's networks are relatively static as IT seeks to minimize the risk of service disruption. The static nature of networks is in stark contrast to the dynamic nature of today's server environment, where server virtualization has greatly increased the number of hosts requiring network connectivity and fundamentally altered assumptions about the physical location of hosts. Prior to virtualization, applications resided on a single server and primarily exchanged traffic with select clients. Today, applications are distributed across multiple virtual machines (VMs), which exchange traffic flows with each other. VMs migrate to optimize and rebalance server workloads, causing the physical end points of existing flows to change (sometimes rapidly) over time. VM migration challenges many aspects of traditional networking, from addressing schemes and namespaces to the basic notion of a segmented, routing-based design. In addition to adopting virtualization technologies, many enterprises today operate an IP converged network for voice, data, and video traffic. While existing networks can provide differentiated QoS levels for different applications, the provisioning of those resources is highly manual. IT must configure each vendor's equipment separately, and adjust parameters such as network bandwidth and QoS on a per-session, per-application basis. Because of its static nature, the network cannot dynamically adapt to changing traffic, application, and user demands. Inconsistent policies: To implement a network-wide policy, IT may have to configure thousands of devices and mechanisms. For example, every time a new virtual machine is brought up, it can take hours, in some cases days, for IT to reconfigure ACLs across the entire network. The complexity of today's networks makes it very difficult for IT to apply a consistent set of access, security, QoS, and other policies to increasingly mobile users, which leaves the enterprise vulnerable to security breaches, non-compliance with regulations, and other negative consequences.

Vendor dependence:

Carriers and enterprises seek to deploy new capabilities and services in rapid response to changing business needs or user demands. However, their ability to respond is hindered by vendors' equipment product cycles, which can range to three years or more. Lack of standard, open interfaces limits the ability of network operators to tailor the network to their individual environments. This mismatch between market requirements and network capabilities has brought the industry to a tipping point. In response, the industry has created the Software-Defined Networking (SDN) architecture and is developing associated standards.

III. OPENFLOW AND SDN

OpenFlow is the first standard communications interface defined between the control and forwarding layers of an SDN architecture. OpenFlow allows direct access to and manipulation of the forwarding plane of network devices such as switches and routers, both physical and virtual (hypervisor-based). It is the absence of an open interface to the forwarding plane that has led to the characterization of today's networking devices as monolithic, closed, and mainframe-like. No other standard protocol does what OpenFlow does, and a protocol like OpenFlow is needed to move network control out of the networking switches to logically centralized control software.

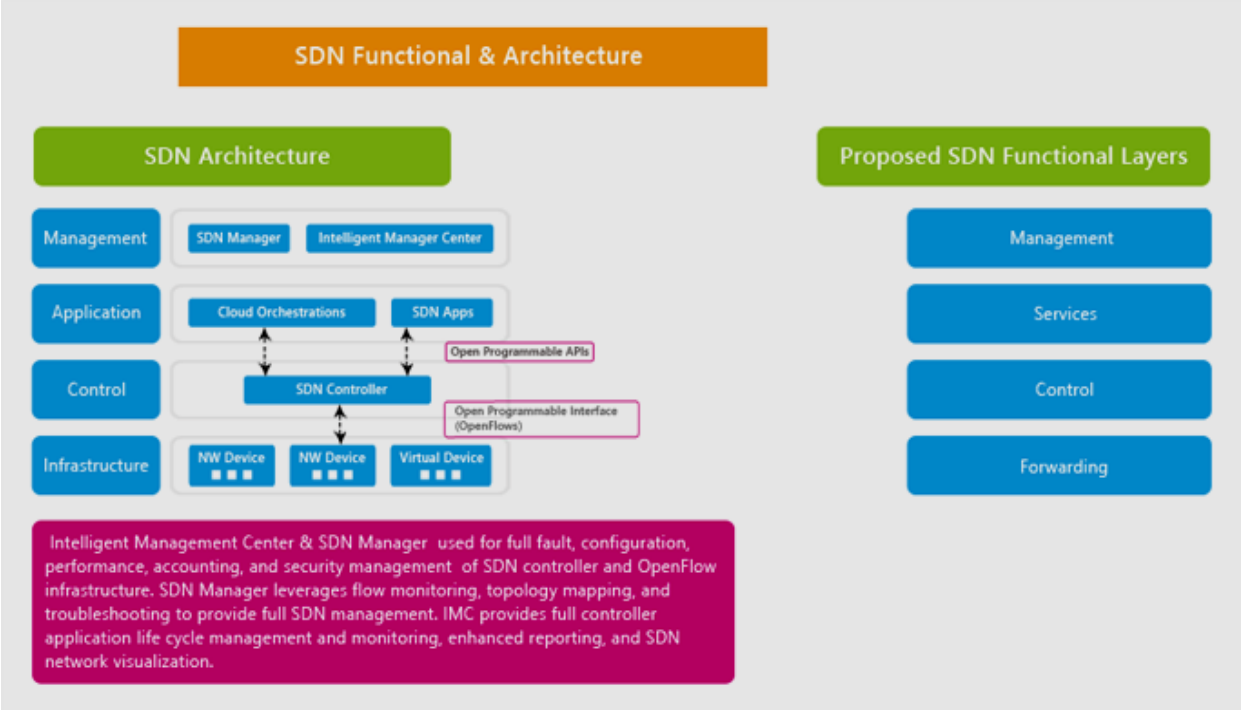

Figure 1 : Layers of Software Defined Networks

Software Defined Networking (SDN) is an emerging network architecture where network control is decoupled from forwarding and is directly programmable. This migration of control, formerly tightly bound in individual network devices, into accessible computing devices enables the underlying infrastructure to be abstracted for applications and network services, which can treat the network as a logical or virtual entity. Figure 1 depicts a logical view of the SDN architecture. Network intelligence is (logically) centralized in software-based SDN controllers, which maintain a global view of the network. As a result, the network appears to the applications and policy engines as a single, logical switch. With SDN, enterprises and carriers gain vendor-independent control over the entire network from a single logical point, which greatly simplifies the network design and operation. SDN also greatly simplifies the network devices themselves, since they no longer need to understand and process thousands of protocol standards but merely accept instructions from the SDN controllers.

Perhaps most importantly, network operators and administrators can programmatically configure this simplified network abstraction rather than having to hand-code tens of thousands of lines of configuration scattered among thousands of devices. In addition, leveraging the SDN controller's centralized intelligence, IT can alter network behavior in real-time and deploy new applications and network services in a matter of hours ordays, rather than the weeks or months needed today. By centralizing network state in the control layer, SDN gives network managers the flexibility to configure, manage, secure, and optimize network resources via dynamic, automated SDN programs. Moreover, they can write these programs themselves and not wait for features to be embedded in vendors' proprietary and closed software environments in the middle of the network. In addition to abstracting the network, SDN architectures support a set of APIs that make it possible to implement common network services, including routing, multicast, security, access control, bandwidth management, traffic engineering, quality of service, processor and storage optimization, energy usage, and all forms of policy management, custom tailored to meet business objectives. For example, an SDN architecture makes it easy to define and enforce consistent policies across both wired and wireless connections on a campus. Likewise, SDN makes it possible to manage the entire network through intelligent orchestration and provisioning systems. The Open Networking Foundation is studying open APIs to promote multi-vendor management, which opens the door for on-demand resource allocation, self-service provisioning, truly virtualized networking, and secure cloud services. Thus, with open APIs between the SDN control and applications layers, business applications can operate on an abstraction of the network, leveraging network services and capabilities without being tied to the details of their implementation. SDN makes the network not so much ―application-aware‖ as ―application-customized‖ and applications not so much ―network-aware‖ as ―network-capability-aware‖. As a result, computing, storage, and network resources can be optimized.

IV. PROPOSED METHOD

1. First we Start select the existing network architecture where in the network there have various devices like Layered switches, Routers, Firewalls, gateways.
2. Then we list all the network devices (Layered switches, Routers, Firewalls, gateways) are from various vendors but all the devices are from Open flow standard.

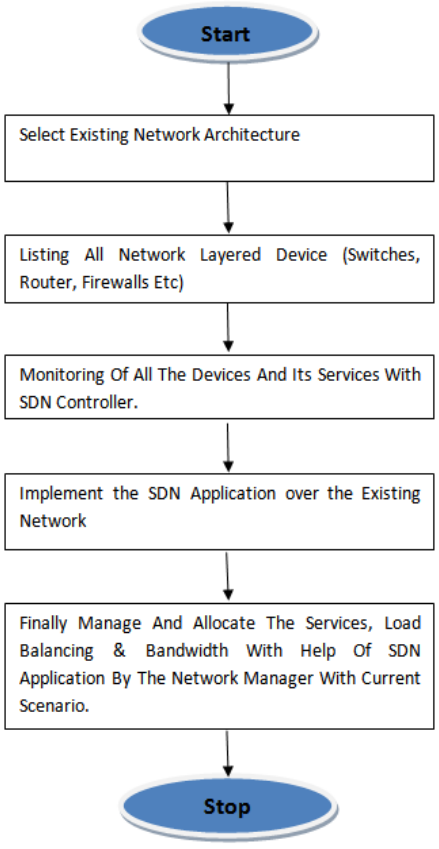

Figure 2: Flow Chart of Proposed methodology.

3. Then we implement the SDN Network Controller engine. This controller provides the services, load balancing, configuration and traffic to the network administrator. *software defined networking* paradigm, and thus has a controller that makes all traffic forwarding decisions and updates low level network switch flow-table entries according to this policy. The *network controller* translates the network policy to actual packet forwarding rules. The network controller establishes a connection to each OpenFlow-capable switch through the OpenFlow protocol, and inserts,deletes, or modifies packet forwarding rules in switches through this connection. The network controller also reacts to packet-in events and switch-join events that come from switches. For packet-in events, the network controller will install relevant forwarding rules in the switch, and for switch-join events, it will establish a new connection with that specific switch. Currently,Procera uses OpenFlow specification version 1.0.0. the flow chart for the same is fig.2.
4. The SDN Controller is handled by the SDN Application. This application provides the centralization of the existing network in the form of GUI based application so that the Network administrator can solve all network related issued.
5. Finally due to the SDN application the network administrator is able to view all the services provided in the organization, load on the devices, bandwidth distribution efficiency, and network device efficiency and simultaneously manage the same accordingly.

V. FUTURE WORK

Our deployments in campus and home networks demonstrate the feasibility of Procera, but more evaluation on performance and scalability is required. As Procera is based on the *software defined networking* paradigm, it also suffers from the inherent delay caused by the interaction ofthe control plane and the data plane: packet for- warding has to wait until the control logic decides on and installs a relevant forwarding rule in the data plane. Recent studies such as DIFANE and DevoFlow on resolving performance and scalability issues in the SDN realm through various mechanisms and algorithms. The ongoing research in SDN mechanisms and protocols should help mitigate some of these problems.

VI. CONCLUSION

Software-defined networking suffers from severe scalability issues towards deployment in larger scale networks such as wide-area networks. The challenges are three-fold: _ A centralized control plane has to store, process, and maintain an unbounded amount of network state information at scale. Moreover, an unbounded number of events emitted should be handled in real time. Since a wide-area network spans a large area geographically, the centralized control plane should consider the propagation delays which may significantly degrade the response time. The relatively high diversity of Open Flow matching rules is a double-edged sword. It may result in a huge number of TCAM rules, which are not cost-effective to be cached in the data plane. We have examined a few early attempts in the research community to address these scalability problems. Generally speaking, these surveyed proposals are learning from the experience of existing large distributed system designs such as DNS. The principles are shared: exploiting locality, hierarchical structuring, replication and synchronization among the nodes. Although solutions to software-defined wide area networks rely on more practical experiences, we think more attention should be paid to thinking about the foundational design principles of software-defined networking.

REFERENCES

[1] A. Lara, A. Kolasani, and B. Ramamurthy, “Network innovation using OpenFlow: A survey,” Communicat ons Surveys Tutorials, IEEE,vol. 16, no. 1, pp. 493–512, First 2014.

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

[3] Y. Jarraya, T. Madi, and M. Debbabi, “A survey and a layered taxonomy of software-defined networking,” Communications Surveys Tutorials, IEEE, vol. PP, no. 99, pp. 1–1, 2014.

[4] Dr.V.V.S.S.S.Balaram, “ Enhancement of Network Administration through Software Defined Networks”. *e-ISSN: 2278-0661,p-ISSN: 2278-8727, Volume 18, Issue 1, Ver. I (Jan – Feb. 2016).*

[5] Pritesh Ranjan, “*A Survey of Past, Present and Future of Software Defined Networking”* Volume 2, Issue 4, April 2014.

[6] R. Jain and S. Paul, “Network virtualization and software defined networking for cloud computing: a survey,” Communications Magazine, IEEE, vol. 51, no. 11, pp. 24–31, 2013.

[7] Diego Kreutz, Member," Software-Defined Networking: A Comprehensive Survey” arXiv:1406.0440v3 [cs.NI] 8 Oct 2014.

[8] M. Canini, D. Venzano, P. Peresini, D. Kostic, and J. Rexford. “A nice way to test openflow applications”. NSDI, Apr, 2012

[9] Sadasivarao, S. Syed, P. Pan, C. Liou, A. Lake, C. Guok, and I. Monga, “Open transport switch: a software defined networking architecture for transport networks,” in Proceedings of ACM SIGCOMM Hot SDN, August2013.

[10] Internet Engineering Task Force (IETF). Proposal: Software Defined Networking Research Group (SDNRG). [Online]. Available: http://trac.tools.ietf.org/group/irtf/trac/wiki/sdnrg

[11] OpenFlow Switch Specification, Version 1.3.0 (Wire Protocol 0x04). [Online].Available: https://www.opennetworking.org/images/stories/downloads/specification/openflow-spec-v1.3.0.pdf
[12] Open vSwitch. [Online]. Available: http://openvswitch.org/
[13] R. Jain and S. Paul, Network virtualization and software defined networking for cloud computing: A survey, IEEE Communications Magazine, vol. 51, no. 11, pp. 24–31, 2013
[14] J. Liao, "SDN System Performance," June 2012. [Online]. Available: http://pica8.org/blogs/?p=201
[15] B. Owens, "OpenFlow switching performance: Not all TCAM is created equal," February 2013. [Online]. Available: http://packetpushers.net/openflow-switching-performance-not-all-tcam-is-created-equal/
[16] Intel Processors, "Software Defined Networking and Softwarebased services with Intel Processors," Intel Corporation,2012.[Online]. Available: http://www.intel.com/content/dam/doc/whitepaper/communications-ia-software-defined-networking-paper.pdf
[17] R. Ramos, M. Martinello, and C. Esteve Rothenberg, "SlickFlow: Resilient source routing in data center networks unlocked by OpenFlow,"in Local Computer Networks (LCN), 2013 IEEE 38th Conference on,Oct 2013, pp. 606–613.
[18] J. T. Ara´ujo, R. Landa, R. G. Clegg, and G. Pavlou, "Software-defined network support for transport resilience," in IEEE NOMS, 2014.
[19] E. Brewer, "Pushing the cap: Strategies for consistency and availability,"Computer, vol. 45, no. 2, pp. 23–29, Feb. 2012.
[20] D. Katz and D. Ward, "Bidirectional Forwarding Detection (BFD),"RFC 5880 (Proposed Standard), Internet Engineering Task Force, Jun.2010. [Online]. Available: http://www.ietf.org/rfc/rfc5880.txt
[21] N. L. M. van Adrichem, B. J. van Asten, and F. A. Kuipers, "Fast recovery in software-defined networks," in Third European Workshop on Software Defined Networks, 2014, pp. –.
[22] N. M. Sahri and K. Okamura, "Fast failover mechanism for software defined networking: Openflow based," in Proceedings of The NinthInternational Conference on Future Internet Technologies, ser. CFI '14. New York, NY, USA: ACM, 2014, pp. 16:1–16:2.
[23] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, ser. IMC '10. New York, NY,USA: ACM, 2010, pp. 267–280.
[24] M. Yu, J. Rexford, M. J. Freedman, and J. Wang, "Scalable flow-based networking with difane," SIGCOMM Comput. Commun. Rev., vol. 41,no. 4, pp. –, Aug. 2010.

