Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 00 (2018) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

The 13th International Conference on Future Networks and Communications (FNC 2018)

SMS: Smart Management Scheme via Software Defined Networks

Ahmad Nahar Quttoum*, Hassan AlSaraireh, Olbi Khadjiev, Marwa Alqudaimat

Computer Engineering Department, Faculty of Engineering, The Hashemite University, Zarqa 13133, Jordan

Abstract

Traffic hikes exhibits points of instability in computer networks, not to mention congestions and its implications of service disruptions, outages and security holes. Such common failures are becoming a big source of concern for most Internet Service Providers (ISPs) who run services that touch almost every aspect of this smart-based life we are living nowadays. Having the network managed by skilled technicians and professional network engineers is not enough to guarantee a well-defined performance of the provided services. Indeed, an update or even a sudden change and/or failure in a segment of the network may lead to service outages that may take hours to get fixed. This work deploys the theme of Software Defined Networks (SDNs) that decouples the network center of intelligence form the rigid and mostly static hardware side, into a simple more flexible programmable scheme. We built a real SDN network, and proved through experiments its performance and flexibility. Compared with the traditional networks and its rigid management schemes, results proved that SDNs can provide for better performance, and lower costs.

Keywords: Software Defined Networks, Resource Management, Data Centers, Dynamic Resource Allocation

1. Introduction and Problem Statement

Computer-based services have emerged in the last few years as a promising approach that touches almost every aspect of life. Advancements in computing and networking technologies proposed a complete new life-style. Indeed, with such smart era of networking and computing technologies, remote computer-based services became an inevitable utility of life. From the clients' perspective, such scheme looks great! True; it provides for flexible life

* Corresponding author. Tel.: +962-777-305402; Fax: +962-5-3903333.
E-mail address: Quttoum@hu.ed.jo

style while saving time and money. In the contrary, this is not the case for Internet Service Providers (ISPs) and their network managers. For ISPs, this imposes more administrative overhead and hard commitments to satisfy. Certainly, having such amount of online-services puts the ISPs in a position in which they are responsible for the network performance and service quality. This includes delays, loses ratios, and reachability.

In Enterprise networks, ISPs usually run mid-size to large networks with predefined performance and security requirements. All guaranteed through Service Level Agreements (SLAs). Enterprise networks come in different scales, settings, and clients' requirements that might be static or dynamic. Static requirements may ease the management overhead at the ISPs side as they require a kind of fixed settings and configurations. However, this not the case for those Enterprise networks with dynamic requirements that changes continuously. Networks like those in residential sites can be considered as an example of static enterprise networks, while those of the academic institutions and commercial sites represent examples of dynamic ones. Usually, a network infrastructure that supports the connectivity services for an academic site like colleges and universities consists of many connecting devices that are interconnected through a set of links. Some of these connecting devices and the add-on services they may run are temporary, and they are fully controlled and directly managed by the ISPs. Moreover, colleges and universities are test-beds for novel research and productive experiments. This requires real-time readiness for instant resource provisions and support. Such dynamic requirements represent an extra overhead and resource management challenge to the ISPs.

To tackle such management challenge and its overhead, proposals in the literature as in [5] [6] [13] [15] [16] presented a set of schemes that suggests amendments to the existing protocols and their functional practices. Such proposals may provide enhancements for certain properties in predefined protocols like the Routing Information Protocol (RIP) and the Open Shortest Path First Protocol (OSPF), but still, it does not provide for *wide-range* management solution that truly reduces the systems' complexity and eases its dynamic configurations overhead.

Data Center Networks (DCNs) is another infrastructure that has rapidly evolved in the last few years as a service-based networking scheme. Nowadays, the heartbeat of several business sites come in DCs, where different parties (i.e. employees, partners, and customers) physically rely on the same data and network resources of a single DCN to interact, collaborate, and create services. Satisfying the performance requirements in such dynamic environments is quit challenging, while demands are growing rapidly, and the needs became emerging to meet the economic and technical growth we are living today. Efficient DCN setting should provide for (1) balanced real-time capacities [2], (2) scalable network fabrics, (4) reliable service levels with a substantial degrees of tolerance against network failures. Therefore, operating in such a scale requires careful network management and access control mechanisms. Indeed, network failures or service disruption states may result in huge losses and negative reputation records for ISPs. Traditional techniques to avoid such problems suggest provisioning the networks with resources for peak loads; hence, such networks run *below* thresholds most of the time though kept ready for any load spikes. Such techniques may lead to *big wastes in resource assets* and *high power costs* [7] [3]!

In this context, the theme of Software Defined Networks (SDNs) can be considered as an adequate candidate to deal with such management challenges. SDNs have recently emerged as a promising approach that proposes novel ways to how networks' resources can be controlled and managed. SDNs leverages the *virtualization technologies* to logically control, monitor, adjust, and tune (i.e. program the packet handling policies) the switching behavior of the underlying connecting devices. Traditionally, such policies are presented as hardware middle-boxes.

Compared to legacy network management platforms, SDN claims to provide several enhancements in terms of resource utilization, setup times, flexibility, scalability and operational costs. It really does! Employing the SDN-based management scheme allows *instant updates* to the network routing tables, and its access control policies [11] [2]. What is more; any updates on the security settings and its related traffic engineering methods can be tuned on the fly.

SDN can also ease the process of implementing middle-boxes by getting their functions integrated with the network controller, such middle-boxes like traffic balancers and firewalls [10] [5]. In SDN, all of this can be done through a simplified graphical user interface developed to directly deal with the network control plane. Without a doubt, this allows for flexible and highly scalable network fabrics.

Compared with the traditional management schemes that required physical interaction with the connecting devices to configure and tune their routing and traffic engineering policies. Through SDN, things become much simpler and way more efficient. In this work, we used the *Microsoft Azure* environment to create a SDN that consists of few Open Virtual Switches (OVSs) interconnecting several servers running varying applications. Decoupling the control plane from the data plane promoted the privilege to dynamically reprogram the network and enabled customized configurations. Hence, the contribution of this work is an emulation of the management scheme of SDN-based networks and show how it:

- Proves its *claims of functionality*, and exploits its *management power*.
- Provides for *challenge abstraction*, through which, the management and dynamic configurations overhead is simplified and all unified via a GUI that allows reprogramming the network policies and tackling traffic updates.
- Improves the *network performance*.
- Reduces the *resource waste* rates and its *costs*.

The reminder of the paper is organized as follows: Section 3 presents the developed SDN environment, followed by sample of the emulation results in Section 4, and finally Section 5 concludes the paper.

3. Software-Defined Networks for a Smart Management Scheme

The resource allocation process and what follows from resource management techniques and its related configuration policies greatly depends on the network traffic schemes, and its varying application requirements. Networks vary according to its type of traffic (i.e. customers) and its provided services. Thus, those of public services like power and telephone companies will certainly differ from others that are concerned with academic and health institutions. Traffic hikes might be expectable to happen at expected predefined periods of time, while it might be sudden at some cases. A power failure at a crowded city with huge population rates, a city like New York for example, can be considered as an example of a sudden incidence that may cause negative effects to the whole internetwork of the city. Indeed, imagine how many hits DHCP (Dynamic Host Configuration Protocol) servers would receive after a network failure in a city like New York! Such an incident can push the whole network to collapse, definitely. Now, if the network managers have the privilege to remotely and centrally reconfigure, reprogram, and tune the whole network settings through a simplified GUI (Graphical User Interface), the situation will be totally different. In a case of network failure, the recovery would be fast and prompt. Indeed, instead of physically visiting the servers' sites and starting the physical troubleshooting tasks to find and fix the problems, with the theme of SDN being deployed, all of this burden will be eliminated to few commands that can be remotely pushed to the network data-path through a simple and user-friendly GUI. Through which, the network manager can redistribute the network resources and reprogram the network policies in a way that strengthens the points of failures.

3.1. SDN Structure

To simplify the management and enable a prompt handling mechanism to the sudden changes and the varying traffic requirements of today's networks, the theme of SDN proposed separating the data-plane (i.e. the forwarding reference) from the network's control-plane. Through such separation technique, as depicted in Figure 1 below, the network intelligence is totally moved to the control-plane side, while leaving the execution tasks to commodity low level forwarding devices (i.e. network switches). Such theme of separation facilitates the deployment of new policies, protocols, and any configuration settings without the need for direct interaction with the hardware side of the network. Moreover, it also allows merging the services that are traditionally provided through standalone middle-boxes to the software side of the network (i.e. the control-plane). Not only increases flexibility, but also reducing the number of hardware devices scattered here and there in the network. Consequently, such a theme helps in reducing cabling, complexity, and power consumption rates.

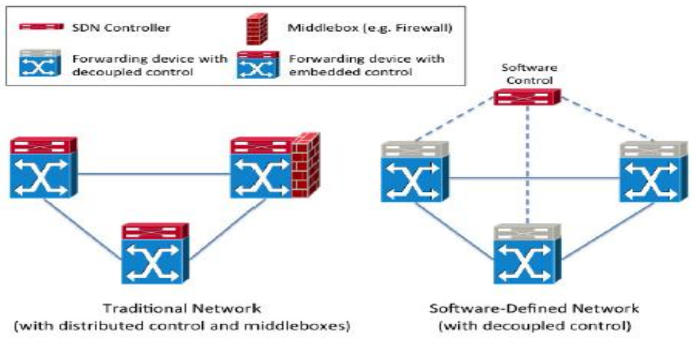
SDN Controller
Middlebox (e.g. Firewall)
Forwarding device with decoupled control
Forwarding device with embedded control
Software Control
Traditional Network (with distributed control and middleboxes)
Software-Defined Network (with decoupled control)

Fig. 1 SDN vs Traditional Networks Architecture [17]

3.2. Control plane – Data plane Communication:

To handle the exchange of information between the two main planes, several standards are proposed in the literature. Among them, we chose the OpenFlow [12] architecture to build our system. An *OpenFlow protocol* is employed to handle the interconnection between the *Controller* (i.e. where the management decisions are made) and the *Switches' Flow Tables* (i.e. OpenFlow Virtual Switches (OVSs)). Similar to switching tables, entries in these flow tables are used to define how the incoming packets are going to be processed (i.e. forwarded). Consequently, when a packet arrives to an OVS, it extracts the packet's header and starts the lookup process to find a match with the table's entries. Once a match is found, the forwarding decision is made accordingly. This may include actions and certain set of instructions that suites the real-time status of the whole network as defined by the network manager. Now, to cover the case that no match is found, at the bottom of the flow entries, the tables are configured with a *default* entry that represents the match for such packets to be processed (i.e. either forwarded or dropped). This guarantees a proper processing for any packet an OVS may receive.

Employing the OpenFlow protocol allowed us to establish a secure communication channel between the two planes, through such channel, we are able to use the controller to *reprogram* the forwarding policies and update the flow-table entries. It is worth to mention that the controller allows the privilege to classify the packets' flows. Through which, the data-plane can be programmed to consult the controller before taking any action in regard to any packet arrives from new traffic flows, this is called *reactive* control mode [14]. Note that such mode may incur more processing delays, however, it providers for higher level of security. In the contrary, a mode that is called *proactive* [8] does not require such referencing between the two planes, instead, policies for such sort of flows might be pushed to other OVSs that reside within the data-plane. This reduces delays and allows for higher throughput rates.

3.3. Experiments

Before building the SDN network in real-life settings over the Microsoft Azure, we chose to build it over an emulation environment.

3.3.1 Mininet Emulation

To simplify the deployment process and the initial development settings, before moving to the real-life settings, our OpenFlow network was first emulated over the Mininet environment [9] to get our configurations tested and verified. Having the work verified, we moved to the hardware side and started developing the SDN network.

3.3.2 OpenFlow Network

To emulate the case that happens after a power failure status and its implications (i.e. when a huge number of networking devices are trying to contact the DHCP servers asking for IP addresses), we used the Http Unbreakable Load King (HULK) tool to generate a huge number of packets that floods the SDN network in a way to create a shape of Denial of Service Attack. At the same time, we ran the same HULK attack on a physical network that we built in the Lab, and then compared the results. The comparison was in terms of network performance, and cost.

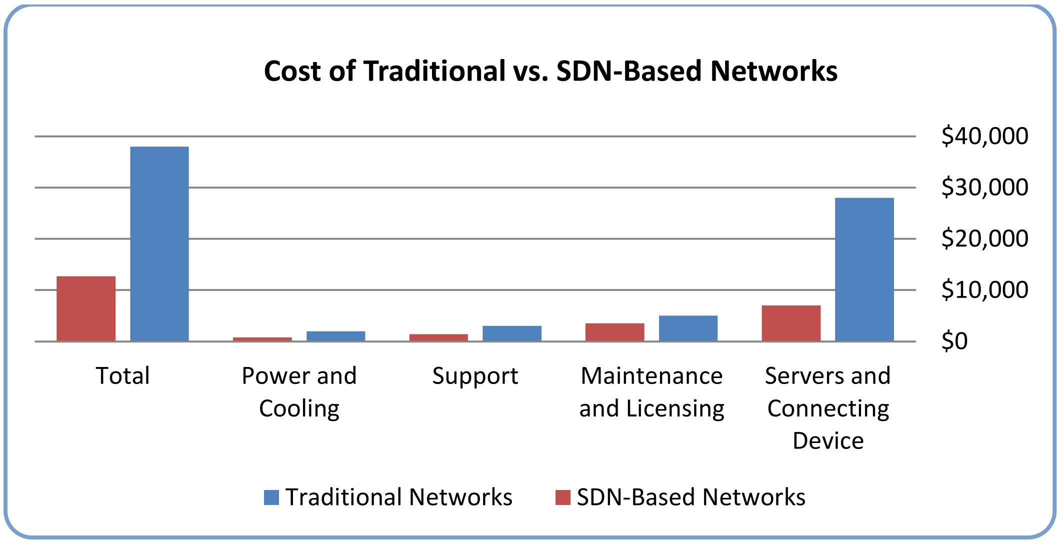

Fig. 2 An estimated cost for the networks that we tested both Traditional and SDN

Figure 2 above shows an estimate cost for the networks we tested. A traditional network that we built in the Lab, and another that is a SD-network. Comparing the costs of the two environments, we can conclude that adopting the SDN based theme provides a significant reduction that may reach a 65% to 70% when compared with that of the traditional networks. Besides, the experiments also shows that an SDN configured network allows for scalable network fabrics that eases the process of adding new nodes to the network. This was not the case in traditional networks that may require physical changes in the network connectivity settings and its hardware devices.

4. Conclusion

Computer networks became one of the fundamental utilities these days. Beside electricity, water, and gas, the Internet service is as important as the aforementioned three essentials. Consequently, relying on the existing management tools might not sufficient enough to handle today's clients' requirements and their carried services. Traditional management schemes require long response times and convey a lot of errors, respectively. Nowadays, this is not acceptable anymore. The theme of SDNs provides a promising management scheme that besides being simple, it is truly flexible, tunable, and cost efficient.

References

[1] Gember A., Prabhu P., Ghadiyali Z., and Akella A. (2012) "Toward software defined middle-box networking."

[2] Mark R., Nate F., Jennifer R., Cole S., and David W. (2012) "Abstractions for network update." *In Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication, SIGCOMM'12, ACM*, pages 323–334, New York, NY, USA.

[3] Shirayanagi H., Yamada H., and Kono K. (2012) "Honey guide: A VM migration-aware network topology for saving energy consumption in data center networks." *In Computers and Communications (ISCC), 2012 IEEE Symposium*, pages 460–467.

[4] Wang R., Butnariu D., and Rexford J. (2011) "Openflow-based server load balancing gone wild." *Workshop of Hot ICE,* Volume (11).

[5] Raza S., Zhu Y., and Chuah C. (2011) "Graceful network state migrations." *IEEE/ACM Trans. on Networking*, volume (19).

[6] Vanbever L., Vissicchio S., Pelsser C., Francois P., and Bonaventure O. (2011) "Seamless network-wide IGP migration," *in ACM SIGCOMM.*

[7] Heller B., Seetharaman S., Mahadevan P., Yiakoumis Y., Sharma P., Banerjee S., and McKeown N. (2010) "Elastictree: Saving energy in data center networks." *In Proceedings of the 7th USENIX conference on Networked systems design and implementation, USENIX Association,* pages 17–17.

[8] Yu M., Rexford J., Freedman M. J., and Wang J. (2010) "Scalable flow-based networking with difane." *In Proceedings of the ACM SIGCOMM 2010 conference*, pages 351–362.

[9] Bob Lantz, Brandon Heller, and Nick McKeown. (2010) "A network in a laptop: rapid prototyping for software-defined networks." *In Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks*.

[10] Handigol N., Seetharaman S., Flajslik M., McKeown N., and Johari R. (2009) "Plug-n-serve: Load-balancing web traffic using openflow." *ACM SIGCOMM Demo*.

[11] Nayak A. K., Reimers A., Feamster N., and Clark R. (2009) "Resonance: Dynamic access control for enterprise networks." *In Proceedings of the 1st ACM workshop on Research on enterprise networking*, pages 11–8.

[12] McKeown N., Anderson T., Balakrishnan H., Parulkar G., Peterson L., Rexford J., Shenker S., and Turner J. (2008) "Openflow: enabling innovation in campus networks." *ACM SIGCOMM Computer Communication Review*, volume 38(2), pages: 69–74.

[13] John J. P., Katz-Bassett E., Krishnamurthy A., Anderson T., and Venkataramani A. (2008) "Consensus routing: The Internet as a distributed system," *in NSDI.*

[14] Casado M., Freedman M. J., Pettit J., Luo J., McKeown N., and Shenker S. (2007) "Ethane: Taking control of the enterprise." *ACM SIGCOMM Computer Communication Review*, volume 37(4), pages: 1–12.

[15] Francois P., Coste P. A., Decraene B., and Bonaventure O. (2007) "Avoiding disruptions during maintenance operations on BGP sessions," *IEEE Transaction on Network and Service Management*.

[16] Francois P., Shand M., and Bonaventure O. (2007) "Disruption-free topology reconfiguration in OSPF networks," *in IEEE INFOCOM.*

[17] Bruno N. A., Marc M., Xuan N. N., Katia O., Thierry T. (2014) "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks." *Communications Surveys and Tutorials, IEEE Communications Society, Institute of Electrical and Electronics Engineers*, 16 (3), pp.1617- 1634.