International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 03 Issue: 09 | Sep-2016 www.irjet.net p-ISSN: 2395-0072

CHALLENGING ISSUES OF SOFTWARE DEFINED NETWORK

Shikha Arya
Assistant Professor
shikha.srms@gmail.com

ABSTRACT: *The fundamental problem with traditional network architecture is that it is static. Users don't have much controlling power and it is just trying to accommodate current networking needs of the user. Therefore which means window for innovation is too narrow with respect to network controlling, virtualization, automation, scalability, programmability and fault tolerance. A revolutionize network technology which can put all the above aspects into practice in future networks is Software Defined Networking. In the study we review a brief history of Software Defined Networking that can help to provide evidential clarity with respect to the understanding of the new technology and more significantly, in understanding the motive that why we need Software Defined Networking*.

Keywords: Software Defined Networking, SDN, SDN models, Control Plane, Data Plane, Open Flow.

INTRODUCTION

Software-defined networking (SDN) promises to bring radical improvements in both cost and functionality to the networking field. At the same time, SDN poses many fundamental questions, including deep architectural issues such as whether control should be centralized or distributed, and whether control and data planes should be separated or share fate. Software-defined networks (SDNs) are widely seen as a promising solution for resolving these challenges. In particular, SDNs promise to provide a multilayer platform that encompasses programmability not only at the forwarding and control planes, but also at the transport layers below and orchestration and services layers above the data and control planes. Early SDN models focused primarily on moving the control plane out of the network elements into "controllers" on the theory that the switching elements could remain simple, general-purpose, and cost-effective while letting the control plane rapidly evolve. Several recent SDN models, on the other hand, include approaches in which control- and data-plane programmability works in concert with existing and future distributed control planes.

SOFTWARE DEFINED NETWORKING

Software-defined networking (SDN) is an architecture purporting to be dynamic, manageable, cost-effective, and adaptable, seeking to be suitable for the high-bandwidth, dynamic nature of today's applications. SDN architectures decouple network control and forwarding functions, enabling network control to become directly programmable and the underlying infrastructure to be abstracted from applications and network services.
The OpenFlow protocol is a foundational element for building SDN solutions. The SDN architecture is:

- Directly programmable: Network control is directly programmable because it is decoupled from forwarding functions.
- Agile: Abstracting control from forwarding lets administrators dynamically adjust network-wide traffic flow to meet changing needs.
- Centrally managed: Network intelligence is (logically) centralized in software-based SDN controllers that maintain a global view of the network, which appears to applications and policy engines as a single, logical switch.
- Programmatically configured: SDN lets network managers configure, manage, secure, and optimize network resources very quickly via dynamic, automated SDN programs, which they can write themselves because the programs do not depend on proprietary software.
- Open standards-based and vendor-neutral: When implemented through open standards, SDN simplifies network design and operation because instructions are provided by SDN controllers instead of multiple, vendor-specific devices and protocols.

SDN ARCHITECTURE

The explosion of mobile devices and content, server virtualization, and advent of cloud services are among the trends driving the networking industry to reexamine traditional network architectures.[21] Many conventional networks are hierarchical, built with tiers of Ethernet switches arranged in a tree structure. This design made sense when client-server computing was dominant, but

such a static architecture is ill-suited to the dynamic computing and storage needs of today's enterprise data centers, campuses, and carrier environments. Some of the key computing trends driving the need for a new network paradigm include:

➢ Changing traffic patterns

Within the enterprise data center, traffic patterns have changed significantly. In contrast to client-server applications where the bulk of the communication occurs between one client and one server, today's applications access different databases and servers, creating a flurry of "east-west" machine-to-machine traffic before returning data to the end user device in the classic "north-south" traffic pattern. At the same time, users are changing network traffic patterns as they push for access to corporate content and applications from any type of device (including their own), connecting from anywhere, at any time. Finally, many enterprise data centers managers are contemplating a utility computing model, which might include a private cloud, public cloud, or some mix of both, resulting in additional traffic across the wide area network.

➢ The "consumerization of IT"

Users are increasingly employing mobile personal devices such as smartphones, tablets, and notebooks to access the corporate network. IT is under pressure to accommodate these personal devices in a fine-grained manner while protecting corporate data and intellectual property and meeting compliance mandates.

➢ The rise of cloud services

Enterprises have enthusiastically embraced both public and private cloud services, resulting in unprecedented growth of these services. Enterprise business units now want the agility to access applications, infrastructure, and other IT resources on demand and à la carte. To add to the complexity, IT's planning for cloud services must be done in an environment of increased security, compliance, and auditing requirements, along with business reorganizations, consolidations, and mergers that can change assumptions overnight. Providing self-service provisioning, whether in a private or public cloud, requires elastic scaling of computing, storage, and network resources, ideally from a common viewpoint and with a common suite of tools.

➢ "Big data" means more bandwidth

Handling today's "big data" or mega datasets requires massive parallel processing on thousands of servers, all of which need direct connections to each other. The rise of mega datasets is fueling a constant demand for additional network capacity in the data center. Operators of hyperscale data center networks face the daunting task of scaling the network to previously unimaginable size, maintaining any-to-any connectivity without going broke.

ARCHITECTURAL COMPONENTS

The following list defines and explains the architectural components:

➢ SDN Application (SDN App)

SDN Applications are programs that explicitly, directly, and programmatically communicate their network requirements and desired network behavior to the SDN Controller via a northbound interface (NBI). In addition they may consume an abstracted view of the network for their internal decision making purposes. An SDN Application consists of one SDN Application Logic and one or more NBI Drivers. SDN Applications may themselves expose another layer of abstracted network control, thus offering one or more higher-level NBIs through respective NBI agents.

➢ SDN Controller

The SDN Controller is a logically centralized entity in charge of (i) translating the requirements from the SDN Application layer down to the SDN Datapaths and (ii) providing the SDN Applications with an abstract view of the network (which may include statistics and events). An SDN Controller consists of one or more NBI Agents, the SDN Control Logic, and the Control to Data-Plane Interface (CDPI) driver. Definition as a logically centralized entity neither prescribes nor precludes implementation details such as the federation of multiple controllers, the hierarchical connection of controllers, communication interfaces between controllers, nor virtualization or slicing of network resources.

➢ SDN Datapath

The SDN Datapath is a logical network device that exposes visibility and uncontested control over its advertised forwarding and data processing capabilities. The logical representation may encompass all or a subset of the physical substrate resources. An SDN

Datapath comprises a CDPI agent and a set of one or more traffic forwarding engines and zero or more traffic processing functions. These engines and functions may include simple forwarding between the datapath's external interfaces or internal traffic processing or termination functions. One or more SDN Datapaths may be contained in a single (physical) network element—an integrated physical combination of communications resources, managed as a unit. An SDN Datapath may also be defined across multiple physical network elements. This logical definition neither prescribes nor precludes implementation details such as the logical to physical mapping, management of shared physical resources, virtualization or slicing of the SDN Datapath, interoperability with non-SDN networking, nor the data processing functionality, which can include OSI layer 4-7 functions.

➢ SDN Control to Data-Plane Interface (CDPI)

The SDN CDPI is the interface defined between an SDN Controller and an SDN Datapath, which provides at least (i) programmatic control of all forwarding operations, (ii) capabilities advertisement, (iii) statistics reporting, and (iv) event notification. One value of SDN lies in the expectation that the CDPI is implemented in an open, vendor-neutral and interoperable way.

➢ SDN Northbound Interfaces (NBI)

SDN NBIs are interfaces between SDN Applications and SDN Controllers and typically provide abstract network views and enable direct expression of network behavior and requirements. This may occur at any level of abstraction (latitude) and across different sets of functionality (longitude). One value of SDN lies in the expectation that these interfaces are implemented in an open, vendor-neutral and interoperable way.

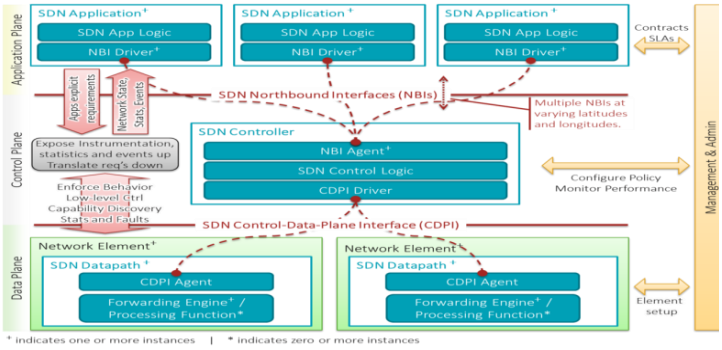

Fig: SDN Architecture

SDN Security Challenges in SDN Environments

Security needs to be everywhere within a software-defined network (SDN). SDN security needs to be built into the architecture, as well as delivered as a service to protect the availability, integrity, and privacy of all connected resources and information. Within the architecture, you need to:

Secure the Controller: as the centralized decision point, access to the SDN Controller needs to be tightly controlled.

1. Protect the Controller: if the SDN Controller goes down (for example, because of a DDoS attack), so goes the network, which means the availability of the SDN Controller needs to be maintained.
2. Establish Trust: protecting the communications throughout the network is critical. This means ensuring theSDN Controller, the applications loaded on it, and the devices it manages are all trusted entities that are operating as they should.
3. Create a Robust Policy Framework: what's needed is a system of checks and balances to make sure the SDN Controllers are doing what you actually want them to do.
4. Conduct Forensics and Remediation: when an incident happens, you must be able to determine what it was, recover, potentially report on it, and then protect against it in the future.

Beyond the architecture itself, how SDN security should be deployed, managed, and controlled in an SDN environment is still very much up for grabs. There are competing approaches – some believe security is best embedded within the network, others feel it is best embedded in servers, storage and other computing devices. Regardless, the solutions need to be designed to create an environment that is more scalable, efficient, and secure. They must be:

- Simple: to deploy, manage and maintain in the highly dynamic SDN environment.
- Cost-effective: to ensure security can be deployed everywhere.
- Secure: to protect against the latest advanced, targeted threats facing your organization.

A new category is emerging for security within next-generation environments called software-defined security (SDSec), which delivers network security enforcement by separating the security control plane from the security processing and forwarding planes, similar to the way SDN abstracts the network control plane from the forwarding plane. The result is a dynamic distributed system that virtualizes the network security enforcement

function, scales like virtual machines and is managed as a single, logical system.

SDSec is an example of network functions virtualization (NFV), which offers a new way to design, deploy, and manage networking services by decoupling the network function, such as firewalling and intrusion detection, from proprietary hardware appliances, so they can run in software. It's designed to consolidate and deliver the networking components needed to support a fully virtualized infrastructure – including virtual servers, storage, and even other networks. This all sounds great in theory, but where does security for the SDN network fit? As we define what may arguably be the "next big thing" for networking, did we leave network security as an afterthought? The new considerations for security in an SDN network are as:

• Programmability - One of the key benefits of the SDN architecture is in its programmability. Similarly, these are the same characteristics that need to be considered for security in an SDN network. Today, security policies are defined for security zones that are static and are tied to physical interfaces. As we move towards a more dynamic SDN network, the security zones become decoupled from the physical plane and in fact, network or host "objects" will be programmatically defined. Flows will need to be dynamically programmed so that appropriate security appliances are in the data path.

• Dynamic policies – Security policies correspondingly become more dynamic, where specific rules are applied based on applications, users, content, devices and network characteristics. Traditional security policies based on IP addresses have already evolved to rich next-generation firewall policies with application, user, and content. Now, on top of this, add another consideration around dynamic context. In an SDN network, traffic and flows can show up on any interface and more data will be needed to express security policies without just relying on traditional network contexts like location or 802.1q tags.

• Automation - Automation will become a foundational requirement for network security. The ability to implement network security policies in an automated manner instead of having to initiate manual changes per device will be critical. This means network security appliances will need robust APIs that enable automated security controls triggered by the controller. Separation of security and network duties will continue to be important in an SDN network. Security policies will continue to be independently pre-defined by security IT administrators, but networking and server operations teams will continue to manage SDN controller and application flows. In other words, the goal with SDN will be seamless insertion of security without delegation of policy creation.

• Tunneled traffic – A key component of the SDN architecture will include network virtualization to run virtual networks over existing physical infrastructures. Network security appliances will need to address this encapsulated traffic. For example, in order to properly inspect this traffic, network security solutions will need to support the ability to decapsulate the traffic, or depend on gateways and switches to translate SDN encapsulation and decapsulation protocols to VLANs for context.

As is typical of an early adopter technology, SDN deployments will be riddled with initial growing pains, including standards and interoperability pain points. Which protocols are ideal, and which switches will interoperate with what controllers? While the specific interoperability aspects of SDN are being worked out, look for network security solution vendors that already exhibit dynamic, programmable characteristics as defined above to ensure appropriate investment protection for your next-generation network.

CONCLUSION

Software Defined Networking (SDN) The goal of Software-Defined Networking is to enable cloud and network engineers and administrators to respond quickly to changing business requirements via a centralized control console. SDN encompasses multiple kinds of network technologies designed to make the network more flexible and agile to support the virtualized server and storage infrastructure of the modern data center and Software defined networking was originally defined an approach to designing, building, and managing networks that separates the network's control (brains) and forwarding (muscle) planes enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services.

REFERENCES

[1] K. Greene. 10 Emerging Technologies, TR10: Software Defined Networking, 2009. Available at: http://www.technologyreview.com/biotech/22120/.

[2] B. Lantz, B. Heller, and N. McKeown. A network in a laptop: rapid prototyping for software-Defined networks. In Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks, Hotnets '10, pages 19:1–19:6, New York, NY, USA, 2010. ACM.

[3] S. Costanzo, L. Galluccio, G. Morabito, S. Palazzo. "Software Defined Wireless Networks:Unbridling SDNs". European Workshop on Software Defined Networking. 2012.

[4] http://www.zdnet.com/10-key-questions-about-software-Definedd-networking-sdn-7000015822/

[5] http://globalconfig.net/software-Definedd-networking-vs-traditional/

[6] http://readwrite.com/2013/04/23/software-Definedd-networking-dn#awesm=~omPg0fn3rysfHX

[7] Vijay K. Gurbani, M Scharf, T.V. Lakshman and V. Hilt. Bell Laboratories, Alcatel-Lucent"Abstracting network state in Software Defined Networks (SDN) for rendezvous services"." Communications (ICC), 2012 IEEE International Conference on Software Defined Networks. (2012).

[8] http://searchcloudprovider.techtarget.com/tip/Three-models-of-SDN-explained

[9] Fl´avio de Oliveira Silva, Jo˜ao Henrique de Souza Pereira, Pedro Frosi Rosa†, and Sergio Takeo Kofuji. "Enabling Future Internet Architecture Research and Experimentation by Using Software Defined Networking". European Workshop on Software Defined Networking. 2012.

[10] M. H. Razaa, S. C. Sivakumar, A. Nafarieha, B. Robertson, “A Comparison of Software Defined Network (SDN) Implementation Strategies”, Elsevier Proc. of 2nd International Workshop on Survivable and Robust Optical Networks (IWSRON). Vol. 32, pp. 1050-1055, 2014.

[11]S. Scott-Hayward, G. O’Callaghan and S. Sezer,”SDN Security: A Survey”, IEEE Proc. of SDN4FNS, Trento Italy, pp. 1 - 7,2013.

[12] S. Shi, G. Gun, “Attacking Software-Defined Networks: A First Feasibility Study”, ACM Proc. of HotSDN, Hong Kong, China, pp. 165-166, 2013.

[13] S. Shin, V. Yegneswaran, P. Porras, G. Gu, “AVANT-GUARD: Scalable and Vigilant Switch Flow anagement in Software-Defined Networks”, ACM Proc. of CCS, Berlin, Germany, pp. 413-424, 2013.

[14] M. McBride, M. Cohn, S. Deshpande, M. Kaushik, M. Mathews, S. Nathan, “SDN Security Considerations in the Data Center”,Open Networking Foundation- ONF SOLUTION BRIEF, 2013.

[15]Y. Zhang, “An adaptive flow counting method for anomaly detection in SDN”, ACM Proc. of CoNEXT, Santa Barbara, California, USA December, 2013, pp. 25-30.