SOFTWARE-DEFINED NETWORKING (SDN) IN MODERN NETWORK MANAGEMENT

Rahma Fitria[1] , Desvina Yulisda [2] Sayed Fachrurrazi [3]

Sistem Informasi Universitas Malikussaleh Lhokseumawe
Jl. Cot Tgk Nie-Reulet, Aceh Utara, 141 Indonesia
email: rahmafitria@unimal.ac.id , desvina.yulisda@unimal.ac.id,
sayed.fachrurrazi@unimal.ac.id

Abstract

Modern computer network implements tasks from routing, monitoring traffic, accessing control and balancing server load. Software-Defined Networking (SDN) has become a new radical model where the network is programmable. The network can be simplified and managed through network programmability. In addition, SDN offers a very good interface between networking devices and the software that control this activity. This paper presents the SDN architecture, ForCES, and OpenFlow protocol to allocate information exchange. Oracle and HP is presented as an organization that built SDN product by having different architecture. Always as a new solution for computer network has been proposed, it still has challenges for researcher to solve the SDN issues that has been raised.

Keywords: *Software-Defined Networking (SDN), programmable networks, ForCES, OpenFlow*

1. INTRODUCTION

Computer network is developed by appliances such as switches, routers and several devices that control traffic such as firewall. Traditional approached of network is developed to perform distributed protocols such as topology routing, traffic observing, and control entry (Ganti et al., 2013). Integrating control and data plane must be fixedly performed as well as operators of networks need to solely configure any protocol on whole separate device. Nowadays, modern computer network evolves in software-defined networks (*SDN*s) that manage packet-processing switches. In order to configure packet-forwarding rules that have been established in every singular switch, *SDN*s enable programmers to straightly control network behavior.

Software Defined Networking (*SDN*) is built to shift traditional network approached to new network paradigm that enable to facilitate

network management. Network intelligence of *SDN* is centralized in the control pane and network appliances become packet forwarding device that an open interface does computation such as *ForCES*, *OpenFlow*, etc. Currently *SDN* becomes attractive to some group of network operators, service provider, and vendors such Cisco, Oracle, HP etc. *SDN* is a current approached that grow rapidly (Astuto et al., 2014). Even though, *SDN* still has some challenges and issue to be addresses. This paper is organized by section II, III, and IV. In Section II, it describes about the basic concepts of Software-Defined Networking. Section III provides the description of two (2) examples of *SDN* product. Finally section IV discusses about *SDN* issues and research challenges.

2. THE BASIC CONCEPTS OF SOFTWARE-DEFINED NETWORKING (SDN)

The Open Networking Foundation (ONF) is an organization that is affiliate with the *SDN* implementation and standardization (Citrix System Inc 2014) (ONF, 2012). Based on the ONF (2012), *SDN* is a dynamic architecture that is easy to manage, not much costly, and easily adapt, so that, it is suitable for the today's apps that have high-bandwidth. This kind of architecture decouples the network authority and forwarding operation that facilitating the network authority to be straightly programmable and the fundamental infrastructure to be reviewed for apps and network services. The *OpenFlow* protocol is one fundamental aspect in order to implement *SDN*.

As stated by ONF (2012), the *SDN* architecture can be determined as follows (*SDN* 101 and Software-Defined Networking 2012) : (1) Directly programmable: As network control is separated from the existing connection, it can be programmed directly. (2) Agile: Separating access from forwarding allows programmer vigorously adapt to network-wide traffic flow in order to reach requirement. (3) Centrally managed: Software-based *SDN* controllers is centrally manage and keep up network with a broad appearance. It can be obvious to appliances and machines as a particular switch. (4) Programmatically configured: *SDN* enables network controller in configuration, management, security, and network optimization resources rapidly by using automated *SDN* programs. (5)

Open standards-based and vendor-neutral: Unlike the computer network that is built by particular protocols and appiances, *SDN* develops simply network design and operation as rules to be conducted by *SDN* controllers.

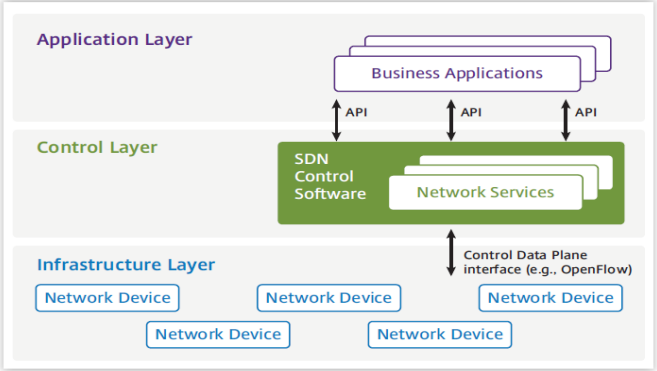

Fig.1. *SDN* architecture as envisioned by the ONF

The figure above describes the basic concepts of *SDN* architecture which are; *(1) Business Applications*, such as video conference is one of the apps that mostly used by user. *(2) Network & Security Services*, allow business apps to execute effortlessly and securely. *(3) Pure SDN Switch*, The functionality of routing protocols in the switch is restricted entirely to the data plane (Citrix Syatem Inc, 2014). *(4) Hybrid Switch*, *SDN* and traditional switching protocols performs synchronously. A network controller allows configuring the *SDN* controller to identify and manage numerous movement progresses while conventional networking protocols is delivered to instruct the remains of the movement on the network. *(5) Hybrid Network* which is a network that employs traditional switches and *SDN* switches, also, executing all that is the similar environment such as pure *SDN* switches or hybrid switches. *(6) Northbound API* enables the control layer and the business application layer to communicate. *(7) Southbound API* enables the control layer and the infrastructure layer to communicate. Protocols allow this communications including *OpenFlow*, (XMPP) and the network configuration protocol.

Network infrastructure of data communication networks compose host interconnected that operate routers, switches and exchanging

communication to bring information to hosts. Routers and switches are limited control of interfaces that is often controlled by specific-vendor. Moreover, network infrastructure is complicated and hard to reconstruct once it has been deployed.

Software-Defined Networking was built to control programing of network data-path. In figure 2, by separating forwarding hardware from the control logic, it effortlessly enables to deploy new protocol and appliances, open network ingenuity and control, and combination of several middleboxes into software management. Contrary to apply and execute protocols on twist of spread device, the network is minimized in order to simplify forwarding device and network controllers.

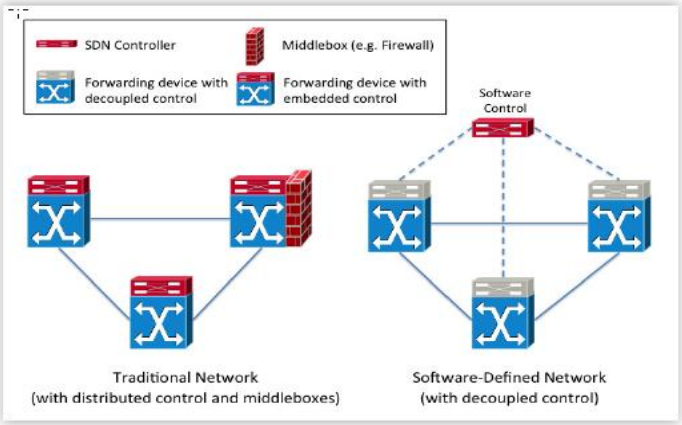

Fig. 2. The traditional network and *SDN* architecture

The present architectures is reviewed to two eminent *SDN* architecture, is called *ForCES* and *OpenFlow* (Astuto, et al., 2013). Both *ForCES* and *OpenFlow* is developed by standard *SDN* rule of dissociation among the command and data planes. Also, it is similar to communication between planes. Nevertheless, both of them have dissimilar in architecture, design, forwarding model and protocol interfaces.

2.1 ForCES (Forwarding and Control Element Separation)

The approach can be defined as the network appliance's centralized architecture has the control feature abstracted from forwarding feature. Still, the network appliance is depicted as one substance. The working group provides the use case in order to integrate current forwarding device with other group of control inside specific network appliance. Hence, the data planes and control are stored inside nearest contiguity such as in similar room or box. Contrary to that control plane is cut wholly from the network tool in "*OpenFlow*-like' *SDN* systems.

The two substances of *ForCES* are the *Forwarding Element (FE)* and the *Control Element (CE)* that communicate by developing the *ForCES* protocol. The responsibility of FE is to provide per-packet handling by implementing underlying hardware. The CE runs command and communicating operation and also utilizes the *ForCES* protocol to train FEs in handling packets. The protocol operates by implementing master-slave model where CEs are masters and FEs are slaves. The greatest developing block of *ForCES* architecture is the *LFB (Logical Function Block)* (Astuto et al., 2014). The LFB can be represented as working block occupying on the FEs that CEs manages by the *ForCES* protocol. The LFB allows the CEs manage the FEs' configuration and their packet process.

2.2 OpenFlow

OpenFlow as well as *ForCES* that is operated by *SDN* rule of disconnected the operation and information forwarding planes allocates communication among the two planes. *OpenFlow* architecture can be described as Figure 3 including the forwarding appliences, *OpenFlow* switch, consist of one or more flow tables and an abstraction layer that communication is secured with a controller by *OpenFlow* protocol (Astuto et al., 2014). Flow tables contain: *(1) match rules*, that is applied to unite arriving packets, also match rules may consist of knowledge that is discovered in the packet header, ingress port, and metadata; *(2) counters*, applied to compile history for specific circulation, for example the amount of the packets received, amount of bytes and range time of the circulation; and *(3) a series of rules* that is used upon a match and give instruction in order to manage matching packets.

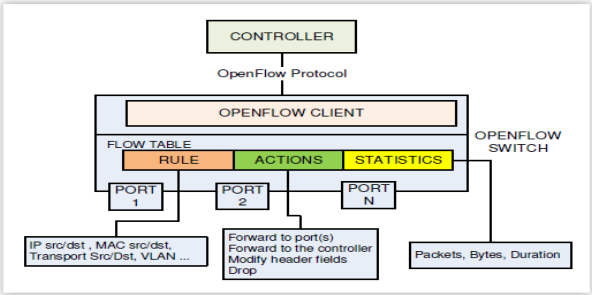

Fig. 3. *OpenFlow* architecture

Packet header rules are elicited and united opposite to the matching fields' portion of the flow table entries while a packet arrives at *OpenFlow* switch. If the entry is match, the switch implements the collection of instruction to be combined with the match flow entry. If the flow doesn't match, the switch relies on the commands represented by the table-miss flow entry. Each flow table must consist of a table-miss entry. This specific entry elaborates a set of instructions to be implemented in handling no match entry.

OpenFlow has both *OpenFlow* and non-*OpenFlow* ports in order to promote non-suitable packets employing normal IP promoting pattern (Astuto et al., 2014). The controller and switch can communicate by *OpenFlow* protocol that interprets group of information to be communicated in these substances in global lik that is secure. Employing the *OpenFlow* protocol a remote controller can simply maintain flow entries from the switch's flow tables (Astuto et al., 2014).

3. DESCRIPTION OF TWO (2) EXAMPLES OF SDN PRODUCT

a. Oracle SDN

Oracle *SDN* (Software Defined Networking) speeds appliance accomplishment and administration flexibility by actively communicating virtual machines (VMs) and servers to networks, storage devices, and other VMs (Hewlett-Packard Development Company, 2012). The network performance can be increased up to 80 Gb/sec server-to-server. Also it can be increased up to 19 times faster live migration, 12 times faster in querying database, and 30 time faster in backups than legacy systems. In result,

Oracle *SDN* maintains virtual flexibility that expenses 50 percent below legacy networking systems.

1) *Oracle SDN: Virtualizing the Data Center Infrastructure;*

Oracle *SDN* can be defined the server *c*onnectivity wholly in software by engaging the basic concepts of virtualization that employs the particular virtual attach which is a software-defined component in assets. It can connect to any virtual machine or server to other resources such as virtual machines, virtual devices, bare metal servers, storage devices and network that is located in any place in the data center. Also, it connects in seconds that provides flexibility and agility without inflexible network configuration to enables control the cloud. Unlike the legacy port and switch-based networking that the server connectivity is employed by complex LAN *configurations.*

2) *Private Virtual Interconnect for IsolatedConnectivity*;

The specific virtual interconnect which is an isolated connectivity component that is employed to connects virtual machines to other data center assets and deployed in seconds. It can employ the private virtual interconnects as well in order to gather the virtual machines, virtual devices, networks, storage devices, and bare metal servers in isolated layer 2 (L2) domains. In contrary, legacy networking need the complex configuration of switches, switch ports, and virtual local area networks (VLANs) to route data among resources (Hewlett-Packard Development Company, 2012). By employing Oracle Virtual Networking, the connections are configured inside Oracle *SDN* itself. As the result, Particular virtual interconnects do not depend on VLAN or port configurations. Also, they are not viewable to external networks except if it is intentionally configured.

3) *Elimination of VLAN Exhaustion;*

As the private virtual interconnect does not depends on traditional networking concepts, so it does not use VLANs or Ethernet address space in order to guarantee privateness from other relations. Consequently,

VLAN enervation is not a matter anymore resources (Hewlett-Packard Development Company, 2012).

4) *Speed and High Scalability;*

With bandwidth of up to 80 Gb/sec per server connection, Oracle *SDN* provides exceptional performance as well as every isolated virtual interconnect also has bandwidth of up to 80 Gb/sec to distribute the business's firmest, most accessible material (Hewlett-Packard Development Company, 2012). One material can provide as many as 1,000 servers and as many as 16,000 private virtual interconnects by automatically routing the fabric enables to configure themself in order to have competent statistics carriage.

5) *Investment Protection and Infrastructure Convergence;*

A sole or dual material components transports both Ethernet and Fibre link movement, causing in full administration easiness and the bottommost physical density (Hewlett-Packard Development Company, 2012). Oracle Virtual Networking secures the remaining network asset as it provides basic Ethernet and Fibre Channel interfaces. The connectivity is viewed as traditional Ethernet network interface cards (NICs) and Fibre Channel host bus adapters (HBAs) within every host.

6) *Redundancy for High Availability;*

Three stages of termination certify higher availability (Hewlett-Packard Development Company, 2012). Firstly for server connection failure that separate data path the traffic is inevitably failed. It is an isolation pat, so that there is no path that able to downgrade the other's functionality. Secondly for a link failure inside the fabric itself, the rest of links route over traffic without any interruptions. Thirdly, all forwarding tables are kept at numerous locations to make sure fabric resilience.

7) *Ability to Deploy Secure Multitenant Environments Rapidly;*

Once various operators share assets, it is often crucial to isolate network and storage data paths to certify segregation (Hewlett-Packard Development Company, 2012). The private virtual interconnect employs as a standalone Layer 2 network that data is viewable only to the assets linked to that isolated virtual interconnect. User apps and information are isolated from other operators' resources as well as data centers inside a shared

environment in order to integrate application SLAs to rapidly reallocate resources once the changing is needed.

8) Integration of Virtual Appliances to Build Virtual Data Centers;

Replacing hardware devices with superior purpose software that operates on virtual machines enables organization to save time, money, power and also space. Oracle *SDN* offers the perfect match to virtual machines, by creating them easy to array, construct, and accomplish (Hewlett-Packard Development Company, 2012). By using Oracle *SDN*, the organization is able to rapidly construct inaccessible connectivity and also deliver up to 80 Gb/sec bandwidth with private virtual interconnects. The users enables to create isolated connections by themselves inside virtual data centers that provides and manages their resources independently without others user is affected.

9) Intuitive Single-Screen Management;

It enables to accomplish all information center connections from a single screen as well as to be visible to private virtual interconnects, virtual devices, virtual switches, physical hosts, network and storage devices. Also, The Oracle Fabric Manager interface enables to manage and control Oracle *SDN*.

10) Ability to Start Small and Grow;

Oracle *SDN* can be started with a minor utilization and mature over time by simply attach new Oracle Virtual Networking components and connect them to build a fabric. As the result, it can begin with a slight amount of servers and mature to a thousand (Hewlett-Packard Development Company, 2012).

b. HP Virtual Application Networks SDN Controller

The HP Virtual Application Networks *SDN* Controller is software that is centralized the control platform for *SDN*. The interface of network structure employs open-standard interfaces and manage protocols (or southbound APIs), such as *OpenFlow* in order to exhibit an abstracted as well as to centralize control plane to network apps. Also they have been developed and united into the controller by providing network services such as network virtualization.

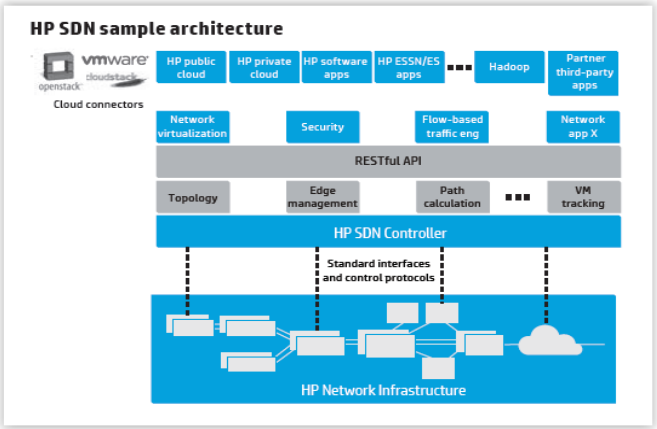

Fig.4 Fig 4.HP *SDN* architecture

The controller is added with vigorous verification and approval techniques that enable HP to exhibit numerous stages of control and entrance to *SDN* apps by existing inside the controller. This technique and others enabling users to increase network flexibility at the same time maintain the network integrity by forbidding unauthorized applications from the unstable and lack performance of network.

The HP Virtual Apps Networks *SDN* Controller is integrated with OpenStack, CloudStack and a RESTful northbound API to exhibit network characteristics and working in order to off-controller *SDN* apps, composition and managing systems, and business apps. The Virtual Application Networks *SDN* Controller offers the policy for a amount of network apps that pull *SDN* to distribute the ability of Virtual App Networks and rise industry alertness (Hewlett-Packard Development Company, 2012).

4. SDN ISSUES AND CHALLENGES

Once proposing a new solution for computer network and *SDN* and protocols such as *OpenFlow* becomes an attractive choice and most organization implement it, also the new challenges arise. This section describes several challenges and issues found by *SDN* as follows: *(1) controller and switch design, (2) scalability and performance in SDNs, (3)*

controller-service interfacing, (4) virtualization and cloud service applications, (5) information centric networking, and (6) enabling heterogeneous networking with SDN (Astuto et al., 2014))

a. Controller and Switch Design

The study that presented on the paper describes one solitary controller enables to switch up to 6 million flows per second while the current research that focus on Beacon controller enables to switch 12.8 million new flows per second in a 12 cores machine with average latency of 24 for each flow. Nevertheless, the logically-centralized controller needs to be physically circulated in order to boost scalability and reliability. The issues are in term of the amount controller required and where to put them within network. The need in dynamic handling of switches to controller is raised in order to add or subtract the set of controllers according to estimation of controller's load. Also, it addresses a technique to dynamically release switches from one to another as the requirement.

b. Software-Defined Internetworking

Environments which are the Internet, request for a control plane that is logically distributed. It will enables themselves engaging autonomous systems (ASes) to be organized individually which is logically centralized and physically distributed controller. The study has proposed software-defined internet architecture in order to disjoin tasks between inter-domain and intra-domain components. The task of intra-domain involves in every domain of boundary routers and associated controller. Modifying inter-domain service models will be restricted to software changing at the inter-domain controllers rather than the complete structure. For examples it can be examined how this architecture is employed to accomplish new Internet services such as information-centric networking, and middlebox service sharing.

c. Controller-Service Interaction

If the controller as a "network operating system", then there should be a obviously definite interface by which apps can contact the fundamental switches and communicate with other apps, and employ system facilities, without needing the app creator to identify the

implementation specifics of the switch (Astuto et al., 2014). Even though there are numerous controllers that obtain their interfaces of application which is happens in the beginning of phases and independent from each other.

Additionally, the northbound API should enable apps to apply dissimilar procedures to the same movement (Astuto et al., 2014). The modularization is proposed in order to certain the instructions set up, so that, it ensures the tasks are not dominated by other rules and accomplished the generalization level employed with a language base on Frenetic. Until a perfect northbound boundary regular occurs, *SDN* apps will carry on to be established in an "ad hoc" fashion and the theory of flexible and portable "network apps" could need to expect (Astuto et al., 2014).

d. Virtualization and Cloud Services

The challenges are comprise express provisioning, competent asset control, and scalability that is able to pointed using *SDN*'s control model (Astuto et al., 2014). In the field of cloud data centers that accommodate Infrastructure as a Service (IaaS) introduce a controlling structure for assests in cloud data centers and points numerous control issues. The authors suggested a data centric and event-driven architecture with open management interfaces, which leverages *SDN* techniques to assimilate network assets into datacenter composition and service provisioning with the purpose of cultivating service-level promises and quicker provision transfer (Astuto et al., 2014).

e. Information-Centric Networking

Information-Centric Networking (ICN) is a fresh model projected for the upcoming design of the Internet, which
aims to rise the productivity of component provision and component availability (Astuto et al., 2014).. The current Internet is information-driven that absorbed on the awareness of location-based addressing and host-to-host communication.

f. Heterogeneous Network Support

Upcoming networks will turn into progressively more heterogeneous, communicating operators and apps over networks reaching from wired, infrastructure-based wireless, to infrastructure-less wireless

(Astuto et al., 2014). In addition, *SDN* enables to deploy and manage network apps and service efficiently. However, Open-Flow of *SDN* that employs centralized control techniques is still have issues and unsuitable to the level of devolution, interruption, and deferment in substructure-less environment.

5. CONCLUSION

This paper provides overview of programmable networks and networks management, which is Software-Defined Networking (*SDN*). It specifically describes *SDN* architecture in detail and also the *OpenFlow* protocol. The characteristics of Oracle *SDN* and HP *SDN* have been offered to market in order to provide solution in managing network. By proposing many benefits of *SDN*, IT organization can shift the way their manage networks to modern network management which is *SDN*. However, as the solution has been proposed, some issues are raised so that it will challenges researcher to improve and solve them.

References

B. N. Astuto, M. Mendonca, X. N. Nguyen, K. Obraczka, T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks." Communications Surveys and Tutorials, IEEE Communicattions Society, Institute of Electrical and Electronics Engineers (IEEE), 2014, 16 (3), pp.1617 - 1634 <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6739370>. <*10.1109/SURV.2014.012214.00180*>. <*hal-00825087v5*>.

Decoding Software Defined Networks," Whipe Paper: Juniper Network, Inc., 2013, < 2000510-001-EN>

Exploring Software-Defined Network with Brocade," White Paper: IP Network, 2014.

G. Ferro, "*OpenFlow* and Software Defined Networking," http://www.ipSpace.net/Webinars.html, (Accessed: 2 December 2014).

Monsanto, J. Reich, N. Foster, J. Rexford, D. Walker, P. Cornell, " Composing Software-Defined Networks," unpublished.

Languages for Software Defined network," Cornell University, Princeton University, U.S. Military Academy.

SDN 101: An Introduction to Software Defined Networking," White Paper: Citrix System, Inc.,2014.

Software-Defined Networking: The New Norm for Networks," ONF White Paper, 2012.

Realizing the power of *SDN* with HP Virtual Application Networks," Technical White Paper: Hewlett-Packard Development Company, L.P., 2012, <4AA4-3871ENW>.

Dinata, R. K. (2017). Analisis Penerapan Teknik Clustering (MSCS) pada Aplikasi SAP/R3 dengan Microsoft Operating System di PT. Arun NGL. *JISKA (Jurnal Informatika Sunan Kalijaga)*, *1*(3), 101-107.

Ula, M., Pratama, A., Asbar, Y., Fuadi, W., Fajri, R., & Hardi, R. (2021, April). A New Model of The Student Attendance Monitoring System Using RFID Technology. In *Journal of Physics: Conference Series* (Vol. 1807, No. 1, p. 012026). IOP Publishing.

Realizing the power of *SDN* with HP Virtual Application Networks," Technical White Paper: Hewlett-Packard Development Company, L.P., 2012, <4AA4-3871ENW>

Software-Defined Networking: Why We Like It and How We Are Building On It," White Paper: Cisco System, Inc., 2013.

V. Ganti, V. Lubsey, M. Shekhar, C. Swan, "Software-Defined Networking Rev.1.0," Open Data Center Alliance, Inc., 2013.