Regular Article

Enhancing Security and Robustness for SDN-Enabled Cloud Networks

Long Tan Le[1,2], **Tran Ngoc Thinh**[1,2]

[1] Ho Chi Minh City University of Technology (HCMUT), Ho Chi Minh City, Vietnam
[2] Vietnam National University Ho Chi Minh City (VNU-HCM), Ho Chi Minh City, Vietnam

Correspondence: Tran Ngoc Thinh, tnthinh@hcmut.edu.vn
Communication: received 30 August 2021, revised 20 September 2021, accepted 21 September 2021
Online publication: 23 October 2021, Digital Object Identifier: 10.21553/rev-jec.294
The associate editor coordinating the review of this article and recommending it for publication was Prof. Tran Manh Ha.

Abstract– **Software-Defined Networking is an emerging network architecture which promises to solve the limitations associated with current cloud computing systems based on traditional network. The main idea behind SDN is to separate control plane from networking devices, thereby providing a centralized control layer integrable to cloud-based infrastructure. The integration of SDN and Cloud Computing brings an immense benefits to network deployment and management, however, this model still faces many critical challenges with regards to availability, scalability and security. In this study, we present a security and robustness SDN-Enabled Cloud model using OpenStack and OpenDaylight. In particular, we design and implement a security clustering-based SDN Controller for monitoring and managing cloud networking, and a hardware platform to accelerate packet processing in virtual switches. We evaluate our proposed model on a practical cloud testbed consisting of several physical and virtual nodes. The experiment results show that the SDN controller cluster significantly improve robustness for the network even in case of being attacked by abnormal network traffic; while the hardware-accelerated switches can be operated in high-performance and well-adapted to the cloud environment.**

Keywords– **Cloud Computing, SDN, OpenStack, OpenDaylight, Network Security.**

1 Introduction

Cloud Computing (CC) is a mature and reliable technology offering a convenient way for network users to access a wide range of "as-a-service" models. The fundamental idea behind CC is to reduce the complexity of underlying infrastructure via abstraction, thereby providing the rapid deployment capability and the ease of management [1]. However, the recent cloud models rely heavily on legacy network architecture, which is now lack of flexibility and manageability due to the unceasing growth of networking devices and data communication.

Software-Defined Networking (SDN) has emerged as a promising paradigm to remove the limitations associated with current cloud network architecture. By separating management functionalities from forwarding devices, SDN shifts the network intelligence into logically centralized software-based controllers, thereby providing the automated configuration in respond to the change of infrastructure [2].

From cloud computing perspective, SDN brings great advantages in many aspects from cost efficiency to application optimization. However, joining SDN to control cloud networks is still a vague concept when there are many critical challenges related to security, availability, and scalability need to address [3]. Notwithstanding various open-source and commercial platforms are well developed for SDN and CC, they have not yet proven the ability to be used as practical platforms for integration of those mentioned technologies. Regarding academic research, many outstanding studies have been proposed on the topic of SDN-Enabled Cloud Computing with impressive effort and contribution [4–6]. However, most research approaches aimed to solve a specific problem, and the usage of SDN in Cloud Computing still has not yet been fully exploited.

In this paper, we propose a SDN-Enabled Cloud Computing model (SDC) for high availability and security leveraging the integration between SDN and OpenStack. Our approach focuses on improving the fault tolerance and performance of SDC in the face of cybersecurity threats. We summarize the major contributions of this paper as follows.

- We have introduced a security and high-availability model for SDC.
- We have designed and implemented a clustering-based SDN Controller with the integration of security functions, and a hardware-based platform for accelerating performance of Open vSwitch in cloud environments.
- We have deployed a cloud testbed using OpenStack and OpenDaylight to carry out experimental evaluation.

The remainder of the paper is organized as follows. Section 2 presents the relevant background and outlines related works about SDN and Cloud Computing. In Section 3, we discuss the concept of our proposed model with the detailed explanation of design and implementation methods. Experimental results are presented in Section 4. Finally, we conclude the paper and suggest the future work in Section 5.

2 Background and Related Works

In this section, we provide a broad view of background knowledge in SDN and Cloud Computing context. Then, we outline the relevant literature regarding the usage of SDN in Cloud environments.

2.1 Cloud Computing

Cloud computing has become a popular buzzword, bringing immense benefits for business and industry over the last decade. The cloud is technically defined as a computation model enabling on-demand network access to shared pools of IT resources over the Internet, thereby minimizing the resource provisioning time and the management effort [1]. Due to the emergence of CC, various architectures (e.g., Public, Private, and Hybrid) and services (e.g., SaaS, PaaS, and IaaS) have been widely applied. In addition, a variety of software and hardware solutions have also been released for deploying and managing cloud environments such as OpenStack [7], Microsoft Azure [8] and VMWare [9], etc.

In this paper, we leverage OpenStack for implementing our proposal model. OpenStack has emerged as a robust open-source cloud management platform ideal for implementing SDC in recent years. This platform is well developed for building cloud infrastructure on standard hardware, and supports various allied projects to control and operate a ubiquitous CC system.

2.2 Software-Defined Networking

The principle of SDN is formed by decoupling traffic forwarding and processing from control, thereby providing logically centralized control and programmability of network services [10]. In SDN, the control plane consists of software-based controllers, responsible for instructing underlying networks. Meanwhile, the data plane is made up of white-box devices simply performing packet forwarding tasks.

Currently, there are many platforms comforting SDN standards. In this paper, we focus on exploiting the use of following SDN platforms:

- *Open vSwitch (OVS)* [11], one of the reputed open-source platform, which is well-suited to a wide range of applications from SDN to Cloud Computing.
- *OpenDaylight (ODL)* [12], one of the most successful open source SDN Controllers providing centralized, programmatic control and network device monitoring.

2.3 SDN-Enabled Cloud Computing

Most of the existing cloud networks are built based on the traditional architecture which is insufficient flexibility to keep pace with dynamic computing resource allocation need in today's enterprises. With the recent advances, SDN has presented it as a perfect candidate to alternate legacy network in cloud data centers.

SDN provides centralized management and control, flexible network functions, and fast packet processing. Those characteristics can help tackle indeed a lot of problems associated with the current cloud models such as centralizing management and control, reducing capital expenditures and operating expenses (CapEx/OpEx), increasing visibility and improving reliability. However, a cloud network based on SDN is being challenged by many critical issues that need to be address, including performance, scalability, virtualization, security, and energy efficiency [3].

Taking performance and security aspects into consideration, the introduction of novel layers in SDN makes this paradigm more agile and flexible, however, it also opens up many network vulnerability surfaces. Especially, the idea of providing a central control plane is obviously subject to the risk of sing point of failure (SPOF). This not only enlarges security issues, but also constrains performance and scalability problems.

Moreover, in both SDN and Cloud Computing, Open vSwitch (OVS) is seen as the main networking component. In cloud environment, OVS is in charge of steering traffic among Virtual Machines (VM) and realizes overlay networks by tunneling protocols. However, OVS is in nature a software-based solution, which is suffered from bad-throughput and low-performance. Therefore, it can become a bottleneck point of system when meeting a sudden increase in network traffic such as flooding attacks.

In this study, we work towards the goal that improves security and high availability for SDC networks.

2.4 Related Works

Recently, some surveys and taxonomies have been presented in concern of SDC context [3, 13–17]. Buyya et al. have many outstanding studies as they introduced the model architecture and vision of SDN in Multi-Cloud Computing [13], and presented an in-depth discussion about the definition, taxonomy, characteristics and research trends of SDC [3]. Moreover, the authors also proposed several SDC frameworks [18, 19], which is used to simulate SDN functions in CC. Even though those tools are helpful for testing SDC environment, it lacks of facilities for real-world deployments.

Many research works leveraged the association between well-known open-source platforms to address the critical challenges of SDC [4, 6, 20–22]. Regarding the high availability of SDC, Son et al. proposed SD-Con [4], a SDC controller built on top of OpenStack and OpenDaylight platforms. Meanwhile, Mayoral et al. [6] presented two SDN orchestration models for CC using single (Se-Arch) and multiple SDN Controllers (ABNO). The authors in [22] also focused on the availability as they replaced the networking components of OpenStack by a cluster of ONOS controllers.

Besides, many studies have been proposed for protecting SDC against DDoS attacks [5, 15, 23, 24]. Krishnan et al. presented CloudSDN [5], a multi-plane collaborative security scheme for SDN–OpenStack. The authors implemented a lightweight monitoring mechanism in forwarding devices to detect attacks, and integrated a third-party analytic engine into SDN control plane to mitigate threats. More recent, Trung et al. [23]

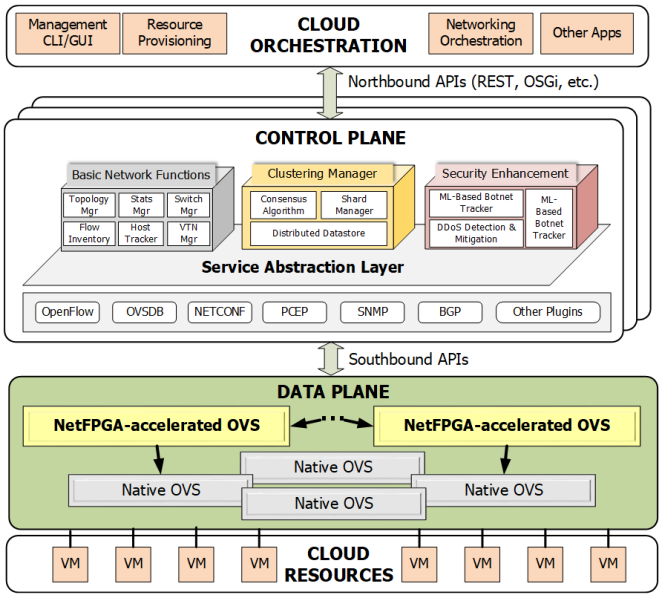

Figure 1. High Availability and Security model for SDN-Enabled Cloud Computing.

introduced a hybrid machine learning approach for classifying networking, along with eHIPF, an IP filtering mechanism to detect attacks. Bhushan et al. [24] presented an attack mitigation strategy based on queuing theory.

To enhance security for SDC, the authors in [25] introduced Brew, a security framework for avoiding the conflict of flow rules in a distrusted SDC environments. Brew consists of several prioritization and classification techniques applying to flow rules to form conflict-free security policies. SeArch [26] is novel NIDS architecture for SDN-Based Cloud IoT environment, focusing on detecting anomalies in IoT devices to mitigate intrusions and stop bottle-neck problems.

We previously proposed ODL-ANTIFLOOD, a comprehensive security solution to protect OpenDaylight controller from saturation attacks [27]. In addition, in [28], we also presented a hardware-accelerating OpenFlow switches with the integration of high-throughput and low-latency SYN Flood defense solution. Those works will be leveraged to use as parts of our proposal model in this study.

3 Methodology

In this section, we discuss the conceptual of our proposal. We begin by explaining the architectural elements, followed by outlining the design and implementation in details.

3.1 Overall Architecture

As discussed in Section 2, we propose in this paper a combination of software and hardware solutions to improve the security and availability for SDC. The conceptual architecture of our model is depicted in Figure 1.

Basically, the architectural elements of our proposed model are mostly adapted from state-of-the-art cloud management platforms (e.g., OpenStack), which can be distributed into three main layers as follows.

- **Cloud Orchestration**: consisting of tools and applications which jointly control, manage and monitor cloud resources. In this work, we will use OpenStack as the management platform for our model.
- **Cloud Resources**: a shared pool of physical and virtual computing resources (e.g. networks, servers, storage, application, and services)
- **SDN-Based Cloud Network**:
  - ***Data Plane***: a set of pure software-based virtual switches associated with the hardware-Accelerated Open vSwitches, providing high-performance connectivity among computing resources in the cloud environment.
  - ***Control Plane***: a set of OpenDaylight controllers, responsible for controlling and managing the cloud network.

We highlight two key points of the model that make our proposal different from related works. Firstly, we employ a secure, clustering-based controller in SDN control plane. Particularly, the cluster is designed and implemented using a well-defined consensus algorithm, along with ODL-ANTIFLOOD [27], a comprehensive solution for protecting control plane from Controller-aimed attacks. The cluster is expected to provide fault-tolerance, decentralized and high availability regardless of SPOF.

Secondly, we introduce a hardware-based architecture for accelerating Open vSwitch, called NetFPGA-Accelerated OVS. The design and implementation of this switch is based on NetFPGA-10G platform [29], which is also part of our previous work [28]. Our solution aims at improving the speed and performance of OVS in cloud datacenters.

3.2 Joint High Availability and Security in SDN Control Plane

To overcome the shortcoming of having a single point of failure, we introduce a cluster model for SDN controllers. In addition, we integrated security functions for protecting system from controller-aimed attacks. This helps enable multiple instances and services to work together as one entity, giving the system an improvement in availability, scalability and data persistence. Figure 2 describes the overview architecture and the control flows between components of our clustering-based SDN Controller.

We utilize OpenDaylight (ODL), one of the best controller platforms for SDN and Cloud Computing, to design and implement the clustering-based system. Current versions of ODL already support clustering implementation based on Raft [30], a well-known consensus algorithm for distributed computing. Figure 2 and Figure 3 depict the workflow of Raft algorithm in the ODL cluster.

Technically, the concept of Raft is based on the agreement of a shared state even in the face of failures to

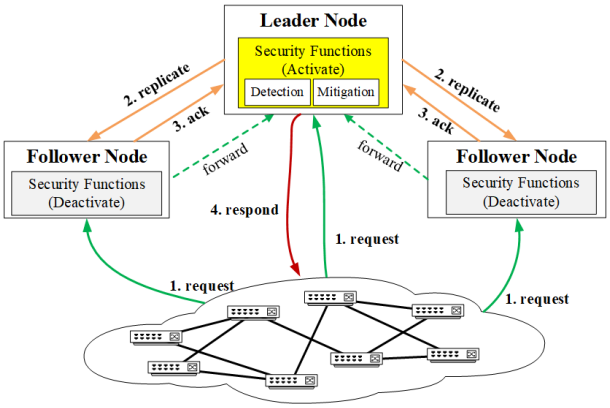

Figure 2. Clustering-based SDN Controller Model.

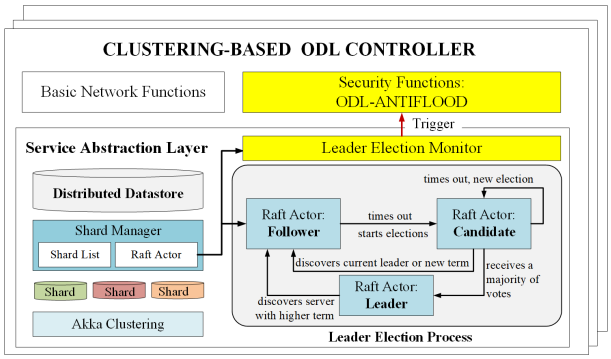

Figure 3. The workflows of ODL Cluster.

maintain overall fault tolerance via three sub-problems:

- *Leader Election*: The cluster system organizes elections to vote for a leader, who will obtain all the responsibility for managing requests from network clients and synchronizing network state among other nodes (the followers).
- *Log Replication*: All requests from clients will be forwarded to the leader and stored in a logs. The log is then replicated to the followers.
- *Safety*: To ensure consistency of the cluster, only the controller which have all the committed entries from the previous terms can become a leader.

Security Enhancement

In the previous work [27], we proposed ODL-Antiflood, a solution for protecting OpenDaylight controller against saturation attacks. The solution includes two security modules Attack Detection and Attack Mitigation. In detection, we use a multi-level statistical-based mechanism to early detect signs of abnormalities in the network. Meanwhile, in mitigation, we collected statistical information from network to build a predictive model and create defense flows to prevent network from anomalies.

The experiment results showed that our methodology is effective and efficient in protecting SDN Controller from popular types of flooding attacks. However, the prior implementation of those functions is only designed and implemented on a single node controller. As shown in Figure 2, to make ODL-ANTIFLOOD work in cluster, we upgrade the defender system by adding

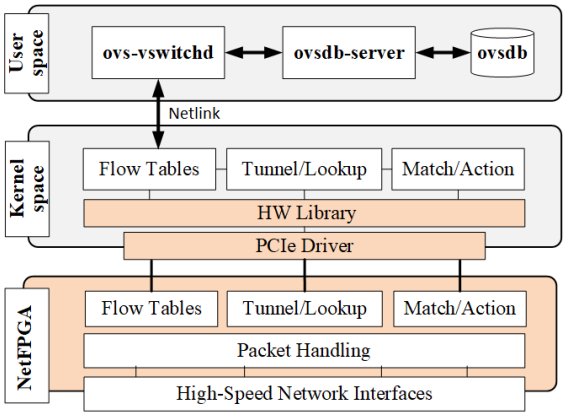

Figure 4. NetFGPA-Accelerate Open vSwitch Architecture.

a trigger mechanism to activate security functions for only the leader. This means that the defender module is still integrated in other members, but it is deactivated.

As described Figure 3, we build a Leader Election Monitor on top of the Shard Manager to observe the raft actor state of each node. If a controller becomes leader, the monitor will create a trigger to wake security functions up in that node. As a result, the leader will be responsible not only for handling request of network clients, but also play as a guardian performing its duties to protect system against attacks.

3.3 Hardware-Acceleration for SDN Data Plane

Figure 4 gives an overview of the NetFPGA-Accelerated OVS architecture. Natively, OVS has three main components: a user space implementing the OpenFlow protocol, a kernel space implementing the forwarding datapath, and a database server storing network information. In OVS, *vswitchd* is a daemon which contains switch's functions, along with a companion kernel datapath for flow-based switching and a osvdb-server for configuring database.

We make use of NetFPGA-10G, a high-speed platform for prototyping networking system, to improve packet processing speed and performance for OVS as shown in Figure 1. In particular, we use NetFPGA-10G to accelerate the datapath, i.e. the kernel space of OVS. The architecture of hardware part in NetFPGA-Accelerated is mapped to the kernel space of the software since it also has the same elements constituting an OpenFlow datapath. For communication between hardware and software, we make an other version of OVS which supports a hardware library for driving the Peripheral Component Interconnect Express (PCI-e) interface.

With NetFPGA-10G cards, OVS can extend the size of flow table, attach virtual network interfaces to gigabit ports, and significantly improve the performance of packet switching functionality. Moreover, The NetFPGA-Accelerated Open vSwitches will be associated with native software-based OVS to form an aggregation network in cloud environments, where they play as core switches, providing transmission in high-speed bandwidth and performance.

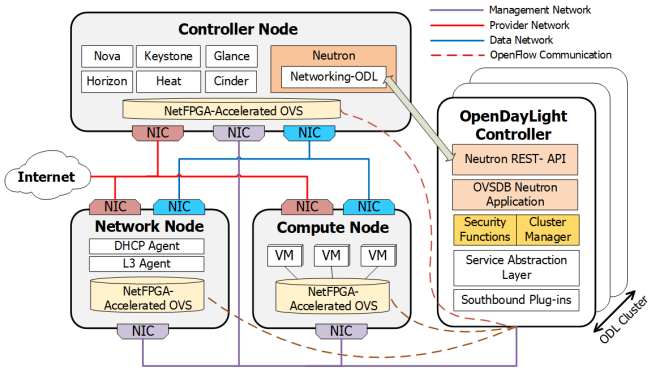

Figure 5. Testbed for SDN-Enabled Cloud Computing Model.

4 Evaluation

The following section describes the evaluation of our proposal model. We first introduce the cloud testbed environment. And then, we conduct test cases to analyze and evaluate the performance of our model.

4.1 Testbed Deployment

Figure 5 describes the logical deployment of our model. As discuss in Section 3, to deploy the SDC, we utilize OpenStack (Stein version) in cooperation with ODL controllers (Neon version). OpenStack supports self-services networking framework called Modular Layer 2 (ML2) neutron plug-in, which allows this platform simultaneously utilizing the variety of technologies in the network layer. For the purpose of enabling SDC, we use OpenDaylight as a back-end SDN controller to manage pool of Open vSwitch in OpenStack environments.

The testbed consists of three nodes (CPU Intel Core i7, 8th Gen, 20GB RAM) for running OpenStack services and three virtual machines (2 Physic Cores, 6GB RAM) for running OpenDaylight cluster. The functionality of each node is listed as follows.

- *Controller node*: the OpenStack orchestration managing cloud nodes, applications and providing user interface (Dashboard).
- *Compute node*: Provides set of Virtualbox-based Virtual Machines (VMs) for resource provision.
- *Network node*: Provides connectivity among nodes, virtual machines, and external networks.
- *ODL cluster*: Provides the cluster of ODL controllers, managing pool of virtual switches in OpenStack

In each OpenStack node, we plug a NetFPGA-10G card to provide network interfaces and associate with OVS, through which they connect to three different networks:

- *Management network* (10.0.0.0/24): used for traffic between the nodes that make up the Openstack infrastructure.
- *Tunnel network* (10.0.1.0/24): used for traffic between virtual machines and the network node.
- *External Network* (10.0.10.0/24): used for virtual machines to access from the Internet through the network node.
- *OpenDaylight network* (172.28.25.0/24): used for ODL controller to communicate with each other in cluster.

4.2 Experimental Results

4.2.1 Clustering-based ODL Controller: We conduct some experiments to evaluate the consensus and fault-tolerant of the cluster when facing a flooding attack. To do that, we carry out the controller-aimed attacks in two scenarios listed as follows.

1) **Scenario 1**: Launch TCP SYN Flooding DDoS Attacks with high-speed rate (Gigabit per second - Gbps) from cloud users, aiming to shutdown one controller in ODL cluster without ODL-ANTIFLOOD.
2) **Scenario 2**: Launch a TCP SYN Flooding DDoS Attack with high-spped rate (Gbps) from cloud users, aiming to shutdown one controller in ODL cluster with ODL-ANTIFLOOD.

Tables I, II, III give the results about ODL cluster activities before, during, and after the attack being launched (without the integration of ODL-ANTIFLOOD). In those tables, we use some following terminologies to show important parameters of cluster.

- *Ip Address*: The IP address assigned for controllers
- *CurrentTerm*: The number of elections organized in cluster system
- *CommitIndex*: The number of acknowledge when a member receives new replicated log from the leader
- *ReplicatedToAllIndex*: The total replicated logs in cluster members
- *InMemoryJournalLogSize*: The amount of log waiting for appending

Taking the first scenario into consideration, in normal situation (i.e. before attacks, Table I), the cluster has a leader (Controller 3) and two followers (Controller 1, Controller 2), and all related parameters such as *CurrentTerm*, *CommitIndex*, *ReplicatedToAllIndex*, and *InMemoryJournalLogSize* are always the same fors all member in the cluster.

When the attack aimed to Controller-2 happens, this controller is shutdown due to CPU saturation ($>$ 90% usage). However, the remaining members are still available, held a new election to vote for a new leader (Table II). Once finishing attack, Controller-2 rehabilitate and rejoin to the cluster (Table III). Also, we launch attacks with various period time, and observe that the recovery time of Controller-2 is approximate 13 seconds regardless of the difference in attack time (Figure 6).

In the second scenario, the cluster keeps working smoothly under normal circumstances, ODL-ANTIFLOOD is integrated and activated in Controller-3 (The leader). Under attacks, the CPU resource and network bandwidth of Controller-2 increases suddenly in short time at the beginning as described in Figure 7 ($>$ 80% CPU Usage and 8000 requests/s), this effect also propagates to the leader when it receives

Table I
ODL Cluster with 3 Nodes Available

	Controller-1	Controller-2	Controller-3
IP Address	172.28.25.215	172.28.25.213	172.28.25.214
ShardName	member-1	member-2	member-3
RaftState	Follower	Follower	Leader
Leader	member-3	member-3	member-3
Term	6	6	6
CommitIndex	40	40	40
ReplicatedToAllIndexx	39	39	39
InMemoryJournalLogSize	1	1	1

Table II
ODL Cluster 2 Nodes Available and 1 Node Shutdown

	Controller-1	Controller-2	Controller-3
IP Address	172.28.25.215	172.28.25.213	172.28.25.214
ShardName	member-1	member-2	member-3
RaftState	Leader	Candidate	Follower
Leader	member-1	null	member-1
Term	19	28	19
CommitIndex	2247	1345	2247
ReplicatedToAllIndexx	1345	-1	1345
InMemoryJournalLogSize	902	1	902

Table III
ODL Cluster With 2 Nodes Available and 1 Node Rejoin

	Controller-1	Controller-2	Controller-3
IP Address	172.28.25.215	172.28.25.213	172.28.25.214
ShardName	member-1	member-2	member-3
RaftState	Leader	Follower	Follower
Leader	member-1	member-1	member-1
Term	35	35	35
CommitIndex	2849	2849	2849
ReplicatedToAllIndexx	2848	2848	2848
InMemoryJournalLogSize	1	1	1

a huge amount of forwarding requests from the follower (> 50% CPU Usage and 4000 requests/s). However, ODL-ANTIFLOOD has the ability to detect this anomaly behavior, it helps the leader realize attacks and apply defense flow rules to the switches in cloud networks. Therefore, the impact of attacks does not cause any serious problems, and the cluster can still stand without any interrupts.

4.2.2 NetFPGA-Accelerated Open vSwitch: To evaluate the efficiency of our NetFPGA-Accelerated Open vSwitch, we conduct experiments to measure its performance in comparison with software-based OVS. In our tests, we generate multiple sets of variant size packet (64-1500 bytes) to forward to the OVS switches. The traffic is flow based on a NIC-OVS-NIC topology, i.e. packets will be transmitted from a OVS physical port to Virtual Machine port, and then back to the physical port. For this purposes, we use 10-Gbps ports of the

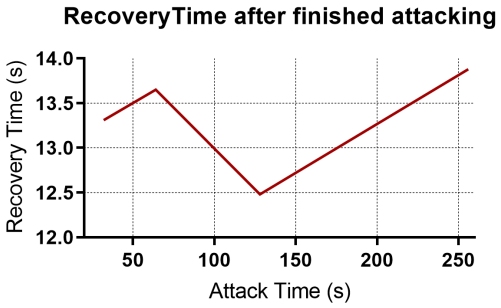

Figure 6. The recovery time of Controller-2 after finishing attack.

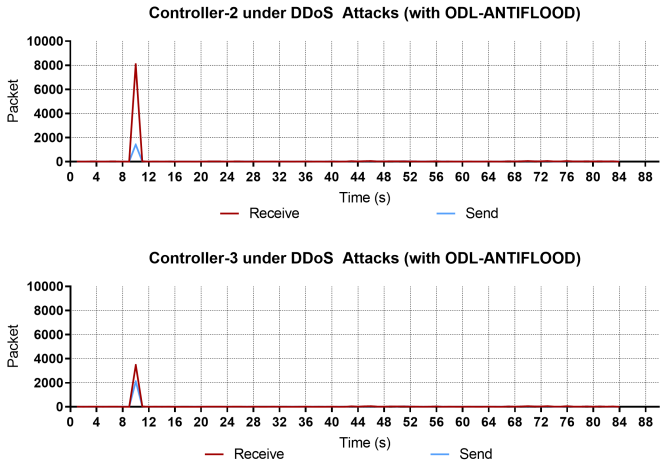

Figure 7. Switch-to-Controller Request/Respond.

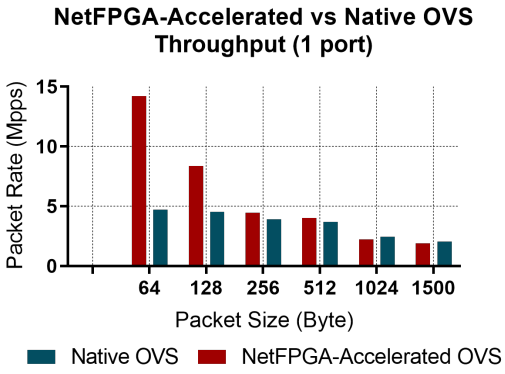

Figure 8. NIC-OVS-NIC Throughput.

NetFPGA-Accelerated OVS, and attach Intel® 82599 NIC cards (each has two 10 Gigabit Ethernet ports) to the pure software OVS.

The measured throughput in one port is shown in Figure 8. As the figure described, the throughput of our switch can reach up to 14 millions packets per second (Mpps) for one port (with 64-byte packets) and 32 Mpps for four ports, outperforming the native OVS (approx. 4 Mpps). In general, the processing of NetFPGA-Accelerated OVS is more efficient than the software-based OVS with the short or medium packets (64-512 bytes). With large packets (1024-1500 bytes), the performance of both switches is relatively equal.

5 Conclusion

In this paper, we presented a high security and availability SDN-Enabled Cloud Computing model deploying in the integration of OpenStack and OpenDaylight. We design and implement a secure, clustering-based SDN Controller for monitoring and managing cloud networks, and use hardware acceleration to improve the speed and performance for cloud network components. We also carry out an evaluation with a cloud testbed consisting of several physical and virtual nodes. The experiment results show that the cluster controller can provide good resistance when it keep working in a high available manner even in the case of being attacks; and quick resilience since if a node is shut down, it can rejoin to the cluster again in short time (approx. 13s). Meanwhile, the hardware-accelerated switches can be operated with high-throughput (max. 14 Mpps) and well-adapted to virtualization environments.

Acknowledgment

This research is funded by Ho Chi Minh City University of Technology - VNU-HCM under grant number C2020-20-33. We acknowledge the support of time and facilities from Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for this study.

References

[1] P. M. Mell and T. Grance, "Sp 800-145. the nist definition of cloud computing," Gaithersburg, MD, USA, Tech. Rep., 2011.

[2] D. Kreutz, F. M. V. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[3] J. Son and R. Buyya, "A taxonomy of software-defined networking (sdn)-enabled cloud computing," *ACM Computing Surveys*, vol. 51, no. 3, May 2018.

[4] J. Son and R. Buyya, "SDCon: integrated control platform for software-defined clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 1, pp. 230–244, Jan 2019.

[5] P. Krishnan and K. Achuthan, "CloudSDN: Enabling SDN Framework for Security and Threat Analytics in Cloud Networks," in *Proceedings of the International Conference on Ubiquitous Communications and Network Computing*. Cham: Springer International Publishing, 2019, pp. 151–172.

[6] A. Mayoral, R. Vilalta, R. Muñoz, R. Casellas, and R. Martínez, "SDN orchestration architectures and their integration with Cloud Computing applications," *Optical Switching and Networking*, vol. 26, pp. 2–13, 2017.

[7] OpenStack, "OpenStack," https://www.openstack.org/.

[8] M. Azure, "Microsoft Azure," https://azure. microsoft. com/en-us/.

[9] VMWare, "VMWare," https://www.vmware.com/.

[10] O. N. Foundation, *SDN Architecture*. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR-521_SDN_Architecture_issue_1.1.pdf

[11] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar, K. Amidon, and M. Casado, "The design and implementation of open vswitch," in *Proceedings of the 12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. Oakland, CA: USENIX Association, May 2015, pp. 117–130.

[12] J. Medved, R. Varga, A. Tkacik, and K. Gray, "OpenDaylight: Towards a Model-Driven SDN Controller architecture," in *Proceeding of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, Jun. 2014, pp. 1–6.

[13] R. Buyya and J. Son, "Software-defined multi-cloud computing: A vision, architectural elements, and future directions," in *Proceedings of the International Conference on Computational Science and Its Applications*, 2018, pp. 3–18.

[14] A. A. Abbasi, A. Abbasi, S. Shamshirband, A. T. Chronopoulos, V. Persico, and A. Pescapè, "Software-defined cloud computing: A systematic review on latest trends and developments," *IEEE Access*, vol. 7, pp. 93 294–93 314, 2019.

[15] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (sdn) and distributed denial of service (ddos) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Communications Surveys Tutorials*, vol. 18, no. 1, pp. 602–622, Firstquarter 2016.

[16] 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, Nov. 2013.

[17] Y. Jararweh, M. Al-Ayyoub, A. Darabseh, E. Benkhelifa, M. Vouk, and A. Rindos, "Software defined cloud: Survey, system and evaluation," *Future Generation Computer Systems*, vol. 58, pp. 56 – 74, 2016.

[18] J. Son, T. He, and R. Buyya, "Cloudsimsdn-nfv: Modeling and simulation of network function virtualization and service function chaining in edge computing environments," *Software: Practice and Experience*, vol. 49, no. 12, pp. 1748–1764, 2019.

[19] J. Son, A. V. Dastjerdi, R. N. Calheiros, X. Ji, Y. Yoon, and R. Buyya, "CloudSimSDN: Modeling and Simulation of Software-Defined Cloud Data Centers," in *Proceedings of the 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, May 2015, pp. 475–484.

[20] T. Hu, Z. Guo, P. Yi, T. Baker, and J. Lan, "Multi-controller based software-defined networking: A survey," *IEEE Access*, vol. 6, pp. 15 980–15 996, 2018.

[21] D. Suh, S. Jang, S. Han, S. Pack, T. Kim, and J. Kwak, "On performance of OpenDaylight clustering," in *Proceedings of the IEEE NetSoft Conference and Workshops (NetSoft)*, Jun. 2016, pp. 407–410.

[22] F. Foresta, W. Cerroni, L. Foschini, G. Davoli, C. Contoli, A. Corradi, and F. Callegati, "Improving OpenStack Networking: Advantages and Performance of Native SDN Integration," in *Proceedings of the IEEE International Conference on Communications (ICC)*, May 2018, pp. 1–6.

[23] T. V. Phan and M. Park, "Efficient distributed denial-of-service attack defense in sdn-based cloud," *IEEE Access*, vol. 7, pp. 18 701–18 714, 2019.

[24] K. Bhushan and B. B. Gupta, "Distributed denial of service (ddos) attack mitigation in software defined network (sdn)-based cloud computing environment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 10, 04 2018.

[25] S. Pisharody, J. Natarajan, A. Chowdhary, A. Alshalan, and D. Huang, "Brew: A security policy analysis framework for distributed sdn-based cloud environments," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 6, pp. 1011–1025, Nov 2019.

[26] T. G. Nguyen, T. V. Phan, B. T. Nguyen, C. So-In, Z. A. Baig, and S. Sanguanpong, "Search: A collaborative and intelligent NIDS architecture for SDN-Based Cloud IoT Networks," *IEEE Access*, vol. 7, pp. 107 678–107 694, 2019.

[27] N. T. Tran, T. L. Le, and M. A. T. Tran, "ODL-ANTIFLOOD: a comprehensive solution for securing

opendaylight controller," in *Proceedings of the International Conference on Advanced Computing and Applications (ACOMP)*, Nov. 2018, pp. 14–21.
[28] D.-M. Ngo, C. Pham-Quoc, T. Ngoc Thinh, and E. Kamioka, "An efficient high-throughput and low-latency SYN flood defender for high-speed networks," *Security and Communication Networks*, vol. 2018, Jan. 2018. [Online]. Available: https://doi.org/10.1155/2018/9562801
[29] NetFPGA10G, "Netfpga 10G," http:// netfpga.org/ 2014/.
[30] D. Ongaro and J. Ousterhout, "In search of an understandable consensus algorithm," in *Proceedings of the 2014 USENIX Conference on USENIX Annual Technical Conference*, ser. USENIX ATC'14. USA: USENIX Association, 2014, p. 305–320.

Long Tan Le received his B.E. degree in Computer Engineering from Ho Chi Minh City University of Technology (HCMUT), Vietnam, in 2018. He is pursuing a MSc degree in Computer Science at HCMUT. Currently, he is working as a researcher and teaching assistant in the Faculty of Computer Science and Engineering, HCMUT. His research interests include Software-Defined Networking, Network Security and Machine Learning Applications for Networking.

Tran Ngoc Thinh received the B.E. degree in Computer Engineering from Ho Chi Minh City University of Technology (HCMUT), Vietnam, in 1999. He received his M.E. and Ph.D. from the King Mongkut's Institute of Technology Ladkrabang, Thailand in 2006 and 2009, respectively. He is currently an Associate Professor at the Faculty of Computer Science and Engineering, HCMUT. His research interests include network security, AI on reconfigurable devices, wireless sensor networks and IoT applications. He is a member of IEEE.