







## Article

# A DDoS Vulnerability Analysis System against Distributed SDN Controllers in a Cloud Computing Environment

Sumit Badotra <sup>1</sup>, Sarvesh Tanwar <sup>2</sup>, Salil Bharany <sup>3,\*</sup>, Ateeq Ur Rehman <sup>4,\*</sup>, Elsayed Tag Eldin <sup>5,\*</sup>, Nivin, AGhamry <sup>6</sup>, and Muhammad Shafiq <sup>7,\*</sup>

<sup>1</sup> Department of Computer Science and Engineering, Lovely Professional University, Phagwara 144411, India  
<sup>2</sup> AIT, Amity University, Noida 201303, India  
<sup>3</sup> Department of Computer Engineering and Technology, Guru Nanak Dev University, Amritsar 143005, India  
<sup>4</sup> Department of Electrical Engineering, Government College University, Lahore 54000, Pakistan  
<sup>5</sup> Faculty of Engineering and Technology, Future University in Egypt, New Cairo 11835, Egypt  
<sup>6</sup> Faculty of Computers and Artificial intelligence, Cairo University, Giza 12613, Egypt  
<sup>7</sup> Department of Information and Communication Engineering, Yeungnam University, Gyeongsan 38541, Korea  
\* Correspondence: salil.bharany@gmail.com (S.B.); ateeq.rehman@gcu.edu.pk (A.U.R.); elsayed.tageldin@fue.edu.eg (E.T.E.); shafiq@ynu.ac.kr (M.S.)

**Abstract:** Software-Defined Networking (SDN) is now a well-established approach in 5G, Internet of Things (IoT) and Cloud Computing. The primary idea behind its immense popularity is the separation of its underlying intelligence from the data-carrying components like routers and switches. The intelligence of the SDN-based networks lies in the central point, popularly known as the SDN controller. It is the central control hub of the SDN-based network, which has full privileges and a global view over the entire network. Providing security to SDN controllers is one such important task. Whenever one wishes to implement SDN into their data center or network, they are required to provide the website to SDN controllers. Several attacks are becoming a hurdle in the exponential growth of SDN, and among all one such attack is a Distributed Denial of Service (DDoS) attack. In a couple of years, several new SDN controllers will be available. Among many, Open Networking Operating System (ONOS) and OpenDayLight (ODL) are two popular SDN controllers laying the foundation for many other controllers. These SDN controllers are now being used by numerous businesses, including Cisco, Juniper, IBM, Google, etc. In this paper, vulnerability analysis is carried out against DDoS attacks on the latest released versions of both ODL and ONOS SDN controllers in real-time cloud data centers. For this, we have considered distributed SDN controllers (located at different locations) on two different clouds (AWS and Azure). These controllers are connected through the Internet and work on different networks. DDoS attacks are bombarded on the distributed SDN controllers, and vulnerability is analyzed. It was observed with experimentation that, under five different scenarios (malicious traffic generated), ODL-3 node cluster controller had performed better than ONOS. In these five different scenarios, the amount of malicious traffic was incrementally increased. It also observed that, in terms of disk utilization, memory utilization, and CPU utilization, the ODL 3-node cluster was way ahead of the SDN controller.

**Keywords:** ODL 3-node; cluster; SDN controllers; DDoS attacks; open network operating system; cloud computing



**Citation:** Badotra, S.; Tanwar, S.; Bharany, S.; Rehman, A.U.; Eldin, E.T.; Ghamry, N.A.; Shafiq, M. A DDoS Vulnerability Analysis System against Distributed SDN Controllers in a Cloud Computing Environment. *Electronics* **2022**, *11*, 3120. <https://doi.org/10.3390/electronics11193120>

Academic Editor: George Angelos Papadopoulos

Received: 18 August 2022

Accepted: 26 September 2022

Published: 29 September 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

Software-Defined Networking (SDN) was introduced in the year of 2008 and since then it is considered a well-established technology from a radically novel concept in the networking sector. A lot of advancement has already been introduced in its architecture. Well-established APIs (Application Programming Interfaces) such as OpenFlow protocol, automation tools and many more are there in its recent advancement [1,2]. The central control point of SDN architecture known as the SDN controller has also obtained new rays

of hope. Starting from NOX, many new varied SDN controllers are there in the market. Since it is the core part, there is a huge responsibility for choosing the best SDN controller for the underlying infrastructure [3,4]. The SDN controller configures the network's complete perspective. It can be stated as an application through which it manages the overall flow of the network traffic. The most important aspect in relevance to SDN architecture is the security of the SDN controller [5]. If the SDN controller's security is compromised, then entire control over the network will be taken over by the intruder. Attackers can manipulate the legitimate traffic from one place to another and sometimes also falsify the original Open daylight (ODL) and Open Networking Operating System (ONOS) both are open-source SDN controllers [6,7]. They are hosted by the Linux foundation. Nowadays, attackers are making use of varied applications to get installed on a controller and potentially reconfigure the network [6]. This can lead the entire network to something unexpected. Therefore, it is mandatory to look for the various security aspects of the SDN controller before you. There are many attacks/threats to which SDN architecture is vulnerable but among all Distributed Denial of Service (DDoS) attacks are the most disruptive [8]. The whole SDN-based network collapse implements SDN into your underlying infrastructure [6,7] also falsifies the original SDN controller. Nowadays, attackers are making use of varied applications to get installed on the controller and potentially reconfigure the network [6]. This can lead the entire network to something unexpected. Therefore, it is mandatory to look for the various security aspects of the SDN controller before you. There are many attacks/threats to which SDN architecture is vulnerable but among all DDoS attacks are the most disruptive [8]. With the bombardment of massive amounts of traffic heading in the direction of the objective, botnets halt the normal functionality of the network. DDoS attacks can be launched on various soft targets in SDN architecture such as APIs, Data Plane, Application layer and SDN central point (controller) [9]. In this paper, we are considering the vulnerability of SDN controllers against DDoS attacks. When the intruder launches the DDoS attack on the SDN controller, a huge amount of traffic is forwarded towards the SDN controller, the flow table of SDN controller has limited memory, and thus it halts its functionality. The legitimate traffic flow also stops in the meantime and therefore at the end the whole SDN based network collapses. ODL and ONOS both are open source SDN controllers [10]. They are hosted by the Linux foundation. Among all the available SDN controllers, both ODL and ONOS are considered to be the most popularly used [10]. There are some well-defined APIs used in both of them. For northbound, REST API is often used, whereas, for southbound, API OpenFlow protocol is used. For both, there are varied versions available. For example, in the case of ODL, since 2014, there are around 13 versions available, whereas, on the other hand, in the case of ONOS, there are around 22 versions available [11]. The latest stable versions for ODL and ONOS are Aluminum (released in September 2020) and Velociraptor (LTS) 2.5 (released in December 2020), respectively [12]. Due to the immense popularity of both the controllers, with every released version, developers are making the best of their efforts to add new security paradigms to them. However, at the same time, intruders are making use of dynamic and new ways to launch attacks on the SDN network [13].

With every passing year due to the immense popularity of SDN and its exponential growth, most of the focus is towards its security [14,15]. As mentioned earlier, since SDN controller is the main control point, thus vulnerability analysis of this is to be addressed properly. In this paper, we have considered ODL and ONOS as study points. In order to check the vulnerability of ODL (Aluminum) and ONOS (2.5), the latest stable released versions against DDoS attacks, different scenarios and tools are used.

### 1.1. Major Contributions

The contributions of this paper are well accomplished and summarized:

- To analyze the DDoS attack vulnerability of ODL and ONOS (stable released versions) by using different penetration tools and scripts on distributed cloud environments (AWS, Azure);
- The network topology is created using the Mininet emulation tool on the host PC;

- Generated malicious network traffic is bombarded on the leader node of every controller and, for this, we have considered five varied scenarios;
- Real-time network analysis is achieved through Wireshark on varied scenarios;
- From experimentation, we have found that a vulnerability system against DDoS attacks on both ODL and ONOS (located distributed) is successful in a distributed cloud environment.

### 1.2. Structure of the Paper

The rest of the article is organized as follows: The work done by other researchers has been summarized in Section 2, and the comparison has been conducted with our proposed work. In Section 3, we present the methodology that was utilized in the course of conducting the research for this particular piece of writing. While the findings are presented in Section 4, further analysis and interpretation of those findings are depicted. In the end, in Section 5, the conclusions along with the future directions to this research work are given.

## 2. Related Work

The authors of the paper [1] stated that the fundamental component of software-defined networking is a controller. As a result, it is critical that the controller has characteristics that improve overall network performance while simultaneously ensuring service continuity in the case of a loss. ODL and ONOS are two prominent open-source distributed controllers. The purpose of this study is to examine how ODL and ONOS manage different failure situations in their data and control planes. The evaluation evaluates both data plane connection and switch failures and recovers various flows in reactive and proactive modes.

The authors of the paper [2] examined the performance of four SDN controllers utilizing the Mininet WiFi platform in different sizes of a simulated wireless network, including throughput, delay, and jitter, as well as packet loss: ODL, POX, ONOS and RYU. The results of this experiment may be used to choose the most suitable controller, taking into account the many characteristics and operational states of the wireless network.

This research [3] took into account the optimal SDN deployment, as well as the choice of a suitable controller platform for evading the limitations of traditional networks and the assessment criteria for the performance of SDN open flow controllers. In addition, the authors surveyed open flow controller performance assessment criteria for SDNs. The survey then moves on to potential literature or research on the performance analysis and evaluation of open flow-based SDN controllers conducted by different scholars and in groups of two or more controllers in relation to different network conditions, network criteria/parameter/metrics, and scaling of network load resources, the most important aspect of which is varying network topology design.

In the conclusion, it aims to draw attention to the present research gaps in the performance bottleneck benchmark for the controller and offers a satisfactory solution after extensive testing. The authors of this study suggest employing SDN-ESRC, a resilient and endogenously secure SDN control plane, to foil exploits of vulnerabilities and backdoors [4]. SDN-ESRC employs a variety of heterogeneous controllers, including RYU, OpenDayLight, and ONOS, to build the control plane. Furthermore, it picks out different heterogeneous controller instances from the controllers set in a dynamic and adaptable manner to identify and repair erroneous control messages. The ESRC's SDN architecture has two flaws as a result of weighing several controllers: (1) an increase in the time required to bring the network up to date, and (2) an inability to guarantee a very high degree of controlled security.

To combat the first problem, SDN-ESRC uses master modification mode to expedite network updates and detect malicious control messages. When it comes to the second problem, SDN-ESRC presents the comparison modification mode, which enables high availability in real time. They provide a methodology for evaluating the security efficacy of ESRC in the context of SDN and conduct theoretical research on ESRC's performance in three prevalent backdoor attack scenarios.

The needs of the most popular open-source NOSs are assessed in this study [5]. Non-functional variables such as accessibility, usability, maturity, security, and interoperability were examined, virtualization, fault checking and debugging, packet forwarding mechanisms, and traffic protection solutions were also included as functional aspects. The Analytical Hierarchy Process is a decision-making aid (AHP) that was utilized to analyze the parameters of the tested NOSs, which included ONOS, ODL, Floodlight, Ryu, POX, and Tungsten. In contrast to the other NOSs studied, we discovered that ODL is the most matched NOS for CDC. When compared to the other NOSs, however, ODL and ONOS provide essentially equal results.

The Internet of Things (IoT) represents a radical shift in how we interact with one another online. It may be used for a variety of things. With the proliferation of IoT devices, DDoS assaults are inevitable [6]. The primary contributions of this paper are the development of a SDN based on the OpenDaylight platform and a novel approach to detecting DDoS attacks in the IoTs using the concept of recurrent neural networks. Both of these contributions are based on the OpenDaylight platform (Neural Networks). In addition, a three-tiered architecture is given for both monitoring for and reacting to distributed denial of service attacks. The programme employs a one-of-a-kind activation mechanism in addition to machine learning and deep learning ideas in order to arrive at a conclusion on the classification of the assault. There are a total of 177 evaluations carried out on the proposed classifier. During the simulation, the tools Mininet and Wireshark were used to assist in the identification of the many DDoS assaults that were carried out on the simulated network. OpenDayLight [7] is a Software-Defined Network controller that is open-source and free to use. It makes it possible to create programmable networks. The OpenDayLight community and the business sector have worked together to create networks that are programmable and interoperable for the benefit of service providers, corporations, institutions, and other organisations. To ensure that the final output is of the highest possible quality despite the ongoing development of the project and the release of new versions, stringent testing is required at all times. It is vital to do regression testing in order to guarantee that modifications to the code will not render currently-used functions inoperable. A powerful selection technique will also be required to assure a lesser number of test cases for regression testing, since such a huge project is certain to have many such situations. This is due to the fact that regression testing is an essential part of the testing process. In light of the findings of our study, we strongly advise making use of combinatorial testing while developing a test suite in order to achieve both a lower total number of test cases and a greater level of test coverage. Networking concepts such as Network Function Virtualization (NFV), Multi-access Edge Computing (MEC), and SDN, amongst others, were introduced as a response to the increasing demands for performance, portability, scalability, and energy efficiency of networks in the era of the IoT. This was done in response to the fact that the number of connected devices in the world is growing at an exponential rate (paper [8]). SDN divides a network into a control plane and a data plane in order to solve the inefficiency, inflexibility, and static configuration of conventional networks by separating the network into a data plane and a control plane. Because the control layer is in charge of the data plane, SDN applications that run at the application layer are able to take use of the flexibility of the network. The controller on the control plane acts as the “brain” of the SDN system and is responsible for managing the network as a whole. The instructions for processing the packets are given to the architecture that lies behind. On the other hand, the administration of large-scale networks necessitates the use of several controllers. This will result in an SDN environment that is fragmented due to the fact that many controllers will attempt to restore the established order. The detailed literature survey is illustrated in Table 1.

**Table 1.** Literature Survey overview.

Paper	Title	About
[1]	An Analysis of the Reliability of the Software-Defined Network Controllers ONOS and ODL	The purpose of this study is to examine how ODL and ONOS manage different failure situations in their data and control planes.
[2]	Comparative Study of the Effectiveness of a Number of SDN Controllers Over a Range of Network Sizes in SDWN	Based on the many features and conditions of the wireless network, the results of this experiment may be utilized to determine the most suitable controller.
[3]	A Comprehensive Study of the Performance of Several Open Flow Software-Defined Network Controllers by Addressing Scalability Metrics Based on Numerous Topology Designs on Software-defined Networks	There are a number of challenges associated with deploying SDN, including determining which controller platform is best for obtaining beyond conventional network constraints and determining how to evaluate the performance of SDN open flow controllers. The inquiry then moves on to the potential literature or work done by a range of academics on the issue of performance analysis and evaluation of open flow-based SDN controllers.
[4]	A Control Plane That Is Both Secure and Resilient for Software-Defined Networks Is Known as SDN-ESRC.	They offer a framework for evaluating SDN-ESRC and theoretically look at how well it protects against three common backdoor attack scenarios. Simulations and testing have been done while SDN-ESRC has been integrated in a prototype system.
[5]	An Evaluation and Analysis of the Available SDN Network Operating Systems with the Purpose of Selecting the Most Appropriate One for Cloud Data Centers	We found that ODL is the most similar NOS to CDC compared to the others we looked at. However, ODL and ONOS provide almost identical findings when evaluated against the other NOSs.
[6]	Detection of Distributed Denial of Service Attacks in the Internet of Things Using Recurrent Neural Networks	The programme classifies the sort of attack using an exclusive activation function and ideas from machine learning and deep learning. We put the suggested classifier through its paces 177 times. Mininet and Wireshark were used throughout the simulation to properly detect the many DDoS attempts on the network.
[7]	Test Design of SDN Controllers from the Perspective of Combinatorial Testing	Programmable networks have been made possible by OpenDayLight, an open-source software-defined network controller. To provide programmable and interoperable networks to service providers, corporations, institutions, and organizations, it has united business and the OpenDayLight community. It is a dynamic project with a variety of releases, all of which necessitate thorough testing to guarantee high-quality software. To make sure that current features are not impacted, efficient regression testing is necessary.
[8]	Open-Source Platforms Used in a Fully Integrated Software-Defined Networking Testbed	Software-defined networking, network function virtualization (NFV), cloud computing, multi-access edge computing (MEC), and network slicing are just a few of the networking concepts developed to improve the performance, portability, scalability, and energy efficiency of networks in the Internet of Things era. SDN separates the network's control plane and data plane to overcome the limitations of conventional networking technologies such static setup, lack of scalability, and inefficiency.

### 3. Proposed Methodology

Every experimentation requires a detailed methodology which illustrates the entire step-by-step procedure to execute the experimentation.

#### 3.1. Emulation Tool

For implementing the topology in the host user, we have made use of an emulation tool known as Mininet. This tool is a powerful tool for creating realistic virtual networks with just simple commands. We can easily connect with the network through its CLI mode. It offers various network topologies which can be used to perform various network experimentation and testing. In our work, we have made use of custom topology with 100 hosts and 60 switches. Mininet also extends its support for multiple versions of OpenFlow protocol. We have made use of OpenFlow version 1.3 to carry out this experimentation.

#### 3.2. Multiple Machines and Distributed Environment

We have considered four different machines situated in varied environments. Machine-1 is a leading SDN controller (ONOS) incorporated in AWS at Chicago.

Machine-2 is an ONOS follower incorporated in AWS at Mumbai. The distributed environment to these different machines will lead to better results in terms of traffic generation and handling huge spikes as well. The fourth one is the Mininet machine, and it is considered as the most important machine because it is the one which is responsible for handling the custom network topology with a corresponding number of switches and hosts. The detailed information regarding these machines and distributed environment is provided in Table 2 and Figure 1.

**Table 2.** A variety of machine specifications.

Name of the Machine	IP Addresses	Specifications
Controller-1 (ONOS Leader (V))	192.142.99.99	Controller installed in the AWS cloud at Chicago
Controller-2 (ONOS follower)	192.35.14.253	Controller installed in the AWS cloud at Mumbai
Controller-3 (ODL follower)	192.73.75.70	Controller installed in the AWS cloud at Europe
Mininet (Virtual Machine)	192.53.105.6	Mininet is installed in a Personnel Computer (PC)

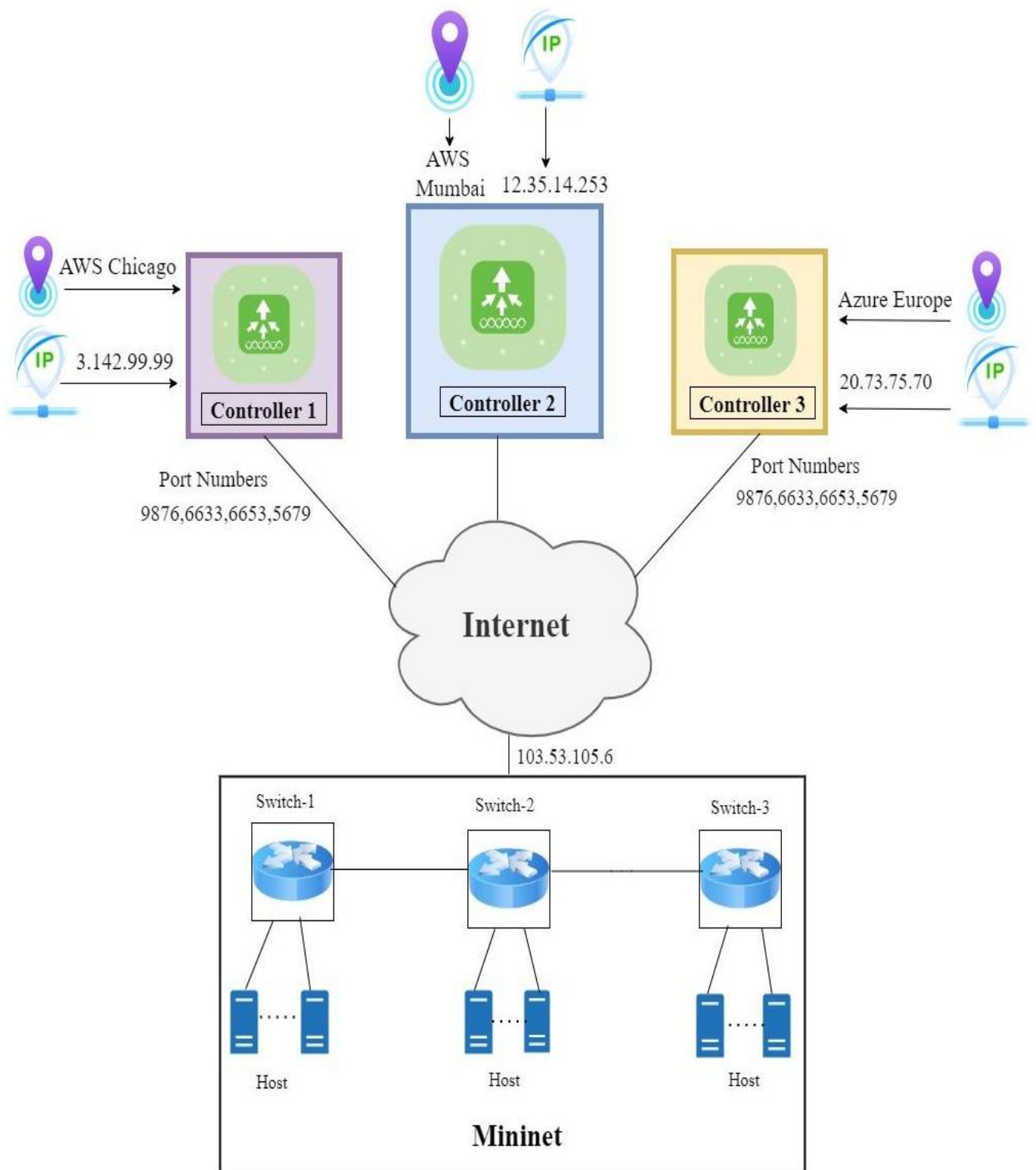
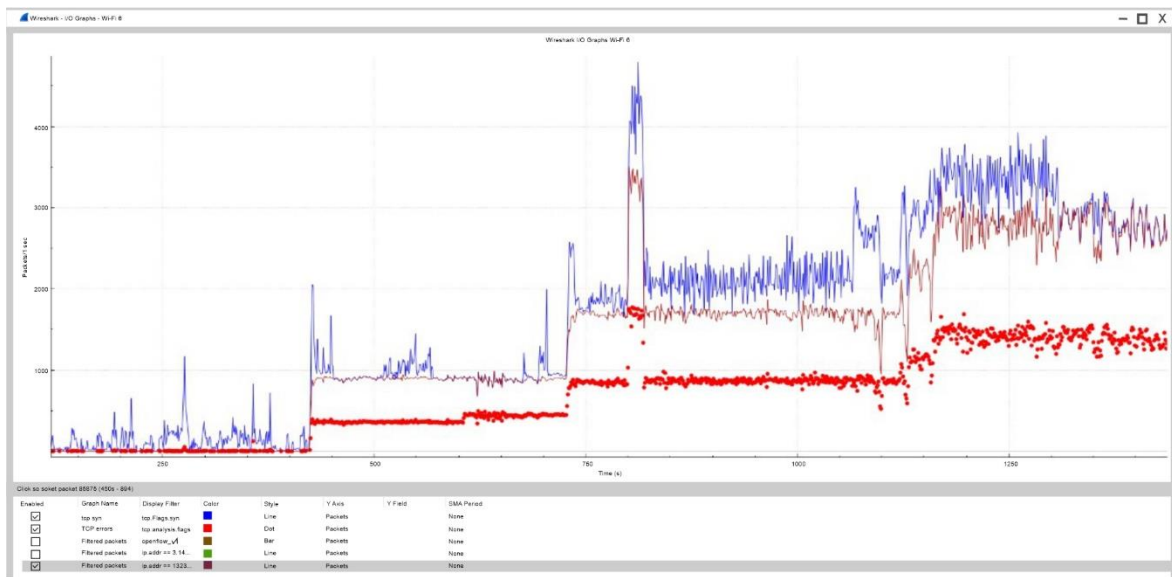


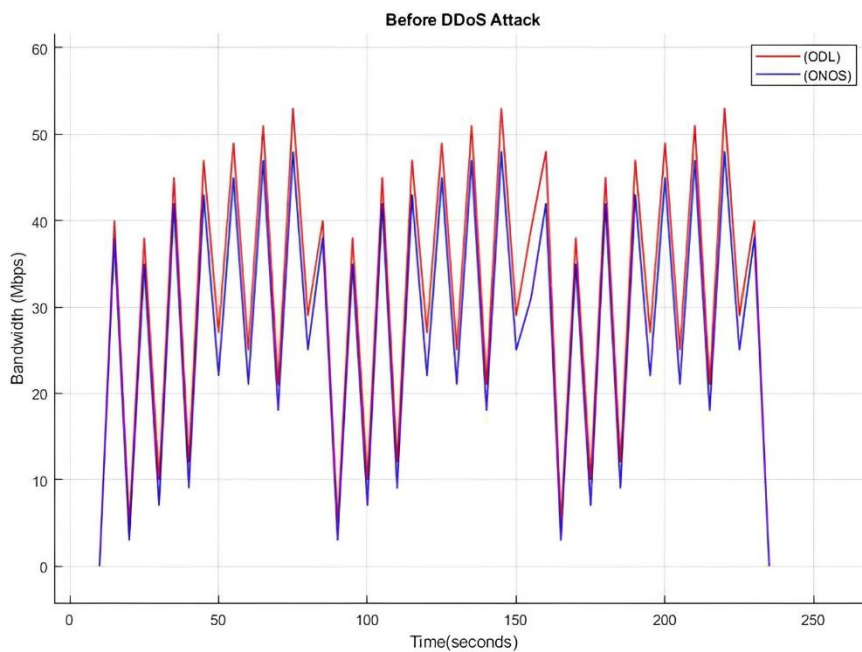
Figure 1. Experimental set.

#### 4. Results of the Experimentation

In this section, results drawn from various experiments are depicted. The parameters that we have considered for comparison and evaluation are CPU, memory, and disk utilization. In Figure 2a,b, the scenario before and after DDoS happened over the emulated environment is depicted. The traffic flow was normal and can be easily observed.



(a)



(b)

**Figure 2.** (a) Experimental Analysis Scenario before the occurrence of DDoS attacks; (b) difference in flow of network traffic once DDoS attacks is launched.

#### 4.1. Scenario before DDoS Attack

When enough devices are compromised, the hacker gives them authorizations to assault; each system begins flooding the target server or network with requests, overwhelming it and creating inefficiency or inability. DDoS attacks take several forms, including volume-based, protocol-based, and application-layer attacks. ‘Even if you have previously been attacked, you may face additional attacks. DDoS attacks are analogous to home invasions. Any unprotected website or web application is vulnerable, and it happens regularly. As a result, if you have had a DDoS attack and only treated the symptoms rather than the underlying flaws and gaps, you are effectively leaving your digital assets open to future DDoS attacks [16–21].



#### 4.2. An Attack Could Be Launched against Any Organization

Whether you are a little business or a major organization, and whether you run a static website, a simple blog, or a popular e-commerce site, you are a possible attack target. Smaller firms and simple websites and web apps may put forth less effort to create powerful DDoS defenses, making them easy targets. DDoS attacks are common and frequently change; they have increased by about 20% over the previous two years. The impact and magnitude of these attacks have increased by around 200 percent as of late. In the first month of 2019, there were roughly as many DDoS attacks as there were in all of 2018. DDoS attacks occur much more frequently than you may think and are always evolving in terms of both form and nature as hackers and cybercriminals develop new techniques for staging DDoS attacks.

#### 4.3. Traffic Range for ODL and ONOS

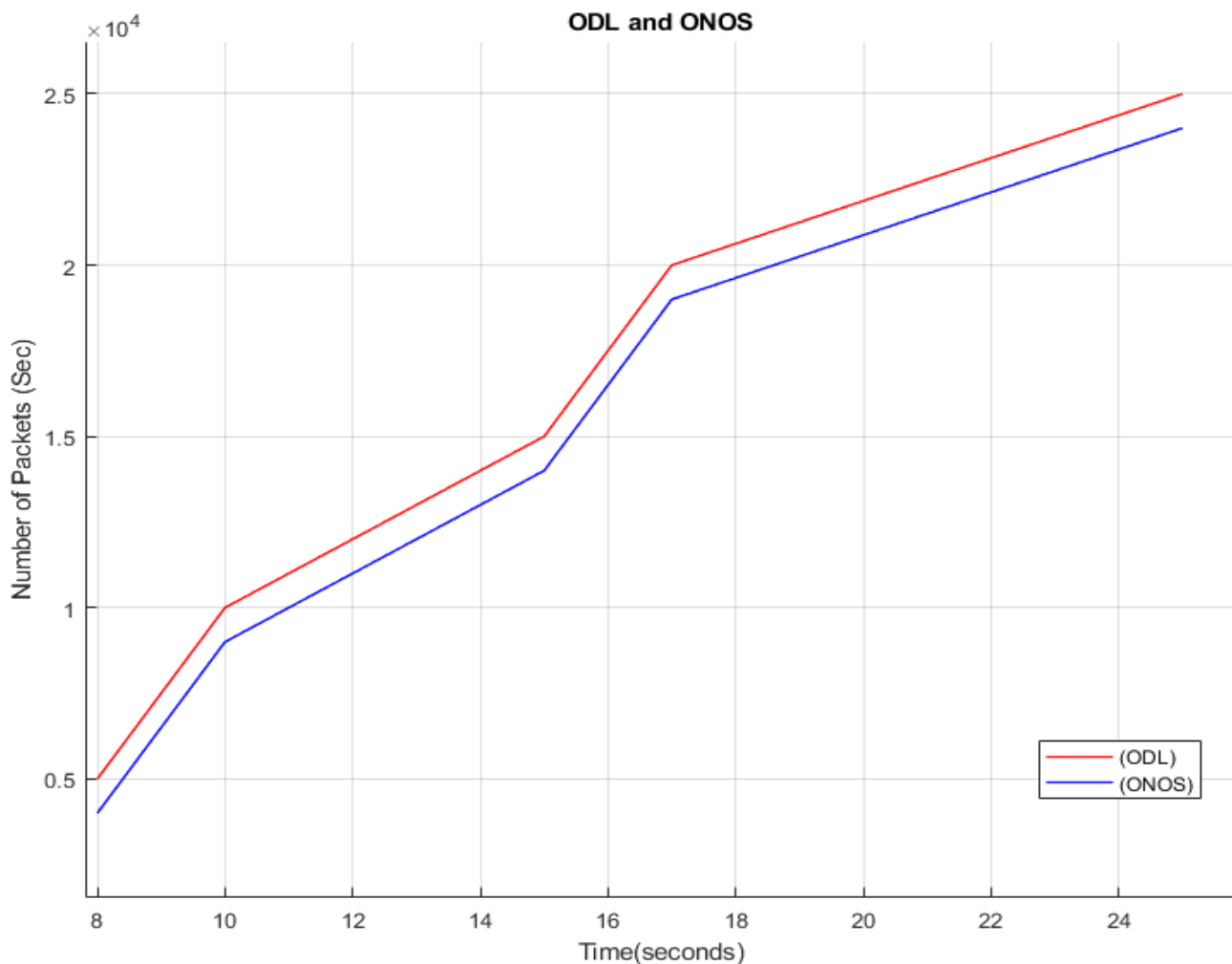
As already stated, businesses can choose from a range of controllers, but two of the most well-known are ODL and ONOS. In Figure 3, the normal traffic analysis is depicted for both of the popular controllers. The time for which the experiment was implemented was around 24 s, and it was found that ODL has transferred the maximum number of packets as compared to ONOS. Although they both are linked to the Linux Foundation, there are several differences between the two that consumers should be aware of. However, before delving deeper into ONOS and ODL, we must first consider the other market participants. Significant rivals ONOS and ODL should be evaluated head-to-head—Corporate vs. Academic—ONOS vs. ODL—Cloud provider vs. Carrier-grade networks. Both ONOS and ODL are built in Java and created for modular use with a flexible infrastructure, and each ONOS partner is also a member of ODL. Aside from that, they are distinguished by a few key qualities. The Eclipse Public License is used by ODL; however, the Apache 2.0 license is used by ONOS. ONOS is more focused on service providers and cloud providers. ONOS is composed of a complex web of interconnected subsystems and scalable telecom system activities, some of which are actively in development. ODL, on the other hand, is based on a strong central abstraction layer and a model-view-control platform.

#### 4.4. Customers to Target

According to the ONF, both ONOS and ODL have hybrid commercial strategies, albeit there are differences in which business attracts which providers. Even though the Whole Stack associated ONOS with telecommunications, it claimed that ODL was primarily concerned with data centers. ODL connects Next Generation Networks (NGN) (OpenFlow and SDN) and Legacy (BGP, SNMP, and so on). Since ONOS places a greater emphasis on performance characteristics and clustering to improve availability and scalability, carriers are frequently more interested in it. ONOS concentrates on carrier-grade networks as a result, and telecommunications companies are included into their initiatives. In comparison with ONOS, ODL has more vendors, such as NES, Cisco, and Juniper.

#### 4.5. CPU Utilization

Figure 4 analyses the effect of the CPU use metric. DDoS assaults, a well publicised phenomenon, pose one of the greatest challenges to the security of modern Internet infrastructure and services. All too often, denial-of-service attacks have a negative impact on and are themselves a target of VOIP services, DNS servers, online games, and e-commerce applications. Online application assaults and denial of service are becoming more common in distributed architecture. While active, the denial-of-service assault lessens the strain on web servers' processing power. Therefore, it is necessary to reduce CPU burden once an attack has been effectively detected.

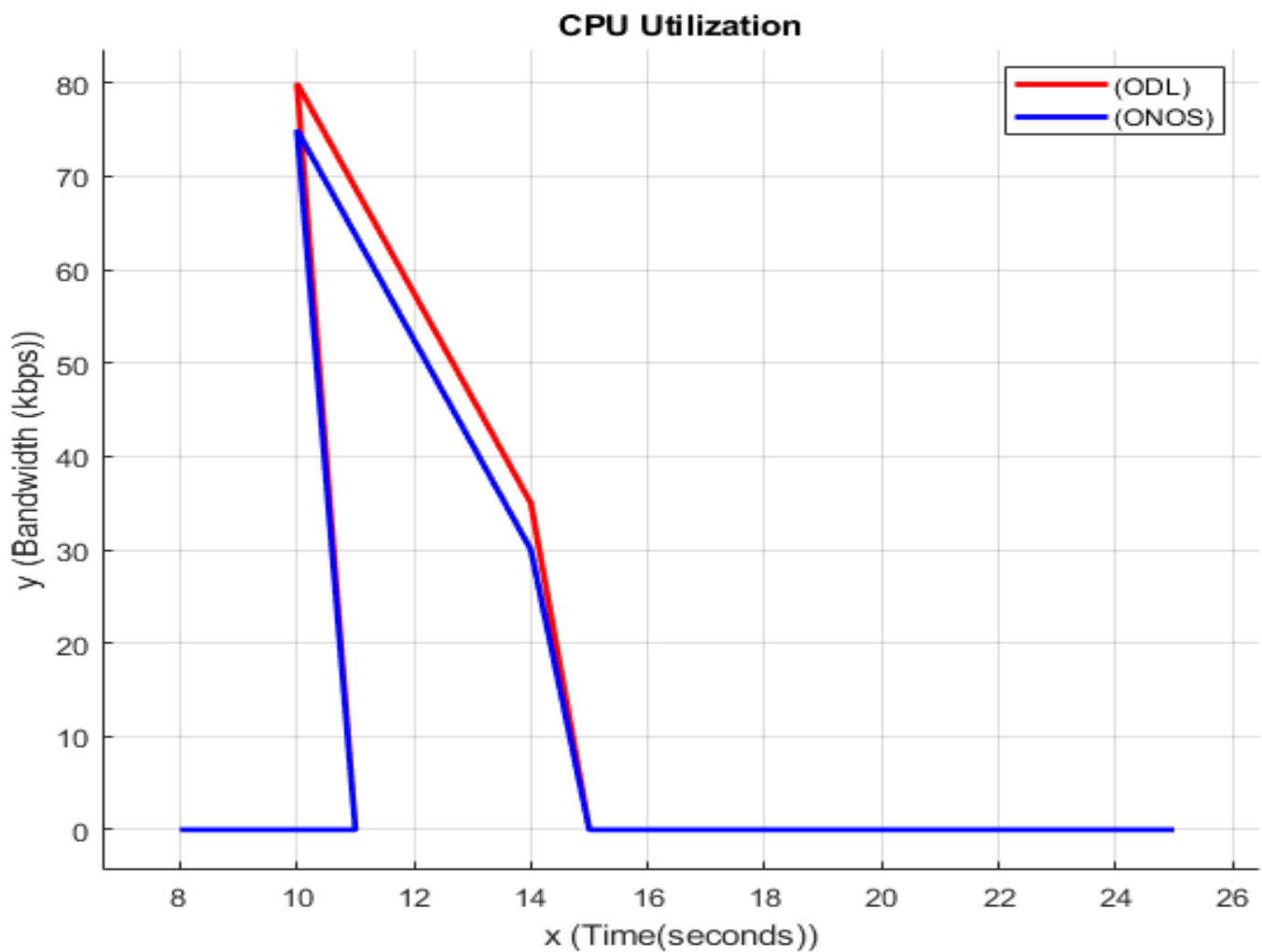


**Figure 3.** ODL and ONOS.

The primary purpose is to reduce the CPU's workload after a SYN Flood attack:

- Permits authorized users to access resources and services after identifying a denial-of-service attack, such as a TCP;
- Lowers and SYN Flood CPU burden;
- To successfully detect denial-of-service attacks, the method employs a threshold and abuse detection strategy. TCP SYN flood attacks are created using the Linux hping tool. The majority of the malicious TCP packets are generated by the hping tool for web servers.

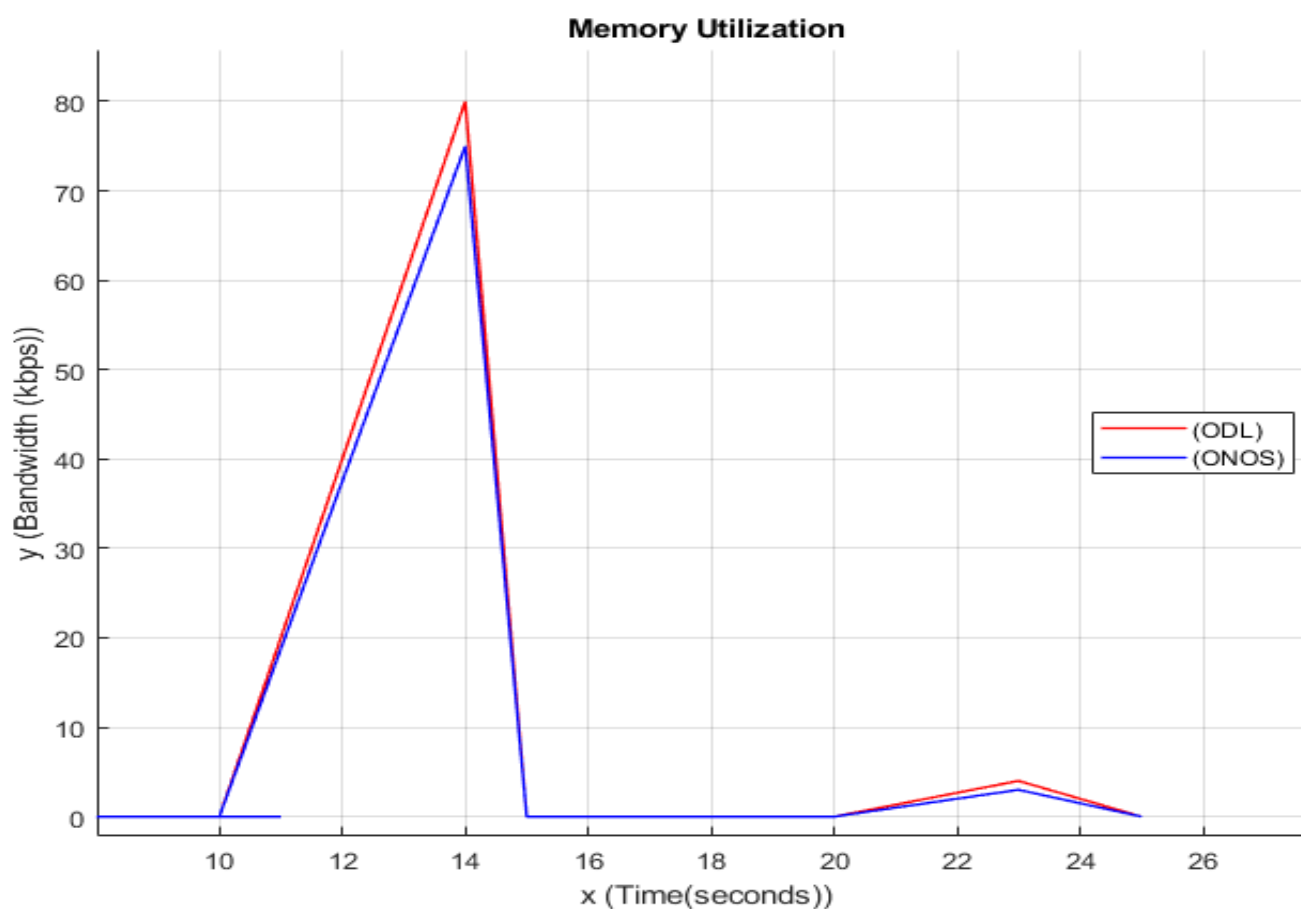
CPU use before, during, and after an attack has been detected is compared to gauge the system's efficacy. The processor uses before, during, and after DoS attacks, such as TCP SYN flood detection. As the numbers demonstrate, CPU use spikes during a TCP SYN Flood assault and then drops down sharply after the attack is identified. Unfortunately, this method can only identify denial-of-service assaults like SYN flooding. The next step is to subject the system to a number of DOS simulations. It was discovered via testing that CPU use was lower than ODL by a significant margin.



**Figure 4.** CPU utilization.

#### 4.6. Memory Utilization

While DDoS attack is being performed on a server, all the resources of that server will be used to its maximum limit like CPU Utilization memory will also get used to the max limit; as you can see in the above graph at a particular time interval, the memory is being used to its max limit, which indicated that the memory is also being affected by the DDoS attack. In the above Figure 5, the  $y$ -axis indicates the Bandwidth in Kbps, and  $x$ -axis indicates Time which is in seconds, and the lines represent both ODL and ONOS value in the time that a certain amount of memory is being utilized. A hardware memory subsystem has multiple levels of storage-based resources and is hierarchical. Numerous sites connect those that deal with scheduling. Memory attacks can use both scheduling- and storage-based contention. An adversary can transfer data from the victim from upper-level storage-based memory resources to lower-level memory resources in the case of a storage-based dispute. In Figure 5 as a result, the victim will have to wait longer for the data to reach the processor core. An attacker can lessen the likelihood that the scheduler would choose the victim's memory requests at a given memory level by tricking it into assigning the attacker's requests a higher priority or by constantly attacking the scheduler with many requests at once in scheduling-based contention.



**Figure 5.** Memory utilization.

#### 4.7. Disk Utilization

While DDoS attack is being performed on a server, all the resources of that server will be used to their maximum limit, like the CPU Utilization Disk, as shown in Figure 6, which will also get used to its maximum limit; as you can see in the above graph, at a particular time interval when the DDoS attack is being performed, the disk is being utilized, which indicated that the disk is also being affected by the DDoS attack. The  $y$ -axis in the graph above indicates the Bandwidth in Kbps, and the  $x$ -axis indicates Time, which is in seconds, and the lines represent both ODL and ONOS values in the time that a certain amount of memory is being utilized. Because of their enhanced dependability, storage capacity, fault tolerance, and other qualities, hard disc drives (HDDs) have grown to be the most popular kind of nonvolatile storage. Because of technological breakthroughs and the constant need to store massive amounts of data, modern computing systems rely largely on HDDs. The detailed scenarios and comparisons are illustrated in Table 3.

Indeed, the researchers discovered that hard disc drives (HDDs) are now an integral component of many commonplace systems, including personal computers, cloud servers, medical bedside monitors, closed-circuit television (CCTV) systems, and automated teller machines (ATMs). As a result, an efficient and easy DDoS attack against HDDs could cause major practical issues for people and organizations. In Table 3, summarization of all the parameters along with the comparison of both controllers is depicted. It can be clearly observed that ODL went down after ONOS in every scenario. ODL has the maximum utilization for all three parameters (Disk, Memory, and CPU). TCP SYN and HTTP type of traffic were used for every scenario.

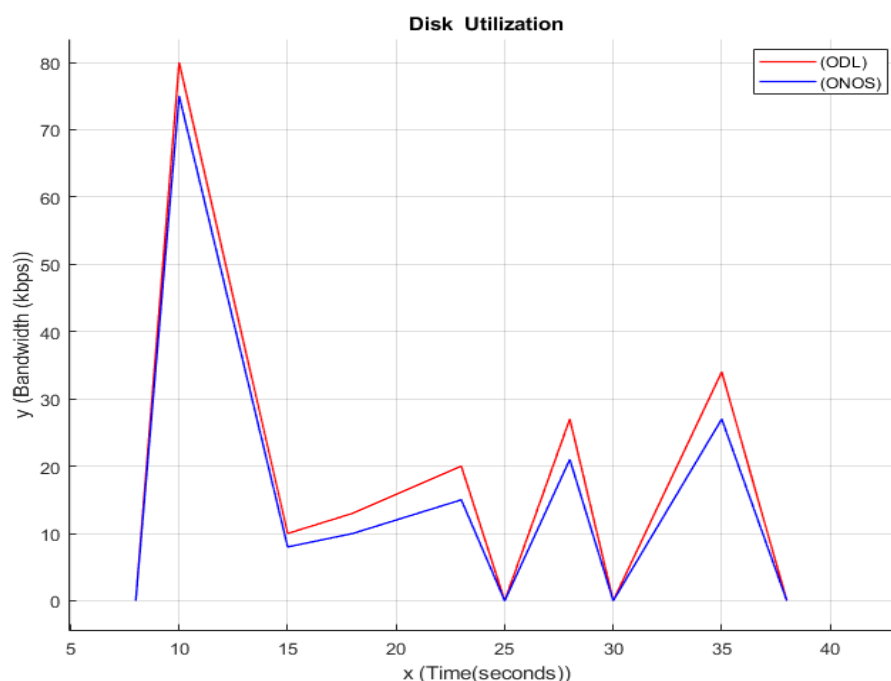


Figure 6. Disk utilization.

Table 3. Comparison of different parameters and scenarios.

Parameters' Scenarios	Controller	No. of Packets/Secs	The Amount of Time in Seconds When the Leader Was Taken Down	Type of Network Traffic	Memory Utilization %	CPU Utilization %	Disk Utilization %
I	ODL	5000	25	TCP SYN and HTTP	96	98	95
	ONOS	5000	23	TCP SYN and HTTP	90	91	92
II	ODL	10,000	20	TCP SYN	84	89	90
	ONOS	10,000	17	TCP SYN	80	84	79
III	ODL	15,000	18	HTTP	72	75	82
	ONOS	15,000	15	HTTP	67	69	63
IV	ODL	20,000	14	TCP SYN	59	70	73
	ONOS	20,000	10	TCP SYN	50	62	54
V	ODL	25,000	11	TCP SYN and HTTP	43	57	46
	ONOS	25,000	08	TCP SYN and HTTP	38	46	35

### 5. Conclusions and Future Scope

Software Defined Networking is a relatively new networking architecture that has become the most widely discussed networking technology in recent years and the latest development in developing digital networks, which aims to break down the traditional connection in the middle of the control surface and the infrastructure surface. This separation aims to make resources more manageable, secure, and controllable. As a result, many controllers such as Beacon, Floodlight, Ryu, ODL, ONOS, NOX, and Pox have been developed. The selection of the finest-fit controller has evolved into an application-specific tool operation due to the extensive range of SDN applications and controllers. In this paper, we have selected two popular SDN controllers for the point of the study (ODL and ONOS). For experimentation, four different machines with varied configurations were considered. ODL-3 node cluster controller and ONOS controller were bombarded under different scenarios. Open source tools were used to generate malicious traffic of DDoS and bombarded the targeted controller. Both SDN controllers were found to be vulnerable to

DDoS attacks, and under different scenarios, it was observed that the ODL-3 node cluster outperforms in terms of disk, memory, and CPU utilization. The time when controllers went down was also high in the case of the ODL-3node cluster controller. We were limited to parameters for comparison. In further study, more parameters and scenarios (more traffic generation) could be considered for future study of points.

**Author Contributions:** Conceptualization, S.B. (Sumit Badotra), S.B. (Salil Bharany), S.T., A.U.R., E.T.E., N.A.G. and M.S.; methodology, S.B. (Sumit Badotra), S.B. (Salil Bharany), S.T., A.U.R., E.T.E., N.A.G. and M.S.; software, S.B. (Sumit Badotra) and S.T.; validation, S.B. (Sumit Badotra) and S.B. (Salil Bharany); formal analysis, S.T., A.U.R. and M.S.; investigation, S.B. (Sumit Badotra) and S.B. (Salil Bharany); resources, M.S. and E.T.E.; data curation, S.B. (Sumit Badotra) and S.B. (Salil Bharany); writing—original draft preparation, S.B. (Sumit Badotra), S.B. (Salil Bharany), S.T., A.U.R., E.T.E., N.A.G. and M.S.; writing—review and editing, S.B. (Sumit Badotra), S.B. (Salil Bharany), S.T., A.U.R., E.T.E., N.A.G. and M.S.; visualization, S.B. (Sumit Badotra) and S.T.; supervision, S.T. and M.S.; project administration, S.B. (Salil Bharany), A.U.R. and E.T.E.; funding acquisition, E.T.E. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was supported by the Future University Researchers Supporting Project No. FUESP-2020/48 at Future University in Egypt, New Cairo 11845, Egypt.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Casado, M.; Koponen, T.; Shenker, S.; Tootoonchian, A. Fabric: A retrospective on evolving SDN. In Proceedings of the First Workshop on Hot Topics in Software Defined Networks, Helsinki, Finland, 13 August 2012; Association for Computing Machinery: New York, NY, USA, 2012; pp. 85–90.
2. Gelberger, A.; Yemini, N.; Giladi, R. Performance analysis of software-defined networking (SDN). In Proceedings of the 2013 IEEE 21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems, San Francisco, CA, USA, 14–16 August 2013; IEEE: Piscataway, NJ, USA; pp. 389–393.
3. Khondoker, R.; Zaalouk, A.; Marx, R.; Bayarou, K. Feature-based comparison and selection of Software Defined Networking (SDN) controllers. In Proceedings of the 2014 World Congress on Computer Applications and Information Systems (WCCAIS), Hammamet, Tunisia, 17–19 January 2014; IEEE: Piscataway, NJ, USA; pp. 1–7.
4. Baktir, A.C.; Ozgovde, A.; Ersoy, C. How Can Edge Computing Benefit From Software-Defined Networking: A Survey, Use Cases, and Future Directions. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2359–2391. [[CrossRef](#)]
5. Sun, S.; Gong, L.; Rong, B.; Lu, K. An intelligent SDN framework for 5G heterogeneous networks. *IEEE Commun. Mag.* **2015**, *53*, 142–147. [[CrossRef](#)]
6. Yoon, C.; Park, T.; Lee, S.; Kang, H.; Shin, S.; Zhang, Z. Enabling security functions with SDN: A feasibility study. *Comput. Netw.* **2015**, *85*, 19–35. [[CrossRef](#)]
7. Badotra, S.; Panda, S.N. SNORT based early DDoS detection system using Opendaylight and open networking operating system in software defined networking. *Clust. Comput.* **2021**, *24*, 501–513. [[CrossRef](#)]
8. Basta, A.; Kellerer, W.; Hoffmann, M.; Morper, H.J.; Hoffmann, K. Applying NFV and SDN to LTE mobile core gateways, the functions placement problem. In Proceedings of the 4th Workshop on All Things Cellular: Operations, Applications, Challenges, Chicago, IL, USA, 22 August 2014; Association for Computing Machinery: New York, NY, USA, 2014; pp. 33–38.
9. Xia, W.; Wen, Y.; Foh, C.H.; Niyato, D.; Xie, H. A survey on software-defined networking. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 27–51. [[CrossRef](#)]
10. Aburukba, R.O.; AliKarrar, M.; Landolsi, T.; El-Fakih, K. Scheduling Internet of Things requests to minimize latency in hybrid Fog–Cloud computing. *Future Gener. Comput. Syst.* **2020**, *111*, 539–551. [[CrossRef](#)]
11. Gupta, A.; Sharma, L.S. Performance Evaluation of Snort and Suricata Intrusion Detection Systems on Ubuntu Server. In Proceedings of the ICRIC 2019, Jammu, India, 8–9 March 2019; Springer: Cham, Switzerland, 2020; pp. 811–821.
12. Shorey, T.; Subbaiah, D.; Goyal, A.; Sakxena, A.; Mishra, A.K. Performance Comparison and Analysis of Slowloris, GoldenEye and Xerxes DDoS Attack Tools. In Proceedings of the 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, 19–22 September 2018; IEEE: Piscataway, NJ, USA; pp. 318–322.
13. Mouradian, C.; Kianpisheh, S.; Abu-Lebdeh, M.; Ebrahimnezhad, F.; Jahromi, N.T.; Glitho, R.H. Application component placement in NFV-based hybrid cloud/fog systems with mobile fog nodes. *IEEE J. Sel. Areas Commun.* **2019**, *37*, 1130–1143. [[CrossRef](#)]
14. Badotra, S.; Panda, S.N. Software-defined networking: A novel approach to networks. In *Handbook of Computer Networks and Cyber Security*; Springer: Cham, Switzerland, 2020; pp. 313–339.
15. Badotra, S.; Panda, S.N.; Datta, P. Detection and Prevention from DDoS Attack Using Software-Defined Security. In *Progress in Advanced Computing and Intelligent Engineering*; Springer: Singapore, 2021; pp. 207–217.

16. Badotra, S.; Nagpal, D.; Panda, S.N.; Tanwar, S.; Bajaj, S. IoT-enabled healthcare network with SDN. In Proceedings of the 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO), Noida, India, 4–5 June 2020; IEEE: Piscataway, NJ, USA; pp. 38–42.
17. Bharany, S.; Kaur, K.; Badotra, S.; Rani, S.; Wozniak, M.; Shafi, J.; Ijaz, M.F. Efficient Middleware for the Portability of PaaS Services Consuming Applications among Heterogeneous Clouds. *Sensors* **2022**, *22*, 5013. [[CrossRef](#)]
18. Badotra, S.; Panda, S.N. Experimental comparison and evaluation of various OpenFlow software defined networking controllers. *Int. J. Appl. Sci. Eng.* **2020**, *17*, 317–324.
19. Bharany, S.; Sharma, S.; Bhatia, S.; Rahmani, M.K.I.; Shuaib, M.; Lashari, S.A. Energy Efficient Clustering Protocol for FANETS Using Moth Flame Optimization. *Sustainability* **2022**, *14*, 6159. [[CrossRef](#)]
20. Betgé-Brezetz, S.; Kamga, G.B.; Tazi, M. Trust support for SDN controllers and virtualized network applications. In Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), London, UK, 13–17 April 2015; IEEE: Piscataway, NJ, USA; pp. 1–5.
21. Isong, B.; Kgogo, T.; Lugayizi, F.; Kankuzi, B. Trust establishment framework between SDN controller and applications. In Proceedings of the 2017 18th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Kanazawa, Japan, 26–28 June 2017; IEEE: Piscataway, NJ, USA; pp. 101–107.