International Journal of Control Theory and Applications

ISSN : 0974-5572

Volume 10 • Number 32 • 2017

Controller Configuration and Security Implementation using ONOS SDN Controller

Vamsi Krishna Patchava[a], Manvi Bhardwaj[b] and Revathi Venkataraman[c]

[a,b]*B.Tech, Final Year, Dept of Computer Science & Engineering, SRM University, Kattankulathur, India*
[c]*Professor, Dept of Computer Science & Engineering, SRM University, Kattankulathur, India*

Abstract: Software Defined Networking (SDN) has paved the way for vendor-neutral controller-centric environment as the abstraction at the lower layers can be performed by any type of controller, irrespective of the programming language they are executed in. Open Network Operating System (ONOS), being a java-based controller, provides a flexible and agile environment at the application level to manage and deploy services. The advancements in ONOS GUI have produced astonishing results with the capability of deploying multiple controllers, more commonly known as clusters. As the packet transmission takes place from the lower level to higher level of the SDN architecture, each switch and node (host) can be individually monitored and traced by a series of traffic patterns and flow rules. The proposed paper aims at enforcing the security functions in ONOS for a reliable data communication. With the help of Eastbound and Westbound interfaces and changes at the OpenFlow protocol level, the flow rules can be confined to the data link layer and also to instill sufficient intelligence to the controller in order to defend the network against common threats in today's network environment such as Distributed Denial of Service (DDoS), Botnets, Malware and IP address spoofing. In this paper, the possibility of several attacks are determined and an adversary model is proposed for the detection and prevention of DDoS attacks at the data plane level.

Keywords: Software Defined Networking (SDN)-Open Network Operating System (ONOS)-clusters-flow rules-OpenFlow-Eastbound-WestBound-Distributed Denial of Service (DDoS)-adversary model.

1. INTRODUCTION

Software-Defined Networking (SDN) is a recent technological trend which is dynamic and much cheaper, making it ideal for high bandwidth applications. This architecture decouples the data plane and the control plane enabling the network to be controlled directly using executional programs with the help of software from a remote location [1].The ONOS (Open Network Operating System) is one of the SDN controllers powered by The Linux Foundation. The aim of this particular project is to create an SDN operating system for different vendors that is designed and modified for achieving scalability, relatively higher performance and high occurrence rate [2]. While ONOS is supported vastly by standard protocols such as OpenFlow, it's architecture is somewhat

indirectly related to them. Coming to the real scenario, ONOS provides its own set of abstractions, models and documentations, which it makes visible to the application programmers. These factors can be deployed during run-time [3].

The apache karaf in the ONOS application can also be accessed in a secure mode, which enforces security policy to every application. Even though this feature allows to authenticate between the controller and the lower-level components of the architecture, there is still a high chance that a malicious node enters into the topology through a possible gateway as the controller is targeted for being the central control point. Due to the OpenFlow Security Analysis conducted as a part of earlier studies, two modeling techniques namely Microsoft's STRIDE methodology and attack trees were proposed, which used the Mininet Framework based on OpenvSwitch, enabled by a POX controller [4]. These techniques have proved to be beneficial for the successful establishment of a secure pathway for the packet transmission from the data plane to the controller. The proposed study brings out a security model for a similar analysis with the help of the Inter-arrival time(IAT) parameter of the packets which is helpful in determining whether the packet is arriving from the correct source and also leaving the decision-making to the ONOS controller.

2. FEW POSSIBLE ATTACKS IN SDN ENVIRONMENT

2.1. Distributed Denial of Service (DDoS)

A Denial of Service attack is any event that diminishes a network's capacity to perform its expected function. These attacks are launched against server resources by preventing authorized users from accessing resources. They pose threats to larger websites such as Amazon and eBay. The effect of these attacks varies from temporarily blocking service availability to permanently distorting information in the network. After gaining access to the network, the attacker may randomize the attention of the system so that he cannot be easily tracked, send invalid data to applications, flood a computer with malicious packets or overuse the resources of the server till exhaustion [5]. The DoS attack uses one computer and one internet connection to flood a targeted resource. Whereas a Distributed DoS attack uses multiple computers and diverse network connections which may be distributed via botnets. A botnet is a group of internet-enabled devices injected with malware and are controlled from a remote location. A bot can be ordered to access a website as a part of the larger DDoS attack [6]. The DDoS attack types fall majorly into the following three categories:

Traffic attacks: These send a huge volume of TCP, UDP and ICPM packets to the target. Due to this, legitimate packets may be lost.

Bandwidth attacks: This attack overloads the target with massive amount of junk data.

Application attacks: Target's system services may be unavailable due to depleted resources at the application layer.

2.2. IP Address Spoofing

IP address spoofing or **IP spoofing** is the creation of IP packets with a false source IP address, for the purpose of hiding the identity of the sender or impersonating another computing system. One technique which a sender may use to maintain anonymity is to use a proxy server. IP spoofing is most frequently used in denial-of-service attacks, where the objective is to flood the target with a massive volume of traffic. Packets with spoofed IP addresses are more difficult to filter since each spoofed packet appears to come from a different address, and they hide the true source of the attack. Denial of service attacks that use spoofing randomly choose addresses from the entire IP address space [7].

2.3. Man-in-the-Middle attack

Supposedly, a conversation is happening between the controller and a switch through an open port, if the conversation is overheard by a third party other than the controller and switch, the information in the form of packets is leaked to an outsider which is a vulnerable threat. It is a type of eavesdropping attack. Man-in-the-middle attackers intercept, send and receive data never meant to be for them without any outside party knowing about the occurrence and even if another party does know about such a happening, it is already too late as the information has been stolen by an external resource or server which does not belong to the existing network topology controlled by an SDN controller [8].

There are several other attacks which are possible in a networking environment such as eavesdropping, data modification, compromised-key attack and other password-based attacks. Since the proposed paper mainly focuses on the occurrence and prevention of DDoS attacks, it is enough to have a basic understanding about the meaning DDoS, botnets, man-in-the-middle scenarios.

3. PROPOSED ARCHITECTURE OF ONOS

Distributed core in the ONOS takes care of clustering. Southbound interface takes care of talking to devices. Applications talk to ONOS using the Northbound interface. ONOS uses a concept called "Intents" for applications to talk to ONOS. Intents are high-level requirements or policies that ONOS translates into flows. An example of an Intent can be a host to host intent where we require one host to talk to another. ONOS calculates the best path and installs flows. When there is any failure in the path between the two hosts, Intent framework in ONOS automatically calculates a new path and installs the flows. In case new path cannot be found, the particular intent goes to failure state. Intents allows the user to not know the lower level details, Intents are equivalent to policies being used by other controllers [9]. This aspect of the ONOS controller can be used to identify a failure path and this indirectly traces the possible occurrences of paths where the DDoS attack is taking place. By calculating the IAT in milliseconds and knowing the right path to reach the controller, DDoS attack can be detected with the help of the OpenFlow tables which are present in each of the nodes.

The network simulator used is ns2 and network animator (NAM) for the detection and prevention of DDoS-effected hosts. Every packet which is being transmitted in the network topology has to pass through the "Intermediate Node" which holds all the flow tables so that they can be accessed from one specific point at any given time. The threshold for IAT is depicted as one millisecond just as the packet generation which is done by Scapy. 50 hosts have been created along with Open Virtual Switches (OVS) induced in a tree topology. Unlike the previous experiments, this attack does not target only one but multiple hosts and three such DDoS-effected hosts are induced and detected in the topology [10].

4. LITERATURE SURVEY

4.1. Resource-Based Attack on Flow Table Limitation in SDN

When it comes to SDN, duplicate switches, often considered as replica, on the data plane forward packets based on the flow rules which are maintained by a central controller. In order to put those rules into implementation, Openv Switches need to write the rules in its flow table. But since the size of the flow table is restricted to a certain space, an issue might arise leading to a performance problem. Adding to that, this type of scalability problem becomes a security flaw related to Distributed Denial of Service (DDoS) attacks, especially the resource attack which exhausts all the content in the flow tables of switches. In the proposed paper, the impact of the resource attack on a software-defined network environment is explored [11]. The resource attack is simulated using SDN with 'Mininet' and' Open Daylight (ODL)', and the effect is analyzed in terms of significant time delay and

bandwidth management. Through thorough evaluation, the importance of maintaining the flow tables taking into consideration their size limitation, is brought out. On top of that, few suggestable solutions were discussed which can address the resource attack and their challenges in the near future to come.

4.2. Premature Detection of DDoS Attacks against SDN Controllers

Even though centralized control is the greatest advantage of SDN, it is also a single point of failure, making it the most devastating disadvantage, if it is made unreachable by a Distributed Denial of Service (DDoS) Attack. To evaluate and overcome this kind of threat, the proposed paper strives to use the central controller of SDN for attack detection and prvention and identifies a solution that is effective when it comes to usage of optimal number of resources such as flow table information in switches or hosts. Additionally, this shows exactly how DDoS attacks can make the controller run out of resources and provides a solution to detect such attacks based on the variation of the destination IP address, which keep fluctuating as the IP addresses which have been used are often dynamic. This technique is able to detect DDoS within the first thousand (1000) packets of the attack traffic [12].

4.3. A DDoS Attack Blocking System using Centralized Traffic Monitoring Control System

With the exponential growth in the number of Internet of Things (IoT) based devices within networked systems, there is need of a means to provide their flexible and secure integration. Software Defined Networking (SDN) is a concept that allows for the centralised control and configuration of network devices, and also provides opportunities for the dynamic control of network traffic. This depicts the use of an SDN gateway as a distributed means of monitoring the traffic originating from and directed to IoT based devices. This gateway can then both detect anomalous behaviour and perform an appropriate response [13].

4.4. Protection Method on SDN using sFlow for SYN Flooding Attacks

In order to strive for high accountability of resources, fault tolerance method is an essential factor for many networks, and SDN is no exception. Things such as link and switch operational failures are generally seen in SDN, affecting the network performance as a whole and might eve lead to a system crash or failure. A recovering method which uses protection for link failures is identified, that is rapid with less requirement for memory. Following the same protection mechanism, the proposed method was able to recover the effects faster without the intervention of the controller by immediately switching to the next traceable safe path which has already been traced when such failure happens. This method is also memory efficient as it sums up the next path based on each link between the hosts or switches instead of each flow, and assigns an ID for every particular link. The simulation results which have been consucted show that this method can recover from link failures within 30 ms, while the restoration needs 58 ms. In heuristic approach, the conveyed method requires around 98 times lesser flow table entries than the protection methods which are based on each flow to store the next traceable paths [14].

5. PROPOSED SYSTEM ARCHITECTURE

A testbed of the SDN-based Attack Prevention architecture is designed and built, which consists of one Controller, one Data Server, one OpenFlow Switch, one Router, one BOTNET, and an Attack Prevention Module to detect DDoS attacks. In this architecture, a packet capture device is connected directly to the OpenFlow switch using a port mirror in order to monitor timestamps of packets going to the Data Server in an observation time. Being able to monitor timestamps results in being able to calculate the Inter-Arrival Time (IAT) parameter that is one of the crucial criteria to detect DDoS [15]. For instance, if the first packet is transmitted after 1 millisecond and

the second packet is transmitted after 3 milliseconds, then the IAT is the difference between the two, which is 2 milliseconds. Using this very parameter, we can detect whether the packet is coming from a malicious node or not.

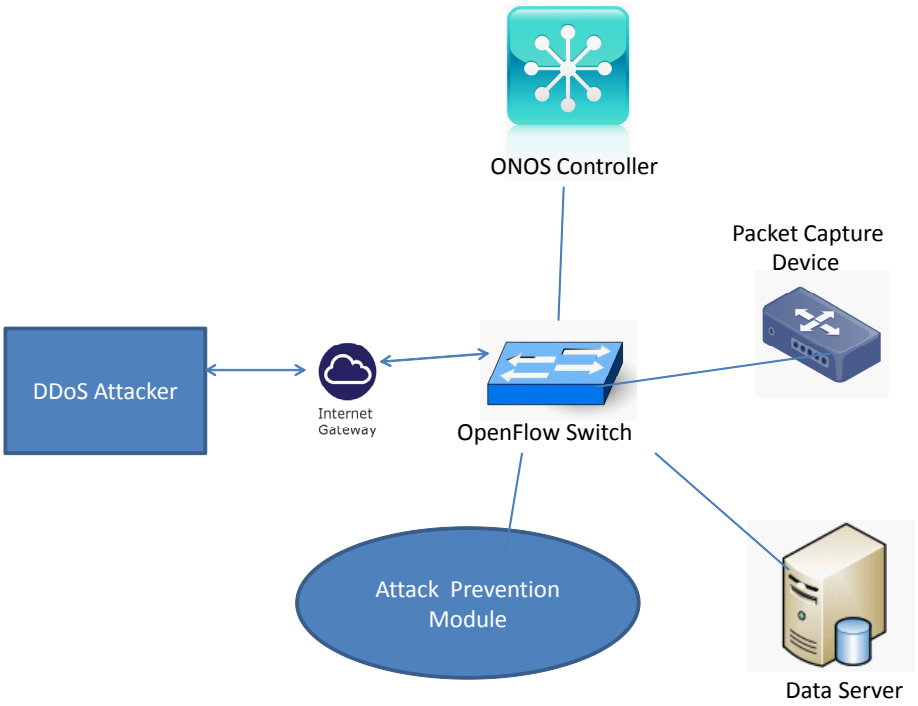

Figure 1

5.1. Purpose of the Attack Prevention Module

The Controller sends request messages to the OpenFlow Switch connected to the Data Server and receives response messages from the switch to collect flow information in flow tables periodically. At the time of Prevention, it extracts flow information from the response messages for calculating the detection parameters.

While the Controller collects flow information from the OpenFlow switch, the packet capture device counts the number of packets passing through it and assigns a timestamp for every packet arriving in a specific period of time. These timestamps are sorted in a formatted list before being sent to the controller. This period of time is equal to the period of time between two send request messages to the switch from the Controller. Secondly, the packet capture device sends the timestamp list to the Attack Prevention Module in the Controller via links [16].

As soon as extracting flow information and receiving the timestamp list are done, this Prevention module calculates the two parameters: the rate of packets having Inter-Arrival Time, the number of flow entries and creates a parameter set. Then, it runs the Attack Prevention Mechanism to make decision whether DDoS occurs or not.

5.2. OpenFlow Switch's Parameter Extraction

All flow entries are stored in flow tables of OpenFlow Switch. The Controller collects flow information by sending an "ofp_flow_stats_request message" and receiving an "ofp_flow_stats message". In the response message, it contains information of individual flows in flow tables of the switch.

To collect information for detecting the DDoS attack, the Controller takes a number of fields of each flow from the body and the match field of the response message as follow:

- *addr_src* - source IP address.
- *addr_dst* - destination IP address.
- *port.tp_src* - source port number.
- *port.tp_dst* - destination port number.
- *packet_count* - number of packets [17].

5.3, Attack Prevention Mechanism

Attack Prevention Mechanism includes setting two rules where attack is ensured to be definitely present and not present, respectively. Based on the traffic analysis above, following parameters come into picture as attributes for attack:

(A1) rate of packets having IAT of smaller or equal to 3 milliseconds, (A2) rate of flows having only one packet per flow, (A3) the number of flow entries to a server within a particular period of time. These are the multiple criteria based on which we can detect possible DDoS attacks in different traffic congestion scenarios. It is important to notice that, since these attributes are monitored for all incoming flows, the system will analyze incoming traffic on the spot and decide whether forwarding 100% of incoming flows through the OpenFlow switch or dropping a portion of all flows, where "part" is rate of (0, 1) in response to detecting a certain degree of Attack risk. Also, the decision of dropping certain percentage of all incoming flows is to reduce traffic, which attacks to an ISP's server in all attack flows. But it may also lead to misdropping normal traffic since that percentage of dropped flows may include flows of both normal traffic and attack traffic. To decrease the possibility of accidentally dropping packets from normal flows, flows of one packet will be dropped at first since risk of 1-packet flows being attack flows is highest.

To detect possible attacks, network administrators may need to establish a very long look-up table, which is a vast domain of various detection and prevention rules. The problem may be much more convoluted when number of inputs increase or when value of inputs vary. And that, in turn, may require a lot of work for a network administrator to afford determining Prevention rules from all combinations of the inputs. In such cases, the existence of an Intermediate node in the topology proves to be beneficial since all the look-up tables of nodes need not be traced [18]. Since every packet flowing to and fro the network has to pass through this intermediate node, the intermediate node uses up the memory in the data server to keep track of all the incomings and outgoings of the packets including the time taken for that packet to arrive a particular node or switch.

Attack Prevention Mechanism Function ()

P1 : packets having inter-arrival time (IAT) upto 3 ms

P2 : packets having one packet per flow.

P3 : number of flow entries directed to the server.

Action: Forwarding all flows or dropping part of the flow.

IF A1 Є [3,10] OR A2 Є [0.9,1] THEN

Action = Dropping 100% of the incoming flows

ELSE IF A1 Є [0.5,3.1] AND A2 Є [0,0.3] THEN

Action = Forwarding all flows

ELSE THEN

Action = DDoS Prevention.

6. GRAPHICAL ANALYSIS

In Figure 2.1, a graph is plotted to show the number of attacking DoS nodes which are detected during simulation time over the total number of nodes present in the network environment. Our analysis has been tested using a total of 20 hosts. This plot shows the comparison between the controller under study, ONOS and Ryu controller [19]. It is crystal clear in the graph depicted that ONOS controller has a greater tendency to detect such malicious nodes that behave as DDoS attackers when compared to Ryu. This is because ONOS consists of Intent framework mechanism. They are specific policies which behave as a stimulating environment and take care of the implementation of various traditional and hybrid network protocols. The graph has been plotted using xgraph feature using the ns2 network simulation tool [20].

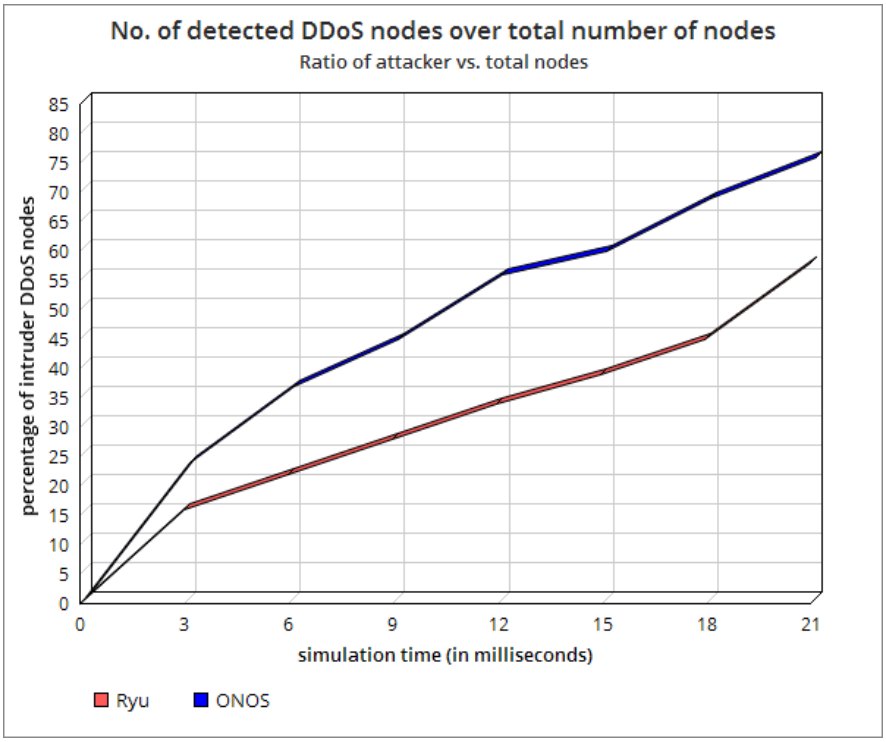

Figure 2.1

Figure 2.2 shows the energy consumption over the detection of the percentage of DDoS intruder nodes. It can be observed that ONOS controller is more efficient in terms of energy consumption in detecting and preventing malicious attacker nodes. This is due to the fact that the nodes stay active only when a packet is passed through them. Rest all the times, they remain inactive. This is a huge benefit for transmitting more packets over the topological network with minimum resources and optimal cost. Also, only the intermediate node which has the responsibility to keep track of all the flow tables of each and every node has the necessity to remain active at all stages of transmission. On the other hand, Ryu controller takes a fairly higher edge when compared to ONOS because of Ryu's inability to remain inactive at considerable situations and due to lack of Intent framework. ONOS has been released at a later stage when compared to Ryu controller, so ONOS obviously has the upper hand in this case.

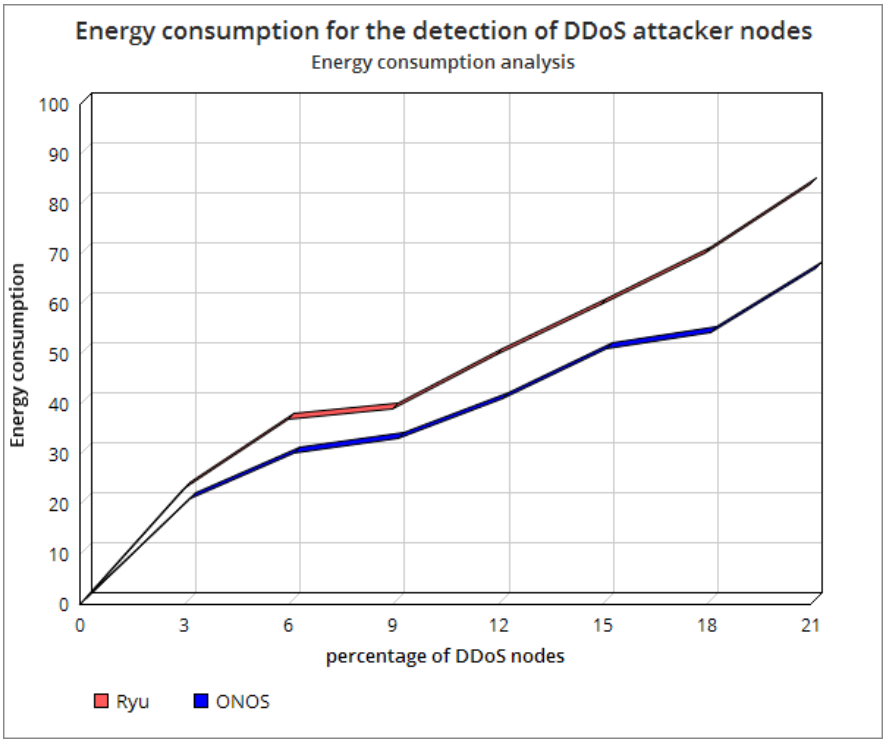

Figure 2.2

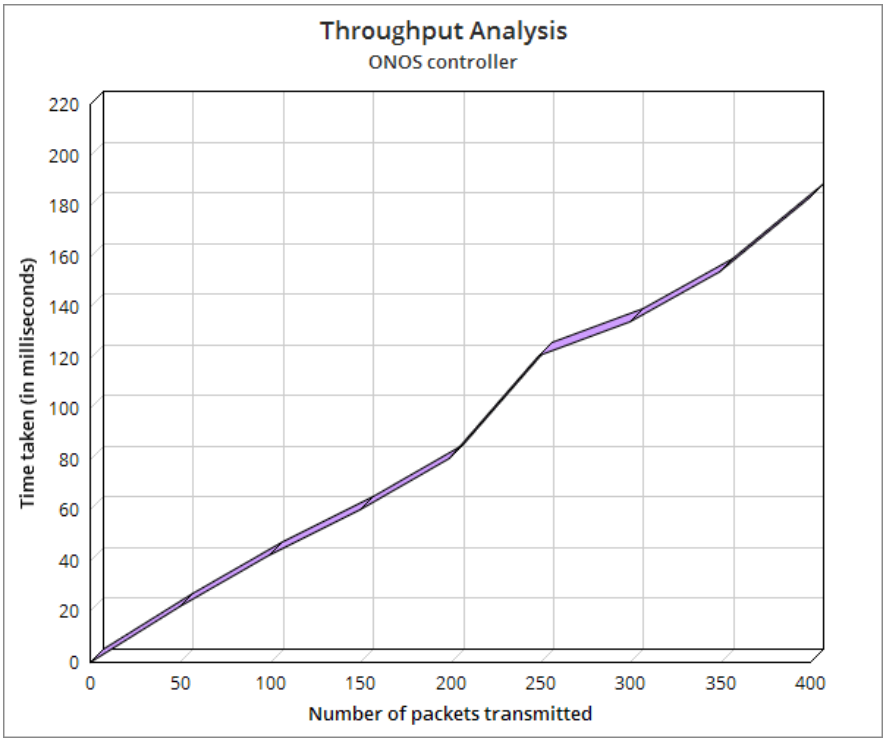

Figure 2.3

Throughput is the number of data packets moved successfully from one host to another [21]. An observation is recorded noting the number of packets that can be transmitted within the specified time, considering that there

is no delay within the network. In Figure 2.3, it can be very well observed that the number of packets transmitted varies constantly with time upto a certain interval. After 80 ms, there is a sudden steep increase in the number of packets. It can be assumed that during this phase, the attacker DDoS nodes have been detected and since these nodes have a higher bandwidth compared to the other hosts, there is a sudden increase in the rate of flow of packets due to more packets being pulled towards these attacker nodes. Once, the prevention mechanism has taken care of the malicious occurences, the plot reaches a steady state increase situation just like how it used to be.

7. CONCLUSION

The main contribution of this paper is a statistical analysis of real network traffic characteristics under both normal and attack situations. Based on this analysis, the criteria and possibility of having a DDoS attack can be efficiently determined with threshold values and high probability. The malicious occurrences of bots can be traced at the data plane layer of the SDN architecture using Inter-arrival time as a criterion for attack detection. By taking into consideration all the flow paths and the OpenFlow table information, it can be deduced that the path through which IAT is more, is retraced back to another path and the host which possesses an abnormal table configuration can be said to be acting as a malicious node. Once, such a node has been detected, it can be labeled as a DoS node. Since every packet has to pass through the Intermediate node, the intermediate node detects these packets coming from DoS nodes and drops them and labels them as malicious occurrences in the data server. This is helpful in order to not let in similar packets from such unauthorized hosts again. The proposed paper also suggests a suitable scenario to prevent such DDoS attack from happening.

8. FUTURE WORKS

SDN is an emerging technological standard and many testbeds have been previously proposed to show the emergence of malicious occurrences in SDN cloud environment. However, with the help of much more effective data centers, network paradigms, proper authentication and authorization, these occurrences are being narrowed down on a vast scale. Security cannot be treated as a single entity and there are many forms of viruses, malicious occurrences, data leaks and plagiarisms which will occur in the near future. This paper shows the possibility of one such major arrack scenario, which is DDoS. The proposed work can be enhanced by taking into consideration all such parameters which can be proven true in the upcoming versions of ONOS architecture. With every new version release of ONOS, the security patches also change. Since SDN has a common three-tiered architecture, the same algorithm can be modified and made possible to be implemented in other SDN controllers such as OpenDayLight (ODL), FloodLight, RoseMary, Ryu, etc.

REFERENCES

[1] https://www.opennetworking.org/sdn-resources/sdn-definition

[2] http://onosproject.org/

[3] https://en.wikipedia.org/wiki/ONOS

[4] "OpenFlow: A Security Analysis" by Rowan Kloti, Vasileios Kotronis, Paul Smith.

[5] https://technet.microsoft.com/en-us/library/cc959354.aspx

[6] https://www.incapsula.com/ddos/ddos-attacks/botnet-ddos.html

[7] http://searchsecurity.techtarget.com/definition/IP-spoofing

[8] "A Man-in-the-Middle attack against OpenDayLight SDN controller" by _Michael Brooks, Baijian Yang.

[9] https://wiki.onosproject.org/display/ONOS/Intent+Framework

[10] "Early detection of DDoS attacks against SDN controllers" Authors : Seyed Mohammad Mousavi, Marc St-Hilare, 2016.

[11] "Assessing the impact of resource attack in Software Defined Networking" by Hiep T.Nguyen Tri, kyungbaek kim.

[12] "SECURITY ANALYSIS OF ONOS SOFTWARE-DEFINED NETWORK PLATFORM" by OLUWADAMILOLA ADENUGA-TAIWO, SHAHRAM SHAH HEYDARI.

[13] "FlowFence: A Denial of Service Defense System for Software Defined Networking" by Andres Felipe Murillo Piedrahita and Sandra Rueda, Diogo M. F. Mattos and Otto Carlos M. B. Duarte.

[14] "Time-based DDoS Detection and Mitigation for SDN Controller" by I Gde Dharma N., M. Fiqri Muthohar, Alvin Prayuda J. D., Priagung K., Deokjai Choi.

[15] "ANALYZING PACKET INTERARRIVAL TIMES DISTRIBUTION TO DETECT NETWORK BOTTLENECKS" by Pal Varga.

[16] http://www.cisco.com/c/en/us/about/press/internet-protocol-journal/back-issues/table-contents-59/161-sdn.html

[17] "A Multi-Criteria-based DDoS-Attack Prevention Solution using Software Defined Networking" by Phan Van Trung, Truong Thu Huong, Dang Van Tuyen, Duong Minh Duc, Alan Marshall.

[18] "Malicious Packet Dropping: How It Might Impact the TCP Performance and How We Can Detect It" by Xiaobing Zhang S. Felix Wu Zhi Fu Tsung-Li Wu.

[19] "Security analysis of OpenDaylight, ONOS, Rosemary and Ryu SDN controllers" by Ramachandra Kamath Arbettu ; Rahamatullah Khondoker ; Kpatcha Bayarou ; Frank Weber.

[20] http://www.nsnam.com/2011/08/xgraph.html

[21] http://searchnetworking.techtarget.com/definition/throughput