DDoS Detection and Mitigation in SDN

Aastha Sanghi (2016H1120159P)

## Abstract

Software Defined Networking(SDN) is considered as future of networking as it is solutions to many frequently faced problem in adhoc networking. Though it promises more secure networking but still there are few security challenges which remain intact or emerges due to decoupling of data plane and control plane such as man-in-middle,DOS etc. One of the major security concern is DOS attack. DOS or denial of service attack can be carried out using large number of UDP ,ICMP, HTTP request packet etc. We have adopted two different prevalent methodologies to mitigate the attack. First, an early detection of attack using flow rate inspection and limiting the bandwidth to secure controller. Second, detection of packets which are spoofed using machine learning techniques based on the feature extracted out of the flows of open switches .We first define accepted architecture of SDN and then both the techniques are discussed in detail in the paper. We have considered OPenFlow as the de facto protocol for communication.Experimental evaluation is done to justify our results. Finally our work has been concluded with future scope.

## Keyword

Software Defined Network(SDN), OpenFlow, Denial of Service(DOS),Security in SDN,Machine Learning, Early Detection

## 1.Introduction

Current prevalent computer networks are facing different problems such as enormous in size, highly heterogeneous and complex to manage.To overcome these problems and to increase the capability of current networks, a novel solution has been introduced that is Software Defined Networks(SDN). In SDN the fundamental idea deals with decoupling of control plane to data plane. Data plane consists switches or routers, which forwards packets whereas control plane consist controller which act as a platform for various network applications. The main motive of SDN lies in logically centralized control with decision making ability based on the global network view [1].

On a positive view, SDN increases the security of the network with the centralized control by resolving conflicts at a global level but the separation of the control and data plane has open different challenges to be taken care such as man-in-the middle attacks, denial of

service (DoS) attacks, and saturation attacks etc. SDN allows network security techniques to be implemented as applications in application plane thus enabling security automation and run time deployment of security techniques [2].

Our main concentration would be resolving flood based attack, that is denial of service(DOS) attack. It is generated by a malicious attacker which will generate a huge amount of fake packets. These packets are concentrated to flood or overload the controller in SDN. As controller is considered as the brain of SDN thus, once it malfunctions the whole network segment can be controlled by the malicious attacker using different methodologies [3].

A DOS attack detection can be achieved by keeping tracks on the flows of switches. If in a flow, attack is detected, different measures can be taken to stop the controller from flooding. But the main challenge is to identify the attacker. IP spoofing makes it more difficult to identify the attacker.

Another challenge is to keep a track on huge amount of packet generated at time of flows [3]. In this paper we have considered RYU controller based network which functions on OpenFlow protocol for communication. Therefore whole network is designed over OpenFlow switches [4].

In below sections we aim at providing a glance on SDN architecture and its security challenges, and further state a solution to deal with DDOS attack problem in SDN networks. Thus in section 2 work done on SDN, security in SDN and Dos attack is discussed. Section 3 proposed system design is discussed with early detection and malicious packet detection using machine learning techniques are discussed. Following section consist our experimental setup and conclusion retrieved from our work.

## 2. Background Study

### 2.1 Overview of SDN

SDN decouples network control and data forwarding layer, and thus further divided into three main layers that are application plane, control plane and data plane [1].The distribution of functionalities in SDN among these planes are described below:

*Application Plane:*

All the network application and services are comprised under this plane such as network management, policy implementation and security services. Applications can request the controller for network stats, status and also for resources usage in different ways. All services need applications coupling with hardware layer of the SDN like topology and link discovery, fire-wall services, domain name services, network address translation services and deploying virtual private networks.

The network information can be provided to applications through the north-bound API of the controller. Hence, OpenFlow is a natural choice to implement network functions in the form of OpenFlow applications. Therefore, a variety of network security services are implemented on top of the OpenFlow controller as security applications [2].

*Control Plane:*

It is basically the Network Operating system that supports the applications of application plane and provide hardware abstraction to these applications [1]. It acts as central logical unit that maintain the global view of the network. The NOS accumulates information using south-bound API for the global view of network. It acts as a perfect interface for coupling applications and hardware to manage the packet flow. In openflow controller provides a determined protocol to communicate with underlying switches and routers [2].

*Data Plane:*

It consists of the hardware components which are used for forwarding the packets based on the instructions (flow rules) from the control plane. Forwarding devices such as routers , switches, virtual switches and access points can be configured for different purposes [2]. Usually hardware components are Openflow switches that consist of flow tables which are used for packet lookup and forwarding. In OpenFlow, the controller comprehends the network complexity, gathers network information through the south-bound API and maintains a logical map of the entire network [3].



Figure 1 SDN architecture

Figure 1. gives a layered view of SDN. Data flow among above planes happens on standard procedure i.e Openflow and also defines data updation procedure in the flow table present in the forwarding switches. Other important terms for defining SDN architecture and working mechanism are mentioned as below:

*Openflow:* It is a standard protocol that determines to add and modify the entry flows of the packet in the switches.Thus the switches in SDN are known as Openflow switches. Openflow switches consist flow table or group table. The communication between switches and the controller happens on the basis of openflow protocol. It also lays the procedure for creating , modifying and removing entries in the flow table [5].

*Flow table:* Each packet has header fields which are used to match the incoming packet with the entry in the flow table in the switch [5]. Both packet header field as well as priority identifies a packet entry in the flow table. Figure 2. [5] shows the fields present in the flow table entry.

| Packet header  Match fields | Priority | Counters | Cookies | Timeout | Instructions |
| --- | --- | --- | --- | --- | --- |

Figure 2. Flow Table entry

### 2.2 Security threats in Control layer

*Unauthorized Access:*

The key idea of SDN is centralized or localized control. With emerging technologies, the implementation of the SDN is distributed to match the high demands. In a distributed architecture, many application can access the control plane and multiple controllers can access the data plane. If an attacker breaches the privacy of the control layer, then the security of network operating system and many other resources are in threat.

*Data Leakage & Modification:*

A diversified actions are mentioned in the OpenFlow switch specification [5] for packet handling. These include add, delete and forward rules to controller. An attacker can analyze the flow stats of packets and create an attack by changing the attributes of the flow stats. For example, the time to process packet via controller will be more than the packet transfer between output port and input port. If the attacker can identify this pattern then he can craft other similar packets and create fake requests and flows in the flow table. This leads to the Denial of Service attack (DoS) which is elaborated in the next section. The relation between data leakage and DoS attack is elaborated in [4].

*Scalability:*

Openflow is a centralized implementation. All the programming is done at control layer and is vulnerable to many attacks. If each and every flow into the table is monitored by controller, then it becomes the bottleneck to handle huge number of switches and data flows. The authors [6] have studied that the capacity of the controller does not match the today’s high speed of flows. This bottleneck will saturate the potentiality and programmability of the controller.

Another drawback for the present controller implementations is the number of hops or forwarding devices between the data layer and control layer that determines the process delay. The total time for the packet transactions increases as the controller scales leading to performance issues. To overcome single point of failure, many implementations have adopted multiple controllers but multiple controllers could not improve the failure recovery [7].

*Configuration Issues:*

As network vulnerabilities are emerging, new security protocols are being designed and implemented. The applicability of these protocols are to all the layers and interfaces of the SDN. Few problems are raised if they are implemented without studying and analyzing the security shortages of the deployment environment. For example, in an implementation like an SDN-based network, it is important for the network operator to match the policies of TLS. Any incorrect use of security features or misconfigurations can impact all the layers of SDN.

*System Level SDN Security:*

Auditing process is the major concern to be satisfied in the industry. It is important to provide a controlled inventory of network devices. This involves the information of the topology of the network, devices running, etc. A switch’s internal logic can be operated in any of the modes like fail-secure mode or fail standalone mode, if the switch is disconnected and reconnected, the switch must operate in the same mode. The duty of the controller to understand the mode in which the switch operated during connection interruption, the impacted flow entries, the forwarding behavior during failures, and the behavior of the controller after reestablishing the connection. The system level design helps in maintains the integrity, confidentiality and availability of information.

## 3. Related Work

### 3.1 DDOS Attack

A DoS or DDoS attack is analogous to a group of people crowding the entry door or gate to a shop or business, and not letting legitimate parties enter into the shop or business, disrupting normal operations. In this case the group of people are the network packets and the door is the controller. The four major components of DDoS attack are the real attacker, compromised hosts called as handlers or masters capable of controlling multiple agents using software programs, the agent hosts who generate a large number of packets towards the victim host, and the target host to which the attack is launched.

A DoS attack could be performed at the infrastructure level with flooding of the flow

table for which limited memory resources are available [8]. In addition, the possibility of Do

S attacks leading to fraudulent rule insertion and rule modification is discussed in [9].

### 3.2 DDOS Detection Techniques

The most popular techniques used to detect flooding or DDoS are the methods to detect the large size of traffic.

Braga et al. [10] discussed a lightweight method to detect DDoS attacks in SDN. The main challenge was to distinguish normal packets from DDoS flooding packets. Different flow features [11]. The features are monitored continuously to detect the DDos attack. The limitation of this approach is to maintain this huge amount for SOM to classify or detect the attack. A separate controller can be implemented to avoid this overhead [12] of traffic monitoring.

A threshold is maintained [13] to detect the abnormal traffic from regular network traffic. It is a simple implementation to identify DDoS attacks as it keeps analyzing the traffic flows. A DDoS alert is triggered by controller if traffic exceeds a certain threshold. Once this alarm or alert is triggered, controller can insert a new flow rule into the table to drop the packets. Similarly, traffic map or patterns can be analyzed regularly to identify abnormal traffic.

Schehlmann et al. proposed COFFEE [14] which utilizes OpenFlow protocol to identify the botnet activities and erase it. It is an iterative implementation. The suspected flows are further validated by sending those packets to controller to extract more features. The network flow is monitored using Cisco technology NetFlow. The detection was done using machine learning algorithms. This method doesn't delay the network traffic until the inspection is completed. After detecting the attack, OpenFlow protocol was implemented by installing higher priority rules.

Feng et al. [15] proposed an intrusion detection method by merging ant colony networks and SVM. Multiclass SVM classification was done using One-against­ all method which trains N classifiers and consults all N classifiers for testing unknown sample. This increases the

testing time, which is critical for detecting intrusion at an earlier stage. Feng et al. [15] concluded that CSVAC (Combining support vectors with Ant colony) shows better results than SVM and Clustering based on Self-Organized Ant Colony Network (CSOACN).

### 3.3 DDOS Mitigation Techniques

DDoS can be mitigated by dynamic and effective response methods to handle DDoS attacks. Monitoring abnormal traffic flows and limiting the traffic rate by the controller are good countermeasures. Active response means to take an offensive action to counter an attack.

An attacker can perform data-to-control saturation attacks where the packets from switches or data plane are flooded to the controller and try to saturate both switch flow table and controller resources [16]. Mitigation mechanisms should ensure that controller and switches have the ability to quickly recover from such flooding. Passive or dormant monitoring agents are proposed to be triggered only when they see the occurrence of fake flooding.The mechanism should also be able to distinguish fake traffic from abnormal traffic.

Koponen et al., proposed FII (Framework for Internet Innovation) [17] to deal with inter domain DoS, based on the IP addressing scheme AIP [18]. AIP comprises details about hosts in packet header related to the host with a global ID, rather than an IP address. This may help eliminating attacks that hijack hosts based on their IP addresses or IP spoofing. They assure the availability of communication channel such that the packet can reach the destination. The approach divides handling DDoS into two parts: Inter and intra-domain attacks. For local, or intra-domain attacks, each domain is given an option to choose their own way of validating local hosts. On the other hand, FII provides a unified method to handle DDoS inter-domains attacks. As a countermeasure, a shut up message (SUM) is alarmed to intruders attacking the network with DDoS attacks.

Flooding or DDoS can also be resolved by flow rule merging and optimizing the flow rules in flow tables. Flow tables can be dynamically flooded with rules to saturate switch memory or to cause buffer overflows and cause it to be closed down or deny service for legitimate traffic or hosts. Hence, the switches must have an ability to dynamically update or re evaluate the flow table rules or merge various flow rules into one. The limitation of this approach is that it is intelligent and complex. Based on the current OpenFlow architecture, such intelligence is not programmed in the switches. Moreover, if such programmability is added to the controller, it will create an extra overhead on the controller.

## 3. System Design

### 3.1 Early Detection

#### 3.1.1 Collecting Flows

SDN allows controller to collect stats of switches flow on a periodic basis. Those stats are use by controller to monitor if an inactive path is there for a period of time. That flow will be removed by controller. Thus that flow information can be use by controller for detection DOS attack detection.

We will extract relevant information from the stats collected at controller such as bitrate. Bitrate for each flow will be recorded and if crosses a threshold then we can assume a node is under attack and take measures to mitigate it.

In SDN, while establishing connection between two hubs, information goes through controller and for subsequent flows packet will go according to instructions once settled by the controller. Thus to know data flow between hosts, stat accumulation is done by controller. We can define the period of stat collection by the controller. Period should not be very big to that attack detection is missed and no very small to have very less information.

#### 3.1.2 Detecting Attack and Attacker

To detect an attack recipient IP address is monitored for incoming packets. If a hst is under attack more packets will flow to it to create a flood effect. A map of victim is created in which if destination IP is facing an attack then it is added in the map otherwise if it already existing its attack count will be incremented with one. If the count against a victim is above a threshold then ingress policies are applied. If for a given time duration, a victim is not recognized then ingress policies will be removed.

As mentioned above a period is defined for the collection of stats. Let's say that as a window size. Window size is kept in time duration of 2 sec. For the mentioned window size, bitrate for each flow will be calculated which is given as :

Bitrate = (Byte\_Count\*8)/(Window\_Size\*1000)

If bitrate increases a attack threshold it is considered as attack. An attack count against each destination switch is captured. An obvious reason for increase in count is when more packet will received from the source and it will reduce unique sender IP in the window.

#### 3.1.3 Mitigation

Mitigation policies is majorly identification if attacker belongs to same domain or the other domain. If attacker id from the same domain where victim is lying then we can apply ingress policies on the port where attacker is connected. And if attacker and victim are lying in different domain then pushback messages mitigation technique is undertaken.

Pushback mitigation techniques basically sends a push back message to other domain controller to limit the rate of packet being sent. Will place an ingress on the port connecting to attacker for a given time period. If during the time period, there is no attack detected then remove the ingress policies from the port for free flow. Ingress policies usually affect queue size, bandwidth allocated and delay parameter. Bandwidth and queue size will be reduced. Delay will be increased for assigned duration.

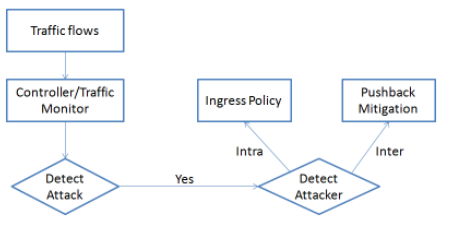


Figure 3: Design for Early Detection of DDoS attack

### 3.2 Detection using Machine Learning Techniques

#### 3.2.1 Collecting Flows

As mentioned in early detection , same methodology is adopted to collect flows. In SDN, while establishing connection between two hubs, information goes through controller and for subsequent flows packet will go according to instructions once settled by the controller. Thus to know data flow between hosts, stat accumulation is done by controller. We can define the period of stat collection by the controller. Period should not be very big to that attack detection is missed and no very small to have very less information.

We will extract relevant information from the stats collected at controller such as byte count, packet count, duration, sending and receiving ports .These information are recorded to find the features which will be fed to our models to train them for identification of spoofed packets. So that these packets can be dropped.

#### 3.2.2 Feature Selection and Extraction

Features are extracted from the a preset time interval. Time interval to collect flow entries is considered as window size. In that window size below features are extracted which are more or less suggested by Braga et al.[3].

1) *Average of Packets per flow (APF):*

As DOS attack uses flood host effect. So there is an increase of packet received at one end. Thus information like packet for each flow can be really helpful. Thus for the window size an average packet count is calculated. The source IP spoofing makes tracing the original attacker very difficult.

APF= Packet\_Count\_per\_Flow/Number\_Flows

2) *Average of Bytes per flow (ABF):*

Another parameter similar to above is byte count per flow which is similarly extracted for each flow and averaged over the number of flows in a window. As one of the peculiarity of DDoS attacks is their payload size. To detect the change in payload size ABF parameter can be used.Payload is often kept very small in order to increase the effect of this kind of attack. ABF is calculated as :

ABF =Byte\_Count\_per\_Flow/Number\_Flows

3) *Average of Duration per flow (ADF):*

Similarly, Duration is another parameter which tells live duration for a flow. And this cumulative duration divided by number of flow gives us the average of duration per flow which can be stated as :

ADF= Duration\_per\_Flow/Number\_Flows

This feature decreases the number of those cases which are not supposed be filtered when there is a small number of packets exchanged between applications.

4) *Percentage of Pair-flows (PPf):*

This is calculated to know the pair-flows in a window. A pair-flow can be described as:

* The source IP of flow 1 must be equal to the destination IP of flow 3;
* The destination IP of flow 1 must be equal to the source IP of flow 3; and

Both flows follow the same communication protocol. A main reason to calculate ppf is DDoS attack increases the number of single-flows into the network as attacker send packets with a fake IP. PPF is computed as:

PPF = (2∗Number\_Pair\_Flows)/Number\_flows

5) *Growth of Single-flows (GSf):*

As mentioned above dos attack increases single flows thus a parameter like growth of single flows can contribute a lot for the detection of malicious packet. When a attack starts GSF could be really high for the flows. It is calculated as:

GSf =(Number\_flows−(2∗Number\_Pair\_flows))/window\_size

6) *Growth of Different Ports (GDP):*

Similar to above parameter, GDP also focus on the increase of port number that are sending data to the victim port. As per IP spoofing fake ports are generated by DDoS attacks. Thus, we compute GDP as:

GDP=Number\_Ports/window\_size

#### 3.2.3 Training and Testing Model

To detect the malicious packets. Machine learning techniques are undertaken. We have opted SVM and Naive Bayes as our models to train and test the features extracted as mentioned above . Both the classifiers are defined as follows:

*Naive Bayes:*

Naive Bayes classifiers are a family of simple probabilistic classifiers based on applying Bayes' theorem with strong (naive) independence assumptions between the features. They assume that the features are independent of each other. The features contribute independently to the probability of classifying a particular tuple into a class label. It is not a single algorithm, it is combination of algorithms. Prior and posterior probabilities are calculated to label a tuple. The advantage of naive bayes is that it can train and give good accuracy for small amounts of training data.

*SVM:*

Simple vector machine are supervised learning algorithms used for regression, outlier detection and classification tasks. It can be interpreted as extension of perceptron. The tuples are represented as vectors in the space and a hyperplane is drawn to distinguish classes in the space. It is generally a binary classifier but can be extended to multiple classes.

It consist two phases training the model and testing the feature extracted from the flow stats. As per fig 4. the process can be enumerated as:

* Traffic flows are collected by the controller after a fixed interval and then the features abf, adf, apf,gdp,gsf and ppf are calculated for that interval from the stats.
* After feature extraction, data pre-processing is done to train models. As we are considering two different models i.e Naive Bayes and SVM, thus pre-processing is done based on specifications corresponding to them.
* After models are trained and they will be applied on feature extracted stream to test.If there a malicious entry. Then a threshold is kept for the count of malicious packet.
* If there an attack is confirmed flow table will be modified to reject the malicious packets and new rules will be added to flow table. This will be the mitigation strategy followed to stop the attack.

### 

Figure 4: Design for DDoS detection using ML techniques

## 4. Experimental Setup

An experimental setup is implemented on Ubuntu 16.04 operating system. VirtualBox is used as virtual machine to perform the network settings. To create a realistic virtual network, running real kernel, switch and application code on a virtual machine, Mininet 2.2.2 has been installed. OpenFlow architecture for SDN has been implemented. Ryu 4.19 is used for programming the controller, generate the traffic and attacks and monitor the packet flows. Python 2.7 is used as the programming language.

A custom topology of two domains namely Domain A and Domain B has been stimulated. Two domains have been chosen to show the inter and intra domain attacks and the solutions to mitigate the attack. Each domain consists of 4 hosts, 3 switches and all the switches of that domain are under one controller, namely controller 1 and controller 2. The below Figure 5 describes a clear picture of the topology. Hping3 is used to create abnormal traffic or high traffic between desired hosts or create attacks between domains or within the domain.

For our first approach of mitigation Bitrate is considered for detecting the attack. Thus values from our simulation is given as below with attack flow threshold as 5 and bitrate threshold as 4000:

| Traffic Type (Rate) | Attack Detected(%) | Attack Missed(%) |
| --- | --- | --- |
| 25% | 91.33 | 8.67 |
| 50% | 96.25 | 3.75 |
| 75% | 98.125 | 1.875 |

Two classifiers SVM and naive bayes are taken for classification task. As SVM takes only binary values and the features are discrete values, we have normalized and binned the features into different bins for the SVM classifier.

| Classifier = SVM | Attack = yes | Attack = no |
| --- | --- | --- |
| Attack = yes | 87 | 18 |
| Attack = no | 23 | 1126 |

Accuracy(SVM) = 96.73

| Classifier = Naive Bayes | Attack = yes | Attack = no |
| --- | --- | --- |
| Attack = yes | 58 | 31 |
| Attack = no | 29 | 943 |

Accuracy(Naive bayes) = 94.44

## 5. Conclusion

In this paper we have opted two methodologies to mitigate the effect of DOS attack.One uses the information of bitrate and other uses features extracted from the flows.

* First technique determine the inter and intra domain attack and mitigate them by changing ingress policies and sending push back message to other controller
* Second techniques change the flow table of the switches to safeguard the network against the attack. Where malicious packets are identified using machine learning techniques

Our work is mainly derived from the prevalent work in the DOS attack and mitigation techniques. And we have tried to show and compare two majorly used techniques of detection and mitigation. Though a lot has been done in machine learning techniques to handle the DDOS attack but still a huge scope is there for improvement.

## 

Figure 5: Custom Topology

## 6. References

[1] Xia, Wenfeng, et al. "A survey on software-defined networking." IEEE Communications Surveys & Tutorials 17.1 (2015): 27-51.

[2] Ahmad, Ijaz, et al. "Security in software defined networks: A survey." IEEE Communications Surveys & Tutorials 17.4 (2015): 2317-2346.

[3] Braga, Rodrigo, Edjard Mota, and Alexandre Passito. "Lightweight DDoS flooding attack detection using NOX/OpenFlow." Local Computer Networks (LCN), 2010 IEEE 35th Conference on. IEEE, 2010.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, “Openflow: enabling innovation in campus networks,” SIGCOMM Comput. Commun. Rev. , vol. 38, no. 2, pp. 69–74, 2008.

[3] Kokila, R. T., S. Thamarai Selvi, and Kannan Govindarajan. "DDoS detection and analysis in SDN-based environment using support vector machine classifier." Advanced Computing (ICoAC), 2014 Sixth International Conference on. IEEE, 2014.

[4] S. Shin and G. Gu, “Attacking software-defined networks: The first feasibility study,” in Proc. 2nd ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw. , 2013, pp. 165–166.

[5] OpenFlow Switch Specification Version 1.4, Open Networking Foundation. [Online]. Available: https://www.opennetworking.org

[6] M. Jarschel et al., “Modeling and performance evaluation of an OpenFlow architecture,” in Proc. 23rd ITCP, 2011, pp. 1–7.

[7] G. Yao, J. Bi, and L. Guo, “On the cascading failures of multi-controllers in software defined networks,” in Proc. 21st IEEE ICNP, Oct. 2013, pp. 1–2.

[8] S. Sezer et al. , “Are we ready for SDN? Implementation challenges for software-defined networks,” IEEE Commun. Mag. , vol. 51, no. 7, pp. 36–43, Jul. 2013.

[9] K. Benton, L. J. Camp, and C. Small, “OpenFlow vulnerability assessment,” in Proc. 2nd ACM IGCOMM Workshop Hot Topics Softw. Defined Netw. , 2013, pp. 151–152.

[10] Rodrigo Braga, Mota Edjard, Passito Alexandre “Lightweight DDoS flooding attack detection using NOX/OpenFlow” , Proceedings of the IEEE Conference on Local Computer Networks (LCN), Denver, CO, USA, 11–14 October 2010 (2010), pp. 408-415.

[11] Yifu Feng, Wang Dongqi, Zhang Bencheng “ Research on the active DDoS filtering algorithm based on IP flow ”, 2009 Fifth International Conference on Natural Computation, IEEE (2009), pp. 628-632.

[12] Shirali-Shahreza and Ganjali Sajad Yashar “Efficient implementation of security applications in OpenFlow controller with FleXam “, HotSDN'13, August 16, 2013a, Hong Kong, China.

[13] YuHunag Chu, Tseng MinChi, Chen YaoTing, Chou YuChie, Chen YanRen “ A novel design for future on-demand service and security”, 12th IEEE International Conference on Communication Technology (ICCT) (2010).

[14] L. Schehlmann and H. Baier, "COFFEE: a Concept based on OpenFlow to Filter and Erase Events of botnet activity at high-speed nodes," Proc. of Lecture Notes in Informatics, vol. p-220, pp. 2225-2239, Sep. 2013.

[15] W. Feng, Q. Zhang, G. Hu, and J. Huang, "Mining network data for intrusion detection through combining SVMs with ant colony networks” , Future Generation Computer Systems, vol. 37, pp. 127-140, July 2014.

[16] Haopei Wang, Lei Xu, Guofei Gu, “Of-guard: a DoS attack prevention extension in software-deﬁned networks”, ONS (2014).

[17] Teemu Koponen, Shenker Scott, Balakrishnan Hari, Feamster Nick, Ganichev Igor, Ghodsi Ali, *et al. “*Architecting for innovation”, Comput Commun Rev, 41 (3) (2011), pp. 24-36.

[18] David G. Andersen, Balakrishnan Hari, Nick Feamster, Koponen Teemu, Moon Daekyeong, Shenker Scott, “Accountable internet protocol (AIP)”, SIGCOMM'08 (August 17–22, 2008)

Seattle, Washington, USA.