T.C.
SAKARYA UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY

MACHINE LEARNING BASED DDOS ATTACK DETECTION FOR SOFTWARE-DEFINED NETWORKS

M.Sc. THESIS

Douglas Omuro MAKORI

Department : COMPUTER and INFORMATION ENGINEERING

Supervisor : Assist. Prof.Dr. Seçkin ARI

February 2018

DECLARATION

I, as a graduate student atSakarya University, solemnly declare thatevery information and data and results in this thesis follows academic laws and that I have not committed any malpractices which include but not limited to falsification and distortion of research findings,thesis written by somebody else and plagiarism that may affect the integrity and credibility of my research.I guarantee that the conclusions in this thesis are honest and based on careful research I conducted under the guidance of my supervisor.

Douglas Omuro MAKORI

ACKNOWLEDGMENTS

I will like to express my sincere gratitude to my supervisor Yrd. Doc. Dr. Seçkin Ari for his unwavering support, patience, encouragement and time throughout my studies and as I undertook this research under his supervision. It was a great experience throughout the research and thesis writing.

I will also give thanks to my family especially my Dad, my brothers and sister for encouraging and supporting me, my friends who are there for me always and of course my late mother who was my inspiration and I am always working to be what you wanted me to be.

TABLE OF CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

ACK	: Acknowledgment message
API	: Application programming interface
DDoS	: Distributed Denial of service attack
DoS	: Denial of service
FN	: False negatives
FP	: False positive
HIDS	: Host intrusion detection system
ICMP	: Internet control message protocol
IDS	: Intrusion detection System
IP	: Internet protocol
IPS	: Intrusion Prevention system
NFV	: Network Function virtualization
OVSDB	: Open vSwitch Database management protocol
SDN	: Software-defined network
SYN	: Synchronization message
TCP	: Transport control Protocol
TLS	: Transport layer security
TP	: True positives
UDP	: User datagram protocol
VLAN	: Virtual local area network
VM	: virtual machine
XMPP	: Extensible Messaging and presence protocol

LIST OF FIGURES

LIST OF TABLES

SUMMARY

Keywords: Software-defined networking, Machine learning, Entropy, intrusion detection

Software-defined networking is a new networking model which decouples the control plane from the forwarding plane which eliminates vertical integration in current legacy networks and provide a global view of the network via a network operating system called a controller. This is going to cut the cost of networking through softwarerization of components previously provided as hardware, it will promote innovation, network security, quality of service, on demand provisioning and load balancing. It is being touted as a future technology especially in the data centers where virtual machines, cloud computing and virtualization arebeing adopted. Despite software-defined networking (SDN)having benefits like anetwork wide view through the controller, its biggest weakness is the controller itself. Attackers can direct massive requests to the controller than it can handle making it crash and unavailable thus rendering the network offline and unusable.

The aim of this thesis is to propose a hybrid lightweight mechanism that runs with the controller to help identify anomalies in the network with use of minimum resources. Because of resource scarcity in SDN a flow intrusion detection system (IDS) is used but since the flow IDS has many false positives, when an attack is detected the flow is forwarded to an entropy calculator module which is based on the TCP 3-way handshake with a threshold set. If the flow is indeed an attack, it'smitigated by a SYN packet counter which blocks the IP address flooding the controller if the number of packets exceed a 50 packets per second.

YAZILIM TANIMLI AĞLAR IÇIN MAKINE ÖGRENME ESASLI DDOS ATTACK ALGILAMA

ÖZET

Anahtar kelimeler: Software-defined networking, Machine learning, Entropy, intrusion detection.

Yazılım tanımlı ağlar dediğimiz kontrol düzlemi eski ağlardaki dikey bütünleşmeyi ortadan kaldıracak ve denetleyici adı verilen bir ağ işletim sistemi vasıtasıyla ağın genel bir görünümünü sağlayacak olan yönlendirme düzleminden ayıran yeni bir ağ modelidir. Bu, önceden donanım olarak sağlanan bileşenlerin yazılımla bilgisayar ortamında kurulumunu azaltacak, yenilik, ağ güvenliği, hizmet kalitesi, isteğe bağlı hazırlama ve yük dengeleme işlemlerini destekleyecektir Özellikle sanal makinelerin, cloud bilgisinin ve sanallaştırmanın kök saldığı veri merkezinde daha da ileri bir teknoloji olarak ön plana çıkıyor. SDN, ağ üzerinden denetleyici aracılığıyla ağ genelinde bir görüş sağlama avantajına sahip olmasına rağmen, bu da onun güçlü zayıflığıdır çünkü denetleyici olmadan bir SDN ağı çalışamaz. Saldırganlar, denetleyiciyi işleyebileceğinden büyük taleplerle hedefliyorlar, dolayısıyla çevrimdışı eleştiriliyorlar.

Bu araştırma, makine öğrenme tabanlı akış IDS'in melez bir mekanizmasının yanı sıra TCP SYN, ACK için denetleyiciyi etkilemeden bir saldırıyı tespit etmek için Entropi sayacı ve denetleyiciye saldıran IP adresleri gerçek zamanlı olarak engellenmiş bir hedefi vardır. SDN, normal bir ağın kaynaklarına sahip değildir, bu nedenle çözüm mümkün olduğunca hafif olmalıdır.

Araştırma, kullanılabilecek bir saldırıyı kapsayan saldırı türlerini ve saldırı senaryosunu uygulamakla birlikte saldırının ilk birkaç saniyede nasıl hafifletilebileceğini gösterecek.

CHAPTER 1. INTRODUCTION

Current legacy networks have evolved into challenging monsters that are difficult to manage and lack the ability to scale to today's needs of mega data centers.Software-defined networking will facilitate their decoupling by separating the data, control and management planes. SDN has enhanced the programmability of networking switches by providing application programming interfaces (API). Programmability has eliminated the need for programming application/vendor specific forwarding for legacy network architectures leading toa well suited, fine-grained efficient algorithmic forwarding decisions that are universally applicable. Software-defined networking has brought scalability and central control which makes it easy to monitor and troubleshoot the SDN networks.

Software-defined networking has led to significant cost reduction for consumers because devices like load balancers and firewalls implemented in current legacy networks which cost thousands of dollars are implemented as software at a fraction of the cost. SDN provides the ability to apply network virtualization based on for example on layer two or layer three features. Virtualizationenablesthe sharing of the same network resources which helps in overall reduction of costs in data centers. Software-defined networking provides Application programming interfaces (APIs) to enhance the programmability of SDN networks. The northbound and southbound APIs are used to write software modules such as firewalls, quality of service and IDS that can be used tocontrol the network systemsinexpensively.This hype, applicability, and popularity of SDN make it a target of hackerswhose intentionis gaining fame and profiteering from access to critical infrastructure where SDN is implemented thus knowing its vulnerabilities is worth finding out as it will help them hack the systems.

1.1. Problem Statement

Security is a pertinent issue in networks because if one gets in control of computer networks, it means they can access nearly all other resourcesat a company's disposal.VLANswere invented to separate different organizational functions from interacting to each other directly to enhance legacy networks'security. SDN supports network virtualization that uses criterions such as packet fields in the packet header which can enable the kind of abstraction thatallows sharing of the physical network infrastructure for multiple users at the same time. Network function virtualization (NFV) which is closely related to and supported by SDN is facilitating softwarerization of network functions such as DNS and caching leading to service innovation and provision of new services.

The decoupling of the control plane from the forwarding plane took away control functions from switches and routers. The controller handles these control functions in software defined networks. The decoupling has simplifiedthe design of network services such astraffic routing, access control, quality of service and security applications through software.ThisSDN ability has resulted in innovation in networks by software being able to innovate faster without waiting for hardware.

In software-defined networks, the controller communicates with the forwarding plane through the southbound application programming interface (API) using a secure transport layer service.All switches have flow tables against which flows match. When a packet arrives at the switch and its header fields do not match against the flow table, the packet is sent to the controller as a *packet-in* message. The controller will process the packet and sends a *packet-out* or *flow-mod* message with particular flow rules which will be installed on the flow table so that next time a similar packet comes to the switch; it is acted upon without a reference to the controller.

The centralization of the controller simplifies network management, but it is a security nightmare.It can be targeted by attackers with massive flooding of requests to knock it offline and make the network unusable. With the controller offline,services will not be available, and the SDN network will not work without the

controller. SDN is susceptible to distributed denial of service attacks(DDoS)which not only target the controller but also target the flow tables in the switch rendering them full at all times so that whenever packets arrive at the switch, they are not processed thus the network is rendered useless. These weaknessesprompt the need to havecontrollers' backups. But these controllers can also be attacked and taken offline depending on the amount of traffic directed at them. As a result of this possibility, there is a need for a system for early detection and mitigation of these attacks.

Network security research has used various techniques ranging from soft computing to machine learning techniques such as supervised, unsupervised, semi-supervised learning, and deep packet inspection. The same case applies to the security of software-defined networks which has been extensively researched mostly due to its "new cool" tag and its observed ability to change the future of networking.

Devices such as firewalls, and software such as Snort IDS implement intrusion detection systems. But we cannot compare them because they both help prevent intrusion but they are different in the way they work; a firewall is just a system that has a predetermined set of rules which allows it to act as a gatekeeper. These pre-determined rules make it a target for attackers who can easily bypass it. Snort uses signatures and a set of rules to help it detect intrusions making it a bit better than a firewall however it has too many false positives as well as false negatives this reduces its efficiency and reliability.

Attackers are changing and advancing their attacking techniques making systems more vulnerable.Machine learning when used on top of the traditional flow and packet inspections systems, enables them to adapt in dealing with new and modified attacks. New techniques that employ machine learning and other artificial intelligence techniques need to be designed solely for malicious intrusion detections in SDN networks.

1.2.Research Questions

Thisthesis seeks to explore weak spots in software-defined networks. In this thesis,is regarded as a single point of failure. Because of the significance of the controller as

the brain of the software-defined network,this study suggests a way to detect intrusions in software-defined networks. This intrusion detector must:

a) Use the least amount of resources.
b) Must identify and mitigate the intrusion early in real time.
c) To find a way to make the intrusion detection system dynamic in its detections
d) Be able to write flow rules to block any source of attacks in the network.

1.3.Research Contributions

This thesis research examines the use of machine learning for detecting and mitigating intrusions in software-defined networks. In this research, we use Distributed denial of service attacks to test the proposed solution.

a) Designing a lightweight module on pox controller that has fivesub-modules including a machine learning based IDS with a model trained and tested by an intrusion dataset. The modeldetects intrusions on a real-time SDN network traffic.
b) Mitigating the distributed denial of service attack through writing flow rules on the switches and blocking any source of malicious traffic.
c) Successful implementation of the detection algorithm in pox controller and Mininet.

CHAPTER 2. SOFTWARE-DEFINED NETWORKS

2.1. Background

Legacy computer networks have two major parts; control plane which has the algorithms that control the movement of packets and the data plane which deals with the forwarding of the data to their destinations. These functions use proprietary licenses as well as vendor-specific hardware.Vertical integration stifles creativity because it's possible for software todevelop faster. But because of coupling together it is impossible for the software to evolve faster than hardware. Software-defined networks solve this bottleneckby decoupling the control and forwarding planesof the legacy network and providing a centralized view of the distributed network systemensuring orchestration and automation of network services(sdxcentral, n.d.).

Software-defined networks provide programmability to the network, an option that has facilitated software to evolve faster than the hardware. SDN has facilitated the design of new network components as software.On-demand provisioning, automated load balancing, streamlined physical infrastructure and the ability to scale network resources according to application and data needs have all been made possible by SDN(Cole, 2013). All these coupled together with network virtualization in servers,SDN is a future technology with promise.

There are two trends currently in SDN: those focusing on the dynamic virtual machine migration and use of hypervisors as well as techniques such as encapsulation and tunneling and those that are striving to achieve software control of the network by using the OpenFlow protocol to manipulate the flow tables in switches(Metzler, 2014). VMware and Open Networking Foundation are for centralization of network control and don't see the role of hardware for some network function in datacenters while on the other hand is Cisco which supportsboth

trends.On April 4, 2017, open networking summit,Google presented espresso SDN pillar which is extending their SDN to the edge of the public internet and as a result making their cloud 25% faster, more available and cost-effective(Hardesty, 2017).

Hardware SDN devices vendors include Cisco, HP, Jupiter, Big switch, and Netgear.Open Networking Foundation (ONF) the organization that is promoting the adoption and implementation of SDN as an open standard developed the OpenFlowopen standard. OpenFlow management and configuration protocol standard a first of its kind is a vendor-neutral interface between the control layer and data layer of the SDN architecture(opennetworking foundation, n.d.).

Software-defined networking architecture is dynamic, manageable, cost-effective and adaptable. These qualities make it suitable for today's bandwidth-hogging data centers that are in need of a flexible solution that can put resources where they are needed. An SDN architecture is considered to be:

a) Programmable- this is facilitated by the decoupling of control and the forwarding planes
b) Agile-This is brought about by the abstraction of the control plane from the forwarding plane which gives network engineers freedom to make network-wide changes in traffic flow to meet their current needs.
c) Central management- SDN has the concept of a controller that's has a network-wide view which appears to all applications and policy engines as a single switch. This visibility makes network management easier.
d) Programmatically configured: SDN gives freedom to network engineers to configure, manage, secure, and optimize network resources very quickly using dynamic custom SDN programs which are vendor independent and can be written by themselves.

SDN is vendor neutral and uses open source standards which makes network design and operation simple because SDN controllers give instructions instead of multiple, vendor-specific devices and protocols.

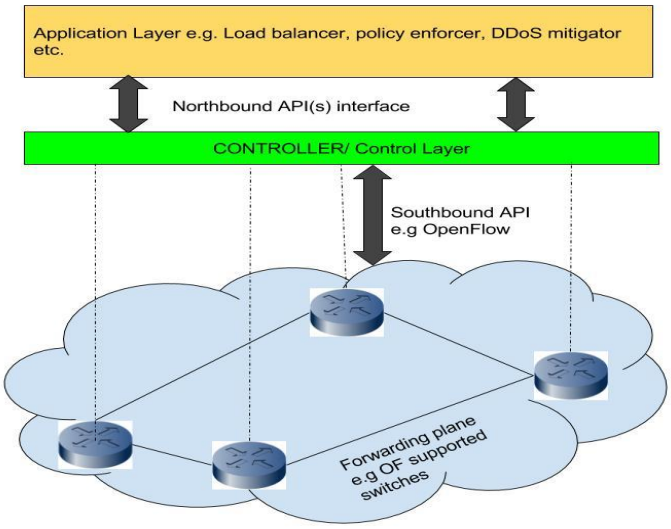

Figure 2.1. SDN Architecture

2.2.Controllers

Controllers are the brain of a software-defined networkthat takes the control plane out of network hardware and runs it as software. The controller facilitates easier integration, administration of applications and automated management of networks. It has a global view of all forwarding devices in the forwarding plane. It also has a way to communicate forwarding instructions to the forwarding devices via the southbound API. It provides abstractions of the network to all the network applications which make developing of network applications easier as developers need only to know how to interface with the controller. The controller is vital to SDN because it performs the control functions that were previously done by switches. If packets do not match against the flow the tables, it's the controller the directs switches on the destination port or address of the packets through the packet-out message. This is possible because the controller has a whole view of the network.Some of the mainstream controllers are include:

2.2.1.OpenDaylight project

OpenDaylight is a community-led and industry-supported open source SDN project initiated by the Linux Foundation with the sole aim to advance software-defined

networking adoption and make a strong case for network function virtualization. It aims to deliver readily deployable controller without any need for other components. It supports add-ons that can add value to its uses. It is java based, and some of its founding members are Big Switch, Cisco, Brocade, Ericson, HP, and IBM. It uses open standards,and as a result of working with open networking foundation, it readily supports OpenFlow though it's open to any future open protocols other than OpenFlow. It provides a northbound API /Restful API which can be used to develop applications network applications easily.

2.2.2.Floodlight controller

It's a java based open source enterprise controller developed by Big Switch. It is easy to use, has a large community of users thus highly likely to get help.It's easy to use and easy to set it up because of the few dependencies. It's multithreaded thereforehas high performance, it supports OpenStack making it easily deployable in cloud computing scenario. It also supports both OpenFlow and non-OpenFlow switches hence highly applicable in every network scenario(Project Floodlight, n.d.).

2.2.3.Pox controller

Pox is a python based open source software-defined networking controller which is very popular for rapid prototyping. It comes with components such as a hub, a layer three switch component, topology discovery and even a spanning tree component which all contribute towards rapid prototyping.

2.3.Secure Channel

The secure channel facilitates the interaction between the controller and the OpenFlow switch as shown in Fig 2.2. It enables the controller to be able to configure and manage the switch by receiving events from the switch and facilitating

sending of packets out of the switch. It uses TLS protocol for secure communication. The protocol is the lifeline of an SDN network because if it the connection fails, the packets which don't match with the switch flow table will not be sent to the controller. If the switch cannot establish a link to the backup controllers, it will fall into the fail secure mode, and all packets that do not match against flow entries are dropped instantly.

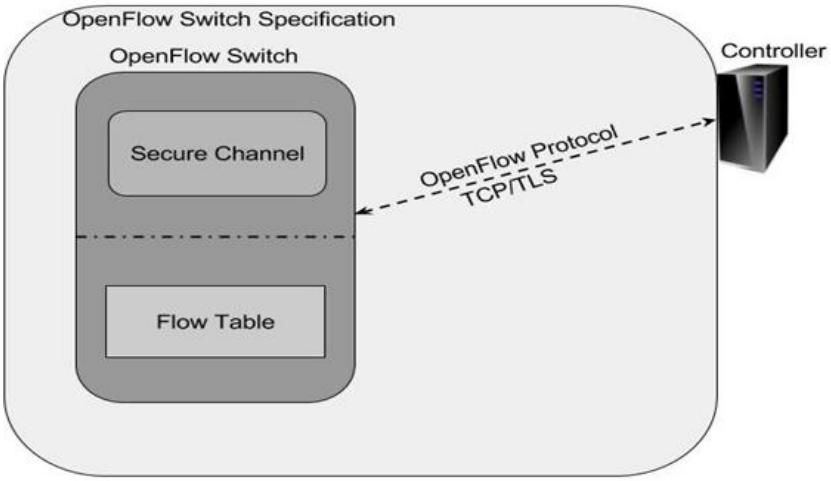

Figure 2.2. Secure channel

2.4.Southbound Interface

The controller communicates with network devices via the southbound API. The API is used to transmit information such as packet handling instructions, notifications on status changes, e.g., if some devices or links are up or down and statistics information such as flow counters or aggregate statistics. OpenFlow is the most common protocol used in the SDN.

2.4.1.OpenFlow

This is the standardized communications protocol that facilitates interaction between the control plane and the forwarding plane. It is the southbound API for software-defined networks. It enables network programmers to modify the behavior of the

switches and routers through writing scripts that run in the controller. OpenFlow protocol allows direct access and manipulation of forwarding plane devices such as switches and routers. Other than openflow other important SDN protocols include:

a) Border gateway protocol for hybrid SDN.
b) NETCONF which is mandatory for configuring OpenFlow enabled devices.
c) MPLS-TP a transport profile for multiprotocol label switching used as a network layer technology in transport networks.
d) Open vSwitch Database Management Protocol (OVSDB) an OpenFlow configuration protocol for managing open vSwitch implementations in SDN.
e) Extensible Messaging and Presence Protocol (XMPP) which is used for messaging and online presence detection all in real time.

Switches communicate with the controllers via OpenFlow messages. They are specified in the OpenFlow specification. When writing any SDN scripts to modify how the forwarding plane works, understanding of OpenFlow messages and events is vital.

2.4.1.1.OpenFlow events

a) Connection Up
   Fired up as result of the establishment of a control channel with the switch.
b) Connection down
   Fired up when the connection to the switch has been terminated.

c) Flow_removed
   This event is raised when the controller receives the flow removed from the switch. The ofp_flow_removed is sent when a table entry is removed due to idle or hard timeout.
d) Statistics events

Raised when the controller receives anOpenFlow statistics reply message (ofp_stats_reply / OFPT_STATS_REPLY) from the switch. The switch repliesto a statistics request (ofp stats_request) from a controller component with this message. For DDoS detection statistics events especially flow statistics is very vital in that by tracking the number of flows which can be obtained by stats request from the controller you can predict a DDoS in an SDN network.

They include:

1. FlowStatsReceived (ofp_flow_stats)
2. PortStatsReceived (ofp_port_stats)
3. QueueStatsReceived (ofp_queue_stats)
4. TableStatsReceived (ofp_table_stats)
5. AggregateFlowStatsReceived (ofp_aggregate_stats_reply)

e) Packet_in

It's fired up by the controller to indicate that a packet arriving at the switch is not matching all entries. Its fire up as a result of OpenFlow packet-in message (ofp_packet_in / OFPT_PACKET_IN).

Its key attributes are:

1. Port (int) – in port
2. Data (bytes) – raw packets
3. ofp (ofp_packet_in) - OpenFlow message which caused this event.

Packet_in is very important especially for DDoS detection especially the packet spoofing DDoS as well as ACK DDoS attacks in that you can monitor the packet-in packet-out to decide if it's a DDoS attack.

2.4.1.2.OpenFlow messages

They facilitate communication between the controller and switches. OpenFlow messages also lead to particular events being invoked. Some of the messages include:

a. Features_request

It’s sent by the controller to the switch. It is composed of just the OpenFlow header.

b. Features_reply

The switch replies to the ofpt_features_request by the controller using this message.

c. Packet_out

The controller sends this message to the switch instructing it to send a packet or enqueue it or even discard it. Its attributes include buffer_id, in_port, actions and the data (bytes).

d. Flow modification message

It’s a message with instructions for modifying the flow table. The crucial fields in flow modification message are hard_timeout, idle_timeout in that they determine how fast flows expire. These fields can be used to write flows or delete flows that are suspect.It’s sent by the controller to the switch.

e. Port modification message

It’s used to modify the behavior of physical ports.

f. Statistics messages- Flow statistics messages include:

1. *Individual Flow Statistics*-Individual flow stats.
2. *Aggregate Flow Statistics*- Contains multiple flows statistics.
3. *Table Statistics*- Containstable information.
4. *Port Statistics*- Containsphysical statistics.

g. Packet-In Message

The packet-in message is sent from the switch to the controller. It’s raised when packets arrive in the datapath or switch and don’t match all fields thus sent to the controller to determine appropriate actions to be performed on the packets. The actions include to forward the packet to a particular port, to drop the packet or modify the packet headers.

h. Flow Removed Message

Flow Removed message is used by the datapath/switch to inform the controller that a flowhas been removed. OFPT_FLOW_REMOVED is used. It has the

following fields:OpenFlow header, openflow_match, cookie field, and packet count and byte count fields. It also has a duration in both nanoseconds and seconds. This message also includes reason field which explains why the flow was removed e.g. due to idle_timeout or hard_timeout.

OpenFlow can create new packets in the network, check port and flow status, and check on table status. As a result, SDN facilitates deployment of applications that perform functions like traffic engineering, network security, quality of service, routing, switching, virtualization, network monitoring, load balancing and many more innovations that can be envisioned by the network developers. All these capabilities were brought by SDN's ability to enable the network to evolve at the speed of the software instead of hardware as is the case in legacy networks.

2.5.OpenFlow Switch Specifications

OpenFlow allows switches to be managed by the controller which is vendor independent. The switch has a flow table which deals with packet forwarding as well as packet lookup functions. It also has a secure TLS channel linking it to the controller which controls the switch via the OpenFlow control. The flow table has flow entries whose packet header values are matched against. It also contains counters as well as actions to be performed on the packets if they match or not match against the entries in the flow table. If packets match against flow table entries, actions canbe to forward the packet over a particular port which can be either physical or virtual port, drop the packet or modify the header fields such as destination IP, Port. If the packets do not match against the flow table entries, they are sent to the controller via *packet-in* which determines what actions to be performed on the packet then sends the instructions to the OpenFlow switches via Packet-out message. OF switch flow entries are demonstrated in Figure 2.3.

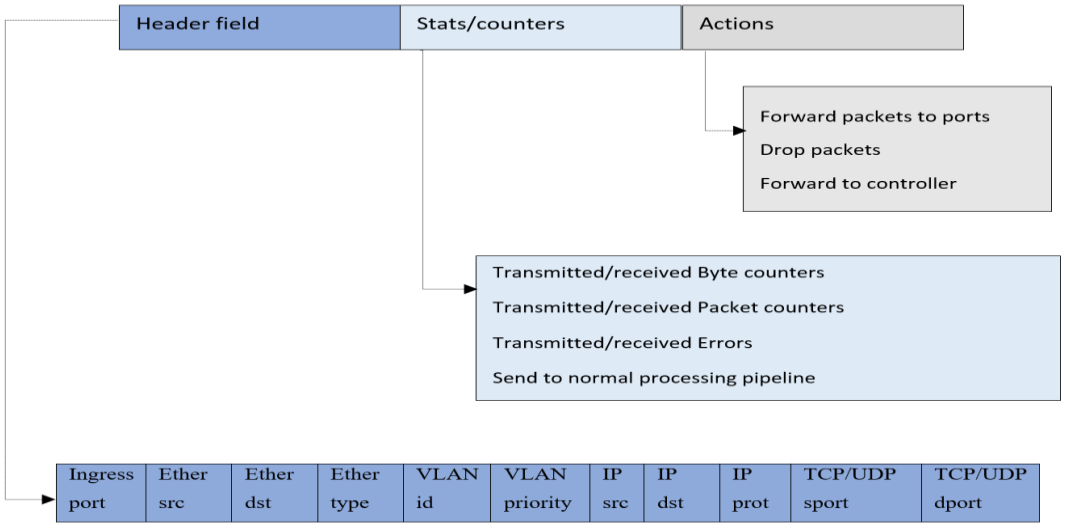

Figure 2.3. Flow table entry

The statistics or counter are provided per table, flow, port and queue as shown in the Table 2.1. The controller can always query the switch and get the counters for these statistics.

Table 2.1. Supported counters list used in statistical messages

Counter		Bits
	Active entries	32
Per Table	Packet lookups	64
	Packet matches	64
	Received packets	64
Per Flow	Received Bytes	64
	Duration in seconds	32
	Duration in nanoseconds	32
	Received packets	64
	Transmitted packets	64
	Received Bytes	64
	Transmitted Bytes	64
	Receive Drops	64
	Transmit Drops	64
Per Port	Receive Errors	64
	Transmit Errors	64
	Receive Frame alignment errors	64
	Receive overrun errors	64
	Receive CRC error	64
	Counter	Bits
	Collisions	64
	Transmit Packets	64
Per queue	Transmit Bytes	64
	Transmit Overrun Errors	64

2.6.Northbound API

SDN applications include load balancers, firewalls, security applications or applications like OpenStack that do orchestration in cloud services. The northbound API provides the network control information to these applications that have very high instance abstractions of the network. The value of SDN is enhanced by the availability of the northbound API on top of which the SDN applications will be developed otherwise the control plane, and forwarding plane are doing the same things. SDN has the upper hand because, with the centralized view of the network by the controller, all control information is in one place andismade available via anAPI.

The API facilitates ease of configuration of networks via applications instead of command-line interfaces, network monitoring, traffic engineering, and security, dropping of suspicious packets, and dynamically changing the quality of service based on the availability of bandwidth or type of customer subscription. Many different players have different approaches to northbound APIs,for example,influential players like Cisco and HP provide the whole SDN stack using existing switches and a new controller and already developed applications that areinflexible to the southbound API. Another class of vendors is Nicira / VMware who control the whole stack of applications, switches and the controller which limits the number of applications offered as well as the number of environments that can be supported.

With the increase in the number of controllers available as well as the programming APIs, open networking foundation established open networking foundation working group in 2012 which was to work towards standardizing the northbound APIs. In the resulting report(Raza, 2013), they agreed upon a standardized North-Bound Interface which will enable developers to concentrate in differentiating their applications instead of porting to different controllers while allowing vendors to focus on performance and availability of rick features to attract new users. The group agreed to provide stable and portable NBI to developers, network services, applications and various controllers.

The proposed northbound interface is composed of the following:

a) User facing Controller APIs
b) Business Logic Implementation
c) Control Plane implementation
d) Device facing protocol adapters

2.7.SDN Security Issues

SDN has a lot of applicability in current networks and future networks like 5g networks. SDN isbeing used for cloud networking, gateway control and backhauling(Schneider, 2015). However, SDN security faces a lot of threats from

malicious applications, attacks from the forwarding plane, attacks from the control network through the northbound APIs in a cloud computing environment(Schneider, 2015). The centralized view of the network is also a single point of attack and failure. The southbound API is also prone to attacks that can compromise its availability as well as performance(Lim, 2015). Despite the advantage of programmabilityand the ability to manage packet forwarding and policy application,SDN security is considered a crucial issue by Lim in (Lim, 2015). Due to SDN infancy and inability to enforce security on a physical topology, an SDN attacker can identify targets,modify content, hide from intrusion detection systems, attack servers, and monitor traffic leaving SDN operators exposed. Lim also Points out that other factors such as the use of TLS for encryption and authentication can leave an SDN network susceptible. Lim points out the possibility of controller flooding and ability to impersonate an OpenFlow switch is a real security headache as is the availability of debugging ports that are not encrypted thus can be used to take control of the switch.He recommends securing and protecting the controller, establishing trust, creating a policy framework to review the ability of controllers to perform the right roles as well as conducting forensics, correcting, recovering the network and protecting the network from attack in the future.

There are several vulnerabilities in the SDN ecosystem. According to (Hinden, 2014) in the article “why take over the hosts when you can take over the network”, the biggest threat to SDN environment is the compromising of the controller because if the controller is compromised, then the whole network is compromised. Effects of SDN controller being compromised include controller subverting new flows, ability to launch a man in the middle attacks, ability to modify content on the network, monitoring of traffic is also possible and sending traffic to compromised nodes unknowingly(Hinden, 2014). The network should be designed with “security everywhere” in mind with all network devices such as routers and switches having security capabilities which will enable security applications to push rules to all those devices. Network designers can also design the networks in a way that it’s easy to isolate any hosts suspected of being compromised. SDN network should also make use security control applications such as access control lists or fully featured

firewalls(Hinden, 2014),(Open Networking Foundation, 2014). One of the many techniques that have been discussed and applied in SDN security is the application of intrusion detection and prevention systems.

2.8.Intrusion Detection

Intrusion detection is an effort to find attempts at compromising the availability, confidentiality as well as the integrity of a resource. It aims to identify any attempts at subverting the already laid down security controls(Berge, n.d.). it is a process of monitoring computer and network system events checking for any signs of incidents that are violations or threats to the laid down standard security policies, use policies as well as computer policies(Mell, 2007).Types of intrusion detection systems include:

2.8.1.Network-based IDS

Network-based IDS attempt to identify any illegal and unauthorized behavior that's unusual in the network. It uses networking devices which collect packets that it processes. The IDS uses laid down criteria to flag suspected traffic. It only works to alert the administrators of imminent danger(Berge, n.d.). An example is Snort IDS. They monitor and analyze all inbound traffic to protect the system from any network-based threats(Techopedia, n.d.). Sometimes network-based IDs monitor and inspect traffic at specific selected points in real-time, to attempt to detect intrusion. These systems are monitor network traffic directed at vulnerable systems in a network. They can detect application, network and transport layer protocol activities(Stallings, 2007). NIDS systems operate on a "wiretapping concept" whereby they inspect the packets from the network stream. The IDS checks for irregular behavior because it comes pre-loaded with attack signatures(SANS Institute, 2000). NIDS systems are highly portable and independent of operating system as pointed out by a SANS institute report. However, these systems have downsides such as: use of attack signatures which makes the system lag behind on latest attacks, they also have a scalability problem especially in high speed networks, and encryption is a problem as the systems cannot scan the protocols according to the SANS institute.

2.8.2.Host-Based IDS

Unlike the network-based IDS, HIDS works to identify any illicit unauthorized behavior on a particular device instead of on a whole network. It is normally installed on each host where it monitors the current operating system and applications. It normally uses a combination of heuristics, rules, signatures and nowadays artificial intelligence techniques like machine learning, deep learning as well as soft computing techniques like fuzzy logic are being deployed for IDS purposes. Examples include Tripwire, AIDE, and OSSEC - Open Source Host-based Intrusion Detection System. These systems use sensors that collect audit trails about the system being monitored. A sensor is needed for each host. This makes HIDS more expensive. They also rely on audit trails which in turn limits these IDS systems. They can also use system logs which are text files where information is recorded whenever system processes operate(SANS Institute, 2000). They are the last layer of defense and the can be customized for application or workflow needs(Beaver, n.d.)

2.9.Intrusion Prevention

It is the act of extending the roles of an intrusion detection system to include the ability to block the malicious unauthorized activities detected(Berge, n.d.). intrusion prevention involves deep packet inspection which can alleviate different network attacks, worms and viruses(Cisco, n.d.). some of the IPS systems have advanced features such as real-time sandboxing and mitigation solutions, global threat intelligence and intelligent security automation(Cisco, n.d.). IPS sit behind the firewall to complement it in filtering out dangerous content(Paloalto networks, n.d.). Actions by Intrusion prevention systems include:

a) Sending alarm to the administrator.
b) Dropping Malicious packets.
c) Blocking traffic from source address.
d) Resetting connection.

Differences between an IDS and IPS are given in Table 2.2. below;

Table 2.2. Differences between IDS and IPS

	IPS	IDS
Type of System	Active, i.e., monitor and defend	Passive, i.e., monitor and notify
Detection Mechanisms used	Statistical anomaly Signature detection Vulnerability facing signature	Signature detection
Position in the network	Inline	Outside communication line

(Simkin, 2017)

Intrusion detections and prevention systems' main agenda is to prevent and mitigate the damage that the detected intrusions could have caused as well as identify the perpetrator of the attacks or new attack patterns. For them to achieve this, they must have the necessary accuracy requirement to differentiate legitimate and illegitimate users.It must be able to carry real-time intrusion detection, be resistant to attacks and perform analysis in a short span of time(Gordeev, n.d.)

2.10.Intrusion detection mechanisms

2.10.1.Signature-based (SBIS)

This intrusion detection mechanism is based on what's known such as specific patterns like malicious instructions sequences malware use, byte sequences, packet headers and payload content which can imply intrusion into a system. It strives to find out the signature that is known. By this way, it takes the analogy of an antivirus software.(Chan, 2013). They are considered to be more accurate and easy to set up than their anomaly counterparts. Their accuracy is as good as their signature database in that if an attack is in the signature database an alarm will be raised if the same attack is in the live traffic packets. The event will be logged for further analysis. Signature-based IDS is as good as its database. If the database is not updated the IDS

becomes obsolete because hackers are always coming up with new mechanisms to bypass the IDS(Gong, 2003).SBIS is based on dictionaries of identifiable signatures which are continuously recorded and stored to keep the system updated(Paloalto networks, n.d.). Signature-based systems normally take a fingerprint of a known attack or malware which can be a byte sequence in a file or a cryptographic hash of a file and store them in a signature database for use in future detections(Zeltser, n.d.). The signature-based systems are well understood, fast, simple to implement and widely available (Cloonan, n.d.).

Significant research has been done on signature-based detection. An android based host-based IDS that takes advantage of android architecture on a system level is proposed by (Ghorbanian, 2013). They propose this system as a way of enhancing the access control of Android since it can monitor events, read and generate logs. Because of significant computational power and time required by signature-based detection, in addition to the rate of change of complexity of attacks, (Yassin, et al., 2014) propose an anomaly based data mining classifier based on naïve Bayes and random forests. The system detects anomalies and help in generating signatures for the IDS. Signature-based systems have been proposed to detect zero day attacks by (Holm, 2014). This is achieved by configuring Snort IDS with an old official rule set. It is able to detect 173 of 183 zero-day attacks. Regular expressions have also been proposed by (Badran, Ahmad, & Abdelgawad, 2008). They implement the system in field programmable gate arrays.Signature detection as used by intrusion prevention systems can be classified into two types:

a) Exploit-facing signatures- They help track individual exploits in the stream that trigger a unique pattern
b) Vulnerability-facing signature- They are broader in nature in that they target vulnerabilities in the target system. They help protect the systems from many types of attacks but it has a high likelihood of false positives(Paloalto networks, n.d.)

2.10.2 Anomaly-based

These IDS normally tends to come up with a statistical model of normal behavior. The model is trained with a lot of data to come up with an accurate model which enables it to alert immediately when an abnormality is detected(Bolzoni, 2009). Anomaly-based detections systems (ABIS) can detect abnormal behavior in the system based on either the number of packets, flows or bytes(Rejchrt, 2013).Anomalies can be caused by any of the following network behaviors: high packet rate, small or large packet size, TCP based anomalies like SYN, ACK, FIN and RST, Fan-in and Fan-Outs, amplification attacks, UDP and ICMP based attacks. Abnormalities can also be caused by host behaviors such as port scans, address scanning, Mass ICMP(Allot Communications, n.d.).

Anomaly-based intrusion detection systems have a potential to detect many myriad behaviors in the network because they take a more generalized approach to identify what a “normal” behavior is. Training the anomaly IDS with data helps it learn to differentiate normal from abnormal behavior thus its ability to detect new and even unknown attacks is enhanced (Gong, 2003).Anomaly-based systems are suited for attacks such as new exploits, stealthy attacks, variations of existing attacks (Gong, 2003). However, anomaly-based systems are prone to too many false positives as a result of the changing nature of networks. They also take long in analyzing, detecting and alerting due to the long duration in analyzing to determine if it’s an anomaly or not. According to DR Gong’s whitepaper;” Deciphering Detection Techniques: Anomaly-Based Intrusion Detection”, anomaly detection is more about measurable effects events rather than how the events are executed which is the primary emphasis of signature-based detection(Gong, 2003).Anomaly detection is more suited to these cases where measurable effects of events have weight than the order of execution of network events. These systems take samples of network traffic randomly and checks against the calculated “normal”. If the selected sample is outside the expected set parameters, the system takes action to deal with the situation(Paloalto networks, n.d.). Anomaly based systems come with disadvantages such as the difficulty in defining all rules. For example, in case of a protocol anomaly detection, all protocols

to be analyzed must be defined, implemented and tested for accuracy. This can be complicated by vendor implementation of the protocols(Foster, n.d.).

The are many anomaly detection studies in literature. These studies employ different techniques to be able to detect anomalies in networks traffic. Machine learning methods such as Naïve Bayes classifier is proposed by (Alaei & Noorbehbahani, 2017) who use it to propose an offline and online active learning system for anomaly detection. An hybrid method of k-Medoids Clustering and Naïve Bayes(Chitrakar & Huang, 2012) to act as an active learning intrusion detection system. A support vector machines and K-Medoids clusteringis proposed by (Chitrakar & Huang, 2012(a)) with the intention of improving on their earlier method that required too much data.t they substituted Naïve Bayes with support vector machines. An heterogeneous anomaly based intrusion detection system has been proposed by (Tran, Vo, & Thinh, 2017). They implement it on both a field programmable array and a graphical processing unit in order to utilize the parallel processing available. They use a back propagation neural network for intrusion detection. They also propose a deep learning anomaly detection system(Van, Thinh, & Sach, 2017(a)). They utilize KDDCUP99 dataset for their model training and testing. Last but not least, (Radoglou-Grammatikis & Sarigiannidis, 2017) propose a flow-based anomaly android device anomaly detection system that monitors the network and collects netflows. They use an artificial neural networkto detect for any intrusion. Lastly, clustering techniques have been proposed by (Nikolova & Veselina Jecheva, 2015). They utilize the flame algorithm for analysis of user activities through a host based intrusion detection system.Applicable areas of protection for anomaly detection include:

2.10.2.1.Crafted payload denial of service

This is when an attacker crafts IP packets which disguise as genuine. The attacker then uses the crafted packets to launch a denial of service attack which can throttle resources such as bandwidth, CPU cycles, Memory. Examples of these DOS include

IP stack crashing, and process table exhaustion. In this situation, an attack can be observed by degradation in the quality of service.(Gong, 2003).

2.10.2.2.Volume based DOS/DDoS

Volume-based attacks go beyond control channel with a high number of packets being flooded to the network by the attacker. Because of the ease with which packets can masquerade in sophisticated attacks so that they are indistinguishable, anomaly detection is more suited as it can detect these complicated attack patterns caused by DDoS(Gong, 2003). It includes UDP and spoofed packet attacks whose intention is to saturate the bandwidth of the attacked site(incapsula, n.d.). there are two stages in DDoS flooding attacks, recruiting the zombies and flooding the victim(Fengxiang & ABE, 2007).

a) TCP SYN flood control packet statistics
b) Volumes of TCP, UDP and ICMP traffic for UDP flooding attacks.
c) The Difference in request and response for packet counts, byte counts as well as the time difference between request and response.
d) Statistics on requests and responses for HTTP in a server DDoS using standard lookups.

2.10.2.3.Protocol and service ports misuse

When attackers try to use obsolete or protocol features that are not used can be an indicator of malicious activity. Installation of backdoors on well-known ports can also be classified as port misuse and can easily be detected by the anomaly detection system.(Gong, 2003).

2.11.Anomaly detection techniques

2.11.1.Protocol anomaly detection

This technique monitors anything abnormal to a protocol format and behavior with regards to its specifications. Through anomaly detection, some aspects of TCP/IP stack can be observed by either doing TCP reassembly or IP defragmentation to check for any unusual behavior especially in Network, transport and application layer by monitoring for any anomalies. Anomaly detection can unearth things like corrupt checksums, how long and consistent are the flags, unusual TCP segmentation overlapping as well as illegal TCP options and usage in transport and network layer protocol. (Gong, 2003). Statistical anomaly detection cannot create a regular network traffic statistics because they don't know how the underlying network functions(Gong, 2003). It is possible to have a protocol anomaly detection because of protocols are well described so determining normal use is very easy(Das, 2002). A protocol normal usage is defined clearly in protocol anomaly detection system. If the usage of the protocol outside the defined intersection, it is considered an anomaly(Lemonnie, 2001). Anomaly detectors also involves disassembly of protocols and checking if they conform to RFC standards. However, considering that many protocols are not designed exactly as specified in RFC, the anomaly detectors need to be more general and flexible as pointed out by Lemonnie.

Protocol anomaly detection is more efficient technique especially in high speed networks because the network traffic to be monitored is low and more static compared to signature detection. It's also capable of detecting unknown and new attacks. This is possible because its trained to know the correct usage which means anything out of specified is an anomaly(Kang, Kang, Oh, & Kim). In case of application layer protocol anomalies, the following anomalies can be observed:

a) Illegal use of commands
b) Running protocols or command on non-standard ports.
c) Unusually long or short field lengths normally an indicator of buffer overflows.

d) Illegal fields values and combinations.

2.11.2.Application payload detection

It is possible with in-depth application protocol analysis to differentiate the logical fields and be able to define behavior constraints between them. It needs a thorough understanding of the application semantics like the encoding for particular fields. Deep inspection can be useful when checking if there is shellcode in some of the fields thus can be used to detect buffer overflow and other exploit attacks that use shellcode. Wang et al describe an application payload detector as one that is suited for attacks that are intended to exploit vulnerabilities of a protocol or those probing or scanning (Wang & Stolfo, 2017). Modelling of the payload is proposed by (Vargiya & Chan, 2003) in which they argue that payload has significant information such as a stream of bytes which can help algorithms in detecting an anomaly. Anomaly payload detection defines a standard model of a payload and the slightest deviation from a normal is an anomaly(Pastrana, Torrano-Gimenez, Nguyen, & Orfila, 2014). Modelling of profile byte efficiency distribution and their standard deviation of the application payload is proposed by (Wang& Stolfo, 2004(a)). Mahalanobis distance is used during detection stage to compare with the precomputed model.

2.11.3.Statistical anomaly/Statistical DDoS

This mechanism uses statistical measures to capture a particular behavior of traffic. Activities such as TCP traffic can be monitored to learn how a normal transfer happens, i.e., the TCP 3-way TCP-handshake, data transfer, and termination of the connection. The timing between packets can also be recorded so that they can be referenced to whenever an attack occurs(Gong, 2003). To do away with false alarms, the system should also try to differentiate between assumed normal long-term learning against short-term spikes in a training environment so that it can operate smoothly in a typical normal environment.

Statistical anomalieshave a higher chance of detecting DDoS attacks as an abnormal behavior due to a burst of traffic whenever an attack occurs. A good statistical Anomaly system should be able to detect some other occurrences that affect traffic without necessarily being attacks such as a flash crowd condition that is harmless and maybe a sudden route change against a real-time Volume based DDoS(Gong, 2003). Statistical anomaly detection systems determine “normal" network activity and then all traffic that falls outside the scope of normal is flagged as anomalous (not normal). The longer the system operates the more it becomes accurate because the analysis is continuous. The Statistical anomaly systems learn from the data to help the make a “normal” pattern that helps improve their detection(Symantec, 2003). Whenever statistical approach is used, features are important. Right features must be fed into the anomaly detection system in order to get accurate detections(Goldberg & Shan, 2015).

Some of the DDoS events that can be detected by statistical technique include:

a) ICMP Flooding attack
b) UDP flooding attacks
c) TCP data segment attacks
d) UDP flooding attack

2.12.Distributed Denial of Service Attacks in SDN

DDoS of service attacks is a problem in today’s networks because of their frequency, ability to take services offline and the lack of effective techniques that can prevent distributed denial of services outright. DDoS attacks target various resources in the targets’ computers such as memory, bandwidth or computing power (Charalampos et al., 2004). Before conducting DDoS attacks, the attackers take over computers whose security is weak by either using methods such as port scanning, topology scanning or permutation scanning then use the compromised computers to spread malware by either using techniques like central source propagation, back-chaining, or autonomous propagation (Charalampos et al., 2004). Once the networked devices are taken over and controlled by the attackers, they can be converted into botnets of master zombies, slave zombies, and reflectors. This architecture enables the

attackersto control and command hundreds of thousands or even millions of networked devices spanning from local to different locations worldwide. The architecturehas helped the attackers control master zombies who in turn control slave zombies and some scenarios slave zombies are mastersof other slave zombies (Charalampos et al., 2004). With the increase in adoption of the internet of things (IoT), DDoS attacks frequency will increase in the future.This is because these devices are using default passwords and very few end users bother to change their default passwords. These unsecured are likely to be exploited by even novice attackers who can get ready made scripts from the internet. Some code scripts for Mirai, a tool that makes it easy to take over IOT devices and form botnets has been released on the dark web(Weise, 2016). According to (Jakob Spooner, 2016), SDN being a unique new networking technology, it has its specific DDoS security concerns such as:

a) Flow decision requests: this happens when a controller is flooded with flow decision requests until the controller computing resources are overwhelmed thus goes offline rendering the rest of the network useless especially when the hard timeout elapses.
b) Flow entry flooding: this is also possible when falsely crafted floors are directed at OpenFlow switches. The switches' hard timeout is set to be long rendering the switches full and cannot make forwarding decisions.
c) Hijacked controller: In software-defined networks, it is possible for an attacker to hijack a controller.

2.13.Previous Work on DDoS Detection and Mitigation in Software-Defined Networks

A system called OF-Guard is proposed by (Haopei Wang, 2014) to detect and prevent attacks that emanate from the controller to the OpenFlowswitches via the southbound API.AVANT-Guard system is suggested by(Shin, Yegneswaran, Porras, & Gu, 2013), aiming to manage switch flow tables to make them withstand overwhelming flooding traffic from the controller. It adds a connection migration module and an actuator module. It also modifies data plane modules to support some

specific features used to detect anomalous traffic from the controller via OpenFlow.Entropy is another method used to detect DDoS attacks in software-defined networks; (Mousavi & St-Hilaire, 2015) propose an entropy-based technique for detecting distributed denial of service attacks. They calculate the probability of an event happening in relation to all events. If the number of events is more than the normal events, then that event can be flagged as an anomaly. In this case, a DDoS attack.(Wang, Liu, Chen, Liu, & Mao, 2017)Propose a system called 'PERM-Guard' that controls the authentications and permissions with the use of identity management to enable the controller to verify the authenticity of flow rules. This will be able to help the controller deal with attacks on the OpenFlow switches by simply sending flowmod messages requiring the switches to do away with flows that are deemed as suspicious. (Berezinski et al., 2015) Also proposean entropy-based network anomaly detection system that where they came up with the entropy through the calculation of the probability of the mass function if a particular vector is in the source and destination address and port, packets and bytes, flow duration andthe out function. This technique is efficient in identifying attacks such as botnets coming from particular sources. It is however not very effective where attacks are from many sources because it's going to report a lot of false positives. They decide to create their dataset because of unavailability of legitimate datasets as well as the lack of proper labeling in available datasets. Building additional SDN modules has been suggested by (Luo et al., 2015). They suggest a module to query for statistics from the switches, module for port statistics, packet filtering module,location binding module as well as a binding module that is used to identify hosts in the network. This approach is efficient in distinguishing compromised hosts within the network which can be very effective in mitigating attacks. The real-time polling of statistics and port stats is essential since they can be used to calculate entropy for anomaly detection. However since some skillful attacks can be able to masquerade as real OpenFlow switches, this system comes up short as there should be a way of identification between the hosts and switches before communication.

2.14.Machine Learning For Intrusion Detection

Machine learning is the act of making computers to take actions without being explicitly programmed(Ng, 2017).according to SAS(SAS, n.d.) machine learning is a data analysis technique that designs models that use algorithms to learn from data enabling computers to predict accurately without being explicitly programmed. Machine learning can be classified as either supervised, unsupervised and reinforced learning.

2.14.1.Supervised machine learning

Supervised machine learning is applicable where the labels and feature of the training set are known but need to be accurately predicted in other unlabelled data. These algorithms use data that is labeled as for example ‘Anomaly or ‘Normal’ traffic as training data to make models which are used to predict class instances in the test dataset. To test the efficiency of an algorithm, the predicted outcomes are compared with real data(Le, 2016). Precision and recall can be used to evaluate the accuracy of the algorithms.

2.14.1.1.Decision trees

Decision tree algorithms actas a decision support tool. They model decisions and their consequences which enables researchers to explore the options available as well as the possible outcomes which in turn helps in forming a clear opinion of the risks of choosing some options as well as their rewards. It is not parameterized, unlike support vector machines.Used for both classification and regression.(Scikit-learn, n.d.) They aim to predict through decision rules learned from data features. It is a very simple algorithm that is easy to use, require minimal input in preparation and has the advantage of being able to work with both numerical and categorical data. However, they make some concepts very difficult to learn as a result of not being able to express them explicitly. There is also a chance of users creating overcomplex decision trees that do not generalize well well with the data at hand. They are also

very susceptible to small changes in the data being used which might result to overly complicated trees being generated leadingin those trees not generalizing well with the data.

The decision trees use entropy and information gain to design trees from the root node. Entropy is used to calculate the uniformity of a sample. Entropy is given by equation 2.1 below:

$$Entropy = -\sum_j p_j log_{2^{p_j}} \qquad (2.1)$$

Decision trees also use Gini index to minimize the probability of misclassification by unquantifying the amount of uncertainity in a single node. Its given by equation 2.2 below.

$$\text{Gini index} = 1 - \sum_j p_j^2 \qquad (2.2)$$

where $p_j$is the probability of class j.

Information gain is also used to gauge at how much a question reduces uncertainty in each particular node. It works to determine which attribute in a given set of training features is the most useful.
It is given by equation below

$$IG(D_p) = I(D_p) - \frac{N_{left}}{N_p} I(D_{left}) - \frac{N_{right}}{N_p} I(D_{right}) \qquad (2.3)$$

where I can be the gini index or entropy while Dp, $D_{left\ and}$ D_{right} are the parent, left and right child node datasets, N_{right} and N_{left} are child nodes of parent node Np (Hong, n.d.)

2.15.Unsupervised Learning

Unsupervised learning is useunlabelled datasets and the algorithms are tasked with finding the relationship between the variables in the dataset. They are algorithms that draw inferences from datasets which are inputs that have no labels(Mathworks, n.d.). Clustering is an example of a Unsupervised learning algorithm:

2.15.1.Clustering

Clustering is concerned with organizing unlabelled datasets in clusters that have similarities between them and are dissimilar from other groups. In clustering proximity measure either the dissimilarity or similarity measure is very important. The criterion function is also important as it affects the quality of the clustering(Ullman, Poggio, Harari, Zysman, & Seibert, 2014). Clusters can be evaluated using inter-cluster cohesion or inter-cluster separation.

2.16. Previous Work On Use of Machine Learning for Intrusion Detection

Because of the changing in nature of networks, they require an intrusion detection system that is dynamic and one that does not depend on signatures to detect as signature-based intrusion detection systems are as good as their signatures. The need for systems that can learn to differentiate a normal from an anomaly in relation to networks has forced a hand in the application of machine learning in network security especially in intrusion detection. Intrusion detection and prevention systems are facing intelligent attacks from sophisticated attackers who know how they can be able to cheat and bypass the IDS systems monitoring the systems. As a result, intelligent systems that can learn new attacks based on the model obtained by training it with relevant data are needed. Because of this need, machine learning (Sangkatsanee, Wattanapongsakorn, & Charnsripinyo, Practical real-time intrusion detection using machine learning approaches, 2011), (Khan et al, 2007),(Mukherjee & Sharma, Intrusion Detection using Naive Bayes Classifier with Feature , 2012), (Chung et al, 2012), fuzzy (Danane & Parvat, 2015), (Kadam & Deshmukh, 2016),(Kumar, Mangathayaru, & Narsimha, 2016), Self organizing maps(Braga,

Mota, &Passito, 2010), (Amanowicz, 2015), deep learning(Niyaz, Sun, & Javaid, 2016),(Tang, Mhamdi, McLernon, Zaidi, & Ghogho, Deep Learning Approach for Network Intrusion Detection in Software Defined Networking, 2016) have been proposed. Support vector machines have been used by(Khan et al., 2007) where they combine SVM and Dynamically growing a self-organizing tree for classification purposes. They also use hierarchical clustering analysis in a bid to reduce the amount of time for training SVM. A real-time machine learning approach is used by (Sangkatsanee et al., 2011)where they collect data features from network packet headers which are in turn classified using machine learning algorithms such as decision trees, Naïve Bayes, Bayesian network among others. They also have a post-processing module that helps deal with false positives and outliers. Naïve Bayes algorithm is used by (Mukherjee & Sharma, 2012) where they use feature selection algorithms such as gain ratio and information gain which reduces the features thus the accuracy of the Naive Bayes algorithm prediction. Deep learning is also a relatively new technique that has been proposed by many researchers as a way of dealing with intrusion detection. (Weiqing & Javaid, 2017) Propose a pox based system that collects traffic with a feature extractor and a classifier which will classify the flows as either attack or normal. Deep learning is also used by (Tang et al., 2016) where they create a network intrusion detection system in the controller which is based on Deep neural networks which they compare against conventional machine learning algorithms with mixed results.

As much as this proposed machine learning and soft computing based intrusion detection systems predict intrusions, they are prone to too many false positives thus rendering the systems unusable(Spathoulas & Katsikas, 2009). To do away with these false positives, systems have to be designed to solve this precision problem by reducing these false positives otherwise there will be many false alarms. Methods such as feature selection(Anusha & Sathiyamoorthy, 2016), (Guo et al., 2010) are proposed to help in doing away with redundant features which reduces the time, cost, and most importantly accuracy of the machine learning classifiers. In addition to these, some machine learning libraries have native tools for feature selection(Scikit-learn, 2017),(Performing attribute selection, n.d.). Feature selection is important

since it can help increase the precision of the various classifiers used in machine learning. For example, when using clustering, it can give two very different results on the same data if the number of features that include redundant features. SOM a neural network like unsupervised algorithm has extended features that help reduce false positives when used for intrusion detection(Alsulaiman et al., 2009). To highlight the importance of feature selection, (Braga et al., 2010), (Tahir, et al., 2015) use it in their bid to increase the precision of their intrusion detection classifiers.

CHAPTER 3. PROPOSED INTRUSION DETECTION SYSTEM

3.1.Insider and Outsider Attack Detection

Attacks occur from outside to the inside of a network, so anaccurate system that helps prevent external threats to a networkis needed. However, there are also insider threats to network security, and these type of attacks also need to be mitigated at all times because they can affect the performance of a network. In normal situations, packet-based intrusion detection system and deep learning based systems are the ones that are more effective in protecting networks from external threats. However, if these techniques are used for software-defined networks, they are going to consume a lot of network resources; something that is not available in software-defined networks. As a result of this, a flow-based intrusion detection system is used in this research to be able to detect the various types of intrusions into the SDN network. A flow based intrusion detection system is preferred since it's not resource intensive thus allowing implementation of a light-weight module on the controller that can detect whenever an intrusion, in this case, a distributed denial of service attack happens. It achieves this through the use of a machine learning algorithm that's trained with data extracted from the Canadian Cybersecurity Institute's modified KDD dataset(Canadian Institute for Cybersecurity, n.d.).

This chapter proposes a novel way that combines what has been discussed in the related literature in the previous chapter. The proposed system has the following components:

a) InformationLogger

This module is concerned with getting the necessary features that are used by our intrusion detection system. It is composed of two sub-modules.

Information logger is a pox controller module that probes for statistics from the OpenFlow enabled switches. It makes sure that switch statistics are collected after every predetermined. This module can be a controller application residing within the controller implying its running over the north-bound API or an application that runs on any hypervisor but with exclusive access to OpenFlow messages exchanged by the controller and the switches as well as the ability to probe an OpenFlow switch. This module sends stats_request to the switch and gets results per flow, port, queue, and table.the results come through the *flow_stats_response* from the switch. In our scenario, we used a per-flow situation for our intrusion detection system.A flow-based intrusion detection system is preferred because of the lightweight nature which is one of the objectives. Compared to a packet-based intrusion detection system, its light, fast and accurate. Packet-based intrusion detection system take a lot of network resources that are not available in software-defined networks. Deep packet inspection based IDS is avoided to make this system light.

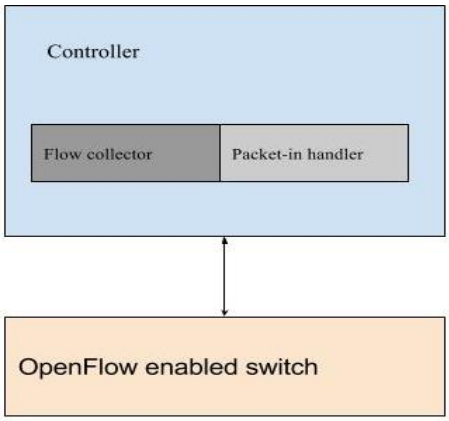

Figure 3.1 Information logger

b)Feature extractor

It is a part of a python pox controller module that receives the flows collected by the flow monitor and extracts them into a format that can be used for prediction by the trained model. It makes use of the same features used by Tang et al in addition to source IP, destination packets and destination bytes.The features used by Tang et al in addition to the destination packets and destination bytes are obtained from the flow statistics while the source IP is extracted from the packet in message. Source IP

is significant in that it enables us to mark particular flows and tract them. It also makes it possible to write a flow blocking that particular spoofed IP address that is flooding the controller through the FlowMod message from the controller. This source seven feature tuple is then sent to a machine learning model for further action. The features used by Tang et al are applied to a deep learning model with excellent results. Considering in machine learning important features are manually encoded and system maps inputs to output while in deep learning features are automatically discovered and acted upon to come up with the output, the features used by Tang et al are relevant in our study.

c) Flow IDS

It is the most critical module for our system which uses the extracted features from the flow extractor. The first phase of detection happens here where a supervised learning algorithm ensemble of KMeans clustering and random forests is used to analyze the flows. In case of an attack, the OpenFlow controller is instructed to send the packets that do the entropy calculation and count. The two algorithms are used together as an ensemble so that we can leverage on their core strengths in order to get a high detection accuracy.

KMeans algorithm involves dividing the data into clusters each having a centroid. The centroids should be placed far as possible from each other. The logic of this simple unsupervised learning algorithm is picking a point on a dataset and association it with the nearest centroid. This procedure is repeated until no any point is left out(MacQueen, 1967).

The objective function for K-Means algorithm is a squared error function:

$$\sum_{i=1}^{k}\sum_{j=1}^{n}\left(\left|\left|x_i - v_j\right|\right|\right)^2 = 1 \qquad (3.1)$$

Where $\left|\left|x_i - v_j\right|\right|$ is the Euclidian distance between a point, x1 and a centroid v_j iterated over K points in the i^{th} cluster for n clusters. In this set up the K-Means clustering algorithm is used to identify the most relevant patterns in the flows that

are extracted and then the ensemble based random tree classifiers are trained on those clusters produced by the K-Means clustering to be able to predict the anomaly accurately.

Random tree classifiers are an ensemble learning method that can be used both for classification and clustering. It is essentially an algorithm with many decision trees(forests) discussed in section (2.14.1.1). All input vectors are put down each of the trees whereby each tree classifies and every tree votes for a class. The forest chooses the classification with most votes. In this research random forest classifier model is applied to each of the clusters by K-Means clustering algorithm. Correlation and strength of trees can affect the error rate of a forest.

Because this system needs to learn a lot from the training dataset, a fixed algorithm is not used as an algorithm that has been trained from thousands of use cases is used so that it can be able to detect accurately in a real dataset.

d) Entropy calculator

Entropy module further works on the works in complementing the flow-based IDS in helping to detect if there is an attack on the network through an entropy calculation. An entropy algorithm is used to calculate the probability/chance of the features being in the flow. It monitors the threeparticular features of the TCP handshake, i.e., SYN, SYN-ACK and ACK to be able to calculate the entropy. Because all the features have an equal chance of appearing in a standard TCP connection, the probabilities are added together. Chance is the probability of any of the three features of the TCP handshake (SYN, SYN-ACK and ACK) appearing. Its calculated by summing the total of each TCP handshake feature divided by the cumulative total of all the three features. The sum of probabilities is used as the entropy value. Entropy is calculated as shown in equation 3.1 and 3.2 below.

$$Chance = \frac{total}{cum.\ feature\ count} \qquad (3.2)$$

Entropy-based on Shannon theory

$$Entropy = \sum_1^n chance * log2\left(\frac{1}{chance}\right) \quad (3.3)$$

Entropy is normal when the outcomes are equal, so whenever the equality outcomes are not equal for all instances, the entropy must go down at all times. This entropy can be a measure of surprise that we can be able to use to detect if there is an attack on the network. If the entropy detects an attack, it confirms the attack as the machine learning model did and thus forwards the source IP address that is flooding packets to the packet counter to enumerate the number of sync packets.

e) Packet counter

When the machine learning based IDS and the entropy determines that it's an attack this moduleenumerates the number of SYN packets per second coming from a flagged flow from a particular IP address and we put a ceiling of 50 packets per second which if the ceiling is exceeded, a FlowMod command is sent by the controller that installs a flow rule that blocks that particular IP address that is flooding packets to the end host. This module is so significant in our research since it enables us to be able to mitigate the DDoS attack. SYN flooding DDoS is used. SYN flooding is a DDoS technique that forges non-existent source addresses which results in the recipient sending the SYN-ACK to the source address which is non-existent resulting in the TCP handshake not being finished because the destination will continue sending SYN-ACK till the resources are overwhelmed, and the network goes offline.

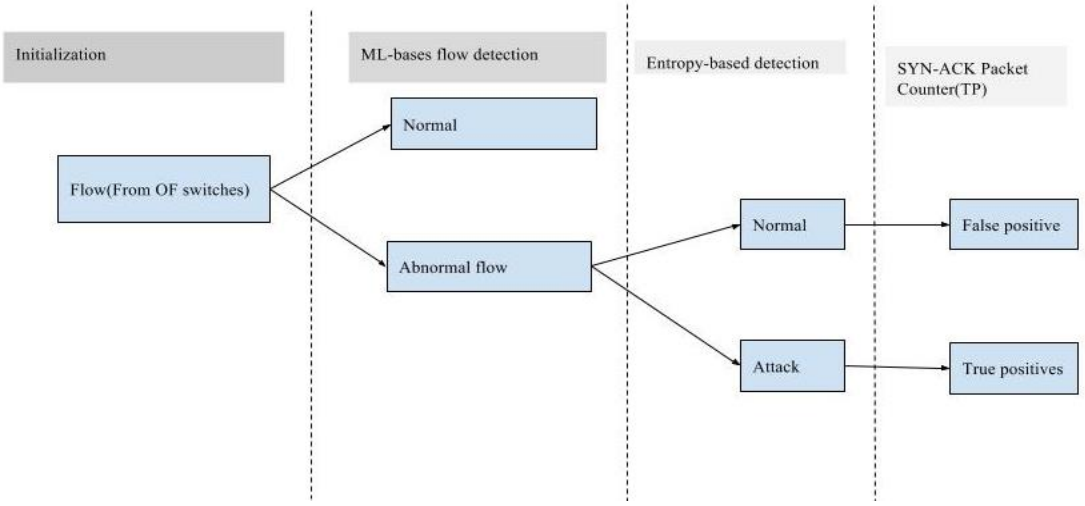

Figure 3.2. Working order of the proposed system

3.3.How The Proposed System Works

This system works when it requests the controller to send a flow_stats_reply message which the controller gets by sending a flow_stats_req to the OpenFlow switch. The pox controller module requests stats in a predetermined time of 5 seconds as used by Mousavi et al.(2015) who determined that that window as the most suitable to enable detection of attacks earlier enough to protect the controller. This module also works with the FlowRemoved message which is a message that is sent by the switchtelling the controller that flow entry has been removed either due to a hard timeout or idle timeout. The FlowRemoved message is only sent if the OFPFF_SEND_FLOW_REM flag is set especially in the FlowMod message. The FlowRemoved message can also result from removing of flow entries using FlowMod message from the controller. FlowRemoved happens due to a timeout, which maybe as a result of idle timeout i.e., when no matching packets over a period or hard timeout that arises when a particular specified time elapse.

After initialization between the switch and the controller, the switch sends stats requests made by the controller on behalf of the information logger. The information logger sends them to the machine learning based flow anomaly detection system which uses a well-trained classification algorithm model to be able to differentiate malicious flows from normal flows. If the ML flow detector determines a flow as

normal, nothing is done otherwise the system requires a further entropy calculation be done on those flows.

Those packets that are flagged as malicious by the machine learning based flow IDS are subjected to further inspection through the use of entropy which is calculated as describe in section 3.1(iv). Entropy is considered to be very accurate when checking anomalies something that it will help in complementing the machine learning based anomaly detection algorithm.Those packets deemed attacks by both the machine learning module and the entropy module have their SYN packets enumerated and when a set threshold is reached a FlowMod message is sent from the controller to block that flooding IP address.

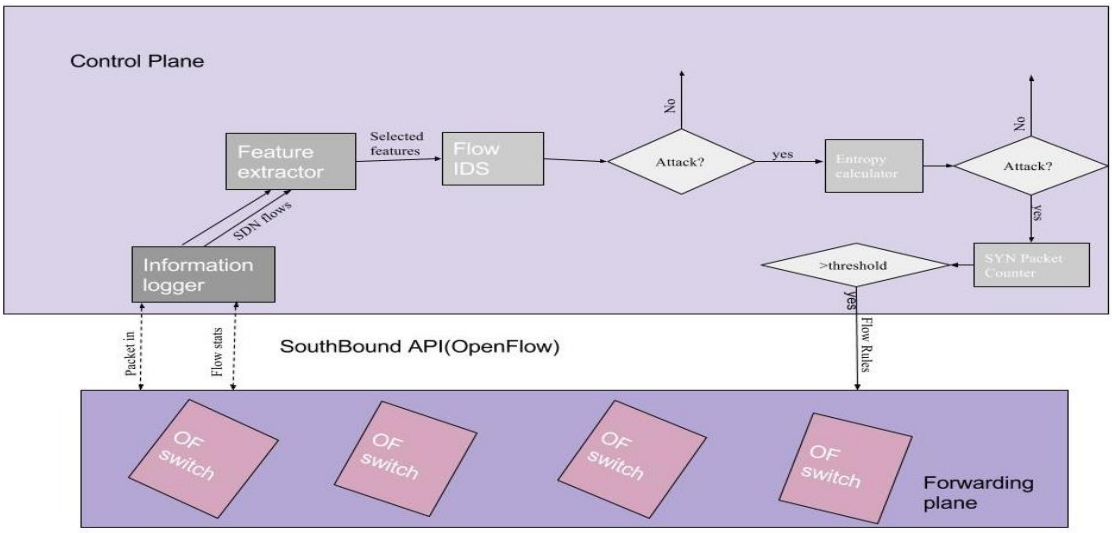

Figure 3.3: Proposed System

CHAPTER 4. SIMULATION, RESULTS, AND EVALUATION

4.1.Testbed Setup

In the following sections, the simulation and experimentation are going to be described. The detection algorithm is implemented in in pox controller which is a fork of NOX controller and is python based thus suitable for prototyping and research purposes since python is easy to learn and also cuts the development time. Mininet is used for providing a virtual network setup that enabled us to develop a virtual network to test our algorithm. We used hping application for network traffic generation.

4.2. Testbed Implementation

POX controller that supports OpenFlow 1.0 protocol is used for implementation. POX controller was preferred because it supports python which is easy to learn and supports fast prototyping. This advantages of python over languages like Java make POX controller appropriate for research purposes. An intrusion detection system is developed as a POX controller application with various modules that process OpenFlow messages exchanged between the switches and the controller and do the processing as explained earlier in the chapter. The application runs in conjunction with the l3_switch component of the pox controller. Errors are logged using the log level option in case of errors popping up while the application is being launched using.

In this prototype application, in order to detect anomalies using the flow based detection, a classification algorithm random forests are trained using Scikit-learn open source python machine learning package.

4.3.Pox Controller

The controller determines how fast and efficient a solution is. Many controllers are available on open source licenses which means that they are usable for free. Controllers such as floodlight and open daylight are available for use with fantastic support, but their learning curve is a bit steep especially considering that they use Java as their development language. The Opendaylight controller is production ready while floodlight controller has great community support as well as support for REST APIs. However, in this research, POX controller is preferred over other controllers because of its relatively straight learning curve. It uses python as a programming language which is easy to learn and being a dynamically typed language. it also cuts down development time by a large percentage. POX controller is also preferred due to the availability of stock components that are available with the controller so no need to redevelop it. It is also used widely in academia for SDN research which emphasizes its quality and applicability in research.

4.4. Mininet

For network simulation purpose, Mininet is usedto simulate thesoftware-definednetwork. Mininet enables creating of virtual networks even on desktop PCs or laptops. In Mininet it’s possible to create virtual switches and hosts and run them through a process based virtualization through the use of kernel namespace, i.e., Linux network namespaces. The use of kernel namespaces enables it to create a new network namespace for each host created in the virtual network. Mininet starts the controller and open-vSwitch processes in virtual machine root namespace by default. Mininet also supports the creation of different topologies for testing of the network experimenting in software-defined networks. Mininet facilitates the rapid prototyping whenever you are working with software-defined networks. It works as a container orchestration tool for emulation of software-defined networks. Starting to use Mininet is as easy as typing the command mn.

Advantages of Mininet

a) Its fast- takes a few seconds to start a network.
b) It’s easy to create custom topologies ranging from a single switch to the data center or back-bone topologies.
c) Can run on a laptop like in this case
d) Its open source
e) Can use customized packet forwarding

Mininet limitations

a) You can’t run software that depends on other operating system kernels since it uses a single Linux kernel for its virtualization of hosts.
b) Mininet hosts share host file system and process IDs spaces thus need for extra care when running daemons requiring configuration of /etc.
c) Configuration and programming the controller is done by the researcher and not by Mininet.

4.5.Traffic Generation

Intrusion detection insoftware-defined networks is being simulated. hping is used for Generation of traffic. Hping is a command-line TCP/IP tool that can be used for packet assembly as well as packet analysis. It works with UDP, TCP, ICMP and RAW-IP protocols. It can be put to many uses such as firewall testing, TCP/IP stack auditing, remote uptime guessing and OS fingerprinting, network testing and even for port scanning.

In traffic generation,the threading module is used so that parallelism can be employed in the module which will enable running many threads at the same time. The random module is also used especially the randInt function of the Random module which is used to generate spoofed IP addresses for non-existent nodes which are used to flood the targets. The first octet has a range from 0-10, the second, third

and fourth have a range from 0-255. OS module is also used to enable hping to set the destination IP, port as well as the destination port and even set the SYN TCP flag.

4.6.Setup of The Network

The simulation was conducted on an HP Envy 13 core i5-6200U CPU, 2.4 GHz with 8 GB RAM. Mininet is used for network emulation as indicated in the diagram below; tree network topology is used to a depth of one and a fan out of 15 hosts. Tree topology is chosen because of its popularity as well as its ease of implementation. It is also the most commonly used topology in data centers.

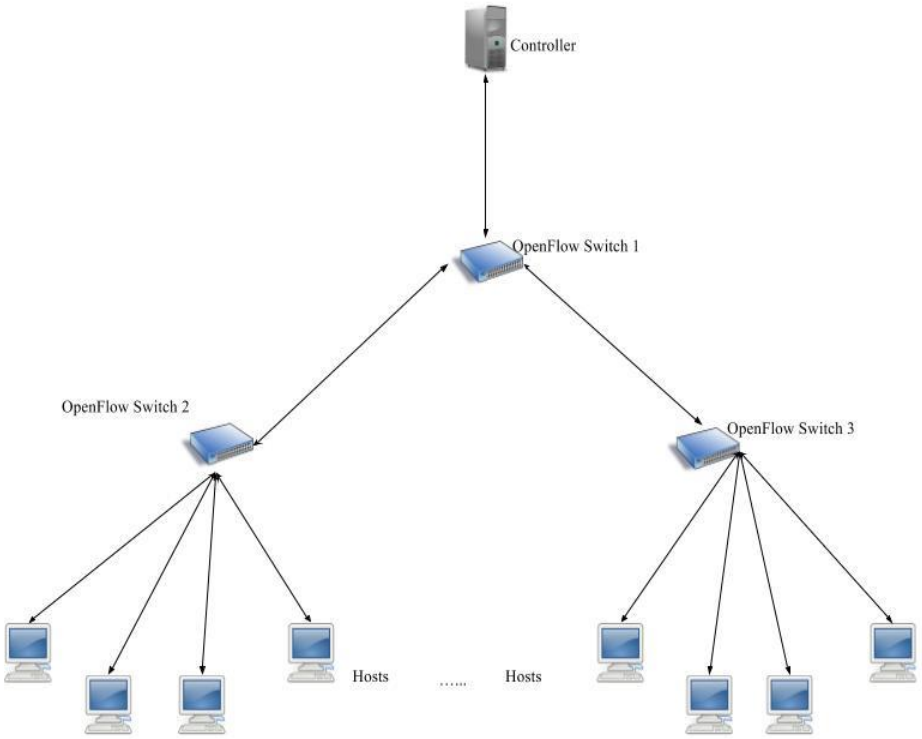

Figure 4.1. Testbed Setup

In the test bed, a total of 15 hosts attached to their respective OpenFlow switches which are in turn connected to a pox controller via a secure channel.

4.7.Dataset

In the simulation, the NSL-KDD dataset(Canadian Institute for Cybersecurity, n.d.) from Canadian Institute of Cybersecurity is used in the training of a machine learning model to help the flow based detection module. The NSL KDD dataset has five attack categories i.e.

a) DoS is the denial of service attack which includes smurf, Neptune, teardrop, pod, land, and bak.
b) R2Lis an unauthorized access to a remote machine which includes warezmaster, warezclient, guess_passwd (guess password), imap, ftp_write,spy, multihop, dict, and phf.
c) U2R includes the following (perl-magic, buffer overflow, eject, rootkit, ffb, loadmodule, perl). It normally encompasses unauthorized access to local superuser privileges.
d) Probe encompasses portsweep, satan, nmap, andipsweep. It can also involve things like port scanning which can be recognized in the system by use of programs such as Nmap.

The advantages of the Canadian Institute Cybersecurity KDD dataset over the KDD 99 dataset include the following:

a) Removal of redundant records in the training datasets which reduces biased results towards the frequency of records.
b) Reduction in the numberof records in the testing and training datasets which enabled the use of all records instead of portions of the record. This has enabled comparison of the results from different intrusion detection researchers using data.
c) Test datasets are duplicate free thus the performance of the learning models is not influenced by the frequency of records.
d) Availability of testing and training datasets in various formats i.e. text files, Weka files (arff) and PCAP formats which makes it easier to use in a different scenario.

4.8.Model Building

For the flow based detection module to function a model has to be built that will enable it to recognize an attack from a normal flow. To achieve this, KDD dataset already divided into train and testing data is used to train the model to enable it to detect attacks accurately.

4.8.1.Data loading, exploration

Count of total labels in two generalized categories either as an attack or normal is indicated in Table 4.1. below.

Table 4.1. Count of individual attack classes in training dataset

Label	Count of label
normal	67343
Attack	58630

Count of labels based on individual attack classes is indicated in Table 4.2. below.

Table 4.2. Count of individual attack classes in training dataset

Attack class Label	Count
normal	67343
DoS	45927
probe	11656
R2L	995
U2R	52

4.8.2 Test Dataset

Count of normal and attack labels in the dataset is indicated in Table 4.3. below.

Table 4.3. Count of normal and attack labels in test dataset

label	Count of label
normal	9711
attack	12833

Count of labels for each attack classes in the test dataset is shown in Table 4.4.

Table 4.4. Count of individual attack classes in Test dataset

Attack class Label	Count
normal	9711
DoS	7458
probe	2754
R2L	2421
U2R	200

By probing the data further,the attack and normal traffic for every protocol in the NSL KDD dataset can be obtained as indicated in Table 4.5.below:

Table 4.5. Protocol feature count in attack and normal dataset

protocol	Attack	normal
icmp	6982	1309
tcp	49089	53600
udp	2559	12434

4.8.3.Feature Selection

While analyzing the data that is used to design and optimize the model, it is observed that some features do not have a significant contribution to the model in that they even affect the model accuracy especially when the insignificant features are many. To avoid inconsistent results that come as a result of using too many features as found out by(Varma, Kumari, & Kumar, 2016), attribute ratio as proposed by (Chae & Choi, 2014) for feature selection is used. The first few features as obtained by applying attribute selection are listed from the most significant in descending order in the Table 4.6. below.

Table 4.6. Most significant features as selected using attribute ratio

Feature	Values
protocol_type_tcp	1000,0
num_shells	326,11353550295854
urgent	173,03983516483518
num_file_creations	62,23362492770388
flag_SF	51,0
num_failed_logins	46,03855641845592
hot	40,77451681709518
logged_in	10,569767441860465
dst_bytes	9,154854355343401
src_bytes	8,464064204948945
duration	7,225829157212557
dst_host_srv_diff_host_rate	5,756880682756574
dst_host_diff_srv_rate	4,83734184897426
num_access_files	4,694879248658319
dst_host_same_src_port_rate	4,393080378884017
num_compromised	4,338539274983927

Feature selection is important especially in this research because if some of the fields or all the fields in the dataset are used for training and testing, they are likely going to result in overfitting which will affect the accuracy of the model for predicting anomaly from normal traffic(Dash & Liu, 1997). Having too many features some of which have minimum effect on model accuracy while the number of training and test cases is small is likely to cause overfitting. This is because it increases the sampling error and reduces generalization ability of the model in question. Some of the fields in the list such as the *flag_SF, num_failed_logins, hot, logged_in* are not so significant to the investigation thus feature selection is applied to make sure accuracy of the model.

K-means clustering and random forest classification algorithms are used to create a model in this research. To avoid overfitting,K-fold cross validation can be used. It involves:

a) Splitting dataset into K equal partitions/folds
b) Use fold 1 as testing set and the union of the other folds as training set.
c) Calculating the test accuracy
d) Repeating step 2 and 3 K times using a different fold as testing each time
e) Use the average testing accuracy as estimate of out of sample accuracy

As indicated in Figure 4.2. a substantial number of features are needed for correct classification. However, there is a number of features optimum for the accuracy of the model beyond which no significant effect on the model accuracy is observed.

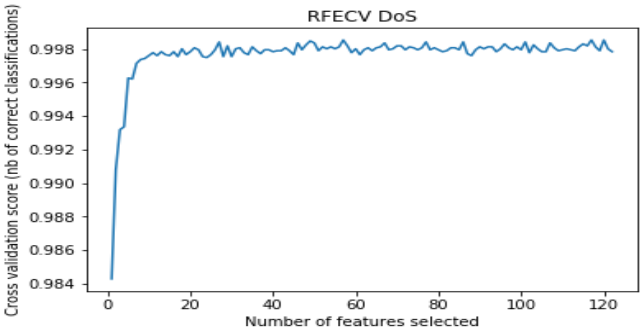

Figure 4.2. Recursive feature elimination by cross-validation

The data is split into 8 clusters with each cluster having both “attack” and “normal” connections and random forest classifiers are trained for the different clusters. Numeric features are used because KMeans does not handle binary features properly.

4.9.Model Evaluation

When evaluating the accuracy of the model, several measures such as precision, recall, and F1-score are used. Because the training data and the test data is from different distributions, the probability thresholds are adjusted by 0.01 for normal and 0.99 for the attack to enable the detection of new attack types which gives a 99% detection rate and around 13-14% False positives.

Model validation techniques are calculated as shown below:

a) Accuracy

$$\text{Accuracy} = \text{TP} + \text{TN}/((\text{TP} + \text{FN}) + (\text{FP}+\text{TN})) \quad (4.1)$$

b) F1 score

$$F1 = 2 * TP/(2 * TP + FP + FN) \quad (4.2)$$

c) False negative rate

$$\text{FN}/\ (\text{FN}\ +\ \text{TP}) \quad (4.3)$$

Where:

True Positive (TP): True is detected as True.

True Negative (TN): False is detected as False.

False Positive (FP): True is detected as False.

False Negative (FN): False is detected as True.

4.9.1. Detection of normal from attack

Table 4.7. Confusion Matrix of model attack detection

	Normal	attack
Normal	8262	1449
attack	182	12651

Accuracy = 0,927653

AUC =0, 918303

False Alarm Rate = 0,149212

Detection Rate = 0,985818

F1 score = 0,939442

Table 4.8. Prediction Model accuracy measurements

	precision	recall	F1 score	count
0.0	0,98	0,85	0,91	9711
1.0	0,90	0,99	0,94	12833
average	0,93	0,93	0,93	22544

Getting a model to predict an individual attack is possible by using a supervised machine learning approach that will differentiate just the denial of service attack from normal flows. Random forest classifiers were trained for each attack types and specifically for Denial of service attacks as shown in Table 4.9. The random forests are chosen because of its unmatched in accuracy when compared to other algorithms, its efficiency on large datasets and helps estimate the most important features in classification, it can estimate accurately missing data while maintaining accuracy when lots of data is missing (Breiman & Cutler, n.d.).

4.9.2. Dos and normal prediction by random forest IDS

Table 4.9. Prediction for DoS using random forest classifier

	normal	Dos
normal	9625	86
DoS	1656	5802

Accuracy = 0,898538

AUC = 0, 88455

False Alarm Rate = 0,00885594

Detection Rate = 0,777957

F1 score = 0,869474

Table 4.10. DoS Model Accuracy Measurements

	precision	recall	F1-score	support
0,0	0,85	0,99	0,92	9711
1,0	0,99	0,78	0,87	7458
average	0,91	0,90	0,90	17169

4.9.3. Probe and Normal detection using a random forest classifier

Probe detection using a random forest model is indicated in Table 4.11. below.

Table 4.11. Prediction of Probe using random forest classifier

	normal	probe
normal	9493	218
probe	946	1475

Accuracy = 0,855216

AUC = 0,856607

False Alarm Rate = 0,133354

Detection Rate = 0,846567

F1 score = 0,869398

Table 4.12. Probe Model Accuracy Measurements

	precision	recall	F1-score	support
0,0	0,81	0,87	0,84	9711
0,1	0,89	0,85	0,87	12833
total	0,86	0,86	0,86	22544

From the above models, the best result was obtained by KMeans clustering used with random forest classifiers which have approximately 98% detection and a low 13-14% false alarm rate. The above model was used as part of the flow IDS that acted on the network information collected by the flow logger and managed to detect anomalies in the data which are further subjected to entropy calculation. This high detection rate is obtained despite decision tree classifiers needing labeled data.

On the other hand, the Random forest model applied to the data has a relatively lower detection rate unlike the previous ensemble of KMeans clustering and random classifiers which posted the best results of any models. This accuracy attributed to the ability for KMeans clustering to take care of missing values especially in fields like protocol type which if one protocol is used there is a chance of missing values when multiple protocol field are used.The DoS and Probe attacks have the highest detection by the algorithm.

4.10.Model Detection and Mitigation in Real Time

As discussed in the previous chapter, the model in section 4.7. composed of KMeans clustering and random forest classifiers is applied directly to the real-time data from the feature extractor described in section 3.2.2.

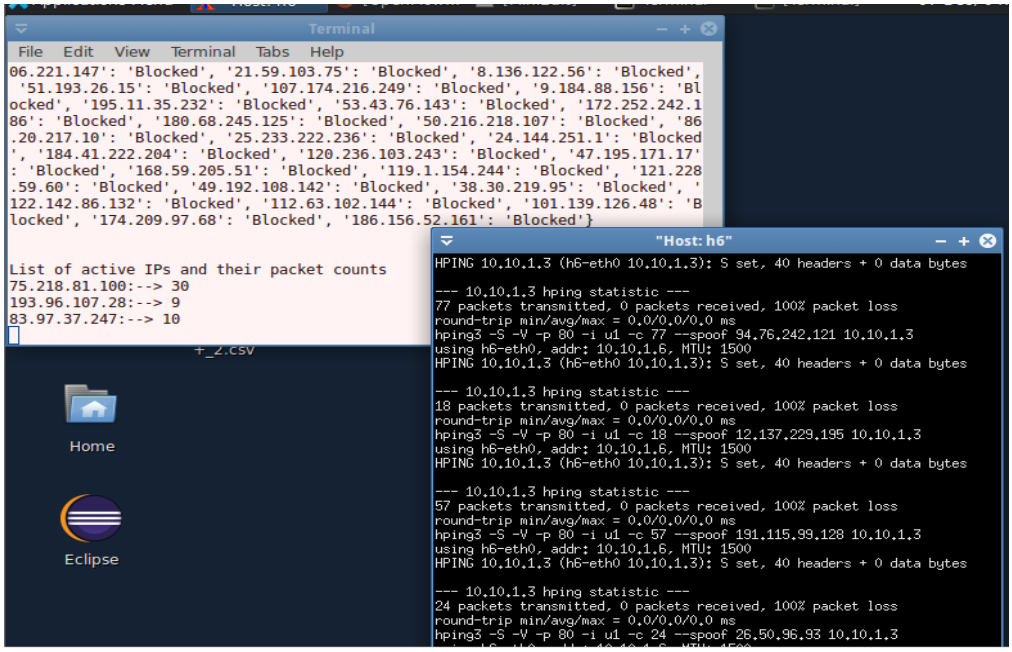

Figure 4.3. Results of simulation indicating real-time blocking of flooding IP addresses by the proposed system

Figure 4.3. shows the count of SYN packets for specific spoofed IP addresses. This count is important because when a count of 50 SYN packets is reached, then a flow is installed to block those IP addresses that are flooding the controller. Figure 4.3. also indicates the list of blocked spoofed IP Addresses. The threshold of 50 packets was arrived at from the work of (Mousavi & St-Hilaire, 2015).

In trying to test the effectiveness of the solution, the attack traffic is increased progressively to determine how much the proposed system can withstand. The system handles well until when the attack crosses 50% when the system reports that the cache is full although the system manages to thwart the attacks and prevent the controller from being knocked offline.It's observed that when traffic reaches 75% a buffer error is displayed but the application can mitigate the attack. This means that hard and soft timeout were adjusted appropriately to deal with a scenario where the amount of traffic is so high like from a botnet or when the number of attackers targeting the controller is very high.The effectiveness of the mitigation method is tasted by the simulation being run while one,two,three,four and five hosts run traffic generation code and note how long it takes for the mitigation script running on the controller detect the particular Fake flooding IP addresses.

The achievements of this research include the meeting of the objectives described in chapter one in that we were able to design a lightweight script that can protect the controller from distributed denial of service attack with minimal resource usage and the system has high accuracy in detection of the attacks in real time as demonstrated in Figure 4.3.

4.11. Conclusion and Future Work

While SDN is changing the networking industryand a promisingfuture technology it currently has mostly research applications and very few industry applications by the major players. With the control plane of the SDN network resting in the controller, it becomes so easy for attackers to target the controller and knock it offline thus render the network unusable. The SDN concept increases the number of attacks that can be targeted at the network, unlike the traditional networks.Protecting the controller by detecting and mitigating against any intrusion was the objective of this research by adding a module that runs on minimum computer resources because SDN is resource starved.

In this research, we utilizea hybrid approach to detect attacks. We made sure that the solution that we applied was suited to SDN based on its specification by utilizing the ability of the controller and the fact that all packets are sent to the controller if they do not match. We also utilized the flow statisticswhich is unique to SDN to help in detecting anomalies. Because SDN does not have enough resources, we use a flow-based intrusion detection system based on machine learningwhich is pre-trained and has a detection accuracy of 98-99%. Because of many false positives involved with machine learning detection, entropy calculation is meant to complement the flow IDS. When the two methods were used together, they enabled the design of a module that ensures the controller can be able to detect and mitigate most of the attacks. With detection accuracy of 99% and false positive rate of 14%, the proposed algorithm is considered accurate and effective as shown with its effectiveness when applied to real-time data on the network.

4.12. Future Work

With false positives of 14%, it’s considered too much when creating a model. Ensemble learning is recommended as it will bring two sides of different algorithms which can help reduce false positives and false negatives. In our solution, entropy will be challenging to calculate when the whole SDN network is being targeted since it can change by a relatively large margin. It is recommended using a system that uses varying thresholds to cater for the different set of attacks.

REFERENCES

Allot Communications. (n.d.). *Network Anomaly Detection*., http://www.allot.com: http://www.allot.com/technology/network-anomaly-detection/Access Date:08.09.2017

Alsulaiman, M. M., Alyahya, A. N., & Alghafis, R. A. (2009). Intrusion Detection System Using Self-Organizing Maps. *Third International Conference on Network and System Security* (pp. 397-402). Gold Coast: IEEE.

Amanowicz, D. J. (2015). Intrusion Detection in Software Defined Networks with Self-organized Maps. *Journal of telecommunications and information technology*.

Anusha, K., & Sathiyamoorthy, E. (2016). Comparative study for feature selection algorithms in intrusion detection system. *Automatic Control and Computer Sciences*, 1-9.

Badran, T. F., Ahmad, H. H., & Abdelgawad, M. (2008). A Reconfigurable Multi-Byte Regular-Expression Matching Architecture for Signature-Based Intrusion Detection. *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications* (pp. 1-4). Damascus: ICTTA.2008.

Beaver, K. (n.d.). *Host-based IDS vs. network-based IDS: Which is better?* Techtarget:http://searchsecurity.techtarget.com/answer/Host-based-IDS-vs-network-based-IDS-Which-is-better. Access Date:09.08.2017.

Berezinski, P., & Szpyrka, B. J. (2015). An Entropy-Based Network Anomaly Detection Method. *Entropy*, 2367-2408.

Berge, M. (n.d.). *what-is-intrusion-detection*. www.sans.org: https://www.sans.org/security-resources/idfaq/what-is-intrusion-detection/1/1, Access Date: 07.12.2017.

Bolzoni, D. (2009). *Revisiting Anomaly-based network intrusion detection systems.* Enschede: Wohrmann Print Service.

Braga, R., Mota, E., & Passito, A. (2010). Lightweight DDoS Flooding Attack Detection Using NOX/OpenFlow. *IEEE Conference on Local Computer Networks* (pp. 408-415). Washington DC: IEEE.

Breiman, L., & Cutler, A. (n.d.). *Random Forests*. https://www.stat.berkeley.edu/: https://www.stat.berkeley.edu/~breiman/RandomForests/cc_home.htm, Access Date: 07.20.2017.

Canadian Institute for Cybersecurity. (n.d.). *NSL-KDD dataset*. http://www.unb.ca: http://www.unb.ca/cic/datasets/nsl.html, Access Date: 08.15.2017.

Chae, H.-s., & Choi, S. H. (2014). Feature Selection for efficient Intrusion detection using attribute ratio. *International Journal of computers and communications*, 22-28.

Charalampos Patrikakis, M. M. (2004). Distributed Denial of Service Attacks. *Internet protocol journal*, VOLUME 7, NUMBER 4.

Chitrakar, R., & Huang, C. (2012). Anomaly based Intrusion Detection using Hybrid Learning Approach of combining k-Medoids Clustering and Naïve Bayes Classification. *2012 8th International Conference on Wireless Communications, Networking and Mobile Computing* (pp. 1-5). Shaghai: WiCOM2012.

Chitrakar, R., & Huang, C. (2012(a)). Anomaly detection using Support Vector Machine classification with k-Medoids clustering. *2012 Third Asian Himalayas International Conference on Internet* (pp. 1-5). Kathmandu: AHICI.2012.

Chung, M., Cho, J., & Moon, J. (2012). An effective denial of service detection method using kernel based data. *IEEE Symposium on Computational Intelligence in Cyber Security*, 9-12.

Cisco. (n.d.). *Cisco IOS Intrusion Prevention System (IPS)*. Cisco: https://www.cisco.com/c/en/us/products/security/ios-intrusion-prevention-system-ips/index.html, Access Date: 09.10.2017.

Cisco. (n.d.). *Next-Generation Intrusion Prevention System*. www.cisco.com: https://www.cisco.com/c/en/us/products/security/ngips, AccessDate: 09.10.2017.

Cloonan, J. (n.d.). *Advanced Malware Detection - Signatures vs. Behavior Analysis*. www.infosecurity-magazine.com: https://www.infosecurity-magazine.com/opinions/malware-detection-signatures/, Access Date: 17.06.2017.

Cole, A. (2013). *SDN and Legacy Network Infrastructure* . enterprisenetworkingplanet: http://www.enterprisenetworkingplanet.com/datacenter/sdn-and-legacy-network-infrastructure.html, Access Date: 09.10.2017

Danane, Y., & Parvat, T. (2015). Intrusion Detection System using Fuzzy Genetic algorithm. *IEEE International Conference on Pervasive Computing (ICPC)*, (pp. 1-5).

Das, K. (2002). *Protocol Anomaly Detection for Network-based Intrusion Detection.* SANS Institute .

Dash, M., & Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis*, 131-156.

Fausett, L., & Kohonen., T. (2017). *Kohonen Self-Organizing Maps*. Access Date:15:08:2017, from mnemstudio: http://mnemstudio.org/neural-networks-kohonen-self-organizing-maps.htm

Fengxiang, Z., & ABE, S. (2007). A Heuristic DDoS Flooding Attack Detection Mechanism Analyses based on the Relationship between Input and Output Traffic Volumes . *16th International Conference on Computer Communications and Networks*, 798 - 802.

Foster, J. (n.d.). *IDS: Signature versus anomaly detection*. http://searchsecurity.techtarget.com: http://searchsecurity.techtarget.com/tip/IDS-Signature-versus-anomaly-detection, Access Date:15.07.2017.

Ghorbanian, M. (2013). Signature-Based Hybrid Intrusion detection system(HIDS) for android devices. *2013 IEEE Business Engineering and Industrial Applications Colloquium (BEIAC)* (pp. 827-631). Langkawi: BEIAC.2013.

Goldberg, D., & Shan, Y. (2015). The Importance of Features for Statistical Anomaly Detection . In *7th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 15.* Santa Clara, California: USENIX Association.

Gong, D. F. (2003). *Deciphering Detection Techniques: Part II Anomaly-Based Intrusion Detection.* McAfee Security.

Gordeev, M. (n.d.). *Intrusion Detection: Techniques and Approaches*. http://www.forum-intrusion.com: http://www.forum-intrusion.com/archive/Intrusion%20Detection%20Techniques%20and%20Approaches.html, Access Date:08.07.2017.

Guo, Y., Wang, B., & Zhao, X. (2010). Feature selection based on Rough set and modified genetic algorithm for intrusion detection. *International Conference on Computer Science & Education* (pp. 1441-1446.). Hefei: IEEE.

Haopei Wang, L. X. (2014). OF-GUARD: A DoS Attack Prevention Extension in Software-Defined Networks. *Usernix*.

Hardesty, L. (2017). *Google Brings SDN to the Public Internet*. sdxcentral: https://www.sdxcentral.com/articles/news/google-brings-sdn-public-internet/2017/04/, Access Date:09.05.2017.

Hinden, R. M. (2014). SDN And Security: Why take over the hosts while you can take the whole network. *RSA conference: Capitalizing on collective intelligence.* San Francisco.

Holm, H. (2014). Signature Based Intrusion Detection for Zero-Day Attacks: (Not) A Closed Chapter? . *2014 47th Hawaii International Conference on System Science* (pp. 4895-4904). Waikoloa: HICSS.2014.

Hong, K. (n.d.). *scikit-learn : Decision Tree Learning I - Entropy, Gini, and Information Gain*. http://www.bogotobogo.com: http://www.bogotobogo.com/python/scikit-learn/scikt_machine_learning_Decision_Tree_Learning_Informatioin_Gain_IG_Impurity_Entropy_Gini_Classification_Error.php, Access Date: 20.08.2017

incapsula. (n.d.). *DDoS Attacks*. incapsula.com: https://www.incapsula.com/ddos/ddos-attacks/, Access date:20.06.2017.

Jakob Spooner, D. S. (2016). A Review of Solutions for SDN-Exclusive Security issues. *International Journal of Advanced Computer Science and Applications, Vol. 7, No. 8,* , 113-122.

Kadam, P. U., & Deshmukh, M. (2016). Real-time intrusion detection with genetic, fuzzy, pattern matching algorithm. *IEEE International Conference on Computing for Sustainable Global Development* , 753-758.

Kang, D.-H., Kang, J. H., Oh, J.-T., & Kim, K.-Y. (n.d.). *Application protocol based anomaly detection for high speed network.* Daejeon: Electronics and Telecommunications Research Institute Electronics and Telecommunications Research Institu.

Khan, L., Awad, M., & Thuraisingham, B. (2007). A new intrusion detection system using support vector machines and Hierachical clustering. *The VLDB Journal*, 507 521.

Kumar, G. R., Mangathayaru, N., & Narsimha, G. (2016). An Approach for Intrusion Detection Using Fuzzy Feature clustering. *International Conference on Engineering & MIS (ICEMIS)*, 1-8.

Landress, A. D. (2016). A hybrid approach to reducing the false positive rate in unsupervised machine learning intrusion detection. *SoutheastCon* (pp. 1-6). Norfolk: IEEE.

Le, J. (2016). *KDNuggets*. The 10 Algorithms Machine Learning Engineers Need to Know: http://www.kdnuggets.com/2016/08/10-algorithms-machine-learning-engineers.html, Access date:08.19.2017.

Lemonnie, E. (2001). *Protocol Anomaly Detection in Network-based IDSs.* Stockholm: Defcom.

Lim, A. (2015). Security Risks in SDN and Other New Software Issues. *RSA Conference: Where The World Talks Security.* Marina Bay Sands, Singapore.

Luo, S., Wu, J., Li, J., & Pei, B. (2015). A Defense Mechanism for Distributed Denial of Service Attack in Software-Defined Networks. *International Conference on Frontier of Computer Science and Technology*.

MacQueen, J. B. (1967). Some Methods for classification and Analysis of Multivariate Observations,. *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability* (pp. 281-297). Berkeley: University of California Press.

Mathworks. (n.d.). *Unsupervised learning: Machine learning technique for finding hidden patterns or intrinsic structures in data*. www.mathworks.com: https://www.mathworks.com/discovery/unsupervised-learning.html, Access Date: 20.08.2017.

Mell, K. S. (2007). *Guide to Intrusion Detection and Prevention Systems (IDPS)* . Gaithersburg: National Institute of Standards and Technology .

Metzler, J. (2014). *SDN and Network Virtualization: A Reality Check.* Network World: https://www.networkworld.com/article/2604023/software-defined-networking/sdn-and-network-virtualization-a-reality-check.html, Access date:20.0.2017.

Morales, K., Martínez, M., Mandow, C. M., & García-Cerezo, B. M. (2012). makori the journal. *journal of computational science*.

Mousavi, S. M., & St-Hilaire, M. (2015). Early Detection of DDoS Attacks against SDN controllers. *International Conference on Computing, Networking and Communications*, 77-81.

Mukherjee, D. S., & Sharma, N. (2012). Intrusion Detection using Naive Bayes Classifier with Feature . *Procedia Technology*, 119 – 128 .

Ng, A. (2017). *Machine learning*. Coursera: https://www.coursera.org/learn/machine-learning

Nikolova, E., & Veselina Jecheva. (2015). Applications of Clustering Methods to Anomaly-Based Intrusion Detection Systems . *International Conference on Database Theory and Application* (pp. 37-41). Jeju: DTA2015.

Niyaz, Q., Sun, W., & Javaid, A. Y. (2016, November 22). *A Deep Learning Based DDoS Detection System in Software-Defined Networking (SDN).*

Open Networking Foundation. (2014). *SDN Architecture.* Palo Alto.

OpenCV. (2017). *Introduction to Support Vector Machines* openCV: http://docs.opencv.org/2.4/doc/tutorials/ml/introduction_to_svm/introduction_to_svm.html,

opennetworking foundation. (n.d.). *Software-Defined Networking (SDN) Definition*. Opennetworking.org: https://www.opennetworking.org/sdn-definition/, Access date:06.05.2017.

Pastrana, S., Torrano-Gimenez, C., Nguyen, H. T., & Orfila, A. (2014). *Anomalous Web Payload Detection: Evaluating the Resilience of 1-grams Based Classifiers.*

Performing attribute selection. (n.d.). weka.wikispaces.com: https://weka.wikispaces.com/Performing+attribute+selection , Access date: 08.07.2017

Project Floodlight. (n.d.). from projectfloodlight.org: http://www.projectfloodlight.org/floodlight/, Access date: 06.18.2017.

Radoglou-Grammatikis, P. I., & Sarigiannidis, P. G. (2017). Flow Anomaly Based Intrusion Detection System for Android Mobile Devices. *6th International Conference on Modern Circuits and Systems Technologies* (pp. 1-4). Thessaloniki: MOCAST2017.

Raza, S. &. (2013). *Northbound Interfaces ,Working Group.* San Francisco: Open Networking Foundation.

Rejchrt, J. (2013 , November 20). *network-anomaly-detection-survey*. Retrieved 10, 2017, from labs.ripe.net: https://labs.ripe.net/Members/jan_rejchrt/

Sangkatsanee, P., Wattanapongsakorn, N., & Charnsripinyo, C. (2011). Practical real-time intrusion detection using machine learning approaches. *Computer Communications*, 2227-2235.

SANS Institute. (2000). *Host vs Network-based intrusion detection systems.* SANS Institute.

SAS. (n.d.). *Machine learning: What it is and why it matters*. www.sas.com: https://www.sas.com/en_us/insights/analytics/machine-learning.html, Access date:07.08.2017

Schneider, P. (2015). *SDN security : Nokia Research perspective* . Nokia Solutions and Networks .

Scikit-learn. (2017). *Feature selection*. scikit-learn.org: http://scikit-learn.org/stable/modules/feature_selection.html, Access Date: 10.08.2017.

Scikit-learn. (n.d.). *Decision trees*. Scikit-learn.org: http://scikit-learn.org/stable/modules/tree.html, Access Date:07.09.2017.

scikit-learn. (n.d.). *Nearest Neighbors*. scikit-learn.org: scikit-learn.org/stable/modules/neighbors.html, Access Date:07.09.2017

scikit-learn. (n.d.). *Nearest Neighbors*. scikit-learn.org: scikit-learn.org/stable/modules/neighbors.html, Access Date:07.09.2017

Scikit-learn. (n.d.). *Support Vector Machines*.Scikit-learn.org: http://scikit-learn.org/stable/modules/svm.html, Access Date:20.09.2017

sdxcentral. (n.d.). *Why SDN or NFV Now?* www.sdxcentral.com: https://www.sdxcentral.com/sdn/definitions/why-sdn-software-defined-networking-or-nfv-network-functions-virtualization-now/, Access Date:15.06.2017

Shin, S., Yegneswaran, V., Porras, P., & Gu, G. (2013). AVANT-GUARD: Scalable and Vigilant Switch Flow Management in Software-Defined Networks. *ACM SIGSAC conference on Computer & communications security*.

Simkin, S. (2017). *What Is An Intrusion Detection System?* https://www.paloaltonetworks.com: https://www.paloaltonetworks.com/cyberpedia/what-is-an-intrusion-detection-system-ids

Spathoulas, G. P., & Katsikas, S. K. (2009). Using a fuzzy inference system to reduce false positives in intrusion detection. *16th International Conference on Systems, Signals and Image Processing*, 1-4.

Symantec. (2003). *Statistical-Based Intrusion Detection*. www.symantec.com:https://www.symantec.com/connect/articles/statistical-based-intrusion-detection, Access Date: 07.08.2017.

Tahir, H. M., Hasan, W., Said, A. M., Zakaria, N. H., Katuk, N., Kabir, N. F., . . . Yahya, N. I. (2015). Hybrid machine learning technique for intrusion detection system. *Proceedings of the 5th International Conference on Computing and Informatics*, (pp. 11-13). Istanbul.

Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A., & Ghogho, M. (2016). Deep Learning Approach for Network Intrusion Detection in Software Defined Networking. *International Conference on Wireless Networks and Mobile Communications (WINCOM)* (pp. 258-263). Fez: IEEE.

Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A., & Ghogho, M. (2016). Deep Learning Approach for Network Intrusion Detection in Software Defined Networking. *International Conference on Wireless Networks and Mobile Communications (WINCOM)* (pp. pp. 258-263.). Fez: WINCOM 2016.

Tran, C., Vo, T. N., & Thinh, T. N. (2017). HA-IDS: A Heterogeneous Anomaly-based Intrusion Detection System. *4th NAFOSTED Conference on Information and Computer Science* (pp. 156-161). Hanoi: NAFOSTED2017.

Ullman, S., Poggio, T., Harari, D., Zysman, D., & Seibert, :. (2014). *Unsupervised learning:Clustering.* Center for Brains, Minds and Machiness.

Van, N. T., Thinh, T. N., & Sach, L. T. (2017(a)). An anomaly-based Network Intrusion Detection System using Deep Learning. *2017 International Conference on System Science and Engineering* (pp. 210-214). Ho Chi Minh City: ICSSE2017.

Vargiya, R., & Chan, P. (2003). *Boundary Detection in Tokenizing Network Application Payload for Anomaly Detection.* Melbourne: Florida Institute of Technology .

Varma, P. R., Kumari, V. V., & Kumar, S. S. (2016). Feature Selection Using Relative Fuzzy Entropy and Ant Colony Optimization Applied to Real-time Intrusion Detection System. *International conference on computational modelling and security* (pp. 503-510). Bengaluru: Elsevier, B.V.

Wang, K., & Stolfo, S. J. (2004(a)). Anomalous Payload-Based Network Intrusion Detection. *Recent Advances in Intrusion Detection. RAID 2004. Lecture Notes in Computer Science* (pp. 203-222). Berlin: Springer.

Wang, M., Liu, J., Chen, J., Liu, X., & Mao, J. (2017). PERM-GUARD: Authenticating the Validity of Flow Rules in Software Defined Networking. *Journal of Signal Processing Systems*, 157-173.

Weston, J. (n.d.). *Support Vector Machine (and Statistical Learning Theory) Tutorial.* Princeton: NEC Labs America.

Yassin, W., Udzir, N. I., Abdullah, A., Abdullah, M. T., Zulzalil, H., & Muda, Z. (2014). Signature-Based Anomaly Intrusion Detection using Integrated Data Mining Classifiers . *2014 International Symposium on Biometrics and Security Technologies* (pp. 232-237). Kuala Lumpur: ISBAST.2014.

RESUME

Douglas Omuro Makori, I was born in 1988 in Kisii Kenya. I attended my primary and secondary school in Kisii Kenya before proceeding to Moi University for my undergraduate degree where I graduated in 2010 with a degree in information technology. I started working at Safaricom Kenya as a help desk support officer and later as a Systems administrator. In 2014 I started and still pursuing a Masters in Computer and information engineering at Sakarya University.