UNIVERSITY OF NAIROBI

College of Biological and Physical Sciences

School of Computing and Informatics

HARDENING THE SOFTWARE DEFINED NETWORK [SDN] CONTROLLER USING BORDER GATEWAY PROTOCOL [BGP]

By: Geoffrey Oguya

P53/86084/2016

Supervisor: Dr. Christopher K. Chepken

A Project Report submitted in partial fulfilment of the requirements to the award of a Master of Science [MSc.] in Distributed Computing Technology degree.

Declaration

I hereby declare that this research report is entirely my own work and to the best of my knowledge, has not been submitted at this or any other university.

Name: _____ Signature: _____ Date: _____

This report has been submitted with my approval as university supervisor.

Name: _____ Signature: _____ Date: _____

Acknowledgement

With much appreciation, I recognize the guidance that was offered by the late Prof. William Okello Odongo around the synthesis of this research's problem statement.

The realized knowledge attained from my graduate lectures at the University of Nairobi who took me through the hands-on laboratory work of the units in Computing Architectures, Computing Services, Computer Forensics and Cyber Security in the course of Distributed Computing Technology.

To my project supervisor, Dr. Christopher Kipchumba Chepken, who offered his time and advice to guide through the application measures of the Research Methodology course work and reinforce the deeper understanding of the expected deliverables from a graduate project.

Abstract

From a necessity, with reference to *Network Facilities Providers [NFPs]*, to adapt to *automated* and *scalable* networks, the computing industry did introduce *Software Defined Networking [SDN]*. It improves on packet transport times, and therefore the *performance* of the network system, by separating the packet switching layer from the packet control layer.

Despite the processing time advantage of *Software Defined Networking [SDN]*, its *OpenFlow protocol* implementation has been prone to *Man-in-the-Middle [MITM]* and *Distributed Denial of Service [DDoS]* cyber-attacks. This *vulnerabilities* have been discovered in the *OpenFlow algorithms*.

Internet Service Providers [ISPs] and *Cloud Service Providers [CSPs]*, therefore, find themselves in a quagmire: on one hand, the impetus to improve the network's processing time parameters by upgrading their systems to a *Software Defined Networking [SDN]* architecture, and on the other hand, is the inhibition to implement this architecture due to the *OpenFlow protocol's* non-resilience to the nefarious security threats.

This research project implemented a *Border Gateway Protocol [BGP]* approach, via the *Software Defined Networking [SDN]* south-bound protocol, in order to realize a hardened secure channel for the *Software Defined Networking [SDN]* controller. This was to provide a viable and more reliable alternative to the default *OpenFlow protocol*.

The *OpenFlow protocol* has been known to be a best performance protocol for packet transmission, and the *Border Gateway Protocol [BGP]*, as from its logic, has been structured to be a best performance protocol for policy driven outputs.

From an empirical approach therefore, the objective in the lab experiment was to compare the performance of the *Border Gateway Protocol [BGP]* to the *OpenFlow protocol*, as a *Software Defined Networking [SDN] south-bound protocol* for both a resilient and reliable network.

Table of Contents

Table of Figures

1. Chapter One: Introduction

The computing industry has realized a greater need for performance, a growing demand for scalability and a requirement for security.

Traditional Networking [TN] is implemented on dedicated hardware running an application specific protocol for that same appliance which tends to hinder integration of dissimilar networks and introduces configuration challenges. The *Network Facilities Providers [NFPs]*, on the other hand, as a requirement does require to build its network on a segmented multi-vendor platform in order to safeguard its services against vendor specific vulnerabilities.

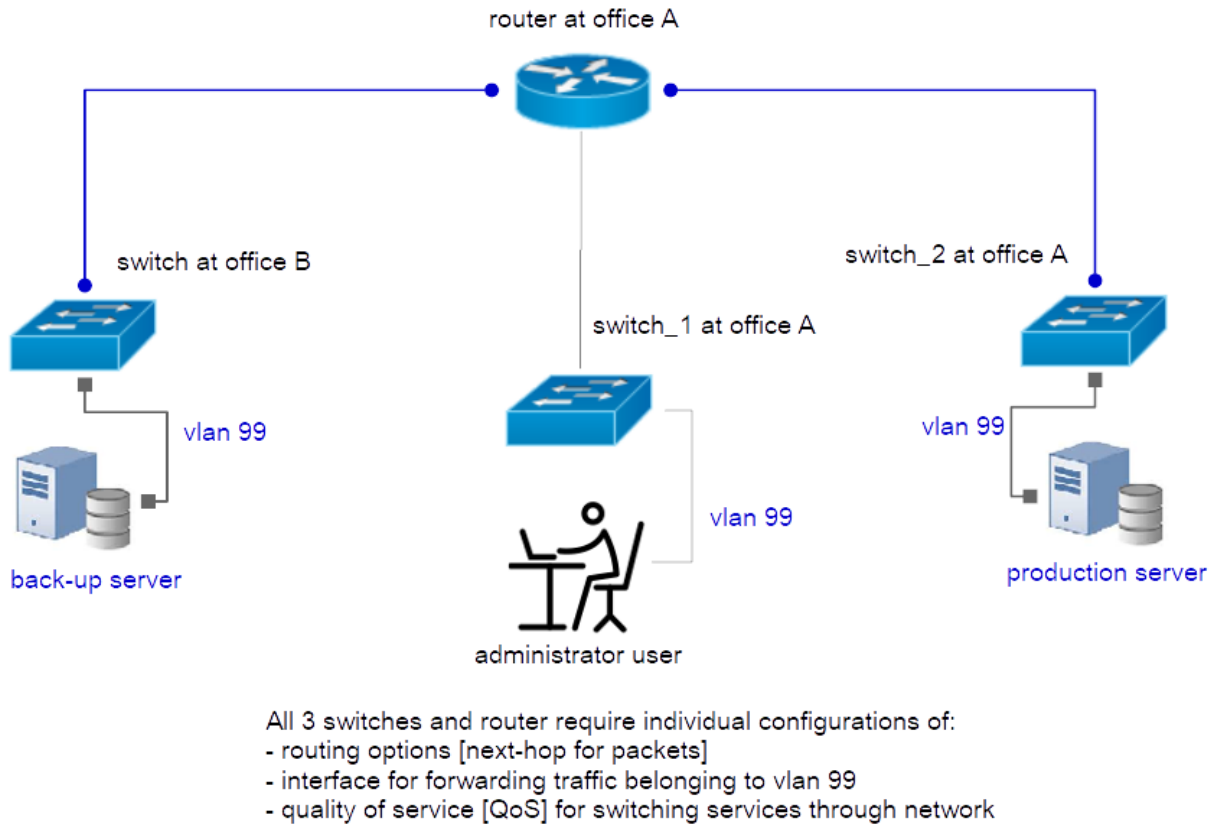


Figure 1.1: Traditional Networking [TN] approach for service delivery

The *Traditional Network* approach requires a myriad of manually executable procedures when introducing or removing a single device from the network. These administrative tasks result in large turn-around times, time loss and sometimes misconfigured policies that lead to network errors.

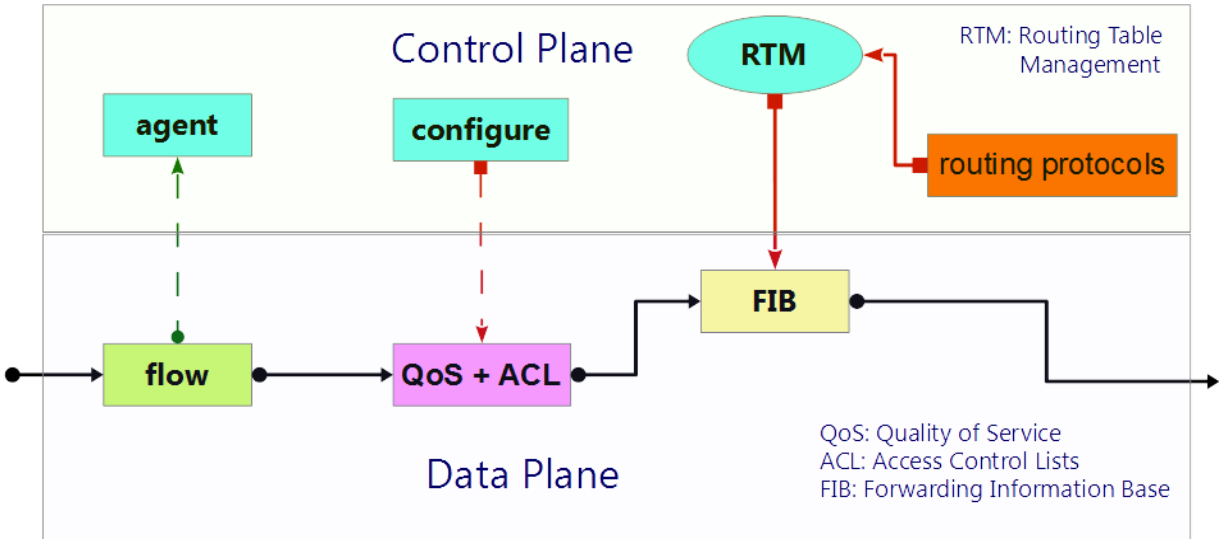


Figure 1.2: Traditional Networking Switching Device Planes

Software Defined Networking [SDN], then is introduced to address the multiple configuration limitations faced by the *Traditional Networks [TN]* and also offer other implementation advantages i.e. cost of switches, automation etc. (Markus N., 2013)

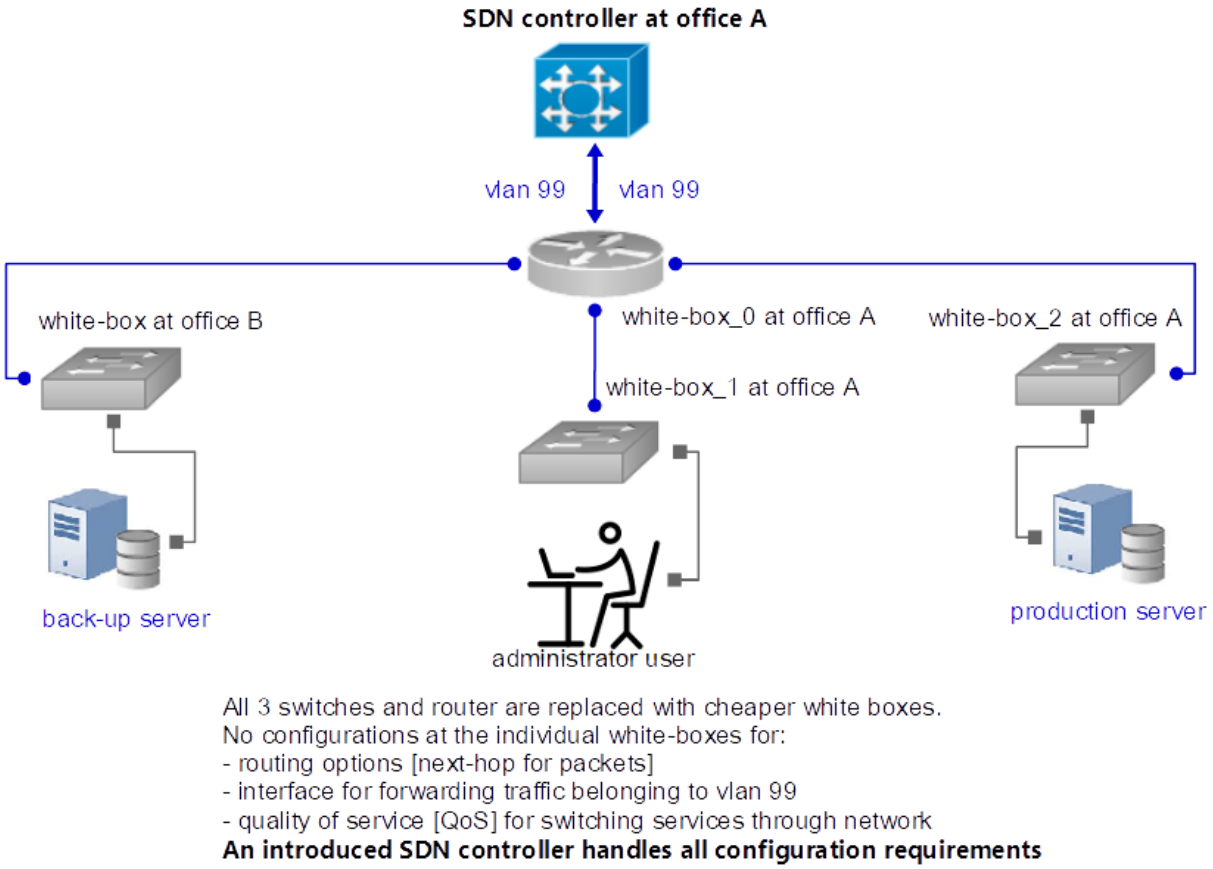


Figure 1.3: Software Defined Networking [SDN] approach for service delivery

Two decades ago, *Software Defined Networking [SDN]* had begun as a probable conceptual model, but has now been practically adopted across 67% of the Data Centers worldwide and 56% of the Data Centers in Africa. SDN does house the future of networking as it converges the agile benefits of the operation elemental units of: computing, networking, virtualization and informational sciences (Buraglio, 2015).

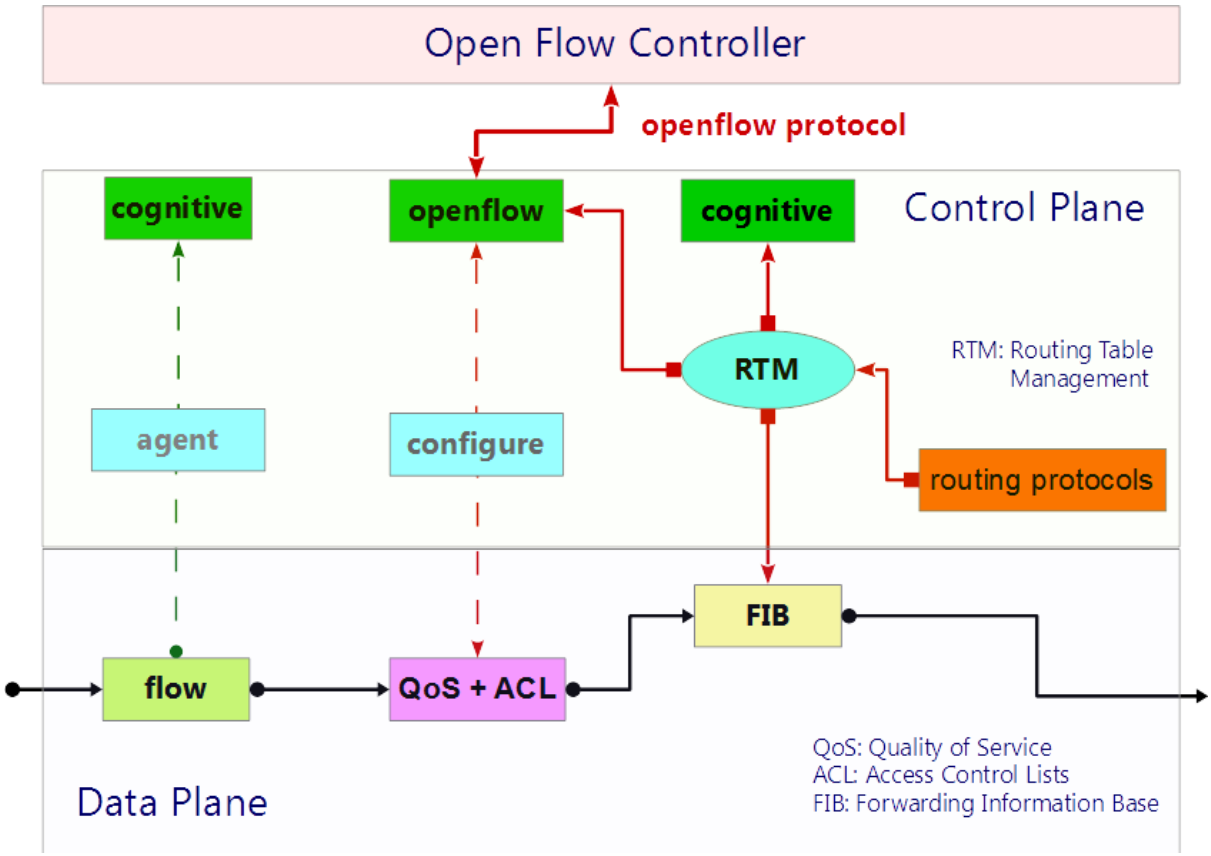


Figure 1.4: Software Defined Network [SDN] Switching Device Planes.

Software Defined Network [SDN] architecture, as realized with *OpenFlow protocol* does allow participants an avenue to implement *Quality of Service [QoS]*, all with the benefits of (Markus N., 2013) :

- Better integration of hardware and software.
- Reduced overhead on *operational expenditures* [OPEX].
- Rapid launch of new service products

It is with this in mind that the *Software Defined Network [SDN] controller* - with reference to its application services - is observed to be a most significant vector, prone to a *Distributed Denial of Service [DDoS]* or *architectural failures* under a security attack.

1.1 Problem Background

Software Defined Networks [SDN] is rapidly taking root across provider networks with the objective being to ride on its efficiency and scalability (Chin et al, 2016) (Lychev et al, 2013) (Ryburn et al 2015).

Software Defined Networks [SDN] in reference to its adoption against *Traditional Networks [TN]* has several operational and infrastructural advantages, however *Software Defined Networks [SDN] OpenFlow protocol* implementation, is currently facing some serious drawbacks of: *vendor dependence*, *managing common APIs*, *scalability concerns*, *support for multiple hypervisors*, *security resilience* etc. (Techtarget, 2016)

The *Networks Facilities Providers' [NFPs]* purpose in shifting from the *Traditional Network [TN]* to *Software Defined Networking [SDN]* is to achieve automation, device segmentation and all the while guarantee security for its services, but if this SDN is implemented with the default *OpenFlow protocol*, the network infrastructure will display weaknesses that are open for exploitation.

As has been captured and illustrated from *Techtarget 2016 forum,* an SDN architecture should be able to provide state-of-the-art performance and secured services to its clients.

Therefore, this research project would aim to converge its efforts towards the research, design, testing, evaluation and analysis of the *Software Defined Network's [SDN]* security resilience concern, with a focus on its *controller* and an alternative southbound protocol to *OpenFlow*.

1.2 Problem Statement

Network Facilities Providers [NFPs] are bound to strict *Service Level Agreements [SLAs]*, and are monitored for compliance by *Regional Regulatory Authorities* i.e. *Communication Authority Kenya [CAK]* if they fail to meet threshold performance contracts as directed.

The *Network Facilities Providers' [NFPs]*, therefore, needs not only meet performance thresholds, but are also required to upgrade their networks to address the evolving technology. The quagmire being, how the *Network Facilities Provider [NFP]*, can upgrade from a *Traditional Network [TN]* to *Software Defined Network [SDN]*, without introducing the documented risks associated with *OpenFlow protocol*.

OpenFlow protocol continues to display security concerns to the *Network Facilities Providers [NFPs]* and has had some documented *vulnerabilities* in its security channel (Lychev et al, 2013)

The research project will, especially, purpose to provide an alternative solution to *OpenFlow protocol*, with a focus on the *Network Facilities Providers' [NFPs]* incertitude to implement *Software Defined Networking [SDN]* at its *network core infrastructure*, due to lack-of, or inadequate protection against the security threat realized in a *DDoS cyber-attack*.

Security in both the operational divisions of networking and the cloud computing domains are on demand for current solutions to the ever evolving threats to its agile systems. These systems also need to present themselves as robust and secure against the proliferation of cyber threats.

OpenFlow protocol, at present, fails to address the working collaborative efforts in security and agility for the *Network Facilities Providers' [NFPs]* evolving infrastructure. *OpenFlow protocol*, has been shown to be: (Ryburn et al, 2016)

- inflexibly centralized affecting packet processing
- manually signaled via L2 that does not scale well with large networks
- not well shielded against *MITM* and *DDoS* attacks

These known and presented vulnerabilities are perilous for *Network Facilities Providers [NFPs]* who would desire to upgrade from *Traditional Networks [TNs]* in order to improve performance, yet fear the risk presented by the associated *OpenFlow protocol* in the SDN implementation and the exposure to cyber criminals (Remes et al, 2014).

This research project aimed to experiment on a better threat mitigation approach in securing the *Software Defined Network [SDN] controller* against the *Distributed Denial of Service [DDoS]* security threat, by an employment of *Border Gateway Protocol [BGP]* in a simulated '*Mininet'* and '*Open Daylight*' Environment.

1.3 Research Objectives

The main purpose of this research project is to integrate the *Border Gateway Protocol [BGP] FlowSpec* flavor into a simulated *Software Defined Network [SDN]* environment, as an alternative southbound protocol to *OpenFlow protocol*, in order to comparatively evaluate its processing time vis-à-vis its security resilience to an injected *Distributed Denial of Service [DDoS]* attack.

In order to align the project to the main objective, the subsequent aims of the research project are:

- To install the open-source *Open Daylight [ODL]* in a virtualized environment, having as its function to simulate the *Software Defined Network [SDN]* controller.
- To configure into the virtualized *Software Defined Network [SDN]* controller environment, a *Border Gateway Protocol [BGP] FlowSpec* flavor, serving as its function an alternative to the southbound *OpenFlow protocol.*
- To create a *Botnet Simulator [BoNeSi]* in a virtualized environment that will serve to simulate an injected *Distributed Denial of Service [DDoS]* attack targeting the *Software Defined Network [SDN]* controller.
- To integrate the open-source *Zabbix* that will serve as monitoring tools for the *Software Defined Network [SDN]* controller's performance measurements and the *Distributed Denial of Service [DDoS]* input parameters.

1.4 Definition of Important Terms

1. ***Traditional Networks [TN]*** are static and inflexible networks having the control and data plane fused into the hardware appliance.
2. ***Software Defined Network [SDN]*** is the decoupling of a network's control plane from the data plane that allows for various abstractions of the infrastructure.
3. ***OpenFlow protocol*** is an open source based protocol that facilitates for routing the data packets in a network to be determined by shared software.
4. ***Border Gateway Protocol [BGP]*** is a standardized protocol that makes the internet work by exchanging routing information between autonomous systems.

5. ***Distributed Denial of Service [DDoS]*** is a malicious attempt to disrupt normal service on a network by flooding its infrastructure with multiple sources of traffic.
6. ***Man-in-The-Middle [MITM]*** attacks are where the perpetrators get in the middle of a communication by eavesdropping on, or to impersonate a relayed information.
7. ***Quality of Service [QoS]*** is the capability of a network to provide better service by giving priority to certain applications that would improve on bandwidth and latency.
8. ***Access Control Lists [ACLs]*** are tables that inform an *Operating System [OS]* which users have rights to access certain systems objects e.g. files or directories.
9. ***Routing Table Manager [RTM]*** is the central repository for all rules that describe the routing protocols that operate under the *Routing and Remote Access Service [RRAS]*. It is used to calculate changes in topology.
10. ***Forwarding Information Base [FIB]*** is also known as a *Forwarding Table* that is used in network bridging to find the proper interface to which the input interface should forward traffic. It optimizes the process of looking up an address.
11. ***Network Facilities Providers [NFPs]*** are entities that provide service to customers via their defined network infrastructures e.g. ISPs, TelCos etc.
12. ***White box*** is a system or device whose internal workings are well understood.
13. ***Virtualization*** is the creation of a virtual version of a resource or device.
14. ***Automation*** is the application of technology to control systems and information systems to handle processes that would have been manually invoked.
15. ***Application Programming Interfaces [APIs]*** are systems of tools and resources in an Operating System enabling developers create software applications.
16. ***Control Plane*** in a router is focused on how a box interacts with its neighbours by tracking topology changes, computing routes and installing forwarding rules.
17. ***Data Plane*** is the work horse of the switching elements by parsing packet headers, managing QoS, filtering, policing, queuing etc.

1.5 List of Abbreviations

- **SDN –** Software Defined Networking
- **BGP –** Border Gateway Protocol
- **DDoS –** Distributed Denial of Service

- **NFP –** Network Facilities Providers
- **ISP –** Internet Service Providers
- **CSP –** Cloud Service Providers
- **MITM –** Man In The Middle
- **API –** Application Programming Interface
- **OPEX –** Operating Expenditure
- **CAPEX –** Capital Expenditure
- **CPU –** Central Processing Unit
- **VLAN –** Virtual Local Area Network
- **QoS –** Quality of Service
- **ACL –** Access Control List
- **RTM –** Routing Table Manager
- **FIB –** Forwarding Information Base
- **SLA –** Service Level Agreement
- **BGP-LS –** BGP Link State
- **PCEP –** Path Computation Element Protocol
- **RPKI –** Router Public Key Infrastructure
- **BoNeSi –** BotNet Simulator
- **TCP –** Transmission Control Protocol
- **UDP –** User Datagram Protocol
- **uRPF –** unicast Reverse Path Forwarding

2. Chapter Two: Literature Review

2.1 Theoretical Basis

The *Internet* has been a great breakthrough, from an experimental research to a universal infrastructure that enables innovation in applications e.g. WEB, *Peer-2-Peer [P2P]*, *Voice over Internet Protocol [VoIP]* etc. The *Traditional Networks [TN]* have been plagued with *closed networks (vendor specific)*, *slow protocol standardization* and *long evolutional delays* that impact on performance (Rexford J., 2012). Service providers have desired to ape the success of the *Internet* on its *Traditional Network [TN]* core infrastructure, and thus the *Software Defined Network [SDN]* research came into play to address *scalability*, *networking* and *computing* improvements.

Software Defined Networks [SDN] has evolved from a theoretical concept into an agile, scalable and deployable architecture (Chin et al, 2016) that has been implemented across data centers worldwide.

Software Defined Networks [SDN] has a unique capability that disaggregates the control of network devices from the data they transport, and the switching software from the actual network hardware. Effectively, this provides a service layer that is more *manageable* and *programmable* than physically reconfiguring networks. Its revolutionary focus is centered on its ability to: (Wang et al, 2015)

- Separate and isolate the forwarding plane from the control plane.
- Centralize the controller and view of the network.
- Program the network by external applications.

At the Google Company, with a focus on of one of its *Points of Presence [PoPs]* in Mombasa [Kenya], *Software Defined Networks [SDN]* has already taken root in their service offering to customers (Giotis et al, 2013): Storage Networks, Cloud Computing Networks, and Content Distribution Networks. Published journals, as referenced in this literature review, remark a global auxesis of a transition from Traditional Networks [TN] to Software Defined Networks [SDN].

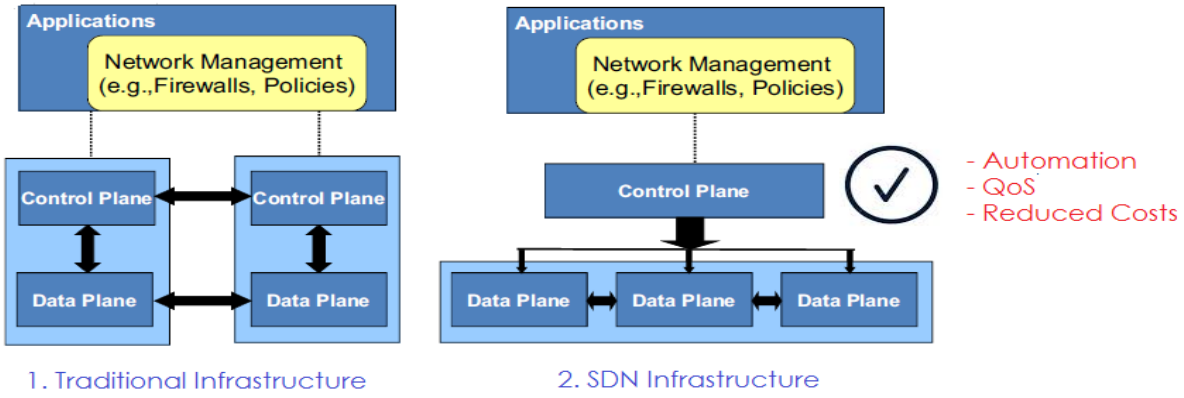


Figure 2.1: Traditional Networking vs. Software Defined Networking

Software Defined Networks [SDN], therefore offers a *unification* of a variety of *Traditional Network [TN]* devices i.e.

- Routers: match longest destination *Internet Protocol [IP]* prefix
- Switches: match destination *Media Access Controller [MAC]* address
- Firewall: match *Internet Protocol [IP]* addresses and *Transmission Control Port [TCP]*

Performance with reference to processing time in *Software Defined Networks [SDN]*, is now addressed. However there is another pillar of networks, security, which needs to be considered. Does the *Software Defined Network [SDN]* meet the service provider's requirement in terms of *security resilience*?

From the statistics shared by the early adopters of *Software Defined Networks [SDN]* (Braga et al, 2010), the *DevOpSec [Software Developers | Operations | Network Security]* Teams (Giotis et al, 2013) within these companies observed and reported an increased rate of cyber-attack on their networks. There has been suggested architectures that mitigate these attacks using various *Algorithms* and *Protocols*.

Implementation of *Software Defined Networks [SDN]*, at the very onset and in its acute inception period, involved linking the *Forwarding Plane* to the *Software Defined Networks [SDN] controller* using the *OpenFlow Protocol* (Braga et al, 2010). The *OpenFlow protocol* which is defined under the *Open Networking Foundation [ONF]* is the most widely employed '*South-Bound API [Application Programing Interface] protocol*' to program both software and hardware (Savage et al, 2015).

The *OpenFlow protocol*, however, without *unicast Reverse Path Forwarding* [uRPF], in a large scale network is vulnerable to source address spoofing, especially because unicast traffic's prime interest is the destination *Internet Protocol [IP]* address, without 'stateful' checking the source *Internet Protocol [IP]* address (Bi et al, 2012).

The *OpenFlow protocol* has been ascertained to permit an easier *Secure Channel Attack* via '*Session Hijacking'* because it is (Ryburn et al, 2016):

- *Inflexibly centralized* affecting packet processing
- *L2 manually signaled* that does not meet large networks' requirements
- Not well shielded against *Man-In-The-Middle [MITM]* and *Distributed Denial of Service [DDoS]* attacks.

Accompanied with an exponential morphing of cyber-attack mechanisms (Wang et al, 2015), it is the intention of this research project to look into amended schemes that would shield the *Network Facilities Providers' [NFPs]* future *Software Defined Networks [SDN]* infrastructure from a *Distributed Denial of Service [DDoS]* cyber-attack.

Distributed Denial of Service [DDoS] attacks, are deployed using various techniques of *flooding*, *amplification*, *protocol exploiting*, *malformed packets*, and leads to a consumption of the networks' resources (Remes et al, 2014): *bandwidth*, *Central Processing Unit [CPU]* and *memory*, and can be classified into 3 groups:

- *Volume based attacks* that are aimed at the system's bandwidth e.g. *User Datagram Protocol [UDP]* and *Internet Control Message Protocol [ICMP]* flooding, spoofed-packets etc.
- *Protocol attacks* that are targeted at the system's resources e.g. *SYN* flooding, ping-of-death, smurfing, fragmented packet attacks etc.
- *Application layer attacks* that target *Operating System [OS]* and application vulnerabilities e.g. *Hyper-Text Transmission Protocol [HTTP]* get | post attacks etc.

Advanced *Distributed Denial of Service [DDoS]* type attacks use *zombie hosts* and *reflectors* to hide the attackers' traces.

The approach would therefore be to research on other proposed algorithms and techniques that alleviate the *Software Defined Networks [SDN] controller* from the *OpenFlow protocol* weakness to *Distributed Denial of Service [DDoS]* attacks.

In order to mitigate against, and ameliorate on *Software Defined Networks [SDN]* delivery, recent publications (Gupta et al, 2015) and (Savage et al, 2015) have proposed replacing the *OpenFlow protocol* with an alternative *Software Defined Networks [SDN]* south-bound *Application Programming Interface [API]:* the *Border Gateway Protocol [BGP]*.

Border Gateway Protocol [BGP] is formed across the *Internet* as a one of the anchors in its foundational structure, and employed in a plethora of *Clos Networks* to interconnect devices participating in various *Autonomous Systems* [AS] (Remes et al, 2014). In the *Traditional Networks [TNs]*, *Border Gateway Protocol [BGP]*, however has been found to have *Man-in-The-Middle [MITM]* and *Distributed Denial of Service [DDoS]* vulnerabilities leading to *Session Hijacking*.

The future directions in *Border Gateway Protocol [BGP]* security, especially focused on *Software Defined Networks [SDN]* are (Butler et al, 2015):

- *Routing frameworks and policies*
- *Attack detection*
- *Data plane protection*
- *Partial deployment*

To improve on the *Border Gateway Protocol [BGP]* offering for *Software Defined Networks [SDN]*, and in order to address the security vulnerabilities, a couple of flavours (under routing frameworks and policies) have been proposed:

2.1.1 RPKI System

It employs the *X.509 certificate base*, which is run by *Regional Internet Registries [RIR]* like *ARIN, AfriNIC etc.*, and is a way to couple an *Internet Protocol [IP]* address range to an *Autonomous System [AS]* through *Cryptographic Signatures*.

Holders generate *Route Origination Authorizations [ROA]* which are described as the signed statements based on *X.509 certificate* that associate *Internet Protocol [IP]* with *Autonomous System Numbers [ASNs]*, and gives the *Autonomous System [AS]* permission to originate | announce the prefix (Remes et al, 2014).

Router Public Key Infrastructure [RPKI], through trust, will check the *Border Gateway Protocol [BGP] advertisements* and filter the results into categories of *valid*, *invalid* and *unknown*.

The *valid status* signifies that both criteria of a 'present *Route Origination Authorizations [ROA]*' and 'matching prefixes' are fulfilled, while the *invalid status* signifies that only the '*Route Origination Authorizations [ROA]* condition' has been met with no 'matching prefixes'. The third, *unknown state* indicates an absence of a *Route Origination Authorizations [ROA]* that should cover the enumerated prefixes.

Over the last couple of years, there has been a push to standardize *secure path validation* for *Border Gateway Protocol [BGP]*, and *securing Route Origination Authorizations [ROA]* with *Router Public Key Infrastructure [RPKI]*. The latter is gaining traction among network operators (Lychev et al, 2013).

2.1.2 BGP LS and PCEP System

Border Gateway Protocol Link State [BGP-LS] is an extension to *Border Gateway Protocol [BGP]* used to distribute the network's link-state topology to external entities e.g. *Software Defined Networks [SDN]*. The papers (Gupta et al, 2015) (Conran et al, 2016) have proposed to employ a mitigation technique against *Man-in-The-Middle [MITM]* through the incorporation of another south-bound *Application Programming Interface [API]*, *Border Gateway Protocol Link State | Path Computation Element Protocol [BGP-LS and PCEP]*.

The solutions offered by *Border Gateway Protocol Link State | Path Computation Element Protocol [BGP-LS and PCEP]* did augment the security resilience against *Man-in-The-Middle [MITM] attacks* when compared to the *OpenFlow protocol*, but did not effectively meet the objective of hardening the secure channel against *Distributed Denial of Service [DDoS]* (Lychev et al, 2013)

2.1.3 BGP FlowSpec System

Border Gateway Protocol [BGP], with reference to an improved *Router Public Key Infrastructure [RPKI]* security scheme, has an *RFC5575* prepared called *BGP FlowSpec* that

alleviates the *Autonomous System [AS]* from these *Distributed Denial of Service [DDoS]* vulnerabilities in the *Traditional Networks [TNs]*.

In this scheme, routers attach their *X.509 based certificates* to the '*Border Gateway Protocol [BGP]* Updates' to verify the source [origin] of the packets.

FlowSpec, which is employed under *Border Gateway Protocol [BGP]*, in summary:

- Exploits much of the *OpenFlow protocol* based SDN controller capabilities of: *complete overview of the network, establishing new data flows* and *gathering various traffic statistics* (Chin et al, 2016)
- Allows for propagation of router rules to a number of routers efficiently via signatures that *rate limit, redirect traffic* or *black hole* requests.
- Allows for the *flexible* and *partial Border Gateway Protocol [BGP]* security deployment, which does co-exist with the *Traditional Networks [TNs]* insecure *Border Gateway Protocol [BGP]* found in areas of the Internet that have not yet deployed *Border Gateway Protocol [BGP] Security*.
- uses the same *granularity* as *Access Control Lists [ACLs]* (Ryburn et al, 2015)
- uses the same *automation* and *best practice* leverage as *Remote Tunneled Black Holes [RTBH]* (Ryburn et al, 2015)

The *Border Gateway Protocol [BGP]*, however, does display some drawbacks when employed in the secure real domain:

- *Random Access Memory [RAM]* resource intensiveness due to *Border Gateway Protocol [BGP] updates* from '*Signature Inclusion*' and '*Byzantine Robustness*' (Butler et al, 2015).
- Complex policy configurations based on *regular expressions [regex]* of *Transmission Control Protocol [TCP-IP]*

The *Border Gateway Protocol [BGP]* and its *flavours* have been widely proposed as security resilient alternatives to the *OpenFlow protocol*. The *Border Gateway Protocol [BGP]* is the routing protocol of the *Global Internet*, as well as for *Network Facilities Providers' [NFPs]* private networks. It also can now carry routes for *Multicast*, *IPv6*, *Virtual Private Networks [VPNs]*, and a variety of other data (Braga et al, 2010).

From (Conran et al, 2016), *Border Gateway Protocol [BGP]* is addressed as a *transfer protoco*l between '*Software Defined Network [SDN] controller*' and '*Forwarding Devices*'. An

integration of *Border Gateway Protocol [BGP]* to *Software Defined Network [SDN]* does offer a number of use cases such as (Bi et al, 2012):

- *Distributed Denial of Service [DDoS] Mitigation*
- *Exception Routing & Forwarding*
- *Graceful Shutdown*
- *Integration with Legacy and Traditional Networks [TNs]*.

It has then also been proposed that *Software Defined Network [SDN]* architecture can then also be integrated with existing *Border Gateway Protocol [BGP]* technologies:

- *Layer3 Virtual Private Networks [RFC4364]*
- *Link State [LS]*
- *Path Computation Element Protocol [PCEP]*
- *FlowSpec [RFC5575]*

2.2 Implementation Concept

This research project sought to show the methodological approach used to develop the problem statement in providing a solution to the vulnerability displayed by the *Software Defined Network [SDN] controller* under the *OpenFlow protocol* to a *Distributed Denial of Service [DDoS]* attack.

The *Software Defined Network [SDN]* can be attacked from the *Secure Channel* or the *Application Programming Interface [API]* attack avenues. Recent papers: (Chin et al, 2016) (Giotis et al, 2013), (Braga et al, 2010) and (Wang et al, 2015) have focused on the *Secure Channel - Detection Mechanism* either using the *entropy statistical approach*, or the *machine learning approach* using *MLP, GAU, KM, Markov, SOM* etc.

In the developing sections, this research proposal paper focused on the avenues of alleviating the inherent attacks on the *Software Defined Network [SDN]* using the *Secure Channel - Communication Protocol*.
This research proposal paper also did focus on the *routing frameworks & policies*, with an employed *BGP FlowSpec protocol* in lieu of the *OpenFlow protocol* for *Software Defined Network [SDN]* controllers.

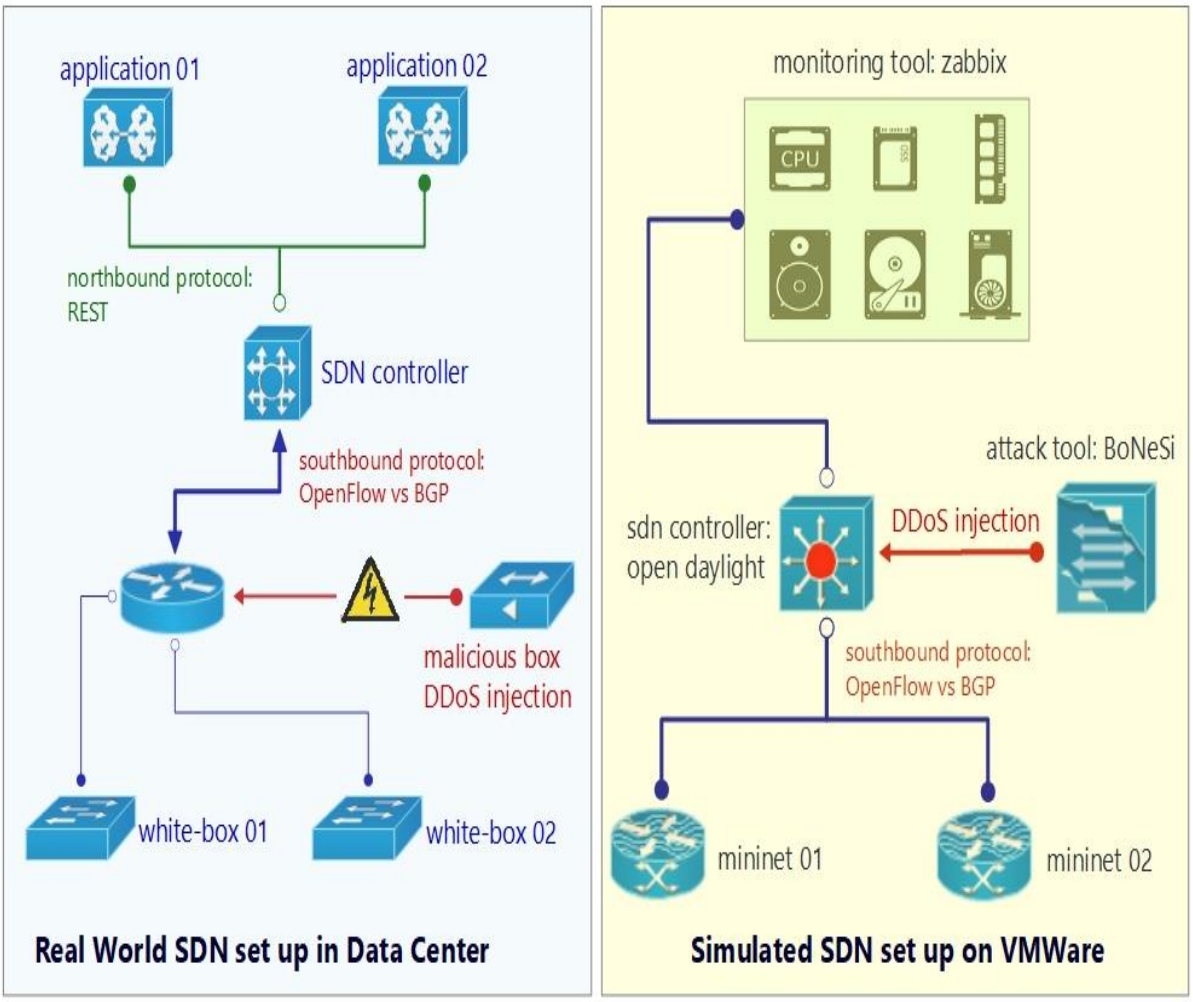


Figure 2.2: Software Defined Networking simulated set up

2.2.1 VMware Workstation

The simulated environment required a virtualized environment. *VMware* was the hosted hypervisor chosen due to:

- Its availability to run on a Windows or Linux *Operating System [OS]*.
- Its support of bridging existing host network adapters to converge all the virtual machines under the simulated *Software Defined Network [SDN]* environment.
- Its scalable set-up of different *virtual machines* to simulate the *Software Defined Network [SDN]* environment on a single machine.

The VMWare workstation hypervisor version 10.0.1 -1379776 was set up in a portable *Personal Computer [PC]* having a 16GB *Random Access Memory [RAM]* to handle the

Central Processing Unit [CPU] workload demands of running 5 virtual machines concurrently.

The virtual network adapter configured in this case under the VMware hypervisor was chosen as 192.168.245.1/24 running in a Windows 10 environment.

2.2.2 Open Daylight

One of the virtual machines under the *VMWare* workstation had an Ubuntu 16.04 Linux server *Operating System [OS]* configured with the Open Daylight Boron SR2 *Software Defined Network [SDN] controller.*

The *Open Daylight [ODL]* software was provisioned with a ready implementation that supported an:

- *Open controller*
- *Virtual overlay network*
- *Protocol plugins*
- *Switch device enhancements*.

The *Boron SR2 – karaf_0.5.2* had an available and ready built architecture, to provision *MiniNet*, which is the aforementioned *virtual overlay network* described, and has been readily embraced for its real world architectural approach.

Within *Open Daylight* there are several *protocols* and *algorithms* used to define and configure the *Software Defined Networking [SDN]* environment e.g. *OpenFlow*, *BGP-LS*, *PCEP*, *YANG – NETCONF* etc.

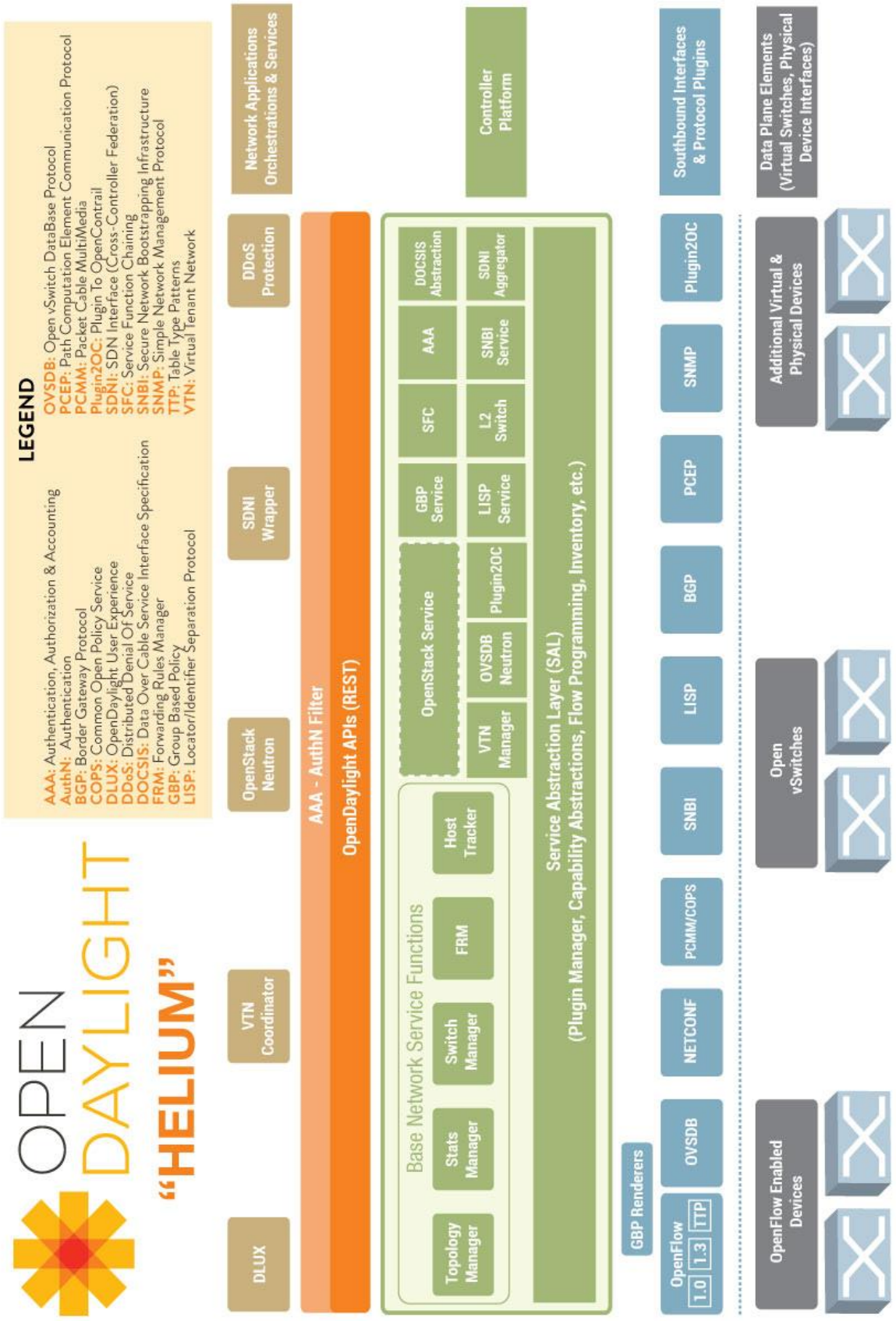


Figure 2.3: the Open Daylight [ODL] system architecture

2.2.3 BoNeSi [BotNet Simulator]

The simulated environment had the requirement to incorporate an attack, a device that would inject *Internet Control Message Protocol [ICMP]* requests akin to a *Distributed Denial of Service [DDoS]* attack.

BoNeSi is an attack network software simulator that allows for the integration into a *virtual* and *real* environment in order to perform penetration testing.

Effectively, to evaluate the performance of the Open Daylight *Software Defined Network [SDN]* controller against these injected *Distributed Denial of Service [DDoS]* attacks, the *BotNet Simulator [BoNeSi]* was the tool employed to flood *Internet Control Message Protocol [ICMP]* requests into the Software *Defined Network [SDN]* controller

The *BotNet Simulator [BoNeSi]* version deployed was the master version from the *GitHub* community.

2.2.4 Zabbix

The *Software Defined Network [SDN]* controller's performance required monitoring with reference to its counter-action to injected *Distributed Denial of Service [DDoS]* attacks.

The *Software Defined Network [SDN]* controller's *Central Processing Unit [CPU] load* and *Random Access Memory [RAM] utilization* were proportional indicators of its performance to the security resilience under the *OpenFlow protocol* vs *BGP FlowSpec protocol* environments.

Zabbix is a software tool used to monitor local or remote system performance by creating timers and graphic displays. It is a high performance polling engine that uses very little *Central Processing Unit [CPU*] and can handle multiple unreachable elements without locking up.

It is employed as performance monitoring tool for network elements e.g. routers, switches, servers etc., and is open source under the *GNU General Public License version 2.0 [GPLv2]*

Zabbix, under version 3.4.8 is implemented as the fifth virtual machine in the test environment.

2.2.5 Quagga

Quagga is a routing software suite supporting the network protocols: *Border Gateway Protocol [BGP], Open Shortest Path Protocol [OSPF]* etc for UNIX platforms.

It is a tool that is readily available and was used in this project to implement the *Border Gateway Protocol [BGP]* at the *Software Defined Network's [SDN]* southbound protocol. Incorporated as the third virtual image under Ubuntu 16.04 with an *Autonomous System Number [ASN]* 64404, it peered with the Open Daylight Controller using the *interior Border Gateway Protocol [iBGP]*

3. Chapter Three: Methodology

This chapter was used to:

- Describe the research design including the system implementation architecture that is employed to evaluate and investigate the research problem statement.
- Align the research design to the guidelines specified in the research objectives.
- Provide justification to the rationale employed for the specific procedures and methods used for data collection and analysis.
- Measure the techniques and milestones specified for the begin-to-finish execution of the project.

In the case of the *Software Defined Network [SDN]* controller's performance; the testing and analysis, especially from a *conceptual research type*, necessitated an *empirical method* to implement the *Border Gateway Protocol [BGP] FlowSpec* in lieu of the *Software Defined Networking [SDN]* default secure channel *OpenFlow protocol*.

The implementation of the research objectives as stated in the Chapter 1 had its formulation of relevant concepts guided by the scientific methods and process found in:

i. Experimental simulations in *the BotNet Simulator [BoNeSi]*, *MiniNet* and *Open Daylight [ODL]*.
ii. *Border Gateway Protocol [BGP]* integration in *Software Defined Network [SDN]* and simulation in *Quagga Software Routing*
iii. Performance, under processing times and security resilience testing, in the *south-bound Application Programming Interface [API]*: *OpenFlow protocol* vs *Border Gateway Protocol [BGP] FlowSpec*.
iv. Monitoring and performance verification in *Zabbix*

The aforementioned tools are open source, robust and readily available.

3.1 System Implementation Architecture

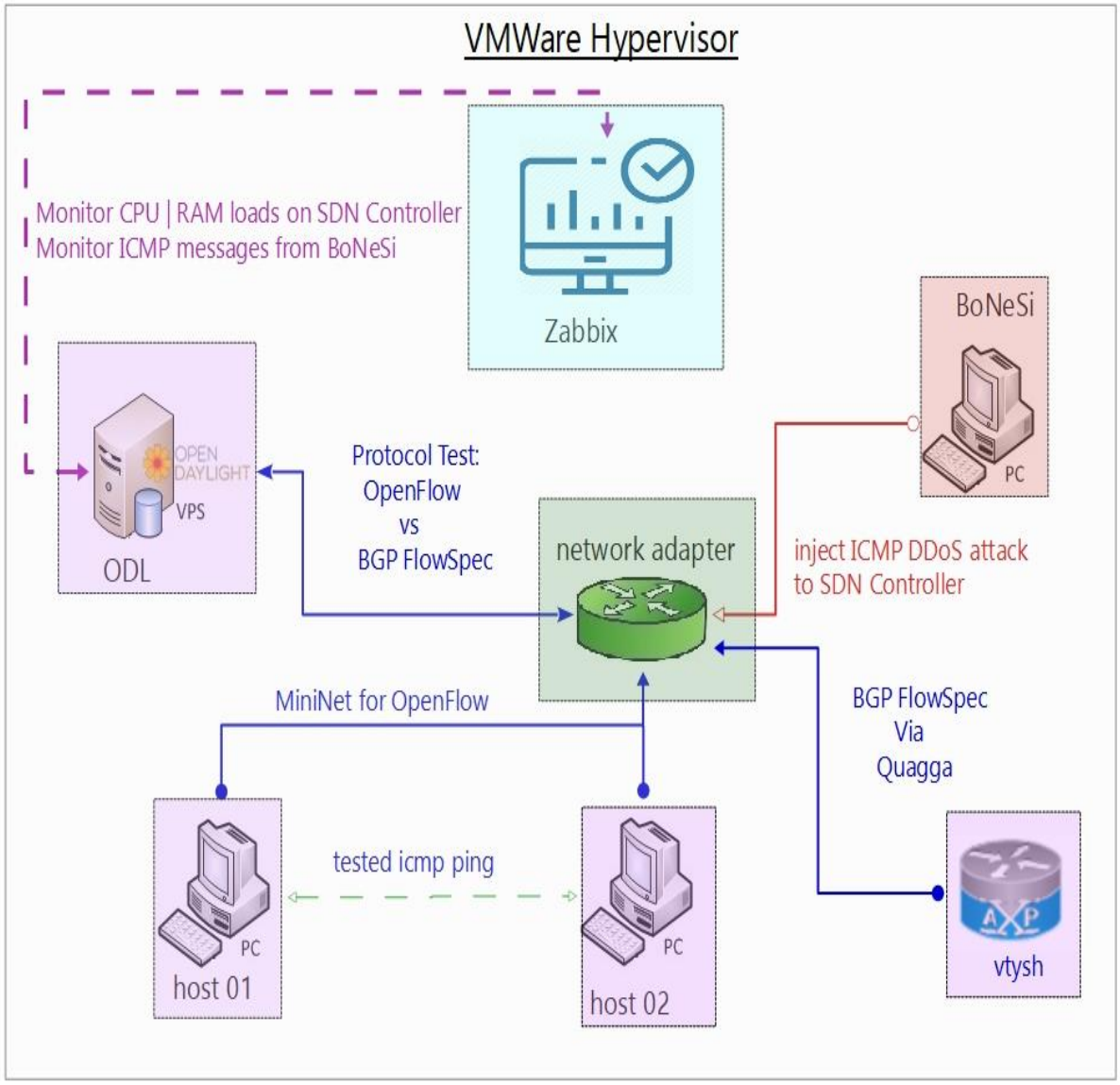


Figure 3.1: system implementation architectural set up

3.2 Research Design

The research project design and scope, sought to obtain the answers to the following questions:

i. What were the *statistics* related to *Software Defined Network [SDN]* penetration within the *Network Facilities Provider [NFPs]* domain in the country?

ii. Were the *Network Facilities Provider [NFPs]* hindered from integrating *Software Defined Network [SDN]* due to performance concerns inherent in the *Software Defined Network [SDN*] *protocols* and *Application Programming Interface [API]*?

iii. Was the alternative and available *Border Gateway Protocol [BGP] FlowSpec* flavor compatible to be deployed as the south-bound protocol for the *Software Defined Network [SDN]* controller?

iv. Was the identified *Border Gateway Protocol [BGP] FlowSpec* flavor better placed to work as a south-bound protocol to the *Software Defined Network [SDN]* controller as compared to the *OpenFlow protocol* at the *Network Facilities Providers [NFPs]* core?

Therefore, the project delved into the enumerated research procedure phases of:

1) A *case study* on the real world *Software Defined Network [SDN]* deployed systems and their performance limitations with respect to the running protocols.
2) A *laboratory experiment* on the *Border Gateway Protocol [BGP] FlowSpec* alternative that improves on *OpenFlow protocols* weaknesses to *Distributed Denial of Service [DDoS]* attacks.

The research was to find out if a deployment of a *Border Gateway Protocol [BGP] FlowSpec*, would effectively harden and secure the south-bound *Software Defined Network [SDN]* controller channel for better mitigation against *DDoS attacks*, as compared to the *OpenFlow protocol*.

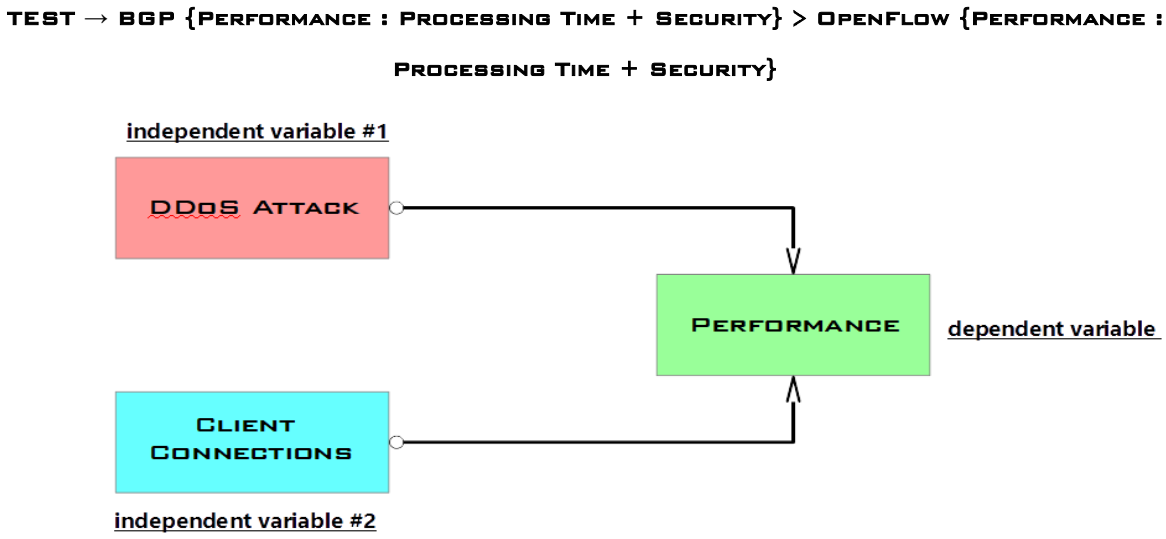


Figure 3.2: conceptual framework

The proposed research study, did have a real world baseline analyzed through issued out *structured questionnaires*, with the *target population*, as from the problem statement, being the *Network Facilities Providers [NFP]*.

The questionnaires were distributed, on a purposive sampling technique, to the local *Network Facilities Provider [NFP]* considering their different *tiers*, classification of *Operating Expenditure [OPEX]*, and their *market-share* with subscriber numbers.

Data collected from these *Network Facilities Provider [NFP]* on the deployed or yet to be deployed *Software Defined Network [SDN]* was an accurate and relevant benchmark to compare with the simulated tests of this research project.

The proposed research study was made in order to implement a *Border Gateway Protocol [BGP]* as an alternative implementation to the *OpenFlow protocol* that would enhance performance of the *Software Defined Network [SDN]* controller in the event of *Distributed Denial of Service [DDoS]* attack.

The assumption made was that an attacker had already found a way to penetrate into the *Software Defined Network [SDN]* system, targeting to *Software Defined Network [SDN]* controller functions. The *simulated attack vector* was carried out by the *BotNet Simulator [BoNeSi]* virtual machine. The injected *Internet Control Message Protocol [ICMP]* simulating a *Distributed Denial of Service [DDoS]* attacking the *Software Defined Network [SDN]* controller was varied, and measurements on the *Software Defined Network [SDN]* controller's performance recorded by the *Zabbix* tool.

The amount of initial *Central Processing Unit [CPU]* load and running processes of each unit was controlled such that it had a negligible effect on the *measurement* by *Zabbix*. Precaution was taken to record the client connections as a control variable for accurate data collected.

3.3 Data Collection

The tests carried out involved procedural tests done for both the *OpenFlow* and *Border Gateway Protocol* [BGP] *FlowSpec protocols* implementations with reference to *Software Defined Network [SDN]* south-bound channel performance.

The control measurements were taken according to:

i. Open Daylight *Software Defined Network [SDN]* controller *Central Processing Unit [CPU]* performance before the *Distributed Denial of Service [DDoS]* attack without client connections [X]

ii. Open Daylight *Software Defined Network [SDN]* controller *Central Processing Unit [CPU]* performance before the *Distributed Denial of Service [DDoS]* attack with client connections [Y]

iii. Open Daylight *Software Defined Network [SDN]* controller *Central Processing Unit [CPU]* performance after the *Distributed Denial of Service [DDoS]* attack Level 1 with client connections [Z1]

iv. Open Daylight *Software Defined Network [SDN]* controller *Central Processing Unit [CPU]* performance after the *Distributed Denial of Service [DDoS]* attack Level N with client connections [Zn]. n = i +1, where I is an integer

The measurements were taken relative to the [attacker's] *Internet Control Message Protocol [ICMP]* packets sent, the [sdn controller's] *Central Processing Unit [CPU]*: *Random Access Memory [RAM]* usage and the [mininet's + quagga's] performance measurement.

The attack situations and scenarios are proposed to be modelled by an external network [bonesi] *Distributed Denial of Service [DDoS]* penetration attack, and a [botnet] compromised attack for the internal network.

The attack is the only varied parameter in both environments, and the other parameters are measured as a consequence of the attack as displayed in the table.

Table 3.1: Comparative Analysis of OpenFlow to BGPFlowSpec in SDN.

Simulated Service			Inputs | Outputs		Results from Thresholds	
Item	Description	Testing Service	Protocol Units		Protocol A: OpenFlow	Protocol B: BGP FlowSpec
BoNeSi	Intruder	DDoS	Ping of Death	SYN flood	controlled	controlled
Open Day Light	SDN Controller	Connectivity	ARP	PID	measured	measured
Mininet	Clients	Uplink to SDN	HTTP	SSH	measured	measured
Zabbix	NOC	Performance	CPU Load	RAM Processing	measured	measured
Zabbix	NOC	Traffic Statistics	KBPS bandwidth	ICMP Reply	measured	measured

The employed simulation tools and instruments:

i. *Windows Operating System based PC*: 16GB RAM | 1TB HDD.
ii. *VMware Hypervisor*: for creating and deploying virtual machines.
iii. *BoNeSi*: a free, open-source tool that is deployed to simulate the *Wide Area Network [WAN]* architecture from which the injected *Distributed Denial of Service [DDoS]* attack will emanate.
iv. *Mininet*: a free, scalable, open-source tool that is deployed within a virtual instance to simulate the Data Center Switches [white-box devices]
v. *Open Daylight*: one of the better tested, readily available *Software Defined Network [SDN]* controllers that can simulate *Border Gateway Protocol [BGP] FlowSpec* configurations.
vi. *Zabbix*: an open-source based operational software to monitor interfaces and packet traffic across the deployed network

4. Chapter Four: Results

4.1 Data Analysis

From the research questions found in the *Research Design* of section 3.2, this research proposal paper sought:

- To tabulate responses received from *Network Facilities Providers [NFPs]* in regard to the *Software Defined Network [SDN]* infrastructure in the real world; its deployment, challenges and improvements.
- To analyze data from the experiments run during the testing of *Software Defined Network [SDN]* architecture in the simulated environments as from the procedure indicated in *Data Collection* of section 3.3.

The statistical data was analyzed via the *factorial formal experimental design*.

The responses were sampled in order to build up a case in addressing *Software Defined Network [SDN]* controller's performance of processing time vis-a-vis security resilience in the Service Provider and *Network Facilities Providers [NFPs]* industry.

The parameters under investigation were *qualitative* in nature: availability, resilience and robustness during and after a *Distributed Denial of Service [DDoS]* attack in the *Software Defined Network [SDN]* environments: with *Border Gateway Protocol [BGP] FlowSpec* and with *OpenFlow protocol*.

The parameters also had variables that were *quantitative* in nature: being the number of *Internet Control Message Protocol [ICMP]* requests, percentage of *Central Processing Unit [CPU]* and capacity of *Random Access Memory [RAM]* usage.

Due to the *qualitative* and *quantitative* aspects of the research proposal's parameters, the *Yin-Yang model* was applicable. The research was guided from an *inductive research approach* and an *experiment based research strategy* through an *exploratory data analysis* of the data collected.

From an increased number of *Internet Control Message Protocol [ICMP]* pings flooded into the *Central Processing Unit [CPU]* of the *Software Defined [SDN] controller*, the performance of *OpenFlow protocol* was compared to that of the *Border Gateway Protocol [BGP] FlowSpec* to ascertain the objectives of the experiment.

4.2 Testing

From the virtual machines configured, which makes up the *Software Defined Network [SDN]* system, all the controls set for the simulated tests under the *OpenFlow Protocol* and *Border Gateway Protocol [BGP]* were modelled and met the requirements set as from the methodology.

4.2.1 Test Environment X: OpenFlow Protocol

Environment X: These results were observed after running the *Software Defined Network [SDN]* controller under the *OpenFlow protocol* and through a set of *19 stages* of a simulated attack. Each stage of the attack, or control as referred here-in, there is an increasing amount of injected *Internet Control Message Protocol [ICMP]* packets in the magnitude of

$$\varphi_n = 9 * 10^{3n}, \text{ where } n = i + 1 \; ; \; i >= 0$$

Table 4.1.1: CPU tests of OpenFlow in SDN.

	Test Environment X	CPU [$ mpstat] OpenFlow			
iteration_1 of 10	description	%usr	%sys	%idl	X_series
Control 1	no client | no attack	41.21	8.04	35.9	0.00
Control 2	with client | no attack	41.21	8.03	35.98	2.00
Control 3	with client | attack 1 (r =9E3)	41.16	8.03	35.95	4.00
Control 4	with client | attack 2 (r =9E6)	41.15	8.05	35.67	6.00
Control 5	with client | attack 3 (r =9E9)	41.09	8.07	35.43	8.00
Control 6	with client | attack 4 (r =9E12)	41.08	8.07	35.04	10.00
Control 7	with client | attack 5 (r =9E15)	41.06	8.07	34.7	12.00
Control 8	with client | attack 6 (r =9E18)	41.14	8.05	34.32	14.00
Control 9	with client | attack 7 (r =9E21)	41.22	8.01	34.27	16.00
Control 10	with client | attack 8 (r =9E24)	41.35	7.97	34.21	18.00
Control 11	with client | attack 9 (r =9E27)	41.64	8.04	34.26	20.00
Control 12	with client | attack 10 (r =9E30)	41.79	8.07	34.06	22.00
Control 13	with client | attack 11 (r =9E33)	42.09	8.14	33.51	24.00
Control 14	with client | attack 12 (r =9E36)	42.12	8.12	33.71	26.00
Control 15	with client | attack 13 (r =9E39)	42.12	7.99	34.49	28.00
Control 16	with client | attack 14 (r =9E42)	42.12	7.98	34.56	30.00
Control 17	with client | attack 15 (r =9E45)	42.14	7.97	34.62	32.00
Control 18	with client | attack 16 (r =9E48)	42.15	7.97	34.66	34.00
Control 19	with client | attack 17 (r =9E51)	42.16	7.96	34.7	36.00
	Gradient	0.037605263			

Table 4.1.2: RAM tests of OpenFlow in SDN.

	Test Environment X	RAM [free] OpenFlow		
iteration_1 of 10	**description**	**used**	**available**	**X_series**
Control 1	no client | no attack	2245170	1247382	0.00
Control 2	with client | no attack	2343204	1191984	2.00
Control 3	with client | attack 1 (r =9E3)	2343120	1158540	4.00
Control 4	with client | attack 2 (r =9E6)	2346688	1106112	6.00
Control 5	with client | attack 3 (r =9E9)	2433448	955908	8.00
Control 6	with client | attack 4 (r =9E12)	2757444	609228	10.00
Control 7	with client | attack 5 (r =9E15)	2724736	758224	12.00
Control 8	with client | attack 6 (r =9E18)	3327908	135332	14.00
Control 9	with client | attack 7 (r =9E21)	2736956	811988	16.00
Control 10	with client | attack 8 (r =9E24)	2827152	751384	18.00
Control 11	with client | attack 9 (r =9E27)	2827060	740112	20.00
Control 12	with client | attack 10 (r =9E30)	3017820	569060	22.00
Control 13	with client | attack 11 (r =9E33)	3439232	154216	24.00
Control 14	with client | attack 12 (r =9E36)	2874968	816016	26.00
Control 15	with client | attack 13 (r =9E39)	2929972	791944	28.00
Control 16	with client | attack 14 (r =9E42)	2926572	800304	30.00
Control 17	with client | attack 15 (r =9E45)	3023560	702252	32.00
Control 18	with client | attack 16 (r =9E48)	3144160	545128	34.00
Control 19	with client | attack 17 (r =9E51)	3271921	516437	36.00
	Gradient	24962.21316		

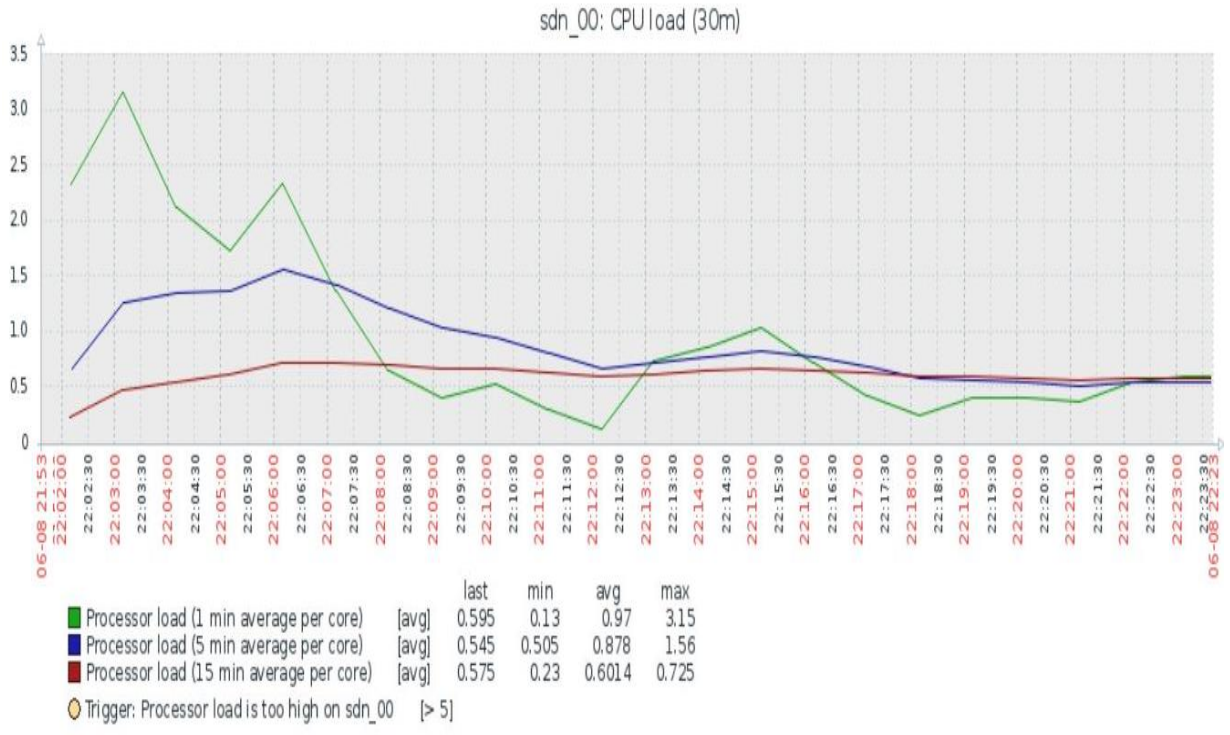


Figure 4.1.1: CPU tests of OpenFlow in SDN visualized on Zabbix

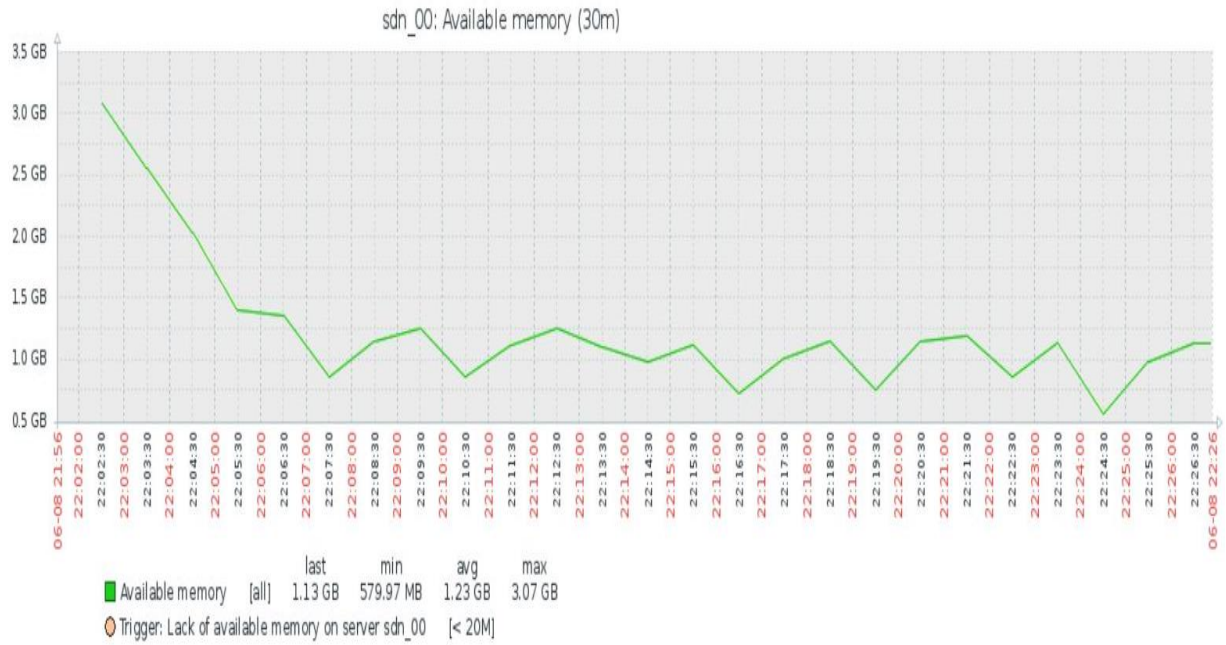


Figure 4.1.2: RAM tests of OpenFlow in SDN visualized on Zabbix

4.2.2 Test Environment Y: Border Gateway Protocol

Environment Y: Using the proposed solution, the *Software Defined Network [SDN]* controller under *Border Gateway Protocol [BGP] FlowSpec* set-up is also subjected to *19 stages* of a simulated attack. The *Distributed Denial of Service [DDoS]* attacks were varied through, $\boldsymbol{\varphi_n = 9 * 10^{3n}}$**, where n = i + 1 ; i >= 0**

This implemented while monitoring Traffic Flow and *Central Processing Unit [CPU]* performance via *Zabbix.*

Table 4.2.1: CPU tests of BGP in SDN.

	Test Environment Y	CPU [mpstat] BGP			
iteration_1 of 10	**description**	**%usr**	**%sys**	**%idl**	**X_series**
Control 1	no client | no attack	41.25	8.21	36.4	0.00
Control 2	with client | no attack	41.25	8.2	36.48	2.00
Control 3	with client | attack 1 (r =9E3)	41.13	8.17	36.63	4.00
Control 4	with client | attack 2 (r =9E6)	41.05	8.14	36.87	6.00
Control 5	with client | attack 3 (r =9E9)	41.03	8.18	36.93	8.00
Control 6	with client | attack 4 (r =9E12)	40.95	8.18	37.04	10.00
Control 7	with client | attack 5 (r =9E15)	40.89	8.18	37.12	12.00
Control 8	with client | attack 6 (r =9E18)	40.86	8.2	37.17	14.00
Control 9	with client | attack 7 (r =9E21)	40.86	8.21	37.19	16.00
Control 10	with client | attack 8 (r =9E24)	40.86	8.22	37.27	18.00
Control 11	with client | attack 9 (r =9E27)	40.81	8.23	37.3	20.00
Control 12	with client | attack 10 (r =9E30)	40.8	8.24	37.32	22.00
Control 13	with client | attack 11 (r =9E33)	40.77	8.26	37.33	24.00
Control 14	with client | attack 12 (r =9E36)	40.72	8.26	37.37	26.00
Control 15	with client | attack 13 (r =9E39)	40.67	8.25	37.4	28.00
Control 16	with client | attack 14 (r =9E42)	40.59	8.25	37.43	30.00
Control 17	with client | attack 15 (r =9E45)	40.5	8.26	37.44	32.00
Control 18	with client | attack 16 (r =9E48)	40.31	8.24	37.23	34.00
Control 19	with client | attack 17 (r =9E51)	40.25	8.22	37.2	36.00
	Gradient	-0.023631579			

		Open Flow				
	Test Environment 1	CPU [$ mpstat] OF			RAM [free] OF	
iteration_1 of 10	**description**	**%sys**	**%usr**	**%idl**	**used**	**available**
Control 1	no client | no attack	76.81	15.03	7.13	2622292	1049612
Control 2	with client | no attack	79.70	13.87	5.64	2676940	986332
Control 3	with client | attack 1 (r =9M)	79.92	13.86	5.39	2677704	950808
Control 4	with client | attack 2 (r =9T)	78.65	14.08	4.87	2683860	942508
Control 5	with client | attack 3 (r =9E)	77.93	14.30	4.71	2641240	931672

		BGP				
	Test Environment 2	CPU [mpstat] SDN			RAM [free] SDN	
iteration_1 of 10	**description**	**%sys**	**%usr**	**%idl**	**used**	**available**
Control 1	no client | no attack	76.81	15.03	7.13	2622292	1049612
Control 2	with client | no attack	72.46	15.01	6.23	2621984	980144
Control 3	with client | attack 1 (r =9M)	73.51	14.92	4.18	2765248	903492
Control 4	with client | attack 2 (r =9T)	71.01	15.19	4.02	2645152	961460
Control 5	with client | attack 3 (r =9E)	69.54	15.51	3.84	2618732	1006072

Table 4.2.2: RAM tests of BGP in SDN.

	Test Environment Y	RAM [free] BGP		
iteration_1 of 10	**description**	**used**	**available**	**X_series**
Control 1	no client | no attack	2607972	999276	0.00
Control 2	with client | no attack	3031708	575232	2.00
Control 3	with client | attack 1 (r =9E3)	2765380	809752	4.00
Control 4	with client | attack 2 (r =9E6)	2344876	1229728	6.00
Control 5	with client | attack 3 (r =9E9)	2339676	1234440	8.00
Control 6	with client | attack 4 (r =9E12)	2335328	1233204	10.00
Control 7	with client | attack 5 (r =9E15)	2971156	597108	12.00
Control 8	with client | attack 6 (r =9E18)	2914228	656228	14.00
Control 9	with client | attack 7 (r =9E21)	2393152	1177216	16.00
Control 10	with client | attack 8 (r =9E24)	2732280	837468	18.00
Control 11	with client | attack 9 (r =9E27)	2914156	654416	20.00
Control 12	with client | attack 10 (r =9E30)	2392704	1175756	22.00
Control 13	with client | attack 11 (r =9E33)	2865240	730360	24.00
Control 14	with client | attack 12 (r =9E36)	2273512	1322308	26.00
Control 15	with client | attack 13 (r =9E39)	2275488	1314332	28.00
Control 16	with client | attack 14 (r =9E42)	2264832	1324600	30.00
Control 17	with client | attack 15 (r =9E45)	2266536	1322688	32.00
Control 18	with client | attack 16 (r =9E48)	2895824	693056	34.00
Control 19	with client | attack 17 (r =9E51)	2267392	1321292	36.00
	Gradient	-8361.807018		

4.3 Evaluation

The *Software Defined Network [SDN]* controller is the core of the infrastructure, and is the most vulnerable point that would allow an attacker to bring down the network.

An initial technical survey, which targeted the Technical and Network Managers at *Internet Service Providers [ISPs]* in the region was done in order to guide:

- The practical approach in identifying solutions to the problem statement.
- The *Software Defined Network [SDN]* infrastructure set-up in the simulated environment.
- The relevance of the proposed solution to the *Internet Service Providers [ISPs]*

Table 4.3.1: Questionnaire results for ISPs on SDN.

SDN Questions to ISPs							
Company	Use SDN	Interest for SDN	Reservations to SDN	How is/would be SDN used	Protection mechanism used	Security threats concern	Desires from SDN
Airtel	NO	YES	Centralized control	Internet and Intranet	Open source IPS	Malware	Routing Traffic routing and QoS
Frontier Optical Networks [FON]	NO	YES	N/A	Internet	OEM Firewall	DDoS	Scalability
Jamii Telecommunications [JTL]	NO	YES	N/A	Cloud and Internet	OEM Firewall	MITM	Vendor neutrality
Liquid Telecom [LTK]	YES	YES	N/A	Cloud and Internet	SDN	Ransomware	N/A
MTN	YES	YES	Interoperability with OEM	Cloud and Internet	OEM Firewall	Ransomware	Local support
ROKE	YES	YES	N/A	Internet	SDN	DDoS	Reduced operational costs
Wananchi Telecom [WTL]	NO	YES	N/A	Internet and Intranet	Open source IPS	No	Show its practical use

From the displayed responses, following the questionnaires sent to the sampled lot of *Internet Service Providers [ISPs]*, there were several observations drawn.

The analysis revealed:

- All of the *Internet Service Provider [ISPs]* did have an interest in *Software Defined Networking [SDN]* features and would readily use it for internet applications.
- The *Internet Service Provider [ISPs]* that had already employed SDN in their infrastructure were 43% of the sample as illustrated in *Figure 4.3.1*

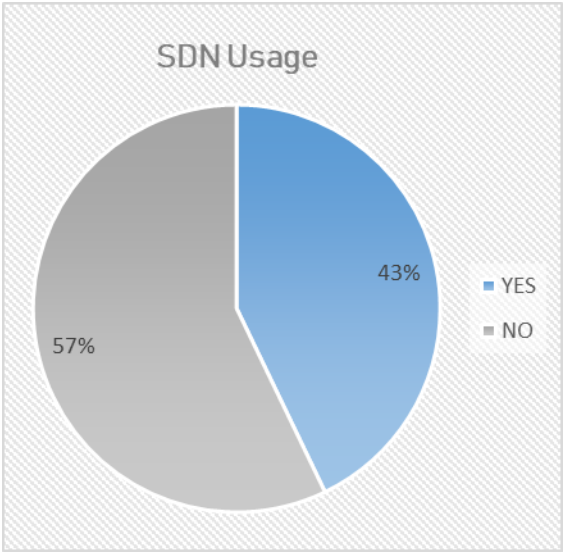


Figure 4.3.1: Pie chart showing Internet Service Provider [ISPs] that are using Software Defined Networking [SDN]

- Two out of the seven *Internet Service Provider [ISPs]*, making 29% of the sample, already employed *Software Defined Networking [SDN]* in their infrastructure to alleviate against attacks as depicted in *Graph 4.3.2*

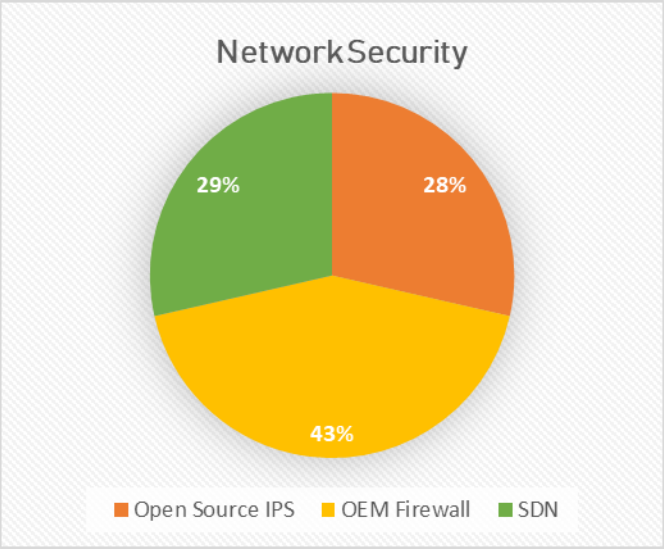


Figure 4.3.2: Pie chart showing the network security device used by the Internet Service Provider [ISPs]

- Two out of the seven *Internet Service Provider [ISPs]*, making 29% of the sample, considered *Distributed Denial of Service [DDoS]* to be their greatest security concern as depicted in *Graph 4.3.3*
- One from the seven *Internet Service Provider [ISPs]*, making 14% of the sample, was not concerned with security threats to their network as illustrated in *Graph 4.3.3*

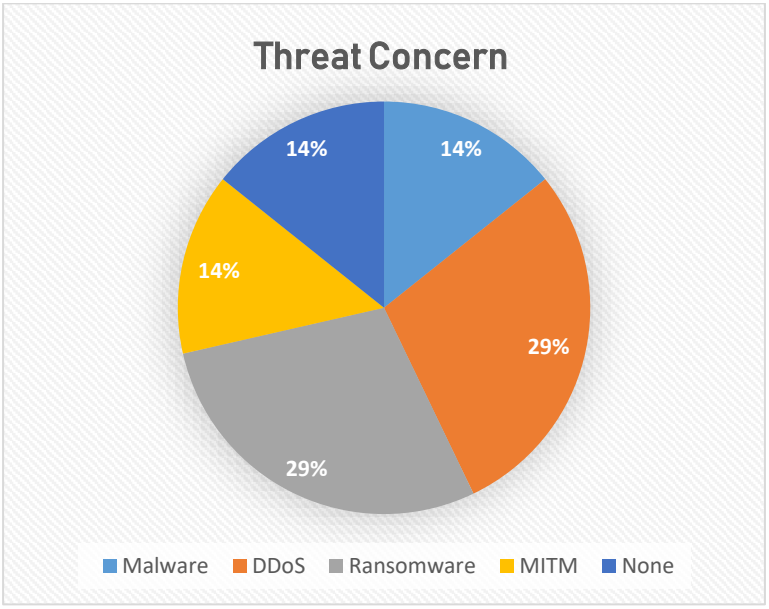


Figure 4.3.3: Pie chart showing the top-most security threat concerns to Internet Service Providers [ISPs]

The results showed an impetus of progression and an arising awareness around *Software Defined Networks [SDN]* in regard to network security.

The experiment was carried out in a simulated environment, where the architecture was configured to mimic the real world environment, and an attacker was introduced to test the robustness of the system.

ANOVA TESTS FROM 19 TREATMENTS

OPENFLOW RAM MEAN $[{}^{ram}_{ofp}X] = 2{,}817{,}952.158$

BGP RAM MEAN $[{}^{ram}_{bgp}X] = 2{,}571{,}128.421$

OPENFLOW RAM VARIANCE $[{}^{ram}_{ofp}\sigma^2] = 106{,}012{,}518{,}426.89$

BGP RAM VARIANCE $[{}^{ram}_{bgp}\sigma^2] = 85{,}659{,}719{,}308.39$

Table 4.3.2: ANOVA tests on RAM under OpenFlow and BGP.

	Test Environment	RAM OpenFlow		RAM BGP	
iterations	description	used	mean deviation [δ^2]	used	mean deviation [δ^2]
Control 1	no client | no attack	2245170	-	2607972	-
Control 2	with client | no attack	2343204	225385813424	3031708	212133548543
Control 3	with client | attack 1 (r =9E3)	2343120	225465578171	2765380	37733675924
Control 4	with client | attack 2 (r =9E6)	2346688	222089906516	2344876	51190158032
Control 5	with client | attack 3 (r =9E9)	2433448	147843447438	2339676	53570223211
Control 6	with client | attack 4 (r =9E12)	2757444	3661237172	2335328	55601838569
Control 7	with client | attack 5 (r =9E15)	2724736	8689252093	2971156	160022063918
Control 8	with client | attack 6 (r =9E18)	3327908	260054960897	2914228	117717321074
Control 9	with client | attack 7 (r =9E21)	2736956	6560377594	2393152	31675606451
Control 10	with client | attack 8 (r =9E24)	2827152	84637095	2732280	25969831397
Control 11	with client | attack 9 (r =9E27)	2827060	82952788	2914156	117667919918
Control 12	with client | attack 10 (r =9E30)	3017820	39947154308	2392704	31835274028
Control 13	with client | attack 11 (r =9E33)	3439232	385988642206	2865240	86501620871
Control 14	with client | attack 12 (r =9E36)	2874968	3250806251	2273512	88575534080
Control 15	with client | attack 13 (r =9E39)	2929972	12548445025	2275488	87403258560
Control 16	with client | attack 14 (r =9E42)	2926572	11798270099	2264832	93817497550
Control 17	with client | attack 15 (r =9E45)	3023560	42274584735	2266536	92776542963
Control 18	with client | attack 16 (r =9E48)	3144160	106411556251	2895824	105427218988
Control 19	with client | attack 17 (r =9E51)	3271921	206087709602	2267392	92255813474
	MEAN [X]	2817952.158		2571128.421	
	VAR [σ^2]	106012518425.89		85659719308.39	

From the two sets of experiments targeting the *Software Defined Network [SDN]* controller's performance under *OpenFlow Protoco*l versus *Border Gateway Protocol [BGP]*, a comparative analysis of the results as illustrated in *Testing 4.2* revealed that:

- The *Central Processing Unit [CPU]* 'user time' increased with a progressive increment of each attack under the OpenFlow environment, when compared to a decreased user time under *Border Gateway Protocol [BGP]* from similar progressive attacks.
- The *Random Access Memory [RAM]* used to provide the application support to counter the injected attack in the *OpenFlow* environment progressively increased with each increased attack on the *Software Defined Network [SDN]* controller. Under the *Border Gateway Protocol [BGP]*, on the other hand, the *Random Access Memory [RAM]* progressively reduced with every increased rate of injected attacks.

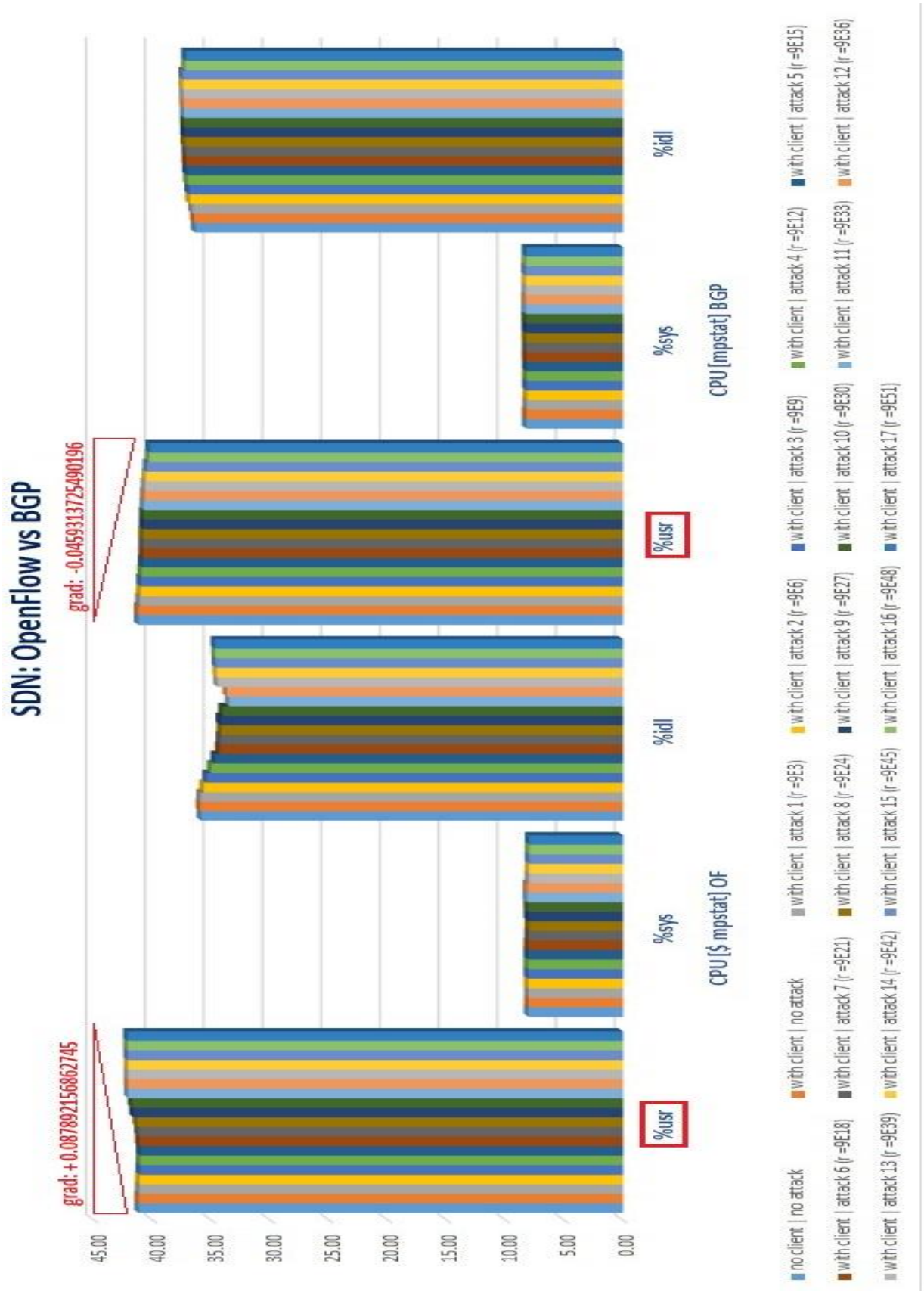


Figure 4.3.4: Comparison of CPU performance between OpenFlow and BGP after 19 control tests.

5 Chapter Five: Discussion

Following the evaluation of chapter four, *Internet Service Providers [ISPs]* are already in the know and have implemented *Software Defined Networks [SDN]*. Not only have they deployed *Software Defined Networks [SDN]* within their infrastructure, but also 29% of the sampled *Internet Service Providers [ISPs]*, use it as the main security gateway into the networks.

From these responses received, after the structured questionnaire issued to the *Internet Service Providers [ISPs]* and with a bearing of the categorized results, as other *ISPs* migrate/evolve their security gateways to the *Software Defined Networks [SDN]*, it is beneficial to analyze the security threats in this new architecture.

With the growing interest in Software Defined Networks, if the *Internet Service Provider [ISP]* is not aware of the inherent dangers in implementing the de facto *Software Defined Network [SDN]* under *OpenFlow protocol*, then they would be opening up their infrastructure to malicious attacks. The *Software Defined Network [SDN]*, by default, for southbound protocols, is configured under the *OpenFlow Protocol* to interconnect the network devices i.e. switches. The *OpenFlow Protocol*, though optimized for use in enhancing performance in processing times with the *Software Defined Network [SDN]* controller, has its vulnerabilities in terms of alleviating security threats in the network and controller domains. Therefore, for an effective and realized performance - processing time to security balance - the *OpenFlow Protocol* needs to be replaced by an alternative southbound layer protocol in *Border Gateway Protocol [BGP]*.

It is with the results, obtained from the laboratory experiment, comparing *Software Defined Network [SDN]* implementation via *OpenFlow Protocol* to *Border Gateway Protocol [BGP]* that the performance of processing times to security tests were tabulated. The *Border Gateway Protocol [BGP]*, proved to be resilient to the introduced *Distributed Denial of Service [DDoS]* attack as compared to the *OpenFlow Protocol*. The *Software Defined Network [SDN]* controller's *Central Processing Unit [CPU]* memory and processing capacities, under *Border Gateway Protocol [BGP]*, as a consequence, were not majorly compromised from the scaled flooding of *Internet Control Message Protocol [ICMP]* packets to the network. Under the *OpenFlow Protocol* implementation, the *Software Defined Network*

[SDN] controller's available memory and processor capacities were observed to progressively reduce with increased attacks.

Border Gateway Protocol [BGP], under the flavor of *FlowSpec* is a viable replacement to *OpenFlow Protocol* in the *Software Defined Network [SDN]* southbound layer due to its performance of processing times and security features. The implication being that *Internet Service Providers [ISPs]* implementing *Software Defined Network [SDN]* in their networks, either for internet services or cloud operations, require to secure the *Software Defined Network [SDN]* controller, and *Border Gateway Protocol [BGP]* is the better alternative when compared to *OpenFlow Protocol*.

Border Gateway Protocol [BGP], as a southbound protocol under *Software Defined Network [SDN]*, however as a limitation does come with its cons:

- A detailed configuration to define *BGP* properties under *OpenDaylight*, may throw off the network engineer doing the optimization due to the complexity in integration.
- When compared to *OpenFlow Protocol* in the performance of processing time measure, *BGP* comes in at second place. *OpenFlow Protocol* has been well engineered for the greater elements of performance, as tested from the simulated environment.

6 Chapter Six: Conclusion

The research project sought to obtain results that determined the security threshold with respect to the performance of the *Software Defined Network [SDN]* controller by comparing the *OpenFlow Protocol* to the *Border Gateway Protocol [BGP]* on the southbound layer in the presence of an attack on the controller.

With the influx of cyber-attacks and the emerging generational technologies that are demanding for automated infrastructure, improving the *Software Defined Network [SDN]* controller channel is paramount to the operations of the *Network Facilities Provider [NFP]* i.e. *Internet Service Providers [ISPs]* in this case.

The research did investigate the *Software Defined Network [SDN]* performance to security trade-offs and analyze through simulated tests that compared the default *OpenFlow Protocol* and the *Border Gateway Protocol [BGP] FlowSpec* in the south-bound *Application Programming Interface [API]*. The tests provided an exploratory and practical insight into the security-performance gap in the current deployment of *Software Defined Network [SDN]* infrastructure.

The lab set-up was set-up successfully to allow for tests to be run by having in place:

- A simulation and configuration of the *Software Defined Network [SDN]* architecture in the lab environment.
- An integration and introduction of an attacker that would attempt to bring down the controller.
- An incorporation and tuning of a monitoring device to capture the output from the *Software Defined Network [SDN]* controller under attack.

From the tests run under the specified control environments, it was shown that the Software Defined Network controller is better secured against *Distributed Denial of Service [DDoS]* attacks by employing the *Border Gateway Protocol [BGP]* in place of the *OpenFlow Protocol*. The performance is measured with reference to the injected attack packets.

7 Recommendations

From the research carried out and implementation from the study:

- The *OpenFlow protocol* is a better performer when compared to the *Border Gateway Protocol [BGP]*.
- The *Border Gateway Protocol [BGP]* is the far superior protocol in security when compared to *OpenFlow protocol*.
- The *Border Gateway Protocol [BGP]* is the better option for the ISP that has or is planning on deploying *Software Defined Network [SDN]* in their network.

The *Internet Service Provider [ISP]* would be better off having *Border Gateway Protocol [BGP]* as a southbound protocol as compared to the default *OpenFlow Protocol*. This enhances and improves on security in the *Internet Service Provider [ISP]* network and still manages to support the performance requirements for operations.

For further research, the recommendation is to identify a *Border Gateway Protocol [BGP]* flavor that would still offer an improved security parameter as is the case with *FlowSpec*, and a greater performance parameter when compared to *OpenFlow protocol.*

8 References

- AKAMAI, (2016), DDoS Attacks, https://www.akamai.com/us/en/resources/ddos-attacks.jsp
- Bi J. et al, (2012), The Challenges of SDN/OpenFlow in an Operational & Large Scale Network.
- Braga R. et al, (2010), Lightweight DDoS Flooding Attack Detection Using NOX – OpenFlow.
- Buraglio N., (2015), SDN: Theory vs. Practice, CODASPY 2016 SDN/NFV workshop, Energy Sciences Network [ESnet].
- Butler K. et al, (2015), A Survey of BGP Security Issues and Solutions.
- CISCO, (2016), BGP, http://www.cisco.com/c/en/us/products/ios-nx-os-software/border-gateway-protocol-bgp/index.html
- Conran M. et al, (2016), BGP has a New Friend – BGP Based SDN.
- Chin T. et al, (2016), An SDN Supported Collaborative Approach for DDoS Flooding Detection and Containment.
- Giotis K. et al, (2013), Combining OpenFlow and SFlow for an Effective and Scalable Anomaly Detection and Mitigation Mechanism on SDN Environments.
- Lychev R. et al, (2013), BGP Security in Partial Deployment.
- Markus N., (2013), SDN – What Can You Do With It In The Enterprise, https://www.sdxcentral.com/articles/contributed/sdn-markus-nispel/2013/04/
- Medved J. et al, (2014), OpenDayLight: Towards a Model Driven SDN Controller Architecture, IEEE, 15th International Symposium in a World of Wireless, Mobile and Multimedia Networks.
- Oates B., (2006), Researching Information Systems and Computing, SAGE publications ltd.
- Ryburn J. et al, (2015), DDoS Mitigation using BGP FlowSpec [Juniper]
- Gupta D. et al, (2015), Inter SDN Controller Communication.
- Savage M. et al, (2015), OpenFlow faces Interoperability Challenges.
- Remes W. et al, (2014), Internet Plumbing for Security Professionals: The State of BGP Security.
- Rexford J., (2012), Software Defined Networking, http://www.cs.princeton.edu/courses/archive/spr12/cos461
- TechTARGET, (2016), Limitations of SDN, http://searchservervitualization.techtarget.com
- Wang H. et al, (2015), Flood Guard: A DDoS Attack Prevention Extension in SDN.

Appendices

A. Questionnaire Form: SDN infrastructure implementation.

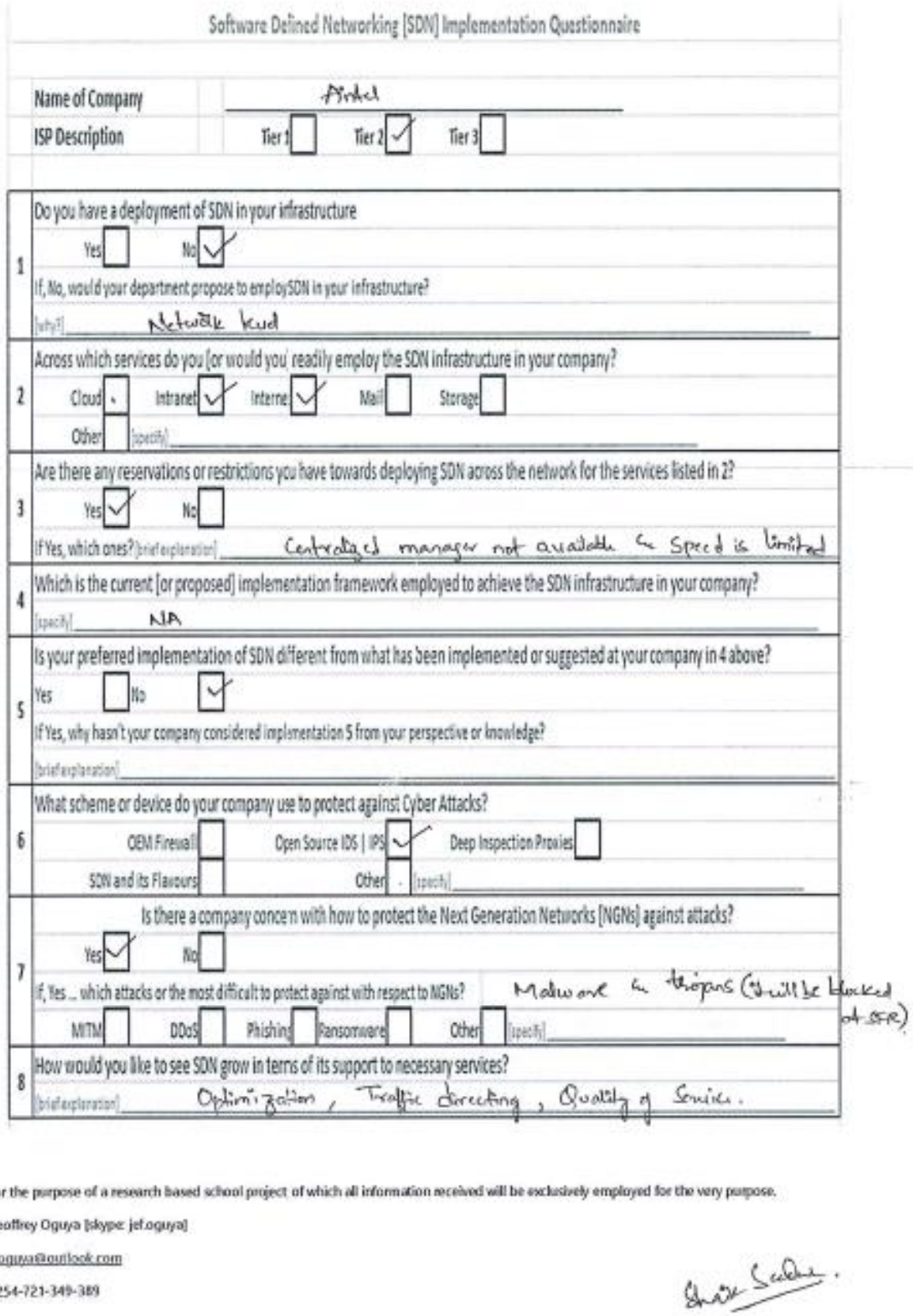
Software Defined Networking [SDN] Implementation Questionnaire

Name of Company _____

ISP Description Tier 1 ☐ Tier 2 ☐ Tier 3 ☐

1	Do you have a deployment of SDN in your infrastructure Yes ☐ No ☐ If, No, would your department propose to employ SDN in your infrastructure? [why?] _____
2	Across which services do you [or would you] readily employ the SDN infrastructure in your company? Cloud ☐ Intranet ☐ Internet ☐ Mail ☐ Storage ☐ Other ☐ [specify] _____
3	Are there any reservations or restrictions you have towards deploying SDN across the network for the services listed in 2? Yes ☐ No ☐ If Yes, which ones? [brief explanation] _____
4	Which is the current [or proposed] implementation framework employed to achieve the SDN infrastructure in your company? [specify] _____
5	Is your preferred implementation of SDN different from what has been implemented or suggested at your company in 4 above? Yes ☐ No ☐ If Yes, why hasn't your company considered implementation 5 from your perspective or knowledge? [brief explanation] _____
6	What scheme or device do your company use to protect against Cyber Attacks? OEM Firewall ☐ Open Source IDS | IPS ☐ Deep Inspection Proxies ☐ SDN and its Flavours ☐ Other ☐ [specify] _____
7	Is there a company concern with how to protect the Next Generation Networks [NGNs] against attacks? Yes ☐ No ☐ If, Yes ... which attacks or the most difficult to protect against with respect to NGNs? MITM ☐ DDoS ☐ Phishing ☐ Ransomware ☐ Other ☐ [specify] _____
8	How would you like to see SDN grow in terms of its support to necessary services? [brief explanation] _____

Figure 8.1: SDN ISP Questionnaire Form



Software Defined Networking [SDN] Implementation Questionnaire

Name of Company: Airtel

ISP Description: Tier 1 ☐ Tier 2 ☑ Tier 3 ☐

1. Do you have a deployment of SDN in your infrastructure
   Yes ☐ No ☑
   If, No, would your department propose to employ SDN in your infrastructure?
   [why?] Network load

2. Across which services do you [or would you] readily employ the SDN infrastructure in your company?
   Cloud ☐ Intranet ☑ Internet ☑ Mail ☐ Storage ☐
   Other ☐ [specify]

3. Are there any reservations or restrictions you have towards deploying SDN across the network for the services listed in 2?
   Yes ☑ No ☐
   If Yes, which ones? [brief explanation] Centralized manager not available & Speed is limited

4. Which is the current [or proposed] implementation framework employed to achieve the SDN infrastructure in your company?
   [specify] NA

5. Is your preferred implementation of SDN different from what has been implemented or suggested at your company in 4 above?
   Yes ☐ No ☑
   If Yes, why hasn't your company considered implementation 5 from your perspective or knowledge?
   [brief explanation]

6. What scheme or device do your company use to protect against Cyber Attacks?
   OEM Firewall ☐ Open Source IDS | IPS ☑ Deep Inspection Proxies ☐
   SDN and its Flavours ☐ Other ☐ [specify]

7. Is there a company concern with how to protect the Next Generation Networks [NGNs] against attacks?
   Yes ☑ No ☐
   If, Yes ... which attacks or the most difficult to protect against with respect to NGNs? Malware & Trojans (will be blocked at SFR)
   MITM ☐ DDoS ☐ Phishing ☐ Ransomware ☐ Other ☐ [specify]

8. How would you like to see SDN grow in terms of its support to necessary services?
   [brief explanation] Optimization, Traffic directing, Quality of Service.

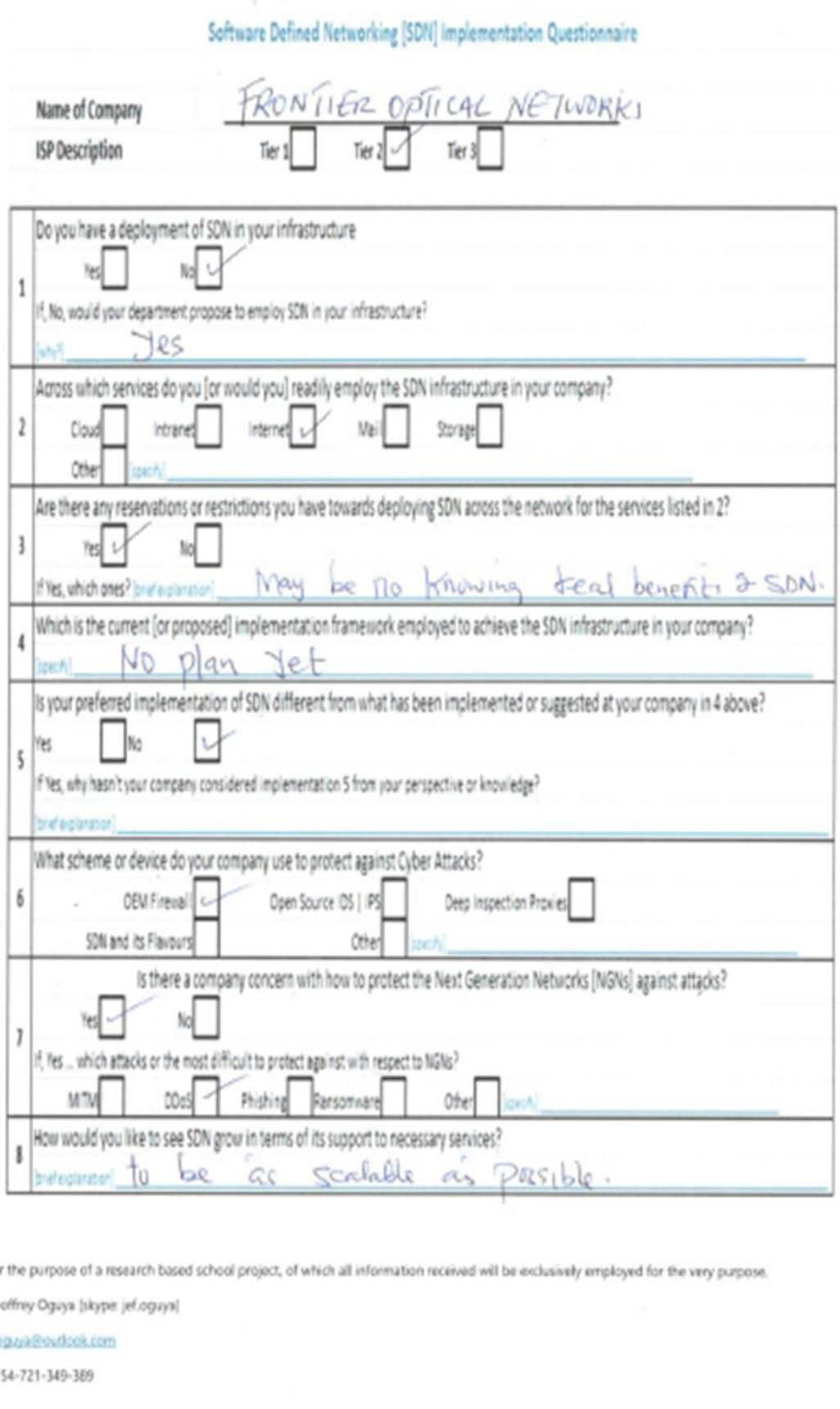
For the purpose of a research based school project of which all information received will be exclusively employed for the very purpose.

Geoffrey Oguya [skype: jef.oguya]

g.oguya@outlook.com

+254-721-349-389

Figure 8.2: SDN ISP Questionnaire Form – AIRTEL



Software Defined Networking [SDN] Implementation Questionnaire

Name of Company: FRONTIER OPTICAL NETWORKS

ISP Description: Tier 1 ☐ Tier 2 ☑ Tier 3 ☐

1. Do you have a deployment of SDN in your infrastructure
   Yes ☐ No ☑
   If, No, would your department propose to employ SDN in your infrastructure?
   [why] Yes

2. Across which services do you [or would you] readily employ the SDN infrastructure in your company?
   Cloud ☐ Intranet ☐ Internet ☑ Mail ☐ Storage ☐
   Other ☐ [specify]

3. Are there any reservations or restrictions you have towards deploying SDN across the network for the services listed in 2?
   Yes ☑ No ☐
   If Yes, which ones? [brief explanation] May be no knowing real benefits of SDN.

4. Which is the current [or proposed] implementation framework employed to achieve the SDN infrastructure in your company?
   [specify] No plan Yet

5. Is your preferred implementation of SDN different from what has been implemented or suggested at your company in 4 above?
   Yes ☐ No ☑
   If Yes, why hasn't your company considered implementation 5 from your perspective or knowledge?
   [brief explanation]

6. What scheme or device do your company use to protect against Cyber Attacks?
   OEM Firewall ☑ Open Source IDS | IPS ☐ Deep Inspection Proxies ☐
   SDN and its Flavours ☐ Other ☐ [specify]

7. Is there a company concern with how to protect the Next Generation Networks [NGNs] against attacks?
   Yes ☑ No ☐
   If, Yes .. which attacks or the most difficult to protect against with respect to NGNs?
   MITM ☐ DDoS ☑ Phishing ☐ Ransomware ☐ Other ☐ [specify]

8. How would you like to see SDN grow in terms of its support to necessary services?
   [brief explanation] to be as scalable as possible.

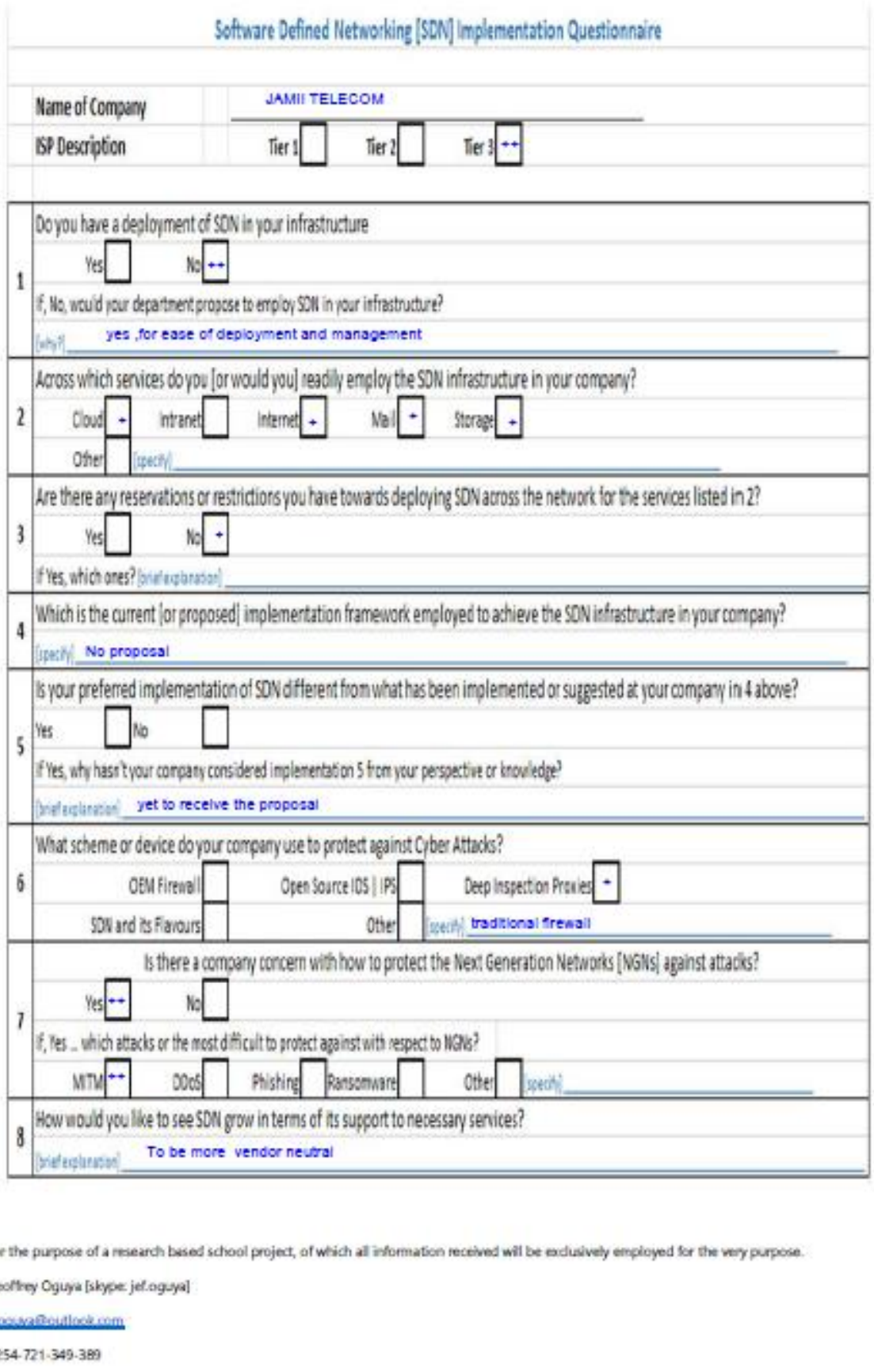
For the purpose of a research based school project, of which all information received will be exclusively employed for the very purpose.

Geoffrey Oguya [skype: jef.oguya]

g.oguya@outlook.com

+254-721-349-389

Figure 8.3: SDN ISP Questionnaire Form – Frontier Optical Networks [FON]



Software Defined Networking [SDN] Implementation Questionnaire

	Name of Company	JAMII TELECOM
	ISP Description	Tier 1 [ ] Tier 2 [ ] Tier 3 [++]
1	Do you have a deployment of SDN in your infrastructure Yes [ ] No [++] If, No, would your department propose to employ SDN in your infrastructure? [why?] yes ,for ease of deployment and management	
2	Across which services do you [or would you] readily employ the SDN infrastructure in your company? Cloud [+] Intranet [ ] Internet [+] Mail [+] Storage [+] Other [ ] [specify]	
3	Are there any reservations or restrictions you have towards deploying SDN across the network for the services listed in 2? Yes [ ] No [+] If Yes, which ones? [brief explanation]	
4	Which is the current [or proposed] implementation framework employed to achieve the SDN infrastructure in your company? [specify] No proposal	
5	Is your preferred implementation of SDN different from what has been implemented or suggested at your company in 4 above? Yes [ ] No [ ] If Yes, why hasn't your company considered implementation 5 from your perspective or knowledge? [brief explanation] yet to receive the proposal	
6	What scheme or device do your company use to protect against Cyber Attacks? OEM Firewall [ ] Open Source IDS | IPS [ ] Deep Inspection Proxies [+] SDN and its Flavours [ ] Other [ ] [specify] traditional firewall	
7	Is there a company concern with how to protect the Next Generation Networks [NGNs] against attacks? Yes [++] No [ ] If, Yes .. which attacks or the most difficult to protect against with respect to NGNs? MITM [++] DDoS [ ] Phishing [ ] Ransomware [ ] Other [ ] [specify]	
8	How would you like to see SDN grow in terms of its support to necessary services? [brief explanation] To be more vendor neutral	

For the purpose of a research based school project, of which all information received will be exclusively employed for the very purpose.

Geoffrey Oguya [skype: jef.oguya]

g.oguya@outlook.com

+254-721-349-389

Figure 8.4: SDN ISP Questionnaire Form – Jamii Telecommunications Ltd [JTL]

Software Defined Networking [SDN] Implementation Questionnaire

Name of Company: Liquid Telecom

ISP Description: Tier 1 [X] Tier 2 [ ] Tier 3 [ ]

1. Do you have a deployment of SDN in your infrastructure
   Yes [X] No [ ]
   If, No, would your department propose to employ SDN in your infrastructure?
   [why?] _____

2. Across which services do you [or would you] readily employ the SDN infrastructure in your company?
   Cloud [X] Intranet [X] Internet [X] Mail [ ] Storage [X]
   Other [ ] [specify] _____

3. Are there any reservations or restrictions you have towards deploying SDN across the network for the services listed in 2?
   Yes [ ] No [X]
   If Yes, which ones? [brief explanation] _____

4. Which is the current [or proposed] implementation framework employed to achieve the SDN infrastructure in your company?
   [specify] Segmented Routing

5. Is your preferred implementation of SDN different from what has been implemented or suggested at your company in 4 above?
   Yes [ ] No [X]
   If Yes, why hasn't your company considered implementation 5 from your perspective or knowledge?
   [brief explanation] _____

6. What scheme or device do your company use to protect against Cyber Attacks?
   OEM Firewall [ ] Open Source IDS | IPS [ ] Deep Inspection Proxies [ ]
   SDN and its Flavours [X] Other [ ] [specify] _____

7. Is there a company concern with how to protect the Next Generation Networks [NGNs] against attacks?
   Yes [ ] No [X]
   If, Yes ... which attacks or the most difficult to protect against with respect to NGNs?
   MITM [ ] DDoS [ ] Phishing [ ] Ransomware [ ] Other [ ] [specify] _____

8. How would you like to see SDN grow in terms of its support to necessary services?
   [brief explanation] not relevant at the moment

For the purpose of a research based school project, of which all information received will be exclusively employed for the very purpose.

Geoffrey Oguya [skype: jef.oguya]

g.oguya@outlook.com

+254-721-349-389

Figure 8.5: SDN ISP Questionnaire Form – Liquid Telecom [LTK]

Software Defined Networking [SDN] Implementation Questionnaire

	Name of Company	MTN Uganda Tier 1
	ISP Description	Tier 1 [ ] Tier 2 [ ] Tier 3 [ ]
1	Do you have a deployment of SDN in your infrastructure Yes [X] No [ ] If, No, would your department propose to employ SDN in your infrastructure? [why?]	
2	Across which services do you [or would you] readily employ the SDN infrastructure in your company? Cloud [X] Intranet [X] Internet [X] Mail [ ] Storage [ ] Other [ ] [specify]	
3	Are there any reservations or restrictions you have towards deploying SDN across the network for the services listed in 2? Yes [X] No [ ] If Yes, which ones? [brief explanation]	integration with multivendor infra
4	Which is the current [or proposed] implementation framework employed to achieve the SDN infrastructure in your company? [specify]	NFV
5	Is your preferred implementation of SDN different from what has been implemented or suggested at your company in 4 above? Yes [ ] No [X] If Yes, why hasn't your company considered implementation 5 from your perspective or knowledge? [brief explanation]	
6	What scheme or device do your company use to protect against Cyber Attacks? OEM Firewall [X] Open Source IDS | IPS [ ] Deep Inspection Proxies [X] SDN and its Flavours [ ] Other [ ] [specify]	
7	Is there a company concern with how to protect the Next Generation Networks [NGNs] against attacks? Yes [X] No [ ] If, Yes ... which attacks or the most difficult to protect against with respect to NGNs? MITM [ ] DDoS [ ] Phishing [X] Ransomware [X] Other [ ] [specify]	
8	How would you like to see SDN grow in terms of its support to necessary services? [brief explanation]	local presence for vendor support

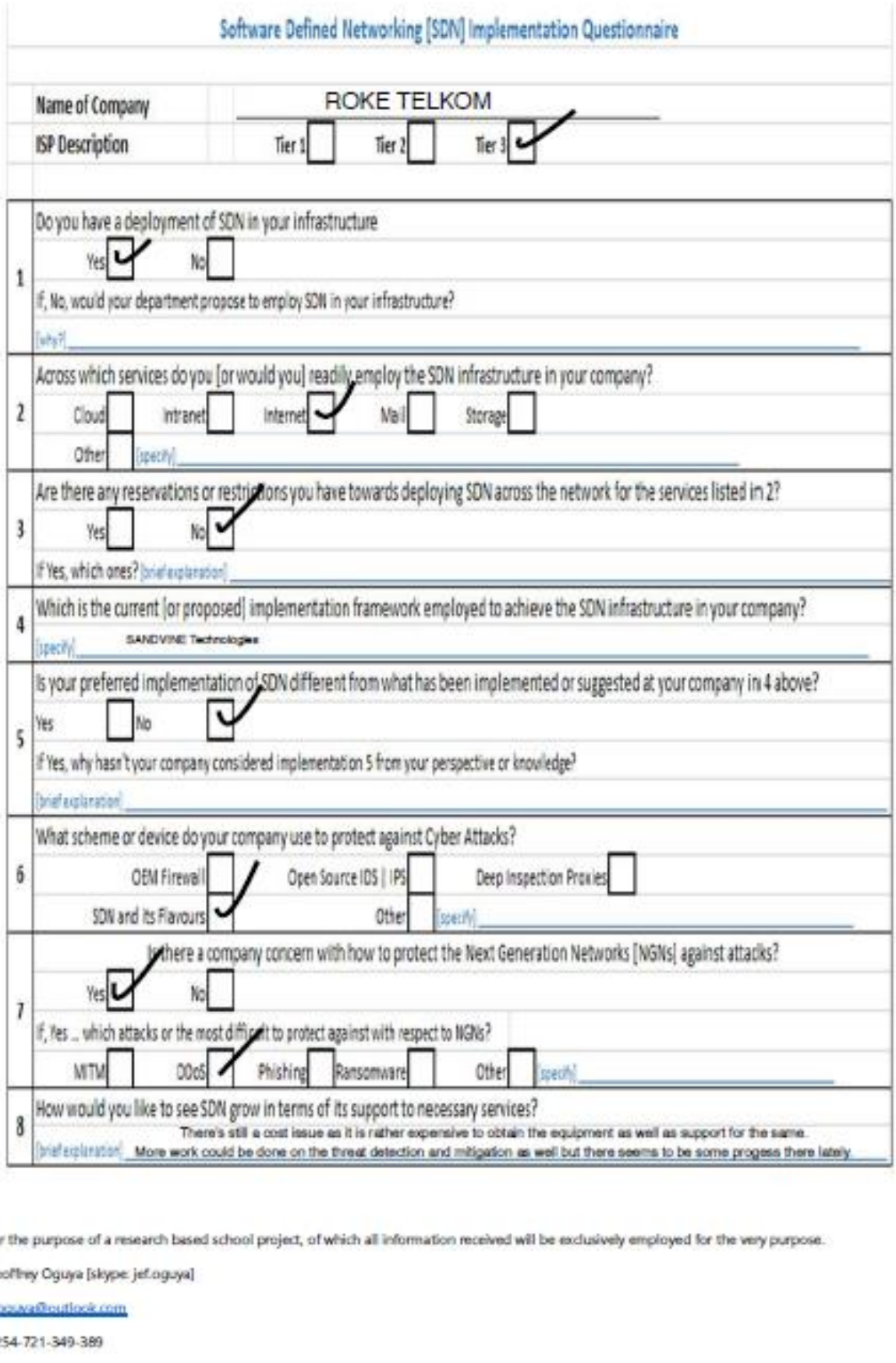
For the purpose of a research based school project, of which all information received will be exclusively employed for the very purpose.

Geoffroy Oguya [skype: jef.oguya]

g.oguya@outlook.com

+254-721-349-389

Figure 8.6: SDN ISP Questionnaire Form – MTN



Software Defined Networking [SDN] Implementation Questionnaire

Name of Company: ROKE TELKOM

ISP Description: Tier 1 ☐ Tier 2 ☐ Tier 3 ☑

1. Do you have a deployment of SDN in your infrastructure
   Yes ☑ No ☐
   If, No, would your department propose to employ SDN in your infrastructure?
   [why?]

2. Across which services do you [or would you] readily employ the SDN infrastructure in your company?
   Cloud ☐ Intranet ☐ Internet ☑ Mail ☐ Storage ☐
   Other ☐ [specify]

3. Are there any reservations or restrictions you have towards deploying SDN across the network for the services listed in 2?
   Yes ☐ No ☑
   If Yes, which ones? [brief explanation]

4. Which is the current [or proposed] implementation framework employed to achieve the SDN infrastructure in your company?
   [specify] SANDVINE Technologies

5. Is your preferred implementation of SDN different from what has been implemented or suggested at your company in 4 above?
   Yes ☐ No ☑
   If Yes, why hasn't your company considered implementation 5 from your perspective or knowledge?
   [brief explanation]

6. What scheme or device do your company use to protect against Cyber Attacks?
   OEM Firewall ☐ Open Source IDS | IPS ☐ Deep Inspection Proxies ☐
   SDN and its Flavours ☑ Other ☐ [specify]

7. Is there a company concern with how to protect the Next Generation Networks [NGNs] against attacks?
   Yes ☑ No ☐
   If, Yes ... which attacks or the most difficult to protect against with respect to NGNs?
   MITM ☐ DDoS ☑ Phishing ☐ Ransomware ☐ Other ☐ [specify]

8. How would you like to see SDN grow in terms of its support to necessary services?
   [brief explanation] There's still a cost issue as it is rather expensive to obtain the equipment as well as support for the same. More work could be done on the threat detection and mitigation as well but there seems to be some progess there lately

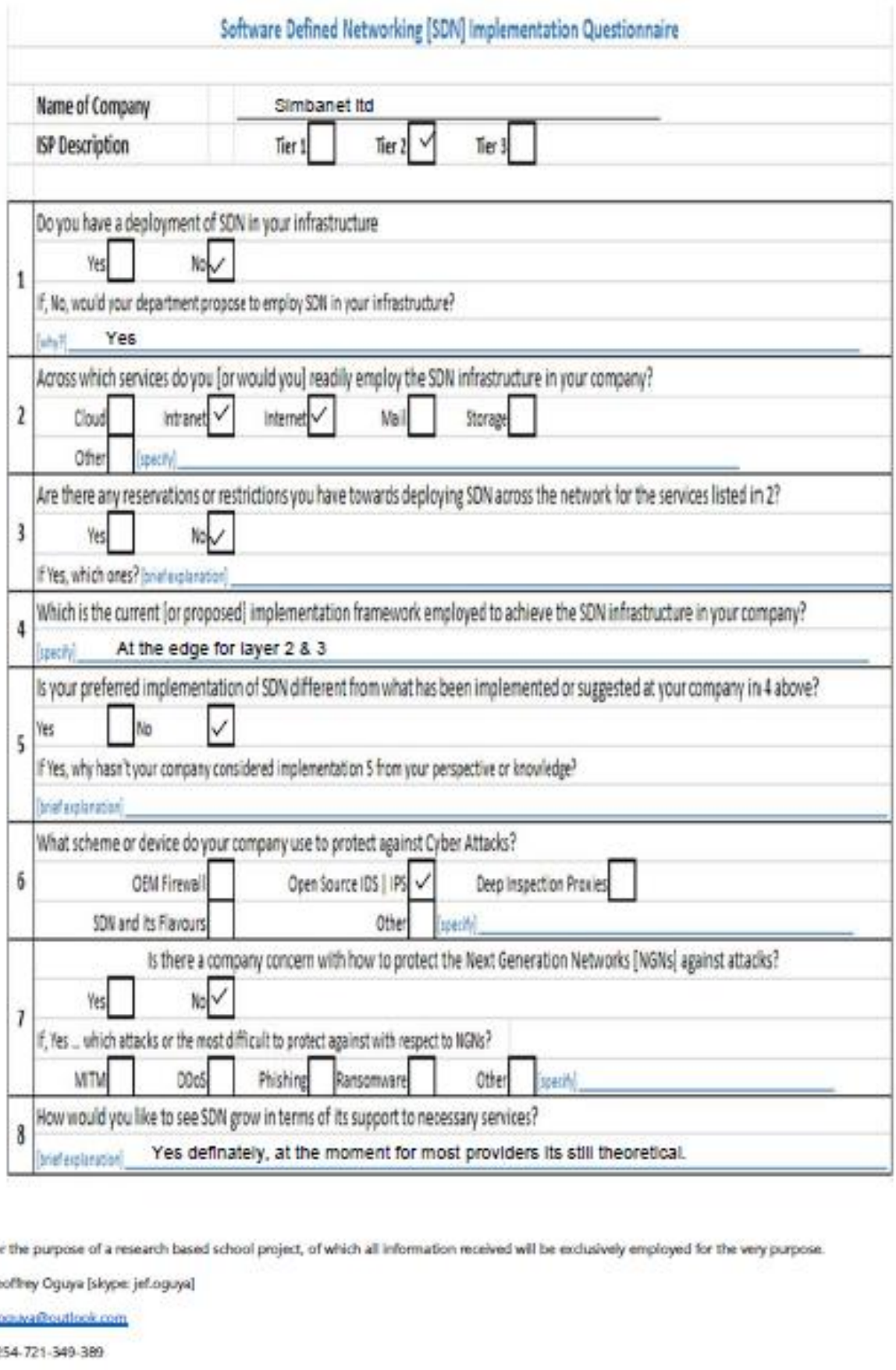
For the purpose of a research based school project, of which all information received will be exclusively employed for the very purpose.

Geoffrey Oguya [skype: jef.oguya]

o.oguya@outlook.com

+254-721-349-389

Figure 8.7: SDN ISP Questionnaire Form – ROKE



Software Defined Networking [SDN] Implementation Questionnaire

Name of Company: Simbanet ltd

ISP Description: Tier 1 [ ] Tier 2 [✓] Tier 3 [ ]

1. Do you have a deployment of SDN in your infrastructure
   Yes [ ] No [✓]
   If, No, would your department propose to employ SDN in your infrastructure?
   [why?] Yes

2. Across which services do you [or would you] readily employ the SDN infrastructure in your company?
   Cloud [ ] Intranet [✓] Internet [✓] Mail [ ] Storage [ ]
   Other [ ] [specify]

3. Are there any reservations or restrictions you have towards deploying SDN across the network for the services listed in 2?
   Yes [ ] No [✓]
   If Yes, which ones? [brief explanation]

4. Which is the current [or proposed] implementation framework employed to achieve the SDN infrastructure in your company?
   [specify] At the edge for layer 2 & 3

5. Is your preferred implementation of SDN different from what has been implemented or suggested at your company in 4 above?
   Yes [ ] No [✓]
   If Yes, why hasn't your company considered implementation 5 from your perspective or knowledge?
   [brief explanation]

6. What scheme or device do your company use to protect against Cyber Attacks?
   OEM Firewall [ ] Open Source IDS | IPS [✓] Deep Inspection Proxies [ ]
   SDN and its Flavours [ ] Other [ ] [specify]

7. Is there a company concern with how to protect the Next Generation Networks [NGNs] against attacks?
   Yes [ ] No [✓]
   If, Yes .. which attacks or the most difficult to protect against with respect to NGNs?
   MITM [ ] DDoS [ ] Phishing [ ] Ransomware [ ] Other [ ] [specify]

8. How would you like to see SDN grow in terms of its support to necessary services?
   [brief explanation] Yes definately, at the moment for most providers its still theoretical.

For the purpose of a research based school project, of which all information received will be exclusively employed for the very purpose.

Geoffrey Oguya [skype: jef.oguya]

g.oguya@outlook.com

+254-721-349-389

Figure 8.8: SDN ISP Questionnaire Form – Wananchi Telecommunications Ltd [WTL]

B. Configuration: SDN architecture and set-up.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- vi: set et smarttab sw=4 tabstop=4: -->
<!--
<snapshot>
    <required-capabilities>
        <!-- openflowjava -->
        <capability>
            urn:opendaylight:params:xml:ns:yang:openflow:switch:connection:provider:impl?module=openflow-switch-connection-provider-impl&amp;revision=2014-03-28
        </capability>
        <capability>
            urn:opendaylight:params:xml:ns:yang:openflow:switch:connection:provider?module=openflow-switch-connection-provider&amp;revision=2014-03-28
        </capability>
        <!-- openflowplugin -->
        <capability>urn:opendaylight:params:xml:ns:yang:config:openflow:plugin:impl?module=openflow-plugin-provider-impl&amp;revision=2015-03-27</capability>
        <capability>urn:opendaylight:params:xml:ns:yang:openflow:api?module=openflow-provider&amp;revision=2015-03-31</capability>
        <capability>urn:opendaylight:params:xml:ns:yang:openflowplugin:extension:api?module=openflowplugin-extension-registry&amp;revision=2015-04-25</capability>
        <!-- binding-broker-impl - provided -->
    </required-capabilities>

    <configuration>

        <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
            <modules xmlns="urn:opendaylight:params:xml:ns:yang:controller:config">
                <!-- default OF-switch-connection-provider (port 6633) -->
                <module>
                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:openflow:switch:connection:provider:impl">
                        prefix:openflow-switch-connection-provider-impl
                    </type>
                    <name>openflow-switch-connection-provider-default-impl</name>
                    <port>6633</port>
                    <!--      Possible transport-protocol options: TCP, TLS, UDP -->
                    <transport-protocol>TCP</transport-protocol>
                    <switch-idle-timeout>15000</switch-idle-timeout>
                </module>
                <!-- default OF-switch-connection-provider (port 6653) -->
                <module>
                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:openflow:switch:connection:provider:impl">
                    <name>openflow-switch-connection-provider-legacy-impl</name>
                    <port>6653</port>
                    <!--      Possible transport-protocol options: TCP, TLS, UDP -->
                    <transport-protocol>TCP</transport-protocol>
                    <switch-idle-timeout>15000</switch-idle-timeout>
                </module>


                <module>
            </modules>

            <services xmlns="urn:opendaylight:params:xml:ns:yang:controller:config">
        </data>

    </configuration>
```

Figure 9.1: SDN south-bound configuration with OpenFlow Protocol

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- vi: set et smarttab sw=4 tabstop=4: -->
<!--
<snapshot>
    <required-capabilities>
    <configuration>

        <data xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
            <modules xmlns="urn:opendaylight:params:xml:ns:yang:controller:config">
                <module>
                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-peer-acceptor</type>
                    <name>bgp-peer-server</name>

                    <!--Default parameters-->
                    <binding-address>192.168.245.132</binding-address>

                    <!--Default binding-port 179-->
                    <binding-port>179</binding-port>

                    <accepting-bgp-dispatcher>
                        <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-dispatcher</type>
                        <name>global-bgp-dispatcher</name>
                    </accepting-bgp-dispatcher>
                    <accepting-peer-registry>
                        <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-peer-registry</type>
                        <name>global-bgp-peer-registry</name>
                    </accepting-peer-registry>
                </module>

                <!--<module>
                <!--<module>
                <!--<module>

                <!--
        <module>

        <module>
                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-peer</type>
                    <name>example-bgp-peer</name>
                    <host>192.168.245.136</host>
                    <holdtimer>180</holdtimer>
                    <retrytimer>10</retrytimer>
            <remote-as>64404</remote-as>
                    <peer-role>ibgp</peer-role>
                    <rib>
                        <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:rib-instance</type>
                        <name>example-bgp-rib</name>
                    </rib>
                    <peer-registry>
                    <rpc-registry>
                    <advertized-table>
                    <advertized-table>
```

Figure 9.2.1: SDN south-bound configuration with Border Gateway Protocol FlowSpec

```
179            <module>
180                <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:rib-impl</type>
181                <name>example-bgp-rib</name>
182                <rib-id>example-bgp-rib</rib-id>
183                <local-as>64404</local-as>
184                <bgp-rib-id>192.168.245.132</bgp-rib-id>
185                <!-- if cluster-id is not present, it's value is the same as bgp-id -->
186                <!-- <cluster-id>192.0.2.3</cluster-id> -->
187                <local-table>
191                <local-table>
195                <local-table>
199                <local-table>
200                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-table-type</type>
201                    <name>ipv4-flowspec</name>
202                </local-table>
203                <local-table>
207                <local-table>
208                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-table-type</type>
209                    <name>ipv4-flowspec-l3vpn</name>
210                </local-table>
211                <local-table>
215                <local-table>
216                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-table-type</type>
217                    <name>ipv4-labeled-unicast</name>
218                </local-table>
219                <local-table>
223                <local-table>
224                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-table-type</type>
225                    <name>ipv4-l3vpn</name>
226                </local-table>
227                <local-table>
231                <local-table>
235                <!--<rib-path-selection-mode>
236                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-path-selection-mode</type>
237                    <name>ipv4-unicast-path-selection-mode</name>
238                </rib-path-selection-mode>-->
239                <extensions>
240                    <type xmlns:ribspi="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:spi">ribspi:extensions</type>
241                    <name>global-rib-extensions</name>
242                </extensions>
243                <bgp-dispatcher>
244                    <type xmlns:prefix="urn:opendaylight:params:xml:ns:yang:controller:bgp:rib:impl">prefix:bgp-dispatcher</type>
245                    <name>global-bgp-dispatcher</name>
246                </bgp-dispatcher>
247                <data-provider>
248                    <type xmlns:binding="urn:opendaylight:params:xml:ns:yang:controller:md:sal:binding">binding:binding-async-data-broker</type>
249                    <name>pingpong-binding-data-broker</name>
250                </data-provider>
251                <dom-data-provider>
252                    <type xmlns:sal="urn:opendaylight:params:xml:ns:yang:controller:md:sal:dom">sal:dom-async-data-broker</type>
253                    <name>pingpong-broker</name>
254                </dom-data-provider>
```

Figure 9.2.2: SDN south-bound configuration with Border Gateway Protocol FlowSpec

C. Configuration: DDoS attack under OpenFlow and BGP

root@smaug:/home/smaug# bonesi -v -p udp -s 1472 -r 9000000 -i bonesi-master/245_subnet.txt 192.168.245.132:6633

```
dstIp:       192.168.245.132
dstPort:     6633
protocol:    17
payloadSize: 1472
rate:        9000000
ips:         bonesi-master/245_subnet.txt
urls:        (null)
useragents:: (null)
stats file:  stats
device:      (null)
maxPackets:  infinite
format:      dotted
toggle:      no
reading file...done
Size of url array: 1
www.google.de/
Number of Useragents: 1
Useragent[0]: Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.8.1.8)
71187 packets in 1.000015 seconds
```

Figure 10.1.1: DDoS attack using UDP flood to test performance of OpenFlow Protocol

```
root@smaug:/home/smaug# bonesi -v -tcp -s 1473 -m 9000000000000 -r 0 -i bonesi-master/245_subnet.txt -d eth0
192.168.245.132:179

dstIp:       192.168.245.132
dstPort:     179
protocol:    6
payloadSize:  1473
MTU:         2043514880
fragment mode: IP
rate:        infinite
ips:         bonesi-master/245_subnet.txt
urls:        (null)
useragents::  (null)
stats file:   stats
device:      eth0
maxPackets:   infinite
format:      dotted
toggle:      no
reading file...done
Size of url array: 1
www.google.de/
Number of Useragents: 1
Useragent[0]: Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.8.1.8)
```

Figure 10.1.2: DDoS attack using TCP flood to test performance of Border Gateway Protocol

D. Configuration: BGP set-up on Quagga

root@smppWAN:/home/smaug# vtysh

smppWAN# show bgp neighbors

BGP neighbor is 192.168.245.132, remote AS 64404, local AS 64404, internal link

BGP version 4, remote router ID 192.168.245.132

BGP state = Established, up for 00:36:01

Last read 00:00:01, hold time is 180, keepalive interval is 60 seconds

Neighbor capabilities:

4 Byte AS: advertised and received

Route refresh: advertised and received (new)

Address family IPv4 Unicast: advertised and received

Address family VPNv4 Unicast: received

Address family IPv6 Unicast: received

Address family Unknown: received

Graceful Restart Capabilty: advertised

Message statistics:

Inq depth is 0

Outq depth is 0

	Sent	Rcvd
Opens:	3	1
Notifications:	0	1
Updates:	3	0
Keepalives:	171	165
Route Refresh:	0	0
Capability:	0	0
Total:	177	167

Minimum time between advertisement runs is 5 seconds

For address family: IPv4 Unicast

Route-Reflector Client

Community attribute sent to this neighbor (both)

0 accepted prefixes

Connections established 3; dropped 2

Last reset 00:45:15, due to BGP Notification received

Local host: 192.168.245.136, Local port: 179

Foreign host: 192.168.245.132, Foreign port: 50288

Nexthop: 192.168.245.136

Nexthop global: ::

Nexthop local: ::

BGP connection: non shared network

Read thread: on Write thread: off

Figure 11.1:Border Gateway Protocol set-up under Quagga

E. Configuration: BGP set-up on OpenDaylight

root@odlSDN:/home/smaug/Karaf/karaf-0.5.2/bin# ./karaf

Apache Karaf starting up. Press Enter to open the shell now...

Exception in thread "Thread-103" io.netty.channel.unix.Errors$NativeIoException: bind() failed:

Address already in use

at io.netty.channel.unix.Errors.newIOException(Errors.java:115)

at io.netty.channel.unix.Socket.bind(Socket.java:204)

at io.netty.channel.epoll.EpollServerSocketChannel.doBind(EpollServerSocketChannel.java:91)

at io.netty.channel.AbstractChannel$AbstractUnsafe.bind(AbstractChannel.java:490)

at io.netty.channel.DefaultChannelPipeline$HeadContext.bind(DefaultChannelPipeline.java:1198)

at io.netty.channel.AbstractChannelHandlerContext.invokeBind(AbstractChannelHandlerContext.java:481)

at io.netty.channel.AbstractChannelHandlerContext.bind(AbstractChannelHandlerContext.java:466)

at io.netty.channel.ChannelDuplexHandler.bind(ChannelDuplexHandler.java:38)

at io.netty.handler.logging.LoggingHandler.bind(LoggingHandler.java:197)

at io.netty.channel.AbstractChannelHandlerContext.invokeBind(AbstractChannelHandlerContext.java:481)

at io.netty.channel.AbstractChannelHandlerContext.bind(AbstractChannelHandlerContext.java:466)

at io.netty.channel.DefaultChannelPipeline.bind(DefaultChannelPipeline.java:944)

at io.netty.channel.AbstractChannel.bind(AbstractChannel.java:203)

at io.netty.bootstrap.AbstractBootstrap$2.run(AbstractBootstrap.java:350)

at io.netty.util.concurrent.SingleThreadEventExecutor.runAllTasks(SingleThreadEventExecutor.java:358)

at io.netty.channel.epoll.EpollEventLoop.run(EpollEventLoop.java:307)

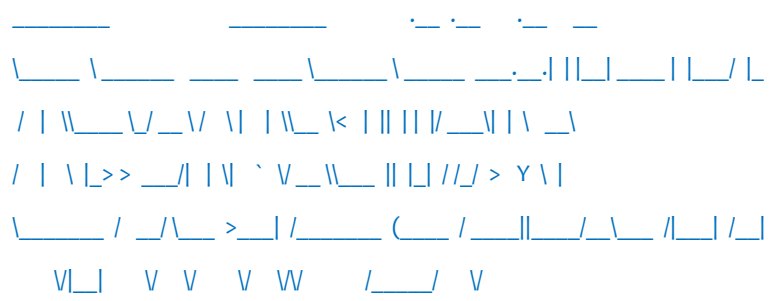
at io.netty.util.concurrent.SingleThreadEventExecutor$2.run(SingleThreadEventExecutor.java:112)

at io.netty.util.concurrent.DefaultThreadFactory$DefaultRunnableDecorator.run(DefaultThreadFactory.java:145)

at java.lang.Thread.run(Thread.java:748)

100% [==]

Karaf started in 131s. Bundle stats: 391 active, 391 total



Hit '<tab>' for a list of available commands

and '[cmd] --help' for help on a specific command.

Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown OpenDaylight.

opendaylight-user@root>bgp:show-stats

Attribute	Value
Object Name	org.opendaylight.controller:instanceName=example-bgp-peer,moduleFactoryName=bgp-peer,type=RuntimeBean
HoldtimeCurrent	180
KeepaliveCurrent	60
SessionDuration	0:00:38:08
SessionState	UP
MessagesStats.ErrorMsgsSent	0
MessagesStats.ErrorMsgsReceived	0
MessagesStats.KeepAliveMsgsSent	Counter32 [_value=38]
MessagesStats.KeepAliveMsgsReceived	Counter32 [_value=39]
MessagesStats.TotalMsgsSent	Counter32 [_value=38]
MessagesStats.TotalMsgsReceived	Counter32 [_value=40]
MessagesStats.UpdateMsgsSent	Counter32 [_value=0]
MessagesStats.UpdateMsgsReceived	Counter32 [_value=1]
PeerPreferences.AddPathCapability	false
PeerPreferences.AS	AsNumber [_value=64404]
PeerPreferences.BgpExtendedMessageCapability	false
PeerPreferences.BgpId	Ipv4Address [_value=192.168.245.136]
PeerPreferences.FourOctetAsCapability	true
PeerPreferences.GrCapability	true
PeerPreferences.Port	PortNumber [_value=179]
PeerPreferences.RouteRefreshCapability	true
SpeakerPreferences.AddPathCapability	true
SpeakerPreferences.AS	AsNumber [_value=64404]
SpeakerPreferences.BgpExtendedMessageCapability	true
SpeakerPreferences.BgpId	Ipv4Address [_value=192.168.245.132]
SpeakerPreferences.FourOctetAsCapability	true

SpeakerPreferences.GrCapability | false

SpeakerPreferences.Port | PortNumber [_value=50288]

SpeakerPreferences.RouteRefreshCapability | true

Figure 12.1:Border Gateway Protocol set-up under OpenDaylight

F. Configuration: OpenFlow set-up on OpenDaylight

root@miniNET:/home/smaug# mn --topo linear,2 --mac --controller=remote,ip=192.168.245.132,port=6633 --switch ovs,protocols=OpenFlow13

*** Creating network

*** Adding controller

*** Adding hosts:

h1 h2

*** Adding switches:

s1 s2

*** Adding links:

(h1, s1) (h2, s2) (s2, s1)

*** Configuring hosts

h1 h2

*** Starting controller

c0

*** Starting 2 switches

s1 s2 ...

*** Starting CLI:

mininet> h1 ping h2

PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.

64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=1.43 ms

64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.263 ms

64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.839 ms

--- 10.0.0.2 ping statistics ---

3 packets transmitted, 3 received, 0% packet loss, time 2001ms

rtt min/avg/max/mdev = 0.263/0.845/1.435/0.479 ms

mininet> links

h1-eth0<->s1-eth1 (OK OK)

h2-eth0<->s2-eth1 (OK OK)

s2-eth2<->s1-eth2 (OK OK)

mininet> pingall

*** Ping: testing ping reachability

h1 -> h2

h2 -> h1

*** Results: 0% dropped (2/2 received)

Figure 13.1:OpenFlow Protocol set-up under OpenDaylight