


## Statement and Confirmation of Own Work

***A signed copy of this form must be submitted with every assignment.  
If the statement is missing your work may not be marked.***

### Student Declaration

I confirm the following details:

<b>Student Name:</b>	Johnson Kenisha Corera
<b>Student ID Number:</b>	c23020001@cicra.edu.lk
<b>Qualification:</b>	Bachelor in Cyber Security
<b>Unit:</b>	SIT325 Advanced Network Security
<b>Centre:</b>	CICRA Campus
<b>Word Count:</b>	1707
<p>I have read and understood both <i>Deakin Academic Misconduct Policy</i> and the <i>Referencing and Bibliographies</i> document. To the best of my knowledge my work has been accurately referenced and all sources cited correctly.</p> <p>I confirm that I have not exceeded the stipulated word limit by more than 10%.</p> <p>I confirm that this is my own work and that I have not colluded or plagiarized any part of it.</p>	
<b>Candidate Signature:</b>	J. 
<b>Date:</b>	03/12/2024

**Table of Content**

Introduction.....	05
Part A.....	06
Part B.....	13
Conclusion.....	19
Reference.....	20

**Table of Figures**

Figure 01.....	06
Figure 02.....	06
Figure 03.....	07
Figure 04.....	07
Figure 05.....	08
Figure 06.....	08
Figure 07.....	09
Figure 08.....	09
Figure 09.....	10
Figure 10.....	11
Figure 11.....	11
Figure 12.....	12

**Table of Acronyms**

<b>Acronyms</b>	<b>Meaning</b>
SDN	Software Defined Network
PbSA	Policy-based Security Architectures
PbSM	Policy-based Security Management
IoT	Internet of Things

## Introduction

- The applicability and challenges encountered in the actual fashioning of working models for real topologies are discussed in this paper, which is based on The Internet Topology Zoo. As for the real-world Ln, DFN and AARNet were formed and evaluated employing tools as Mininet, GraphML and ONOS. An evaluation involved script setup, the debugging of Python codes, and the integration of other support to topology with Software-Defined Networking (SDN) platforms. This research also examines policy-based security architectures of (PbSA), open networking devices and the theoretical underpinning of SDN. Understanding those advantages, disadvantages, and factors that affect its performance will provide a comprehensive picture of the theoretical and practical interconnection between theoretical concepts and actual network design.

## Part A

- The very first thing I did was browse through what The Internet Topology Zoo offered which was a great number of real data sets of various network topologies. Therefore, I began by creating a copy of the mentioned GitHub repository.

```
kenisha@kenisha-virtual-machine:~$ cd 6.1p
kenisha@kenisha-virtual-machine:~/6.1p$ git clone http://github.com/sjas/assessing-mininet.git
Cloning into 'assessing-mininet'...
warning: redirecting to https://github.com/sjas/assessing-mininet.git/
remote: Enumerating objects: 1273, done.
remote: Total 1273 (delta 0), reused 0 (delta 0), pack-reused 1273 (from 1)
Receiving objects: 100% (1273/1273), 2.79 MiB | 681.00 KiB/s, done.
Resolving deltas: 100% (716/716), done.
```

*Figure 01: evaluating mininets by cloning*

- My focus was on two different topologies: the AARNet topology that gives the representation of the Australian universities and the 57 nodes topologically representing the German network called DFN. I achieved it by naming the parser folder and then using the wget tool to download a file called Aarnet.graphml.

```
kenisha@kenisha-virtual-machine:~/6.1p$ cd assessing-mininet/parser
kenisha@kenisha-virtual-machine:~/6.1p/assessing-mininet/parser$ wget http://www.topology-zoo.org/files/Aarnet.graphml
--2025-01-07 22:55:52-- http://www.topology-zoo.org/files/Aarnet.graphml
Resolving www.topology-zoo.org (www.topology-zoo.org)... 129.127.10.249
Connecting to www.topology-zoo.org (www.topology-zoo.org)|129.127.10.249|:80... connected.
```

*Figure 02: Installation*

```

51 #set bw to 10mbit if nothing was specified otherwise on startup
52 if bandwidth_argument == '':
53     bandwidth_argument = '10';
54
55 # ... and link all corresponding switches with each other
56 temp4 = self.addLink(
57     temp4 += id_node_name_dict[src_id]
58     temp4 += ' '
59     temp4 += id_node_name_dict[dst_id]
60     temp4 += ' '
61     temp4 += bandwidth_argument
62     temp4 += ' '
63     temp4 += str(latency)
64     temp4 += 'ms')
65 temp4 += '\n'
66 # next line so I dont have to look up other possible settings
67 temp += "ms", loss=0, max_queue_size=1000, use_htb=True)"
68 tempstring4 += temp4
69
70 outputstring_to_be_exported += tempstring4
71 outputstring_to_be_exported += outputstring_4a
72
73 # this is kind of dirty, due to having to use mixed ' ' ""
74 temp5 = "controller_ip = "
75 temp5 += controller_ip
76 temp5 += "\n"
77 tempstring5 += temp5
78
79 outputstring_to_be_exported += tempstring5
80 outputstring_to_be_exported += outputstring_4b
81
82
83 # GENERATION FINISHED, WRITE STRING TO FILE
84 outfile = ''
85 if output_file_name == '':
86     output_file_name = input_file_name + '.generated-Mininet-Topo.py'
87
88 outfile = open(output_file_name, 'w')
89 outfile.write(outputstring_to_be_exported)
90 outfile.close()
91
92 print("Topology generation SUCCESSFUL!")

```

Figure 03: python code

- I ran the code again and got the output file Aarnet.py which will help to set up the network automatically.

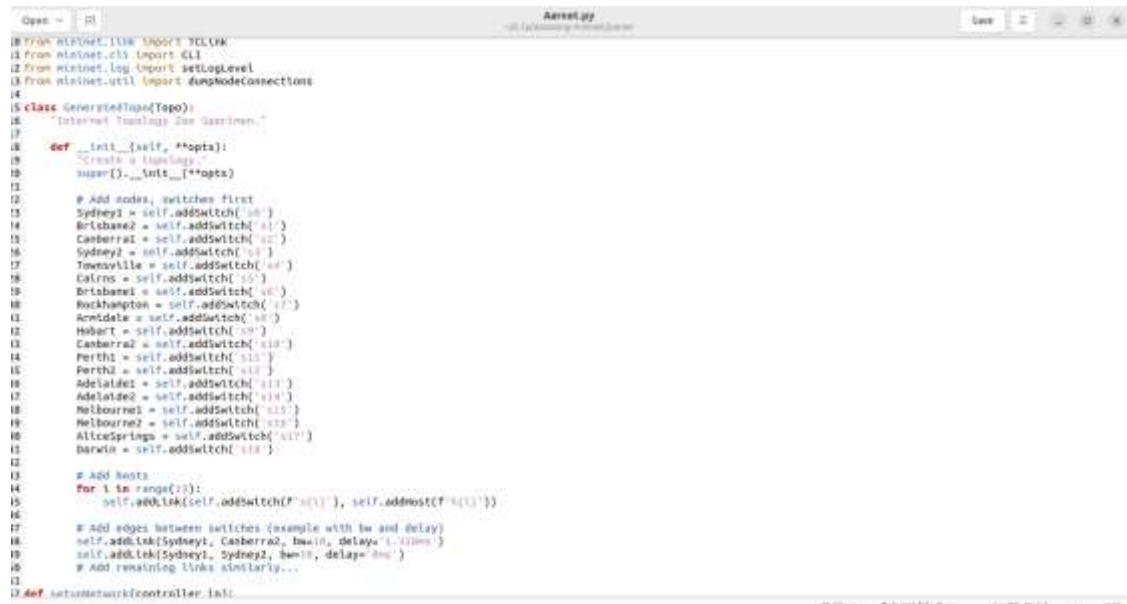
```

kenishaghenlaha-virtual-machine: ~/Aarnet/GraphML-Topo-to-Mininet-Network-Generator
$ python3 GraphML-Topo-to-Mininet-Network-Generator.py -f
Aarnet.graphml -o Aarnet.py
Topology generation SUCCESSFUL!

```

Figure 04: creating a network topology is a successful thing

- To run the Aarnet.py Python file, I had to make some alteration on the code, which include the file path change and brackets addition.

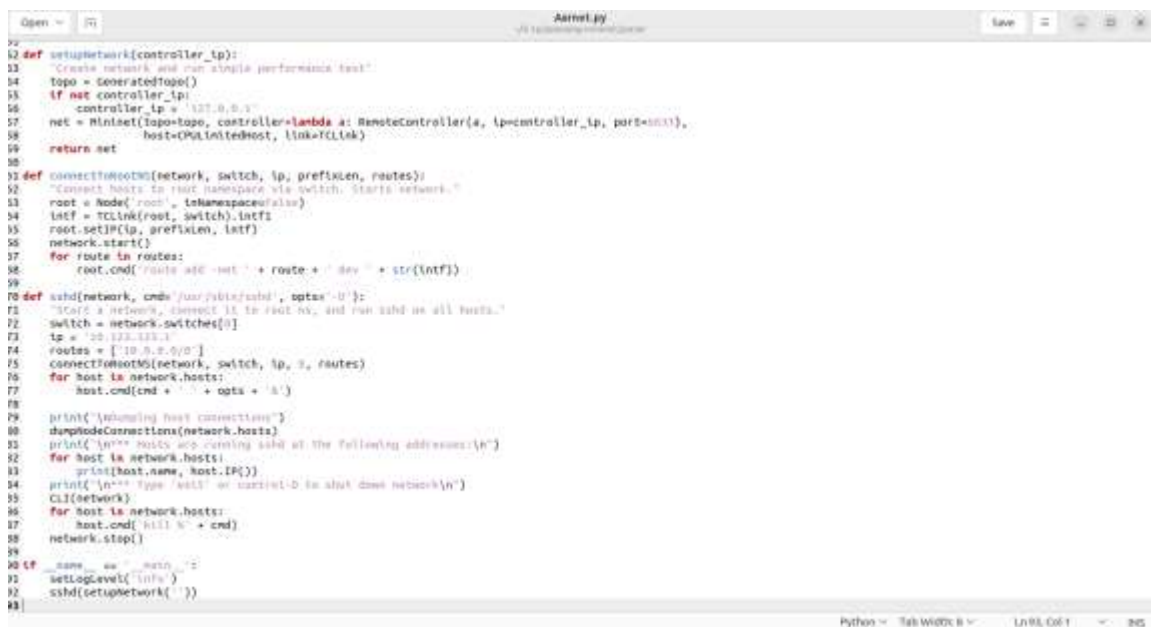


```

1 from mininet.log import TCLINK
2 from mininet.cli import CLI
3 from mininet.log import setLogLevel
4 from mininet.util import dumpNodeConnections
5 class GeneratedTopo(Topo):
6     "Internet Topology Zoo Baseline."
7
8     def __init__(self, *opts):
9         "Create a topology."
10        super().__init__(*opts)
11
12        # Add nodes, switches first
13        Sydney1 = self.addSwitch('s0')
14        Brisbane2 = self.addSwitch('s1')
15        Canberra1 = self.addSwitch('s2')
16        Sydney2 = self.addSwitch('s3')
17        Townsville = self.addSwitch('s4')
18        Cairns = self.addSwitch('s5')
19        Brisbane1 = self.addSwitch('s6')
20        Rockhampton = self.addSwitch('s7')
21        Armidale = self.addSwitch('s8')
22        Hobart = self.addSwitch('s9')
23        Canberra2 = self.addSwitch('s10')
24        Perth1 = self.addSwitch('s11')
25        Perth2 = self.addSwitch('s12')
26        Adelaide1 = self.addSwitch('s13')
27        Adelaide2 = self.addSwitch('s14')
28        Melbourne1 = self.addSwitch('s15')
29        Melbourne2 = self.addSwitch('s16')
30        AliceSprings = self.addSwitch('s17')
31        Darwin = self.addSwitch('s18')
32
33        # Add hosts
34        for i in range(10):
35            self.addLink(self.addSwitch(f's{i}'), self.addHost(f'h{i}'))
36
37        # Add edges between switches (example with bw and delay)
38        self.addLink(Sydney1, Canberra2, bw=10, delay='1.1ms')
39        self.addLink(Sydney1, Sydney2, bw=10, delay='1ms')
40        # Add remaining links similarly...
41
42 def setupNetwork(controller_ip):
43     "Create network and run simple performance test."
44     topo = GeneratedTopo()
45     if not controller_ip:
46         controller_ip = '10.0.0.1'
47     net = Mininet(topo=topo, controller=lambda a: RemoteController(a, ip=controller_ip, port=8033),
48                  host=CPUIntendedHost, link=TCLink)
49     return net
50
51 def connectToRootNS(network, switch, ip, prefixlen, routes):
52     "Connect hosts to root namespace via switch. Starts network."
53     root = Node('root', inNamespace=False)
54     intf = TCLink(root, switch).intf1
55     root.setIP(ip, prefixlen, intf)
56     network.start()
57     for route in routes:
58         root.cmd('route add -net ' + route + ' dev ' + str(intf))
59
60 def sshd(network, cmd="/usr/sbin/sshd", opts='-D'):
61     "Start a network, connect it to root NS, and run sshd on all hosts."
62     switch = network.switches[0]
63     ip = '10.0.0.1'
64     routes = ['10.0.0.0/8']
65     connectToRootNS(network, switch, ip, 8, routes)
66     for host in network.hosts:
67         host.cmd(cmd + ' ' + opts + ' &')
68
69     print('Initiating host connections')
70     dumpNodeConnections(network.hosts)
71     print('**** hosts are coming alive at the following addresses:\n')
72     for host in network.hosts:
73         print(host.name, host.IP())
74     print('**** type \'kill -s SIGINT 0\' to shut down network\n')
75     CLI(network)
76     for host in network.hosts:
77         host.cmd('kill -s ' + cmd)
78     network.stop()
79
80 if __name__ == '__main__':
81     setLogLevel('info')
82     sshd(setupNetwork(''))
83

```

Figure 05: Path addition to enable Python 3



```

92 def setupNetwork(controller_ip):
93     "Create network and run simple performance test."
94     topo = GeneratedTopo()
95     if not controller_ip:
96         controller_ip = '10.0.0.1'
97     net = Mininet(topo=topo, controller=lambda a: RemoteController(a, ip=controller_ip, port=8033),
98                  host=CPUIntendedHost, link=TCLink)
99     return net
100
101 def connectToRootNS(network, switch, ip, prefixlen, routes):
102     "Connect hosts to root namespace via switch. Starts network."
103     root = Node('root', inNamespace=False)
104     intf = TCLink(root, switch).intf1
105     root.setIP(ip, prefixlen, intf)
106     network.start()
107     for route in routes:
108         root.cmd('route add -net ' + route + ' dev ' + str(intf))
109
110 def sshd(network, cmd="/usr/sbin/sshd", opts='-D'):
111     "Start a network, connect it to root NS, and run sshd on all hosts."
112     switch = network.switches[0]
113     ip = '10.0.0.1'
114     routes = ['10.0.0.0/8']
115     connectToRootNS(network, switch, ip, 8, routes)
116     for host in network.hosts:
117         host.cmd(cmd + ' ' + opts + ' &')
118
119     print('Initiating host connections')
120     dumpNodeConnections(network.hosts)
121     print('**** hosts are coming alive at the following addresses:\n')
122     for host in network.hosts:
123         print(host.name, host.IP())
124     print('**** type \'kill -s SIGINT 0\' to shut down network\n')
125     CLI(network)
126     for host in network.hosts:
127         host.cmd('kill -s ' + cmd)
128     network.stop()
129
130 if __name__ == '__main__':
131     setLogLevel('info')
132     sshd(setupNetwork(''))
133

```

Figure 06: updating the print methods with brackets



- Following that, the Aarnet.py script was run to build the topology and connect with an ONOS program.

```

kenisha@kenisha-virtual-machine:~/A ip/assessing-nininet/parser$ sudo python3 Aarnet.py
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h0 h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16 h17 h18
*** Adding switches:
s0 s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 s18
*** Adding links:
(s0, h0) (10.00Mbit 0ms delay) (10.00Mbit 0ms delay) (s0, s3) (10.00Mbit 1.328ms delay) (10.00Mbit 1.328ms delay) (s0, s10) (s1, h1) (s2, h2) (s3, h3) (s4, h4) (s5, h5) (s6, h6) (s7, h7) (s8, h8) (s9, h9) (s10, h10) (s11, h11) (s12, h12) (s13, h13) (s14, h14) (s15, h15) (s16, h16) (s17, h17) (s18, h18)
*** Configuring hosts
h0 (cfs max/1000000us) h1 (cfs max/1000000us) h2 (cfs max/1000000us) h3 (cfs max/1000000us) h4 (cfs max/1000000us) h5 (cfs max/1000000us) h6 (cfs max/1000000us) h7 (cfs max/1000000us) h8 (cfs max/1000000us) h9 (cfs max/1000000us) h10 (cfs max/1000000us) h11 (cfs max/1000000us) h12 (cfs max/1000000us) h13 (cfs max/1000000us) h14 (cfs max/1000000us) h15 (cfs max/1000000us) h16 (cfs max/1000000us) h17 (cfs max/1000000us) h18 (cfs max/1000000us)
*** Starting controller
c0
*** Starting 19 switches
s0 s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 s18 ... (10.00Mbit 0ms delay) (10.00Mbit 1.328ms delay) (10.00Mbit 0ms delay) (10.00Mbit 1.328ms delay)

```

Figure 07: executing Aarnet.py

- At last, I launched the ONOS in Docker.

```

root@kenisha-virtual-machine:/home/kenisha# docker pull onosproject/onos:2.7.0
2.7.0: Pulling from onosproject/onos
Digest: sha256:bc844aaf64e03834c7043bad83faeccc0af8984e9b4887c13e1a307a7c2
Status: Image is up to date for onosproject/onos:2.7.0
root@kenisha-virtual-machine:/home/kenisha# docker run -t -d -p 8181:8181 -p 5005:5005 -p 830:830 --name onos_new22 onosproject/onos:2.7.0
29a98b4ba686f1715ccc89783ff63c481caee722eb3e962c632354044271ca
root@kenisha-virtual-machine:/home/kenisha# docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS
29a98b4ba686   onosproject/onos:2.7.0              "/bin/onos-service ."   14 seconds ago Up 13 seconds 0.0.0.0:830->830/tcp, :::830->830/tcp, 6640/tcp, 0.0.0.0:5005->5005/tcp, :::5005->5005/tcp, 0.0.0.0:8181->8181/tcp, :::8181->8181/tcp, 6653/tcp, 0.0.0.0:8181->8181/tcp, :::8181->8181/tcp, 9879/tcp
root@kenisha-virtual-machine:/home/kenisha# sudo docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS        NAMES
29a98b4ba686   onosproject/onos:2.7.0              "/bin/onos-service ."   14 seconds ago Up 13 seconds 0.0.0.0:830->830/tcp, :::830->830/tcp, 6640/tcp, 0.0.0.0:5005->5005/tcp, :::5005->5005/tcp, 0.0.0.0:8181->8181/tcp, :::8181->8181/tcp, 6653/tcp, 0.0.0.0:8181->8181/tcp, :::8181->8181/tcp, 9879/tcp
root@kenisha-virtual-machine:/home/kenisha#

```

Figure 08: run the docker

- The ONOS was logged into by using a web browser and connected to the network topology of the given topology.



*Figure 09: ONOS login page*

- I was then able to get to the main page after logging in to the system and saw that there were no devices associated with the SDN.



Figure 10: Main page

- Then, at last, I executed the generated script.

```
kenisha@kenisha-virtual-machine:~/6.ip/assessing-mininet/parser$ sudo python3 Aarnet.py
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h0 h1 h2 h3 h4 h5 h6 h7 h8 h9 h10 h11 h12 h13 h14 h15 h16 h17 h18
*** Adding switches:
s0 s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 s18
*** Adding links:
(s0, h0) (10.00Mbit 0ms delay) (10.00Mbit 0ms delay) (s0, s3) (10.00Mbit 1.328ms delay) (10.00Mbit 1.328ms delay) (s0, s10) (s1, h1) (s2, h2) (s3, h3) (s4, h4) (s5, h5) (s6, h6) (s7, h7) (s8, h8) (s9, h9) (s10, h10) (s11, h11) (s12, h12) (s13, h13) (s14, h14) (s15, h15) (s16, h16) (s17, h17) (s18, h18)
*** Configuring hosts
h0 (cfs max/1000000us) h1 (cfs max/1000000us) h2 (cfs max/1000000us) h3 (cfs max/1000000us) h4 (cfs max/1000000us) h5 (cfs max/1000000us) h6 (cfs max/1000000us) h7 (cfs max/1000000us) h8 (cfs max/1000000us) h9 (cfs max/1000000us) h10 (cfs max/1000000us) h11 (cfs max/1000000us) h12 (cfs max/1000000us) h13 (cfs max/1000000us) h14 (cfs max/1000000us) h15 (cfs max/1000000us) h16 (cfs max/1000000us) h17 (cfs max/1000000us) h18 (cfs max/1000000us)
*** Starting controller
c0
*** Starting 19 switches
s0 s1 s2 s3 s4 s5 s6 s7 s8 s9 s10 s11 s12 s13 s14 s15 s16 s17 s18 ...(10.00Mbit 0ms delay) (10.00Mbit 1.328ms delay) (10.00Mbit 0ms delay) (10.00Mbit 1.328ms delay)
```

Figure 11: executing the script Aarnet.py

- Even after running the code I could not connect the network to the topology, for some reason or the other.



*Figure 12: The topological connection failed.*

## Part B

**Q1. There is a push towards creating networking devices that are open, or run Linux inside, such that software can be written to extend or enhance the capabilities of devices. What are the ways in which this is a good idea? And are there ways in which this is a bad idea?**

- While the concept of open sourcing networking hardware, or at the very least, running Linux on it provides its advantages and disadvantages. On the positive side, open networking devices offer significant and specific flexibility. It also reveals that the developers of applications have about control rights in changing or improving aspects of devices that are important for the users' requirements as well as the evolution of device characteristics. Also, it is publicly designed for multi-entity collaboration within the many-community; therefore, tools and solutions are invented and disseminated as soon as possible. Moreover, Linux is supported virtually on everything and is well known, so hosting this OS on the networking devices resembles what many developers face.
- However, it will also have its disadvantages. With reference to interacting parties, one might regard openness of the gadgets as the underlying source of security threats. What this means is that the network is not immune to change, and as everybody knows when change occurs it brings along words like; hacking, virus, Trojan horse, and other elements that compromise network security. It also means that there is a question as to the absence of homogeneity of the software environment of the device, or on the use of different levels of security. A software that depends on the community may also be less successful because it will not be able to patch severe security holes as quickly as proprietary software, which is usually backed by in house development and testing personnel.
- There are several benefits to using Linux-based open networking devices, including the following: The open networking devices that are based on Linux are liberty flooded and contain more creativity than some risks that needs be taken into consideration. (Anerousis, 2021)

**Q2. What is your understanding about policy-based security architectures?**

- The approach of Policy-based Security Architectures (PbSA) is becoming more and more feasible to enforce security policies in a network automatically. These designs allow a company to apply rules to user, traffic and other important processes. One could free themselves from the rigidity of conventional solutions in network security since PbSA, as one of the techniques in network security, eliminates security policies from the physical implementation of the network infrastructure. (ATN, 2001)
- PbSA addresses issues concerning the do's and don'ts in a certain context through making of policies. Most of these rules are normally set using one or more of several factors including the network segment, device class, or user identity. Once the policies have been set, the enforcement mechanism employs use of both, software and hardware in enforcing compliance from the network layer and downwards.
- Another benefit of PbSA is that it is centralized compared with a decentralized system in which discrete clients of the huge dispersed combine demand the responsibility for its protection. From the above analysis, it was evident that the use of policies in the system provide relatively more flexibility to the security management in that they can be changed while the modifications can be easily disseminated to other systems.
- Yet, as the complexity of the network services is growing, it becomes a task to set up these rules and ensure that they are implemented correctly in every network layer. This means that in order to be able to monitor and ensure compliance it requires the right tools and methodology.

**Q3. . In software-defined networks, policy-based security management or architecture (PbSA) is an ideal way to dynamically control the network. We observe that this enables security capabilities intelligently and enhance fine-grained control over end user behavior. Explain more about PbSA.**

- PbSA which is Policy-Based Security Architecture is a method that can provide dynamic and accurate security regulation in Software Defined Network. An SDN partially restricts network flow and policy efforts by decentralizing the data plane from the control plane. This is employed by PbSA to develop a security policy that defines what network resources are available and how.
- Fully explained in the concept of PbSA design, policies are described in such a manner, which does not dictate a specific hardware and/or platform. They could allow, block, redirect or encapsulate traffic using source or destination IP address, user or the application. Since such policies have been defined, all these network devices will behave in a manner that is in compliance with these policies if the SDN controller is given full control. (Strassner, 2004)
- For SDN networks, PbSA's greatest strength is its ability to learn and operate network behaviors in real time. He further pointed that PbSA also employ proactive measures to maintain organizational compliance with conformance by resisting attacks and controlling policies locally to the current network conditions or threats. This increased control granularity is especially useful in complex security conditions like those of cloud computing environment, or The Internet of Things environment.
- In order not to experience fail setup that leads to formation of more voids, they require well developed policy management systems for proper utilization and interactively, they require profound knowledge on fine grained network construction and security models.

**Q4. Please give the generic workflow of most of the existing policy-based security management proposals.**

- Several crucial steps are included in the general workflow of the majority of current policy-based security management (PbSM) proposals: These are policy definition, translation of policies, dissemination of policies, implementation of policies and auditing of policies.
- Policy Definition: This is the first whereby to meet a given organizations' security demands, the necessary policies such as firewall, anti-viral, and encryption among others will be formed. Policies could be organizational like data protections policies, traffic filtering policies, access control policies among others. They're often presented in a format that is easily interpretable by both a computer and a human.
- Policy Translation: Subsequently, these policies are changed in a manner that can be understood by lower tiers of the network. In this procedure top-level policies are translated to policies that can be implemented at various levels or devices like switches, firewalls, routers, and SDN controllers. (Claassen, 2002)
- Distribution of Policies: these policies are translated and then put into the network enforcement points by the translators that deal with such work. This may involve changing the set policies in the security products, or moving the configurations to the network device among others. In SDN this is handled by the SDN controller wherein every component of the network is properly set.
- Policy Enforcement: In this step, the omissions that put into operation the settings of the distributed policies are performed. All the elements as hubs, switches, firewall as well as routers also enforce policies and apply traffic flow, access permission or any other conditions according to the policies that have been set.
- Policy Monitoring: The last activity here is to monitor in order to ensure that policies are implemented as they should and also the network behavior is as per the requirements. Here, notifications and tracking are meant for discovering that a policy has been violated or there might be a security issue.



## **Q5. Why the analysis of the policy based SDN security architecture is essential?**

- For the following reasons, the policy based SDN security architecture is equally important for the research. First of all, SDN redesigns the handling and protection of networks by partitioning the control plane and data plane, while enabling the centralized administration of the web. As a result of that, PbSA has a solution that makes network security rules more realistic and achievable, subject to the current circumstances within the network at a given moment and the evolving threat scenarios.
- It is important to think about these designs so that we might be in a position to discern if the policies are correctly defined or not, and if they are being correctly implemented or not. Due to policy gaps or lack of, awareness in the implementation of the policies, denial-of-service attacks affects the network availability, confidentiality and integrity. In this manner, organizations may guarantee that the operational SDN environment conforms to these standards and obtains a conceptualization of the architecture, hazards and inadequacies in policy definitions. (Hu, 2020)
- Moreover, as a network scales up and becomes more realistic, it actually becomes quite an issue to enforce policies that are in place. Policy processing may be analyzed and bottlenecks or restrictions exposed; resources are allocated to keep the network at optimum, while security is increased. Second, the consequences of insecurity affecting the systems are more severe bearing in mind that the current trending adoption of SDN in data centres and cloud solutions. It is therefore possible to obtain basic protection against complex attacks and the extendibility of the architecture to cater for a diverse and growing network beyond the reach of the fairly simple threat model effectively in the PbSA.

**Q6. Explain how number of policy rules, rule matching probability of flow in policy-table, flow arrival rate, and the impact of rule expressions on throughput impacts the performance of PbSA.**

- This paper motivate that PbSA behavior and performance, considerably depend on the quantity of policy rules and on the probability that the rules matched, the arrival rate of flow, relative to the complexity of the rule expressions. Number of Rules in the Policy: Thus for the VL increasing with the increase in the number of tuples in the sub base, it is quite clear from the results obtained that the system processing overhead is also growing in tandem with it. That is, the greater the number of rules, the more parameters each packet or flow is compared against, which negatively affects the flow through per time and decision making duration.
- The rule matching probability means identifying the extent to which a certain flow corresponds to a policy rule. The chance for the table is something that cannot be thrown to the extreme periphery of possibility recognition. The reason for a high likelihood of rule matching is that the PbSA can easily match the relevant security policies at the earliest, high load carrying capacity and efficient management of services. Where matching probability is low, the system needs to search through the rules more closely and this takes time to sort through leading to low throughput.
- Flow Arrival Rate: The flow velocity by which fresh flows get to the PbSA is also another aspect that determines any given performance. In other words, for the PbSA to handle more packets within a certain time frame of time, the arrival rates within a single high flow will be challenging. When the capacity of the flow cannot be met by the capacity of the system then the corresponding areas experience congestion and reduced productivity. The impact of rule expressions the degree of policy enforcement slows down when there is a policy containing rules with complex expression that needs deep packed inspection or in case it was established that more than two conditions are needed. Due to this reason, much time is consumed when attempting to explain the need for evaluating each packet, thus compromising the throughput of PbSA. (Anerousis, 2021)

## Conclusion

- This work, which has been conducted using The Internet Topology Zoo as the framework, shows and implies that the development of methods to construct functional models suitable for the investigation of real-world topologies using currently existing tools and platforms such as Mininet, GraphML, and ONOS, has profound theoretical and practical implications. The assessment procedure of integrating SDN platforms, debugging Python scripts and studying of policy based security architectures (PbSA) reveal the challenges, which appears to map the theoretical concepts of networks into operational use. However, Linux-powered open networking devices can offer such advantages as flexibility, the opportunity for teamwork, and creativity; at the same time, possible security issues and possible challenges related to maintaining a stable and coherent software environment. PbSA also turns out to be useful for the reliable implementation of proving compliance with network security regulations and enforcing them in complex and constantly evolving networks. However, when trying to achieve the best outcome, there is always the need to have well-established policies and guidelines and well-defined policies to be enforced across the organization.
- The kind of policy rules implemented affects the PbSA's throughput and efficiency as does the number of policy rules that have to be matched, arrival rates of flows and the complexity of the rules imposed. In Software-Defined Networks (SDN), it is possible to allocate resources more efficiently, scale up SDN easier while at the same time making it more secure if these factors are given consideration. Lastly, this study makes an emphasis to a fact that the evaluation and upgrading of policy-based architectures has to be ongoing and presented in response to the rising challenges of implementing subsequent policies to meet the demand of more complicated topological structures for networks.

## Reference

- 'The Origin and Evolution of Open Programmable Networks and SDN'  
[https://www.researchgate.net/publication/349515725\\_The\\_Origin\\_and\\_Evolution\\_of\\_Open\\_Programmable\\_Networks\\_and\\_SDN](https://www.researchgate.net/publication/349515725_The_Origin_and_Evolution_of_Open_Programmable_Networks_and_SDN)
- Policy based management system, Aeronautical Telecommunication Network (ATN) Seminar.  
[SP22.pdf](#)
- Policy-Based network management  
<https://www.sciencedirect.com/book/9781558608597/policy-based-network-management/#book-info>.
- Impact of network security on SDN controller performance  
<https://open.uct.ac.za/login>
- A survey on Software-Defined Network and Open Flow  
[A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation | IEEE Journals & Magazine | IEEE Xplore](#)
- Improving network management with software defined networking,' IEEE Communications Magazine  
<https://ieeexplore.ieee.org/document/6461195>