Access Rates Analysis Of Routing Stack Transformation

Vincent Omollo, Dr. K. Langat, Dr. S. Musyoki, E. Ogari, P. Nyakomita

Abstract: The security of data is a very important aspect of every institution. One of the methods of boosting data security is by use of routing stack transformation. This approach successfully hides router inter-networking operating system from potential hackers by giving wrong or inaccurate results about the target operating system (including its manufacturer, model and version). This process normally involves changing the system kernel and modifying the transmission control protocol/Internet protocol (TCP/IP) parameters. In this paper, we investigated the effects of the above changes in data transmission rates (bit rates). This was done by use of experimentations before and after these alterations. A number of packets were sent from source and the time they take to reach the target is measured. This information was then used to determine the access rates in kilobits per second across a given communication link. We found that the network speeds were slightly reduced due to the TCP/IP and kernel changes. This could be attributed to the fact that network devices now took a long time trying to learn the format of the new packets, hence the subsequent delays. Due to this, a number of ways of boosting network speed are suggested. The simulated results tend to agree with the real trend observed in that in both scenarios, network access rates were reduced after routing stack transformation.

Key words: Routing stack, TCP/IP, flags, kernel, throughput

I. INTRODUCTION

The main concern of the internet during its establishment was to enable expensive resources such as mainframe computers to be shared [1]. The internet has developed into a global avenue to carry out many tasks including business, entertainment, academic research, communication and news delivery. The most utilized feature is the retrieval of data [2]. However, the security of the data being transmitted in computer networks continues to be a major issue. Many networks have been compromised and this has resulted in theft of crucial organizational information. This can be attributed to the flaws that do exist in the architecture of some of the networking devices. Altering the design of the internet involves the danger of introducing new opportunities for attacks [3]. For example the Content-Centric Networking (CCN) routers are required to keep the state of communication process. This information can be used to launch a denial of service attack. The browsers also keep crucial information in their caches, including usernames and passwords. This data can be used by attackers to infringe into users' privacy and the consequent damages. Many attacks succeed because most users fail to utilize computer access control features to their full capacity [4]. This therefore means that network intruders can rarely launch brute force mechanisms against such systems. Instead they spend time in coming up with techniques to take advantage of the weak configuration and coding in the target hosts [5].

- *Vincent Omollo, Dr. K. Langat, Dr. S. Musyoki, E. Ogari, P. Nyakomita*
- *Email: vincentyoung88@gmail.com*
- *Department of Telecommunication and Information Engineering,*
- *Jomo Kenyatta University of Agriculture and Technology.*

One of the ways of securing networks is hiding the crucial details about the target host from potential intruders [6]. In this approach, the system kernel, TCP/IP options and flags are re-structured to hide the target host system basic information. The ultimate goal is to give network scanners such as Nmap inaccurate or no information at all about the target IP address host. This information include the device manufacturer, model, version, open ports, services running at the system and its internetworking operating system. However, since these changes deals with the system kernel and TCP/IP framework, both of which determine the format in which packets are transmitted across data communication network, their effect on the network speeds is worth evaluating. The Maximum Transmission Unit (MTU), for example, controls the maximum Ethernet packet size the network device will be sending. The limit of this unit is essential majorly because internet works in packets. Therefore although bigger packets can be constructed and sent, the Internet Service Provider (ISP) and the internet backbone routers and other networking devices will chop (fragment) any packets larger than their limit. These individual components are then re-assembled by the receiving target equipment before they are read. The TCP Maximum Segment Size (MSS) is another crucial component. It specifies the maximum amount of TCP data in single IP datagram that the local system can accept. The IP datagram can be fragmented into multiple packets when it is sent. Theoretically, this value can be as large as 65495. However, such a large value is never practically used. Instead systems use the outgoing interface MTU size, less 40 as their preferred MSS. Another equally vital TCP part is the window size. Computers in a network communicate by sending packets of data forth and back. When they do so, they use TCP. A computer can begin a connection with small packet size. The receiving computer might respond with larger packets. This requires the sending machine to scale down the size of the packets to a value below the previous one. The receiving computer again responds with a larger packet size. This cycle continues until the transmitting machine or the receiving device meets their maximum packet size. However, too large window sizes slow down network connections. Therefore the effects of routing stack transformation on these components were the subject of this paper.

II. METHODOLOGY

Introduction

Access rates analysis of the routing stack transformation is presented in this paper. In this section, we explain the series of actions that we followed to achieve our objectives. As already stated, the routing stack transformation is achieved by re-structuring the kernel and TCP/IP format.In figure 1 below, the structure of the Unix kernel is presented [6].

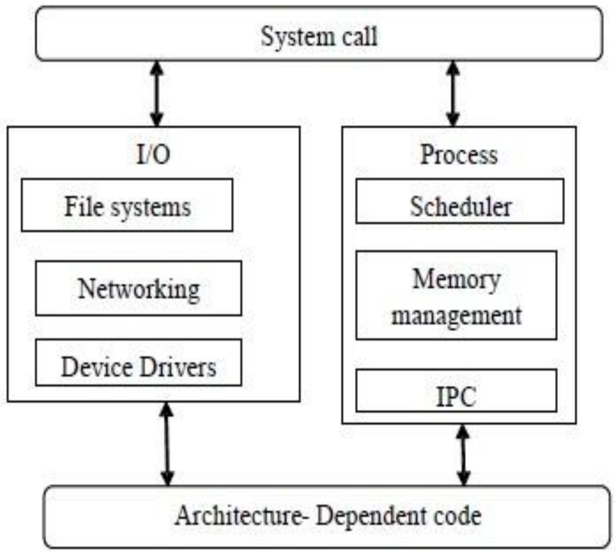


Figure 1: Architecture of the kernel

The user applications interact with the kernel through the system calls. The file system component keeps track of the entire disk map. The networking component provides networking functionalities using various protocols. The device drivers control the various hardware devices. The process scheduler manages how the various processes access the processor. The memory manager keeps track of the memory map by allocating and de-allocating memory to files and directories. The inter-process communication enables the various processes to interact with one another. This architecture also features the Input/ Output (I/O) component which are the various peripheral devices attached to the computer system. For this paper, the focus is on the networking component. The modular router developed in [6] is as shown in figure 2 below.

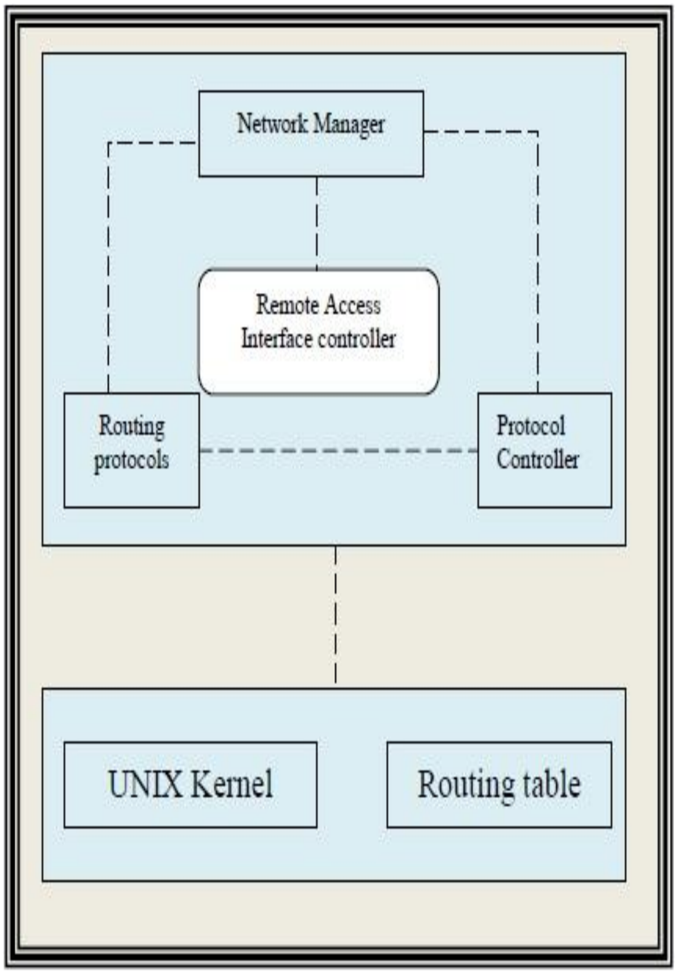


Figure 2: Modular router design

The network manager takes control of the entire communication process. It defines the interfaces through which the various routing protocols can be accessed. It also communicates directly with the networking component of the underlying system kernel. Its configuration may include access control lists, encryption description and even link level cryptography to secure data being passed in those interfaces [6]. The remote access interface controller determines the link through which remote administration activities may be carried out on the router. It also determines whether the local loop address can be used to provide access to the network manager. The routing protocols are sets of rules for determining how networked devices exchange information. These can be broadly be categorized into distance vector routing protocols or link state protocols. With distance vector routing protocols, when a new router is connected in the network, it takes the effort of every router on the network to update its routing table. It therefore has to advertise its presence in the network, normally by using a broadcast message. It utilizes a distance to a remote network to find the best path. It also uses hop counts, tick counts (1/18) or bandwidth of links. What is so clear for these links is that they have slow convergence time. Network updates are more frequent than link state protocols, and such may lead to congested traffic. Examples of these protocols are Routing Information Protocol (RIP) and Interior Gateway Routing protocol

(IGRP). The default administration distance 100 (RIP) or 120 (IGRP) while the maximum hop count is 15 (RIP) or 255 (IGRP) [6]. With link state protocols, when a new router is connected to the network, it updates its routing table by the help of the adjacent router. As such, traffic levels are reduced. It maintains three tables (directly attached neighbours, topology of entire network, and routing table) to achieve its route selection functionalities. Examples of these protocols are the Open Shortest Path First (OSPF) and the Netware Link State Protocol (NLSP). The maximum administration distance is 110. For this paper, RIP and OSPF were selected to be representatives of the two routing protocols discussed above. Figure 3 below shows how the RIP module was configured [6]. As can be seen, there are two networks allowed for communication. This means that any device that may want to interact with the RIP daemon must use any of these two networks. Figure 4 below shows the configuration of the OSPF daemon. The configuration looks similar to the one in Figure 3 above, only that we have added a new parameter called area Identification (ID).

```
! -*- rip -*-
!
! RIPd sample configuration file
!
! $Id: ripd.conf.sample,v 1.1 2002/12/13 20:15:30 eng.
Vinny Exp $
!
hostname ripd
password zebra
!
! debug rip events
! debug rip packet
!
router rip
! network 11.0.0.0/8
! network 11.0.0.1/8
! network eth0
! route 10.0.0.0/8
! distribute-list private-only in eth0
!
!access-list private-only permit 10.0.0.0/8
!access-list private-only deny any
!
!log file /var/log/omosh/ripd.log
!
```

Figure 3: Configuration of the RIP daemon

```
! hostname protocol_ospf
password omollo
enable password omollo
!
interface eth0
!
interface lo
!
interface wlan0
!
router ospf
network 192.168.1.0/24 area 0.0.0.5
network 192.168.10.0/24 area 0.0.0.5
network 192.168.20.0/24 area 0.0.0.5
network 192.168.30.0/24 area 0.0.0.5
!
line vty
!
end
```

Figure 4: Configuration of the OSPF daemon

For the OSPF daemon, we have included four networks. We then ran simulations in Common Open Research Emulator (CORE) before and after routing stack transformation (re-structuring the TCP/IP flags and options, kernel components) as implemented in [6].

III. RESULTS AND DISCUSSIONS

In this section, we will present the results that we obtained together with probable explanations for the changes observed. Figure 5 below shows simulated results before routing stack transformation. The simulated access rates are shown along the networks. For example the access rate along network 10.0.1.2/24 is 1.354 Kbps while that along network 10.0.0.20/24 is 3.729 Kbps. Figure 6 below shows the simulation results obtained after routing stack transformation were performed. Comparing Figures 5 and 6, it can be observed that there were significant reductions in the network access rates. For example, the bit rate along network 10.0.1.2/24 is now 0.697 Kbps while that along network 10.0.0.20/24 is 0.259 Kbps.

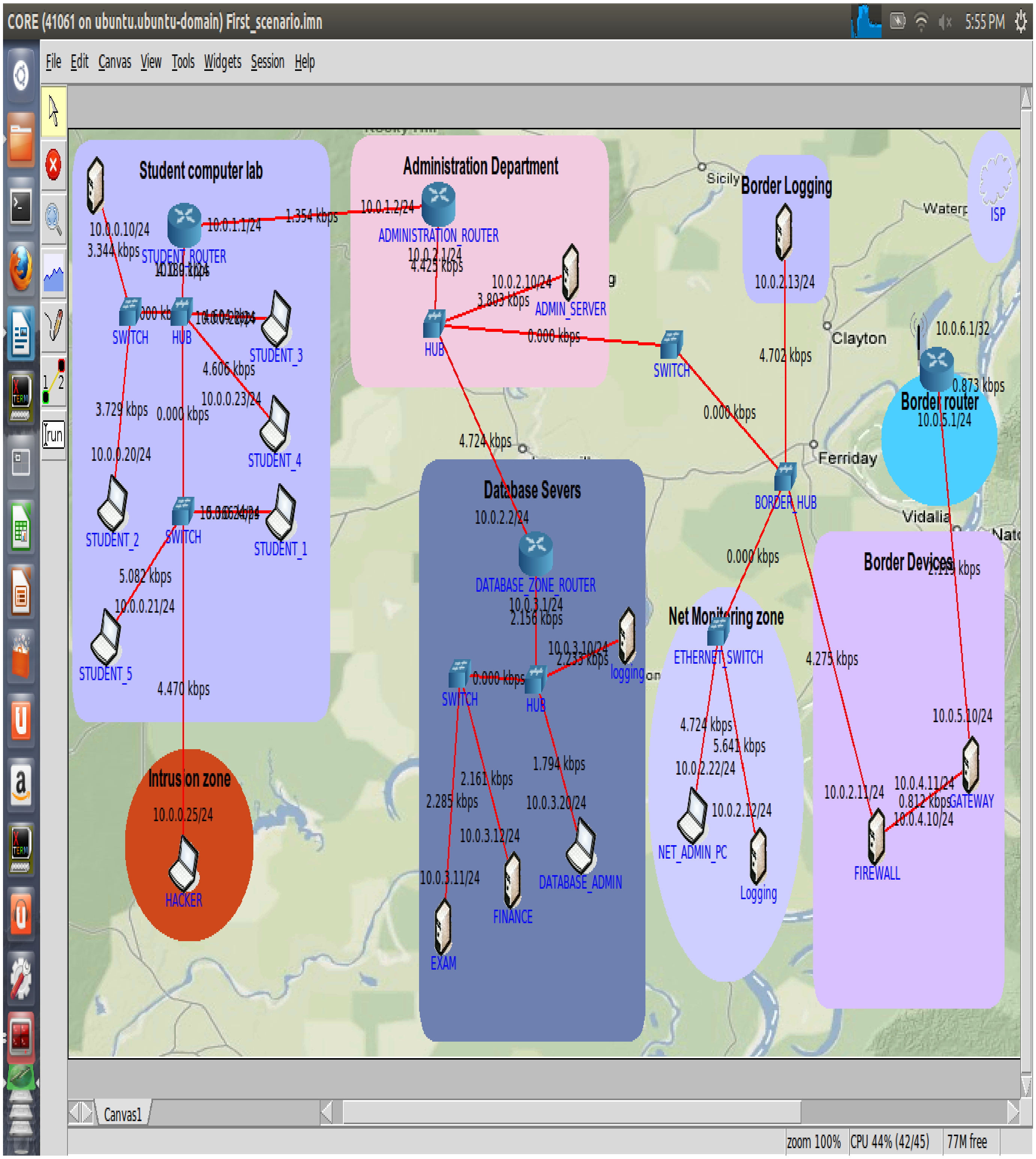


Figure 5: Throughput Simulation responses before routing stack transformation

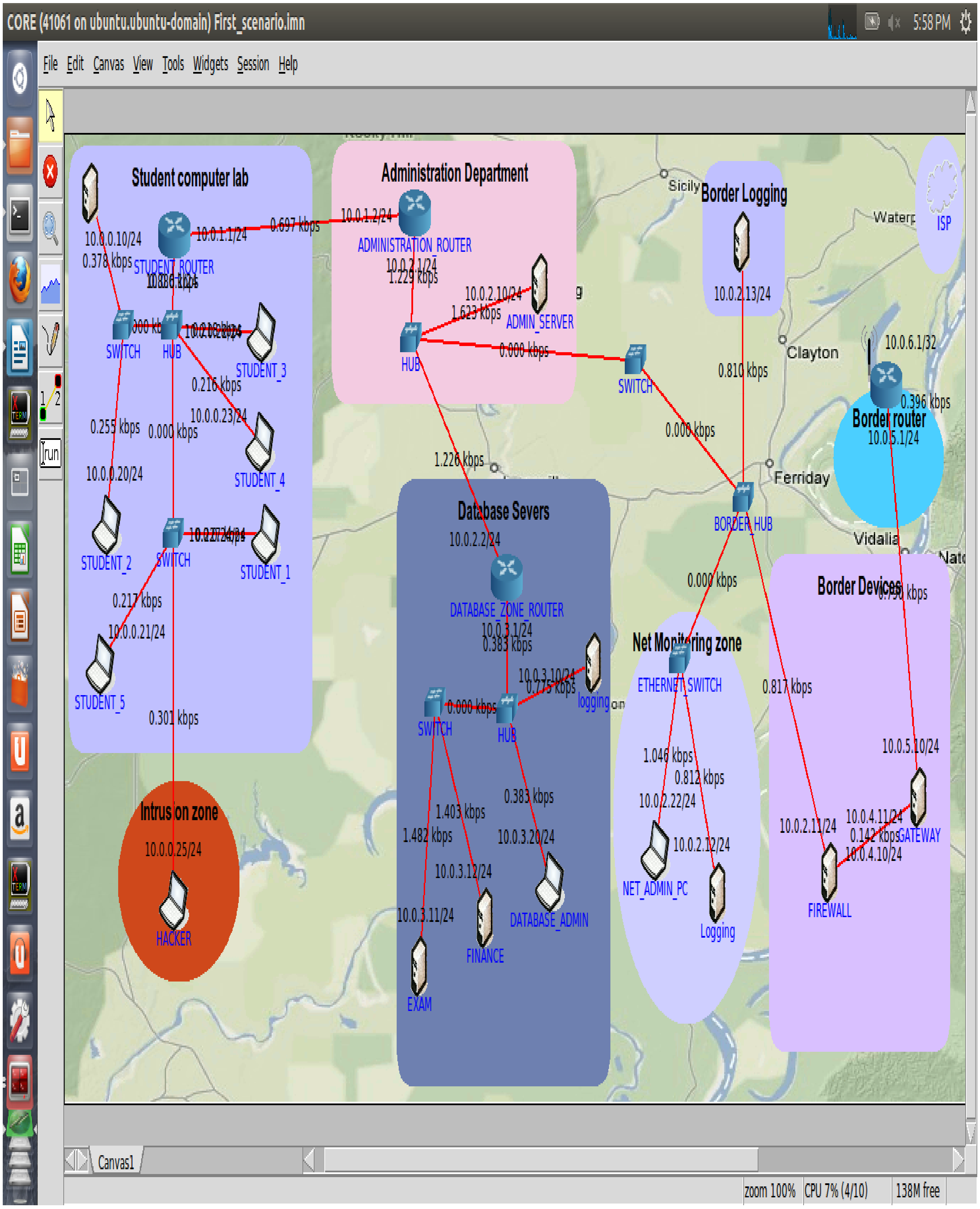


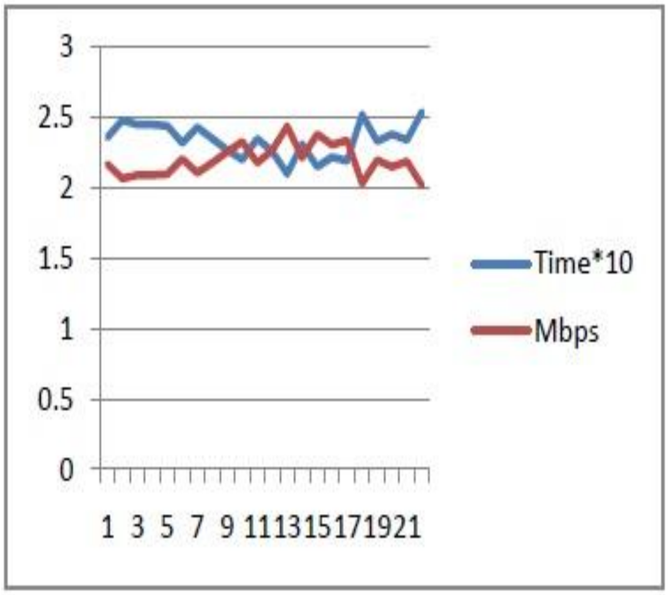
Figure 6: Throughput Simulation responses after routing stack transformation

These reductions can be attributed to the fact that network devices now take some time trying to understand the new format of the TCP/IP packets. These changes affect the crucial manner in which the TCP/IP packets are packaged to travel along the network. Particularly, larger MSS, window size and PTU all lead to slower network connections. This could be the case especially when the ip_don't_defragment flag is set, meaning that the destination machine has no capacity to slice the IP packets so that they can be transmitted effectively across faster networks. The kernel snippet responsible for the defragmentation was found by [6] to be as shown in Table 1 below.

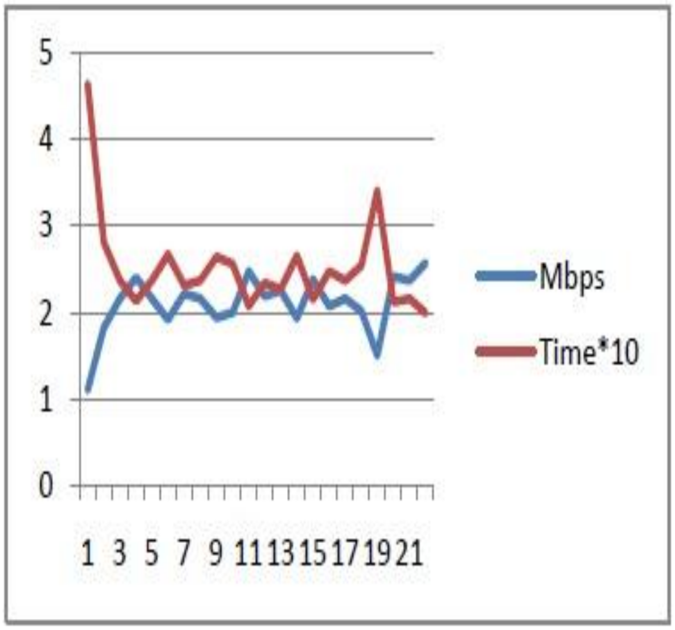
```
static inline
int ip_dont_fragment(struct sock *sk, struct dst_entry
*dst)
{
return 0;
return (inet_sk(sk)->pmtudisc == IP_PMTUDISC_DO ||
(inet_sk(sk)->pmtudisc == IP_PMTUDISC_WANT &&
!(dst_metric(dst, RTAX_LOCK)&(1<<RTAX_MTU))));
}
```

Table 1: Kernel snippet for setting null to defragmentation

To validate the simulated results, a prototype was developed and tested under similar conditions. Graphs 1 and 2 below shows the obtained responses.



Graph 1: Network throughput before Routing stack transformations



Graph 2: Network throughput after Routing stack transformations

From the graphs, it was observed that network access rates, were significantly reduced. Graph 2 shows a sharp reduction for the initial packets sent. This can be due to the fact that discovery of the new TCP/IP packets architecture had to take place, hence the subsequent delays.

IV. CONCLUSION

Routing stack transformation provides a convenient way of encapsulating target host crucial information from the spying eyes of network intruders. With this mechanism, network scanners such as Nmap return inaccurate or no information at all about the target host. This greatly reduces the chances of specific attacks targeted at particular internetworking operating systems (IOS). However, as the results of this paper have shown, these transformations come with reduced network access rates. There is therefore a tradeoff between security and network speeds. The focus of future research therefore lies in devising mechanisms of encapsulating router information while at the same time maintaining the network access rates. Also the number of TCP/IP flags and options to be re-structured were combined using probabilistic approach. Future works should hence deal with mathematical models of achieving these optimal combinations.

References

[1]. B. M. Leiner, V. G. Cerf, D. D. Clark, R. E. Kahn, L. Kleinrock, D. C. Lynch, J. Postel, L. G. Roberts, and S. Wolff, "A brief history of the Internet," SIGCOMM Computer Communication Review, vol. 39, no. 5, pp. 22–31, 2009.

[2]. Cisco Systems, Inc. "Cisco visual networking index: Forecast and methodology", 2009–2014. White Paper, 2010.

[3]. Tobias Lauinger, "Security & Scalability of Content-Centric Networking", 2010.

[4]. J. Yan, A. Blackwell, R. Anderson, and A. Grant. "Password memorability and security: Empirical results". IEEE Security and Privacy, 2(5):25 – 31, September – October 2004.

[5]. CERT Coordination Center. "Incident and vulnerability trends". Technical report, CERT, May 2003.

[6]. Vincent Omollo, Dr. Langat, Dr. Musyoki, "Secure LAN Architecture by using routing stack transformation", 2013.