National College of Ireland

Routing Table Size Aware Dynamic Routing for Maximization of Throughput of an SDN

Research Project
MSc Cloud Computing

Siddharth Sachin Kulkarni
x16110293

School of Computing
National College of Ireland

Supervisor: Dr. Paul Hayes

National College of Ireland
Project Submission Sheet – 2016/2017
School of Computing

Student Name:	Siddharth Sachin Kulkarni
Student ID:	x16110293
Programme:	MSc Cloud Computing
Year:	2017
Module:	Research Project
Lecturer:	Dr. Paul Hayes
Submission Due Date:	16/08/2017
Project Title:	Routing Table Size Aware Dynamic Routing for Maximization of Throughput of an SDN
Word Count:	4480

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:	
Date:	19th September 2017

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
3. Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Routing Table Size Aware Dynamic Routing for Maximization of Throughput of an SDN

Siddharth Sachin Kulkarni
x16110293
Research Project in MSc Cloud Computing

19th September 2017

Abstract

Virtual Networks are more commonly used for the purpose of research, engineering, etc. The rise in the concept of virtual networks gave birth to SDNs. These are programmable networks which can be configured each time based on the needs of the user. SDNs are made up of virtual routers, switches, controllers and virtual hosts. The main aim of this research is to increase the throughput of virtual networks by implementing a dynamic routing algorithm. This algorithm uses the routing table size as one of the parameters while deciding the order in which the packets are transmitted from the source device to the destination device in a virtual router. The port numbers of the source and destination are also recorded in the process. Packets that need to be transmitted are checked for the completeness of the information in the routing table and then sent to their destinations. The systems that are available currently do not consider the size of the routing table while implementing routing algorithms to increase the throughput of the system. While some authors have explored the possibility of incremental deployment for increasing the throughput, it has still not been solved taking the port numbers of source and destination. This paper implements an algorithm called the Route Mapper which makes use of a map table. The map table has all the details about the packet including the information that can be found in the routing table. Through this research, we can clearly observe that a substantial increase in the throughput of the network can be overserved when the proposed algorithm is implemented.

Keywords *Software Defined Networking, Routing, Throughput Maximization, Routing Table Search*

1 Introduction

Software Defined Networking is a fairly new concept in the field of networking. Few of the key features of SDN are that they are extremely cost effective, dynamic and manageable. Due to this, SDNs are compatible for applications that are of high bandwidth. In SDNs, the control plane and the data plane are decoupled and the control plane is centralized. (McKeown; 2009) The control plane sends signals to the data plane regarding the destination of the packets and the most appropriate paths to be followed in order for the packet to reach its destination quickly. The architecture of SDN allows network engineers

to directly program it which is one of the advantages of decoupling the forwarding functions from the data plane. This gives scope for network engineers to adjust the flow of traffic in the network in order to adjust to the networks requirements. The control plane of an SDN centrally controls the data planes and hence it appears as one switch. The control plane has the complete global view of the network. Network engineers have the option to dynamically configure the security, resources and manage them by making use of network programs. By use of SDN, the network architecture becomes more simplified and clear to understand. (*Software-Defined Networking (SDN) Definition*; n.d.)

Mininet is a network emulator which enables researchers, developers, network engineers, etc. to deploy large sized networks on simple computers and virtual machines in-spite of having limited resources (Figure 1). It allows users to run code without having to modify it. (Kaur et al.; 2014) This helps in the elimination of the set-up and maintenance costs that would generally be incurred while using physical network devices. The hardware test beds are extremely expensive and need expert skills in order to operate. Simulations help in solving this problem partially since most of the simulators do not give much performance. Mininet makes use of the OpenFlow controller and is completely open source. Real world network situations can be emulated using this tool since it also allows creation of virtual hosts, switches, controllers, routers, etc. The OpenFlow protocol is used as the communication protocol between the controller and the virtual devices.

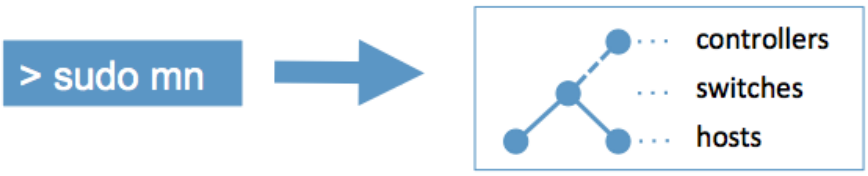

Figure 1: The Mininet Network Emulator

Routers are devices that connect networks and decides the destination and path of the packets. The routers in networks are generally installed at the gateways in networks. Each router has a routing table which contains destination address of the packets and the most efficient router that the packet should follow in order to reach the destination in minimum time. It also contains information about the next router that the packet will come across in the route to the destination. The function of routing is linked to the network layer of the OSI model. Switches that belong to the third layer are capable of the function of routing. (Rouse; n.d.) The size of the routing table in a router varies from the size and capacity of the router and the manufacturing company. The process of routing consists of three main entities; the sending host, the intermediate routers or the routers that are encountered on the path to the destination and the receiving host. During the process of routing, the packets are transferred from higher level protocols like TCP, ICMP, etc. to lower level protocol like the IP protocol. (*Routing Processes*; n.d.) The working of a virtual router can be seen in Figure 2

In virtual networks, the function of routing is performed by a virtual router or a vRouter which mimics the functions of a hardware device in the network. Virtual routers are part of the concept of Network Functions Virtualization (NFV). (*Virtual Router* -

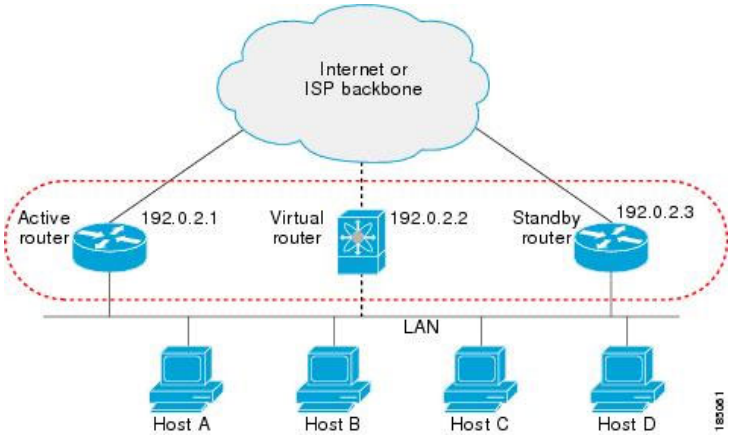

Figure 2: A Virtual Router

Definition; 2017) By the implementation of virtual routing, the IP protocol gets unbound from the hardware. This simply means that the router is free to traverse around the network or the data centre rather than being bound at a single gateway or node. (*Virtual Router - Definition*; 2017) The vRouters are basically software frameworks for the task of routing. They can be used in place of a physical router in order to perform the functions of routing in a LAN. (*What is a Virtual Router? - Definition from Techopedia*; 2017)

Routing tables are installed on IP nodes. As stated earlier, the routing tables contain crucial information about the packets (Figure 9). The IP protocol checks the routing table for information regarding the shortest path to the destination in order to ensure delivery of packets over the network in minimal time. If the route field for a particular packet is found to be null in the routing table, a routing error is sent by the IP protocol to the higher level protocols like TCP, ICMP, etc. (*Routing Processes*; n.d.) Along with the general information about the packets, extra entries are added to the routing table through the higher level protocols and also dynamically via communication with the other routers in the network. A routing table typically consists of values like the Network ID, Network Mask, Next Hop, Interface, Metric, etc. (*IP Routing Table*; 2017)

Network Destination	Netmask	Gateway	Interface	Metric	Purpose
0.0.0.0	0.0.0.0	157.55.16.1	157.55.27.90	1	Default Route
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0.1	1	Loopback Network
157.55.16.0	255.255.240.0	157.55.27.90	157.55.27.90	1	Directly Attached Network
157.55.27.90	255.255.255.255	127.0.0.1	127.0.0.1	1	Local Host
157.55.255.255	255.255.255.255	157.55.27.90	157.55.27.90	1	Network Broadcast
224.0.0.0	224.0.0.0	157.55.27.90	157.55.27.90	1	Multicast Address
255.255.255.255	255.255.255.255	157.55.27.90	157.55.27.90	1	Limited Broadcast

Figure 3: Routing Table of Windows 2000

2 Related Work

Installing and configuration of networks requires individuals having high amount of skills and expertise in the networking area. (Kreutz et al.; 2013) In situations involving interactions between different nodes of the network, a more in-depth knowledge is necessary and certain components of simulation need to be incorporated. (Sezer et al.; 2013) Computer networks generally consist of number of components such as routers, switches, hosts, etc. along with many other protocols that are complex in nature. Network engineers set up these components of the network and ensure that various policies are incorporated to make sure that the signals react to various events and incidents in the network. (Nunes et al.; 2014) Engineers have very limited tools in hand while operating with networks. Another major challenge faced by researchers and engineers of networks is the Internet Ossification. This means that, owing to the large size of the internet, it is extremely difficult for it to evolve - both physically and also by performance. Evolution of the internet is extremely important since the applications running on the internet are constantly becoming more complex and need new types of resources. (Nunes et al.; 2014) This has led to the development of the concept of Software Defined Networking. SDN in simple terms would mean networks that are programmable. It is a fairly new concept in networking in which the hardware in a network is separated from the control planes and this in turn helps in the evolution of the internet.

2.1 Software Defined Networking

According to Sezer et al. (2013), SDN primarily has four key attributes: (i) The control plane is detached from the data plane. The devices on the network become simple elements that forward packets since they do not have the control plane. (ii) The controller has complete visibility of the network and is centralized. The decisions regarding the forwarding of signals is made based on the flow rather than the destination of the packets. Thus, this increases the flexibility of the network. (iii) The network components in the controller and in the data plane have a common open interface. The control plane is moved to an external entity. and (iv) The network is programmable using external softwares. This key attribute of the network is considered as one of the main value propositions of Software Defined Networking. (Kreutz et al.; 2015)

SDNs mainly operate using standardized APIs (Application Programming Interfaces) in order to help network engineers to reconfigure the handling of data and resources in a network. This helps applications running on networks to easily update their network components. Software Defined Networking is now being extensively used in the fields of virtualization and cloud computing. (Kirkpatrick; 2013) Applications running in a Software Defined Networking Environment can configure the switches of a network by simply making use of different APIs as per their individual requirements. This is possible due to the unique manner in which the different components of the network like routers, switches, etc. are deployed. This majorly changes the limitations of the networking infrastructures.

Ethernets and switches found in networks have flow tables that help in implementation of various features of the network like building of firewalls, improving the QoS, collecting the network data for generating the statistics, etc. (Kim and Feamster; 2013) The flow tables belonging to the switches and routers of different vendors varies but they all have

certain similar properties. All these similar properties can be found in the OpenFlow switches which are one of the key elements of Software Defined Networking. (McKeown et al.; 2008) The OpenFlow switches make use of an open protocol in order for the purpose of programming the flow tables of the virtual switches and router in an SDN. (Yamanaka et al.; 2014) The traffic of the network can be divided into research flow and product flow by the administrator. According to McKeown et al. (2008), the OpenFlow switches are fundamentally made up of 3 segments: (i) Flow table which is consists of directions to the switch on how to process the flow. (ii) A secure connecting channel between the switch and the controller. and (iii) The protocol for the communication between the controller and the switch called as the OpenFlow protocol.

OpenFlow mainly has three different types of operation: Reactive, Proactive and Hybrid. (Liu et al.; 2015) The reactive mode reacts to the signals in a network or the traffic. Any new flow that enters the switch is searched in the flow table and if no entry is seen, a new OFP packet-in packet is created and sent to the controller in order to get instructions. The proactive mode, unlike reactive mode speculates the different types of traffic matches that could enter the switch and populates the flow table with it. This results in timely delivery of the packets in the network. The third mode i.e. Hybrid mode is a combination of both the proactive mode and the reactive mode of operation in networks.

2.2 Routing

The process of transferring of packets from one host to another is referred to as routing. Routers are the devices that generally help in the process of routing. (Nascimento et al.; 2011) The signals or information that are being sent through the network are made up of packets. These are the most fundamental units of a transfer of signals or information. Modern SDNs have centralized RCPs or Routing Control Platforms. These help in increasing the routing flexibility of the networks. This also results in increase in the security of the router. (Rothenberg et al.; 2012) The virtual routers found in SDNs follow the RouteFlow protocol. This protocol is responsible for the operation of the control logic of the OpenFlow switches and is primarily made up of Virtual Machines. The main objective of the RouteFlow protocol is to enable remote IP routing services centrally since the data and control planes are already separated in SDNs. The messages that are transmitted using the routing protocol can either be sent to the physical devices or can be retained in the virtual networks. (Nascimento et al.; 2011) The Forwarding Information Base (FIB) is generated by the routers based on the different routing protocols like OSPF, Routing Information Protocol, Border Gateway Protocol, etc. (Lin et al.; 2013) All routers make use of routing tables in order to store information like addresses of other devices in the network and the most efficient path to send packets to those devices. The size of the routing table varies based on the size and the brand of the router. Routers in smaller network generally have smaller size routing tables whereas routers in large data centre have routing. Some routers that are being used at a large scale even have sizes of 5,12,000 routes. The fields that can be generally found in typical routers are, metric, interface, destination, next hop and network mask. The destination of the packets can sometimes be IP address of the receiving device or it can be a network ID. The next hop field refers to the next link or router that will be met by the packet while being transmitted to the destination device. The metric field is fairly important since it holds

information about the best route to be followed by the packet in order to reach the destination. This varies based on the routing algorithm that is being used in the process of routing. The best path is often the route that takes the least amount of time for the packets to reach. This helps in ensuring the efficiency of the network.

The throughput of a network is determined based on the successful delivery of the packets from the source to the destination. The throughput of a network also varies based on the time consumed by the packet to reach the destination address from the source.

3 Methodology

In this project, the mininet network emulator has been used for the purpose of emulation of an SDN. Mininet has been installed onto a Ubuntu 64-bit virtual machine and has been allocated 3 GB of RAM. PuTTY has been used for the purpose of terminal emulation and as a file transfer application for the network. The wireshark packet analyser has been used in order to analyse the traffic in the network and accordingly generate graphs. Wireshark (Figure 4) allows developers to filter the network traffic based on the protocols being followed by the signals and hence take appropriate decisions regarding the research.

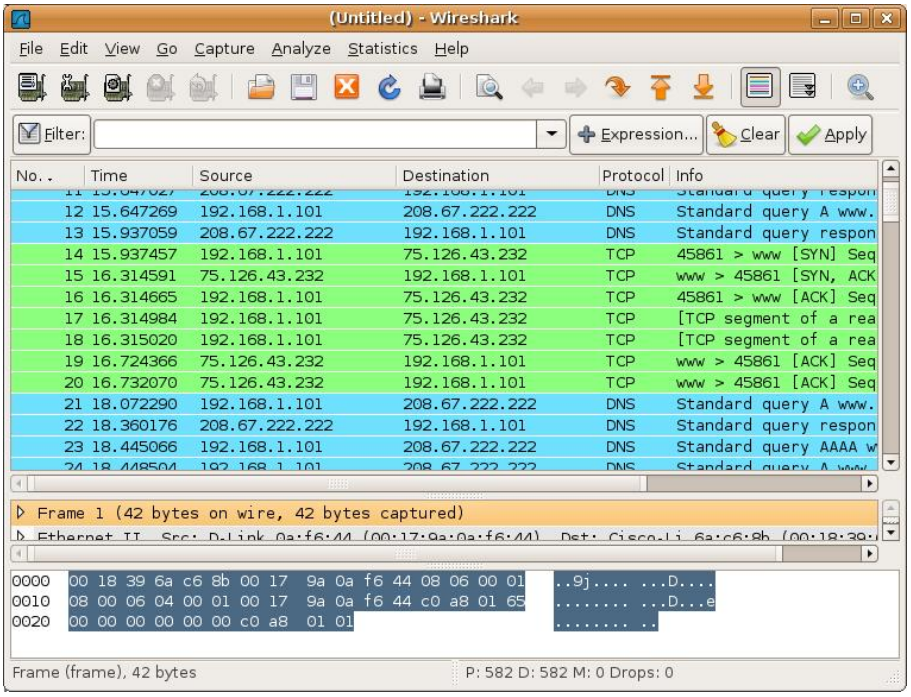

Figure 4: Traffic as seen in Wireshark Network Analyser

Mininet enables the users to add switches, hosts, controllers, routers and other network devices virtually based upon our need. This feature of mininet was particularly helpful for us since our primary focus was to analyse the routing table size of the router and then dynamically add network elements. The routing table contains crucial and important information regarding the packets and the signals being transmitted.

In Figure 5, you can see a sample routing table. This routing table belongs to the kernel of the virtual machine that has been set up for the purpose of this project.

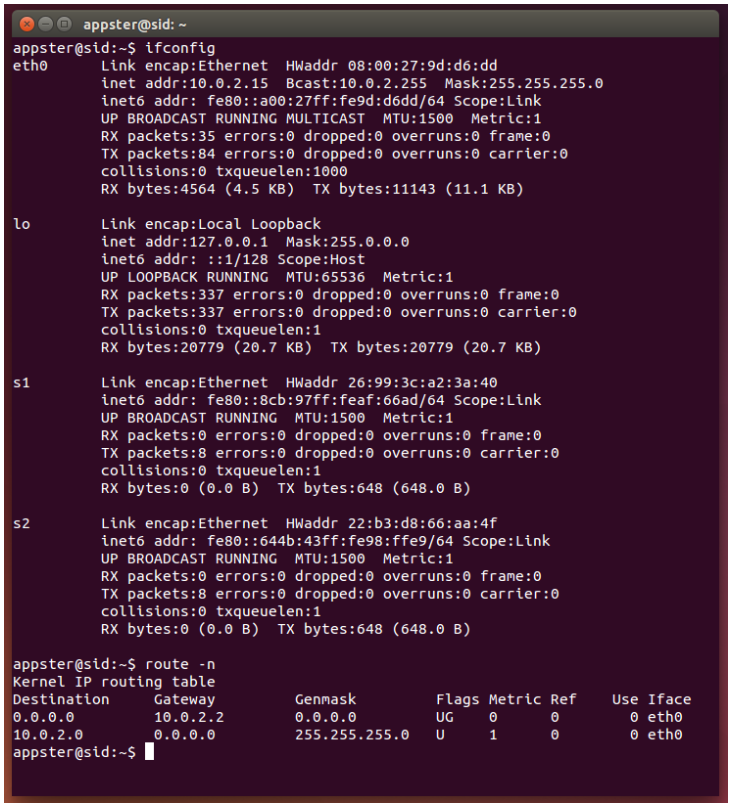

Figure 5: Routing Table and IP Addresses of the configured VM

On execution of the command ifconfig, the complete IP configuration of the system is listed. The IP address of the system as seen in the eth0 section is 10.0.2.15, meaning, any packets created by the virtual machine sid will have the source address set to 10.0.2.15. Similarly, the packets that are being sent to the virtual machine will have the same IP address set as their destination. Since the virtual machine will accept any inbound packets bearing their destination address as 10.0.2.15, any packets found on the ethernet with this destination address will be sent to the virtual machine. This is referred to as Address Resolution Protocol in networking. The most common types of traffic that computer generally encounter are unicast traffic. Different protocols like SSH, HTTP, POP3, etc. come under this category of traffic.

On observing the routing table found in the lower part of the image, we can observe that the IP address of the virtual machine (10.0.2.15) falls in the address space of 10.0.2.0/24. Since this line is visible in the routing table, it simply means that the virtual machine has been successfully connected to the network. The ping command can be used in order to test the ICMP protocol (Internet Control Message Protocol). If a static route is found in the routing table, it is a direction being given to the router to send the packets through that particular path. If no static route is found, the computers will use the default gateway for the purpose of transmission of packets in the network which might not always be an efficient choice. Figure 6 shows the different types of protocols that are available for routing.

For every instance found in the routing table, a bit-wise logical AND is performed over the IP address and the network mask. This is then checked against the Network

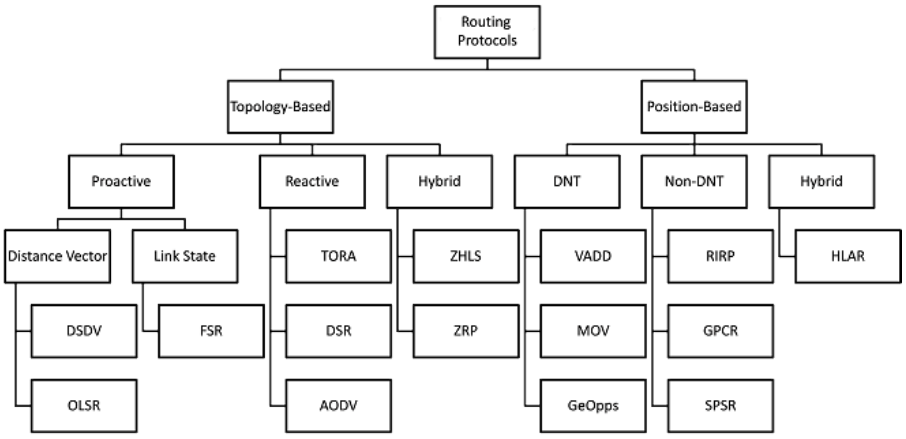

Figure 6: The different routing protocols

ID table for a match. Once checked, the path that has the longest similarity is chosen as the most appropriate path for the transfer of packets. If multiple instances of longest matches exist, the instance with the lowest metric is chosen as the most efficient path to the destination since that route would ensure fast transfer of the packets without any loss.

Many a times, the routing table is found to be missing a few entries. When the router encounters such a situation, it is in a position where the transfer of packets is not possible since the routing table is missing important information. In such cases, the IP protocol sends an error message to the higher level protocols such as TCP, IMCP. This process consumes a lot of time and hence reduces the efficiency of the network.

In order to eliminate this problem, the implemented solution first checks the size of the routing table and checks the entries in it. If the entries in the table are found to have missing values, then the implemented dynamic routing algorithm completes the transfer of other packets and while waiting for the error response from higher level protocols. As soon as the other packets are transferred, the packet having the missing value is transmitted in order to complete the entire process of transmission. While doing this entire process, the port number of the source and destination are also recorded by the system for the purpose of identifying the reason for the discrepancy in the routing table.

4 Implementation

This algorithm makes use of a unique map table in order to increase the throughput of the network using dynamic routing. The details regarding the packets in queue for transmission and their routing tables are put up into the map table. Each packet in queue is assigned an ID in order to ensure that the packets are easily distinguished based on

their time of arrival into the map table. The map table is then searched to check which entry into the table has the largest number of entries in their routing table. By default this packet is picked and sent for transmission while the packets having missing values are sent an error message to the protocols of higher level.

Prior to this step, Ubuntu 14.04 has been installed in the Virtual Box and has been set up. Once installed, mininet has been installed in the Ubuntu virtual machine. The mininet stand alone virtual machine can also be used for this purpose but I have installed the mininet application into another virtual machine in order to gain more flexibility with the machine and to be able to store programs locally and run them from the terminal.

```
for r in [0, x - 1] do
  Calculate Sr
  for each s, 0 ≤ s ≤ d - 1 do
    for each prefix p ∈ Gr do
      Calculate hs (p)
      Assign key p to the routetable hs(p)
    end for
    Calculate the shortest path value spvl
  end for
  Find the min-max routes spvl and set V (r) = j
  Assign all the keys in Gi to maptable using hs
end for
```

Figure 7: The proposed algorithm "Route Mapper"

Once the above two steps are completed, the map table containing values about the packets and their routing table information need to be searched in order to check which has the most number of prefixes. Since it would be difficult to manually search and compare each value with the next one, the greedy search method has been used which consists of two stages; the first stage and the refine stage. In the first stage, we do not always achieve the most accurate output since this step involves only the variables that have half contribution towards calculating the computation load. The refine stage assigns values to the packets based on their computation load in order to ensure that the packets that have the most computation load are sent first while the packets awaiting response from higher level protocols receive the necessary information. The map table can undergo 3 types of operations: (i) Inserting new information whenever a new packet arrives for transmission. (ii) Deleting the existing entries if the map table is unable to find information that is missing from the routing table. (iii) Modify the routing information of the entries that are already existing in the table. The deletion and updation of the routing information inside the map table happens in a simple manner. However, whenever a new packet arrives into the network, if the packets currently in the table are already in transmission, it would be difficult to interrupt the process and then add the new entry since the packets would already be in the process of being transmitted. Hence, the

best possible way to solve this problem would be to create another new map table and parallelly start executing it once the packets are ready for transmission.

5 Evaluation

When a packet arrives for transmission the length of its prefix is matched by the router to check for the longest similarity. The similarity is achieved by performing a logical bit-wise AND over the IP address and the subnet mask and comparing it with the Network ID column in the routing table. The last four bits of the destination address is hashed to check if the source and destination addresses belong to the same address space. Four router elements and their routing tables have been used for the purpose of this experiment. The routing tables of routers installed at virtual controllers, virtual switches, virtual hosts and virtual routers have been taken in to consider for the purpose of analysis. All the routing tables are hashed both in a normal routing environment and in the improvised dynamic routing environment that has been proposed through this paper. The routing throughput and the memory size is compared assuming that the data regarding routing is stored in 64 bits. The routing throughput is evaluated using the Wireshark network analyser and the same is used to produce data regarding the network traffic observed during the time of capture. The impacts of using multiple map tables together is yet to be observed. The case studies that are taken into consideration below are the various.

5.1 Experiment : Simplified fast look-up of the Routing Table

For every instance where a packet arrived at a virtual switch in order to be transmitted to another destination in the network, the lookup function was evoked. In case of a virtual network, the complete address of the destination and source devices is needed for the lookup function to be executed. In case of a physical network, the function only needs the identifier of the connection for successful delivery of the packets. When multiple packets are being forwarded from the source to the destination through the same router, only the first packet contains the full address of the destination while other packets just have the identifier specified. Such a type of lookup is generally observed in ATM machines.

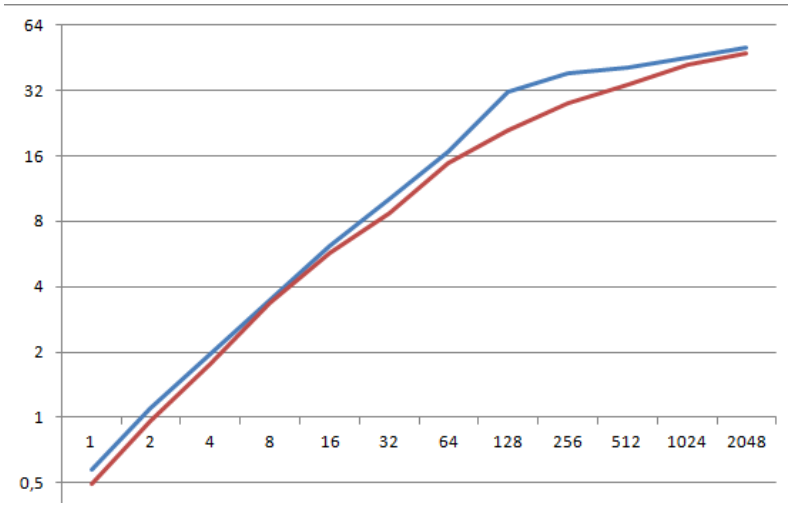

Figure 8: Graph showing comparison between the present system and difference after implementing the new algorithm. (Blue line indicates the old system and red line indicates the throughput of the routemapper algorithm

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.0.2.2        0.0.0.0         UG    0      0        0 eth0
10.0.2.0        *               255.255.255.0   U     1      0        0 eth0
```

Figure 9: Routing table of the switch


```
"No.","Time","Source","Destination","Protocol","Length","Info"
"1","0.000000000","127.0.0.1","127.0.0.1","TCP","74","48198 > 6633 [SYN] Seq=0 Win=43690 Len=0 MSS=65495 SACK_PERM=1 TSval=4842 TSecr=0 WS=512"
"2","0.000020000","127.0.0.1","127.0.0.1","TCP","54","6633 > 48198 [RST, ACK] Seq=1 Ack=1 Win=0 Len=0"
"3","0.129170000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x41b4  A daisy.ubuntu.com"
"4","0.129181000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0xfb72  AAAA daisy.ubuntu.com"
"5","0.152556000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0xfb72 "
"6","0.152586000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0x41b4  A 162.213.33.164 A 162.213.33.133"
"7","0.153129000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x6126  A daisy.ubuntu.com"
"8","0.153137000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x27e5  AAAA daisy.ubuntu.com"
"9","0.169845000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0x6126  A 162.213.33.164 A 162.213.33.133"
"10","0.172392000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0x27e5 "
"11","0.212301000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0xc53b  A daisy.ubuntu.com"
"12","0.212309000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x579e  AAAA daisy.ubuntu.com"
"13","0.226612000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0xc53b  A 162.213.33.164 A 162.213.33.133"
"14","0.230480000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0x579e "
"15","0.231055000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x51d7  A daisy.ubuntu.com"
"16","0.231060000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x0fb3  AAAA daisy.ubuntu.com"
"17","0.250091000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0x51d7  A 162.213.33.164 A 162.213.33.133"
"18","0.254874000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0x0fb3 "
"19","0.300771000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0xd963  A daisy.ubuntu.com"
"20","0.300778000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x9b1e  AAAA daisy.ubuntu.com"
"21","0.317494000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0xd963  A 162.213.33.164 A 162.213.33.133"
"22","0.317800000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0x9b1e "
"23","0.318387000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x41c2  A daisy.ubuntu.com"
"24","0.318391000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x6b6e  AAAA daisy.ubuntu.com"
"25","0.338004000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0x41c2  A 162.213.33.164 A 162.213.33.133"
"26","0.340887000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0x6b6e "
"27","0.403853000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x7970  A daisy.ubuntu.com"
"28","0.403862000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x1f4f  AAAA daisy.ubuntu.com"
"29","0.419786000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0x7970  A 162.213.33.164 A 162.213.33.133"
"30","0.424254000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0x1f4f "
"31","0.424965000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x6d9b  A daisy.ubuntu.com"
"32","0.424973000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0xe3d7  AAAA daisy.ubuntu.com"
"33","0.440413000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0x6d9b  A 162.213.33.164 A 162.213.33.133"
"34","0.447348000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0xe3d7 "
"35","0.494065000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x2111  A daisy.ubuntu.com"
"36","0.494073000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x272a  AAAA daisy.ubuntu.com"
"37","0.507502000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0x2111  A 162.213.33.164 A 162.213.33.133"
"38","0.517712000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0x272a "
"39","0.518271000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x5917  A daisy.ubuntu.com"
"40","0.518276000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0x7fe8  AAAA daisy.ubuntu.com"
"41","0.536662000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0x5917  A 162.213.33.164 A 162.213.33.133"
"42","0.541927000","127.0.1.1","127.0.0.1","DNS","137","Standard query response 0x7fe8 "
"43","0.588971000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0xa16f  A daisy.ubuntu.com"
"44","0.588979000","127.0.0.1","127.0.1.1","DNS","76","Standard query 0xc312  AAAA daisy.ubuntu.com"
"45","0.603379000","127.0.1.1","127.0.0.1","DNS","108","Standard query response 0xa16f  A 162.213.33.164 A 162.213.33.133"
```


Figure 10: Data set used for running the tests

6 Conclusion and Future Work

Software Defined Networking is one of the few ways using which the field of internet communication can evolve and expand. While new programs and APIs help in making the job of a network engineer easier, it is equally important to optimize and improvise the existing network topologies since it would help in saving a lot of time while ensuring 100 per cent utilization of the network resources. Hybrid SDNs are very unique since they have the pros of both Proactive and Reactive SDNs. By implementing the algorithm that has been proposed in this research, it would be clearly possible to increase the throughput of networks. Through this research, I have successfully demonstrated that the throughput of a network can be increased simply by implementing a new algorithm for routing which I have names as "Route Mapper". The algorithm takes into consideration the size of the routing table in order to more efficiently utilize the routers and reduce the number of packet losses and delays in transmission. The algorithm also records the port number of both the source and the destination devices.

One of the future works of the project could be improvising the algorithm so that it can dynamically update the map tables as and when a new packet arrives at the switch in order to be transmitted to the destination. The current algorithm does now allow new packets to be added to the table while the older ones are still in the process of being transmitted. The research can also be further developed by trying to implement the algorithm for devices that are on a Wide Area Network.

Acknowledgements

I would like to take this opportunity to sincerely thank my mentor **Dr.Paul Hayes** who has been constantly guiding me throughout the course of this research right from the first day till the day of submission. I would also like to thank him for all the suggestions and ideas given for improvising the idea of the research.

I would like to express my gratitude to my course director and the head of Cloud Competency Centre **Dr.Horacio Gonzalez-Velez** for helping in the early stages of research and in the process of decision of topics.

Finally, I would like to thank my parents for their continuous encouragement and support and for always having believed in me. I would also like to thank my friends for their help and assistance during my research.

References

IP Routing Table (2017).
URL: *https://technet.microsoft.com/en-us/library/cc958823.aspx*

Kaur, K., Singh, J. and Ghumman, N. S. (2014). Mininet as software defined networking testing platform, *International Conference on Communication, Computing & Systems (ICCCS)*, pp. 139–42.

Kim, H. and Feamster, N. (2013). Improving network management with software defined networking, *IEEE Communications Magazine* **51**(2): 114–119.

Kirkpatrick, K. (2013). Software-defined networking, *Communications of the ACM* **56**(9): 16–19.

Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S. and Uhlig, S. (2015). Software-defined networking: A comprehensive survey, *Proceedings of the IEEE* **103**(1): 14–76.

Kreutz, D., Ramos, F. and Verissimo, P. (2013). Towards secure and dependable software-defined networks, *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, ACM, pp. 55–60.

Lin, P., Hart, J., Krishnaswamy, U., Murakami, T., Kobayashi, M., Al-Shabibi, A., Wang, K.-C. and Bi, J. (2013). Seamless interworking of sdn and ip, *ACM SIGCOMM computer communication review*, Vol. 43, ACM, pp. 475–476.

Liu, Y., Hecker, A., Guerzoni, R., Despotovic, Z. and Beker, S. (2015). On optimal hierarchical sdn, *Communications (ICC), 2015 IEEE International Conference on*, IEEE, pp. 5374–5379.

McKeown, N. (2009). Software-defined networking, *INFOCOM keynote talk* **17**(2): 30–32.

McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S. and Turner, J. (2008). Openflow: enabling innovation in campus networks, *ACM SIGCOMM Computer Communication Review* **38**(2): 69–74.

Nascimento, M. R., Rothenberg, C. E., Salvador, M. R., Corrêa, C. N., de Lucena, S. C. and Magalhães, M. F. (2011). Virtual routers as a service: the routeflow approach leveraging software-defined networks, *Proceedings of the 6th International Conference on Future Internet Technologies*, ACM, pp. 34–37.

Nunes, B. A. A., Mendonca, M., Nguyen, X.-N., Obraczka, K. and Turletti, T. (2014). A survey of software-defined networking: Past, present, and future of programmable networks, *IEEE Communications Surveys & Tutorials* **16**(3): 1617–1634.

Rothenberg, C. E., Nascimento, M. R., Salvador, M. R., Corrêa, C. N. A., Cunha de Lucena, S. and Raszuk, R. (2012). Revisiting routing control platforms with the eyes and muscles of software-defined networking, *Proceedings of the first workshop on Hot topics in software defined networks*, ACM, pp. 13–18.

Rouse, M. (n.d.). What is router? - definition from whatis.com.
URL: *http://searchnetworking.techtarget.com/definition/router*

Routing Processes (n.d.).
URL: *https://technet.microsoft.com/en-us/library/cc958831.aspx*

Sezer, S., Scott-Hayward, S., Chouhan, P. K., Fraser, B., Lake, D., Finnegan, J., Viljoen, N., Miller, M. and Rao, N. (2013). Are we ready for sdn? implementation challenges for software-defined networks, *IEEE Communications Magazine* **51**(7): 36–43.

Software-Defined Networking (SDN) Definition (n.d.).
URL: *https://www.opennetworking.org/sdn-resources/sdn-definition*

Virtual Router - Definition (2017).
URL: *https://www.sdxcentral.com/nfv/definitions/whats-a-virtual-router-vrouter/*

What is a Virtual Router? - Definition from Techopedia (2017).
URL: *https://www.techopedia.com/definition/26162/virtual-router*

Yamanaka, H., Kawai, E., Ishii, S. and Shimojo, S. (2014). Openflow network virtualization on multiple administration infrastructures, *Proc. Open Netw. Summit*, pp. 1–2.