Turnitin Originality Report

Final Report by Dharmesh Bhanushali

From Project Report (EE297B Spring 2016)

Processed on 05-May-2016 01:29 PDT
ID: 668551007
Word Count: 11462

Similarity Index	Similarity by Source	
6%	Internet Sources:	2%
	Publications:	1%
	Student Papers:	5%

sources:

1. 1% match (student papers from 14-Aug-2015)
Submitted to CSU, San Jose State University on 2015-08-14

2. 1% match (student papers from 19-Jun-2015)
Submitted to CSU, San Jose State University on 2015-06-19

3. 1% match (student papers from 13-Mar-2015)
Submitted to CSU, San Jose State University on 2015-03-13

4. < 1% match (student papers from 29-Nov-2015)
Submitted to CSU, San Jose State University on 2015-11-29

5. < 1% match (student papers from 12-Aug-2015)
Submitted to CSU, San Jose State University on 2015-08-12

6. < 1% match (student papers from 31-Jul-2015)
Submitted to Higher Education Commission Pakistan on 2015-07-31

7. < 1% match (student papers from 01-Oct-2015)
Submitted to University of Hertfordshire on 2015-10-01

8. < 1% match (Internet from 02-Mar-2016)
http://www.thinkmind.org/download.php?articleid=iccgi_2011_7_30_10145

9. < 1% match (Internet from 25-Apr-2015)
http://www.atltelecom.com/download/download.php?id=52&mode=dynamic&method=download&referer=

< 1% match (Internet from 02-Aug-2015)

10 http://www.brocade.com/content/dam/common/documents/content-types/configuration-guide/netiron-05800b-switchingguide.pdf

11 < 1% match (Internet from 07-Mar-2014)
http://www.ece.virginia.edu/~mv/pubs/tutorials/ANTS2012/ANTS2012-Tutorial-PartI.pdf

12 < 1% match (Internet from 15-Nov-2010)
http://www.giikorea.co.kr/toc_en/min58767-ip-multimedia_toc.html

13 < 1% match (publications)
Liu, Ruoyan, Huaxi Gu, Yawen Chen, and Haibo Zhang. "Analyzing Packet-Level Routing in Data Centers", 2013 IEEE 11th International Conference on Dependable Autonomic and Secure Computing, 2013.

14 < 1% match (student papers from 26-Sep-2013)
Submitted to INTI International University on 2013-09-26

15 < 1% match (publications)
Son, Jungmin, Amir Vahid Dastjerdi, Rodrigo N. Calheiros, Xiaohui Ji, Young Yoon, and Rajkumar Buyya. "CloudSimSDN: Modeling and Simulation of Software-Defined Cloud Data Centers", 2015 15th IEEE/ACM International Symposium on Cluster Cloud and Grid Computing, 2015.

16 < 1% match (Internet from 29-Jan-2016)
http://www.academypublisher.com/jnw/vol08/no06/jnw0806.pdf

17 < 1% match (Internet from 05-Mar-2016)
https://greencloud.gforge.uni.lu/ftp/comm-metrics.pdf

18 < 1% match (Internet from 31-May-2015)
http://mirror.eu.oneandone.net/ubuntu/ubuntu/dists/trusty/universe/i18n/Translation-en

19 < 1% match (Internet from 28-Dec-2015)
http://conferences.sigcomm.org/sigcomm/2011/papers/sigcomm/p350.pdf

20 < 1% match (Internet from 23-Mar-2016)
http://faculty.nps.edu/xie/papers/safe-rr-infocom14.pdf

21 < 1% match (Internet from 11-Mar-2016)
https://ediss.uni-goettingen.de/bitstream/handle/11858/00-1735-0000-0022-5F4A-3/Thesis.pdf?sequence=1

22 < 1% match (Internet from 11-Apr-2016)
http://www.greenorbs.org/people/liu/guodeke/tpds-shuffle.pdf

23 < 1% match (publications)
Data Communications and Transmission Principles, 1997.

24 < 1% match (student papers from 08-Apr-2015)
Submitted to SUNY Institute of Technology at Utica/Rome on 2015-04-08

25 < 1% match (Internet from 10-Aug-2015)
http://www.researchgate.net/publication/236279819_Routing_Algorithm_An_Overview

paper text:

Analysis of SDN based ECMP for

5**Data Center Networks A Project Report Presented to The faculty of the Department of Electrical Engineering San José State University In Partial Fulfillment of the Requirements for the Degree Master of Science By**

Nirmal Karia 010018723 EE297B Section 01, Spring 2016 nirmalkaria@gmail.com (669) 300-7019 Dharmesh Bhanushali 010020114 EE297B Section 09, Spring 2016 dharmeshdpb@gmail.com (669) 225-8750 Department Approval

____ __________________________________ Dr. Balaji Venkatraman (signature)

4**Project Advisor __________________________________ Dr. Nader Mir** (signature) **Project Co-Advisor __________________________________ Dr. Thuy T. Le** (signature) **Graduate Advisor Department of Electrical Engineering Charles W. Davidson College of Engineering San José State University San Jose, CA 95192-0084**

__________________ Date __________________ Date __________________ Date TABLE OF CONTENTS

 ABSTRACT SDN is a rising phenomenon in today's computer network architecture. SDN splits the control plane and data plane of a network architecture. This split at each forwarding device is consolidated and handled by a centralized entity. This centralized control plane provides an open interface to forwarding devices and data plane. With lack of port densities and increased costs in data center architecture, fat tree architecture with high speeds links are more preferable. For multi-path routing data center architectures use ECMP as routing protocol to forward flows. Due to lack of dynamic scheduling capabilities and flexibility of ECMP, ECMP cannot not account for fair bandwidth utilization of a link. In this report, we performed two experiments in a fat tree topology based on SDN environment. The first is based on ECMP in fat tree topology. The second experiment involves the use of optimized ECMP in fat tree topology with the use of SDN based environment. The use of ECMP through open flow protocol (SDN based environment) made it possible to dynamically adjust flows. Thus, with the help of centralized control it is possible to dynamically assign traffic loads as per links bandwidth utilization. Thus the use of central controller can significantly outperform the traditional ECMP algorithm in data center networks. CHAPTER 1 - INTRODUCTION 1.1 DATA CENTER NETWORK HISTORY With the fast improvement of big data, distributed computing and Internet, for example, colleges, research labs and organizations, are building tremendous Data center to bolster their applications. Those applications, for example, scientific and data analysis, information investigation, warehousing and huge scale system administrations, require significant intra-cluster data transfer capacity (bandwidth). As those applications keep on expanding, scaling the limit and handling capacity of server farms has turned into a test issue. So as to adjust to existing network conditions, make proficient utilization of network resources, specialists use traffic engineering to manage flows for forwarding paths. Figure 1:- Timeline of Data Center Evolution Today's data centers constitute by a huge number of machines with huge total data transfer capacity requests. The conventional Internet correspondence are controllable through an unassuming number of given host sets. There are especially a little measure of ways between correspondence host sets. The data centers are always showing signs of change over both time and space. Those properties intrinsic in the server farm makes the customary network design and protocols no more suitable. An average data enter topology consists of three level trees of switches or switches. For instance, a three-layered topology

8**has a core** layer **in the** foundation **of the tree, an** aggregate layer **in the** center **and an edge** layer **at the leaves of the tree.**

Those designs have different ways points of interest, which can convey full transfer speed between discretionary hosts. To decrease costs, specialists proposed an enhanced design, Fat Tree topology. Most extensive IP systems work Open Shortest Path First (OSPF) to select a shortest path for every pair of correspondence host. In any case, if we use OSPF in traditional data center architecture, it cannot utilize multipath advantages of this data center topology. In server farm systems, Equal Cost Multipath (ECMP) is regularly used to statically select flows across multiple path available. As indicated by the stream's hash esteem, distinctive scopes of stream are sent to various way. This characterized mapping of streams to ways does not represent data transfer capacity use of current system. ECMP cannot understand reasonable utilization of connection's data transfer capacity. To tackle this issue, we utilize Open Flow convention to enhance the ECMP protocol. In this report, we created a data center topology in an SDN based environment calculated the bandwidth utilization of links in different scenarios and evaluated the performance of two experiments. We used ECMP in the first experiment in traditional data center topology and evaluated the results based on bandwidth utilization of links. In the next experiment we used ECMP in SDN based environment, where flow handling is managed by a central controller and compared the results based on bandwidth requires at the core links. Depending upon the bandwidth used at links and flow being forwarded in both experiment we evaluated the results. Most data center topologies have been composed as Fat Tree topology. These multiestablished trees have numerous equivalent cost paths between all host and destination pairs. A key test is to powerfully forward streams along these equivalent cost paths to enhance the throughput of Data centers. ECMP has been broadly utilized as a part of data center architecture to statically forward streams over numerous equivalent cost ways. This static mapping of streams to ways does not represent either current system usage or stream size, with coming about some connections data transmission use surpass limit while other still sit out of gear. To take care of this issue, we proposed SDN-based ECMP calculation to acknowledge dynamic stream forwarding as indicated by connections transmission capacity use. SDN is another system design that permits system overseers to control system movement adaptable. There is less research work in server farm system of SDN engineering. Consequently, we explored how to utilize the adaptability of SDN innovation to enhance the throughput of server farm systems. There are three fundamental focuses for a SDN-based server farm engineering. They are physical topology, forwarding flows as per bandwidth utilization and routing protocol. They are not autonomous, that is, the execution of one will be impacted by the decisions of others. 1.2 TOPOLOGY Bandwidth is turning into the bottleneck for adaptability of large scale data center topology networks. Existing answers for explaining this bottleneck basically utilize progressive systems of switches. With the costly, non-item switches at the highest point of the hierarches convention, one issue is that the port thickness of top of the line switches limits general group size. Furthermore, it 6 brings about high cost. Late methodologies, for example, Fat Tree and are Clos topologies that utilization different center changes to give full transmission capacity between any pair of hosts in the system. Those topologies are interconnected by a few layers of changes to evade imperfections in port densities which exists in business switches. Late research advocates the even development of server farm systems. These exploration introduce a server farm correspondence design that influences ware Ethernet changes to convey adaptable transmission capacity for extensive scale groups. The consequences of their tests demonstrate that this topology can convey versatile data transmission at fundamentally bring down expense than existing strategies. 1.3 ROUTING The technique we used for multi path routing is ECMP. There is another way of static VLANS also. Both of these ECMP and static VLANS properly utilize the benefit of multiple

13**equal cost path in a fat tree**

architecture. They perform load balancing and select a best path among all the

13**equal cost path in a fat tree**

topology. This helps to achieve multi-path routing in an efficient manner and flows are forwarded along the same path and order is maintained. The first step is to find the equal cost path in fat tree topology. Then flows are forwarded statically across multiple available paths. This static flow forwarding does not take into consideration efficient bandwidth utilization and packet size which results in collision and also effect on throughput. The biggest limitation of ECMP is that if, there two or more flow which happen to take the same path available there is possibility that the bandwidth utilization of core links might exceed the threshold. The problem with ECMP is dynamic flow selection so we suggest a use of centralized scheduler which will overcome the issue of ECMP in data center network 7 1.4 NECESSITY

1**OF DATA CENTERS The** primary prerequisite **of most** associations **is business** coherence; **if there is a** framework disturbance, **IT operations may**

get to be impeded which can affect accessibility of administrations to clients. To minimize any odds of interruption, accessibility of solid base is an unquestionable requirement. Moreover, data security is likewise vital and henceforth a server farm must offer a protected domain with controlled access to decrease breaks. A server farm in this way ought to guarantee its usefulness and uprightness. The Telecommunication Industry determines the necessities for telecom server farms including both undertaking server farms web server farms. It additionally indicates natural necessities for server farm particular gear. For powerful server farm operation, it is important to give a fundamental domain appropriate to establishment of hardware in an office. Telecom server farms require building pieces which are dull in nature to give incorporated hardware and building designing and simple development and versatility. A few server farms convey remote frameworks to get to and oversee server farm hardware, which are run utilizing computerized scripts to perform nonexclusive operations which don't require nearness of work force. Such server farms are worked without lighting and are known as "Light-out" server farms. This takes into account to find server farms in remote scantily populated regions, in this manner expanding vitality productivity, decreasing staff costs and evoking noxious assaults on base. Maturing server farms and quick IT development has provoked associations to exploit vitality and execution based efficiencies of current gear and perform server farm change. This procedure is an incorporated methodology comprising of a few ventures did all the while as against the customary serial server farm approach. Data Center Transformation are as per the following: 8 • Standardization and Consolidation: It involves supplanting matured gear with more up to date ones which give enhanced limit and execution, alongside institutionalization of systems administration and administration stages for simple sensibility. • Virtualization: It can be utilized to supplant servers in server farms and can be utilized to make desktops which are then facilitated in server farms and leased on a membership premise. It brings down capital and operation costs. • Automation: It helps fixing, setup, obligingness and discharge administration helps server farms run all the more productively. The necessity of server farms are characterized with the utilization of TIA Tiers. Level one recognizes an essential data center with non-excess dissemination way and a normal high accessibility estimation of 99.68 percent. Level two is a server room which serves repetitive limit foundation and a normal accessibility of 99.73 percent. Level 3 is an extensive data center with numerous information appropriation ways, all base being double fueled and having an accessibility of 99.97 percent. Level 4 must surpass all level 3 prerequisites and what's more ought to have all cooling framework as double fueled, flaw tolerant foundation with the capacity to store electrical force with a normal accessibility of 99.99 percent. 1.5 MODERN DATA CENTER DESIGN Data centers by and large continue running by significant organizations or government bolstered masters. Keep running of the server farms can have

1**from one** space **to a complete building. Most equipment are server-like mounted in rack** pantries. **Servers** change **in size from single units to** immense self-rulingly **standing** stockpiling **units which are** on occasion **as** extensive **as the racks.** Colossal server farms **even make use of**

transportation holders involving

1**1000's of servers. Instead of repairing** solitary **servers, the entire** holder **is**

supplanted in the midst of overhauls. A keep running of the Data Center contains a strong, all around fabricated building lodging stockpiling devices, servers, web accessibility and expansive cabling and frameworks organization equipment. It moreover includes cooling rigging and establishment to supply power, nearby robotized fire soaking structures. It is urgent to take fortifications once in a while to

1**ensure operability and high availability. The more** fundamental programming **and hardware, more** attempts **are** required **for**

security. Checking, astute extension measurement and organization of server farms' essential

1**systems can be** refined **through Infrastructure** organization. **It enables** consistent watching **and** directing **of all IT systems and** office base **using special** programming, **hardware and sensors. It** recognizes **and** murder danger **sources to ensure high** openness **of** benefits, furthermore **to** perceive hole **in** overabundance **of** establishment **and** give widely inclusive **data on power** usage **to measure** sufficiency. Correspondence **inside the datacenter** relies **on** upon **IP** tradition **based**

framework involving switches

1**and switches that** trade movement **between the internal servers to the outside world**

CHAPTER 2. FAT TREE TOPOLOGY 2.1 SELECTION OF THE TOPOLOGY Fat-Tree Topology is best suited and widely used at present in the Data Center Networks. Before deciding the the appropriate Topology to be used in the Data Centers. Let us learn about the different types of Topologies that can be used in the Data centers, their methodology of functioning and how they do into different networks. There are many topology structures from basic direct connections to the complex routing networks. Topology structures like Point-to-Point is the basic type of structural connections between the two direct hosts. As the number of networking devices in the network increase this starts getting complicated because we need to manage the the exact number of physical connections that of the number of devices in the network. Figure 1: DCN Architecture Secondly, we have Bus topology. It consists of a single main line bus which is connected to all the devices in the network and whole topology fails if the bus gets a lag or fails to function properly. We then have Star and ring Topology which is the smart device centric type of Topology. These topologies have a smart device like a router or a switch in between and are connected to the hosts. The difference between a start topology and a ring topology is that in the star the central device is the vital part, all hosts and the bus connection fail if the central device fails. Whereas in Ring topology, all the hosts are connected to the next one and they form a ring with their connection manner. This gives the point of failure to be the failure of any one host in the ring. However, with the complex growing networks it was very challenging to get a reliable and consistent topology deign. Tree Topology served the purpose later. This structure was first named as Hierarchical Topology. It is most widely used structure in the networks at present. It is actually evolved from the cumulative properties of the Star and Bus topology. Structural features of the star topology were developed. Instead of keeping one central device, all the layer 2 and layer 3 devices were given importance and they were the building blocks of the core layer. On the other hand, properties of bus topology were inherited and the connection format was built in a systematic hierarchical manner. Actually, the topology separates the network into numerous layers of network. For the most part in LANs, the blue print of the devices in the network is divided in three layers of networking devices. The bottom layer is the access-layer. It consists of the Edge Routers/Switches which are connected to the host beneath them. Edge devices are also known as T.O.R switches- Top of the rack switches Hosts are the initiators, serving them is the sole and foremost function of the topology. As the number of hosts increase or decreases the upper layers have to adjust themselves and vary according to the math on which the topology efficiency is built. The center layer is called 1 2 Aggregation Switch layer. This basically provides the functionality of the distribution layer, which fills in as the intermediate building block between Edge layer and Core layer. These devices are responsible for organizing the connections form the core layer and distributing them to the underlying Edge devices. The upper-most layer is the Core layer, and it the back bone of the network. It is the root of the tree and outlets all the nodes from it. All the Core devices are connected with the bus with all the respective underneath devices and this is the basic Tree structure. This was the best invention but failed if the root failed and whole topology went scrap after that until the root device is up again. This gave rise to the flawless solution of connecting a single tree structure with more than one root. Even if one root is down, the network is still reachable using the other root. The Fat-tree topology was invented by “Charles E Leiserson” from MIT. He proposed the most efficient and flawless network structure for the Data Center communication. The thickness of the branch is the term for the bandwidth available on that medium. High bandwidth links are known thicker and low bandwidth links are known as skinny links. Fat-tree Topology provides with the backward compatibility within the defined structure. Serves the best support for the Ethernet layer (Layer-2). It is cost effective and consumes very less power which is the basic need nowadays, with the ever growing networks. It’s built out of the low cost infrastructure and emits Low heat. The line speed can be user defined and the network proves to serve to the best according to whatever is defined. 2.2 FAT TREE ARCHITECTURE Knowing that Fat-tree works on constant bandwidth on all the bisection. Every layer still provides equal aggregate bandwidth. All the ports support constant speed on the hosts. In the case of uniformly distributed packets on all the paths that are available, we can notice that all the devices 1 3 will transmit at the same speed of line. The methodology and the mathematical calculations for making an efficiently working topology is as follows. The server racks to be interconnected in a fat tree topology (Three Layer topology). • Each of the

11**pod consists of (k/2) * 2 servers** and 2 **-layers of (k/2)** port **-k switches. • Each** of the **edge**

switches must connect

21**with (k/2)** server **and (k/2)** aggregate **switches. • Each** of the aggregate switches must connect with (k/2) edge and (k/2) core switch. • All of the (k/2) * 2 core switch has to respectively connect with k pods each. Figure 3:- Example of a Fat-Tree topology 2.3 NETWORK UTILIZATION The following analysis shows the Network Utilization if the Fat-tree topology that we have used in this research project. We can derive that the bisection bandwidth of fat-tree structural networks in ideal case is 1.536 Gb/s. It shows all the following aspects in the table below such as, Aggregate Bandwidth of the network, Percentage of bisection bandwidth for the tree in ideal case, two level table, classification of the flow and scheduling method of the flow. 1 4 Figure 4 :- Network Utilization table for Fat-tree Topology

25**2.4 MULTIPATH - ROUTING**

Definition – “Multipath routing is the routing technique of using multiple alternative paths through a network, which can yield a variety of benefits such as fault tolerance, increased bandwidth, or improved security. The multiple paths computed might be overlapped, edge-disjointed or node disjointed with each other.” Explanation - Methodology for using multiple different routing paths in a network to deliver the packets from a node to node. This technique actually considers best path and the successive best paths, for making the routing decisions. As a result, it gives an assortment of advantages such as better throughput (bandwidth), fault tolerance and enhanced security. All the different paths figured may be overlapping, edge-incoherent or hub disconnected within themselves. These are the three vital components needed to properly implement multipath routing algorithm are Multipath Calculation algorithm that derives the logic for available multi-paths. Secondly an algorithm based on Multipath Forwarding capabilities that makes sure about the chosen path and guarantees to deliver the amount of traffic to the destination allotted on the path. Finally, the algorithm should execute a proper end-to end protocol which can determine and decide the multiple paths to be used. The Important aspects to be considered while designing the multipath routing algorithm are to maintain the fault tolerance, the bandwidth availability and the load balancing in the designed network. To make full usage of available multiple paths for routing the packets in the network is commonly known as Concurrent Multipath Routing (CMR). This is achieved by making utilizing the resources by different carriers simultaneously, distributing the packet load all over the different available paths according to the resources available on each path, designing fast discovery system for all the path in case of failure. 2.5 “EQUAL COST MULTIPATH ROUTING PROTOCOL (ECMP)” Definition – “Equal-cost multi-path routing (ECMP) is a routing strategy where next-hop packet forwarding to a single destination can occur over multiple best paths which tie for top place in routing metric calculations. This Multi-path routing is used in conjunction with most routing protocols, because it is a per-hop decision limited to a single router. It substantially increases the bandwidth by load-balancing traffic over multiple paths.” Explanation - ECMP proves to be an essential load balancing routing algorithm implemented in the networks. Despite using only single best path to route the packets to the destination, it uses all the different best paths that are available. ECMP figure outs the possible best path and checks their available resources for finalizing the routing decisions and when these multiple best paths are obtained, it sends out the traffic with available bandwidth. Figure 5 :- ECMP based routing in Data Centers Network The working of ECMP varies based on different layers it is implemented on. ECMP is functioned to work differently on Layer 2 and it probably functions different on Layer 3. We, in this research have used Layer 2 functioning of ECMP and using open Vswitch. We implement ECMP on our topology constructed using 2 core switches, 3 pairs of Aggregate switches (total 6 aggregate switches) and same 3 pairs of Edge switches or Top Of the Rack switches (6 in total). Basically we break the arbitrary packet traffic and distribute it over a path and making sure that the respective single flow remain on that particular path for whole of its lifetime. This in turn helps to choose the hashing algorithm to be used and selecting the header information. Hence, we organize the in order delivery of the packet traffic at the destination. This information in header which is encapsulated on this sort of packets consists of the

> 9**MAC address** of **the source, Mac address** of **the destination, IP address** of **the source, IP address** of **the destination,** DSCP, Source **port,**

Destination port and the next hop. Load balancing for Layer 3 is achieved by separating the information load over multiple links. The probable measure we have to take care of here is by making sure that the individual TCP connection is established and maintained over a single link for achieving best performance in the built network. Hence, Layer 3 distribution of load must include

> 16**Source IP, Destination IP, Source** TCP **port, Destination** TCP **port and** lastly **the** protocol number. Forwarding **in**

ECMP may cause jitter if the paths are chosen randomly and packets are sent over the path. Hence, we try to send the packets destined for a fix source-destination pair on the same path till the ongoing session is executed completely. CHAPTER 3 SOFTWARE DEFINED NETWORK (SDN) 3.1 TERMINOLOGY IN SDN Definition – “Software defined networking (SDN) is an approach to computer networking that allows network administrators to manage network services through abstraction of higher-level functionality. This is done by decoupling the system that makes decisions about where traffic is sent (the control plane) from the underlying systems that forward traffic to the selected destination (the data plane).” Figure 6 :- Basic SDN structure with separate Data and Control plane Explanation- To cope up with the ever increasing Inter-Network and the complexity, it is important to maintain the performance and efficiency of the network. SDN proves out to be the solution which was the epitome of success in providing the ultimate network performance. Basically, SDN is built over three pillars of innovativeness: • The data plane and control plane separation. • The control plane Programmability. • Application programming interfaces (APIs) standardization. Normally for computer networks, packet flow is taken care of by the specifically devoted devices like routers and switches and other networking devices working together in a networks based and operated by different ISPs (Internet Service Providers). The functional characters and the rules which drives all these sort of devices can be organized and mapped in three basic sub-categories known as planes. 3.1.1 The Control Plane This is designed to carry the signaling traffic and maintain open flow table. It performs tasks like routing setup, deciding the routing path from source to destination, QOS, etc. Management of the system and configuration of the same is also an important function of the control plane. 3.1.2 The Data Plane This depends on the information derived by the control plane. According to the routes derived by control plane, the data plane switches data in the form of packets from source to destination over a selected path. 3.1.3 The Management Plane Maintains the policy in the network and admins the network. It is like a sub-part of the control plane because even this one has to take care about the performance of the network. Figure 7 :- OpenFlow Protocol and Data Layer Interaction The open flow model and design of the control plane allows to dynamically access all the resources and admin control of the network. SDN was actually invented to overcome the flaws in the old school routing algorithms like smartness of the switches, which is visible in every switch and each of them is equipped to function on a specific routing-algorithm. For example, we take distancevector algorithm which will make a neighborhood routing table. Be that as it may, those switches are normally expensive. Moreover, since the forwarding table of every switch is built in a disseminated way, it gets difficult to troubleshoot and derive the best-path for routing. Figure 8 :- Structure and planning of a Software Defined Network A rigid firmware serves as the platform for implementation of all the three planes. Firmware is the combination of hardware and the software. SDN separates

20**the data plane** with **the control plane,** this control **plane is**

actually a centralized software in the network that works as a brain of the network. The vital role of the control plane is to build and maintain the routing table for each source-destination pair. Whereas, the vital role of the Data Plane is to implement that routing table and route the packet and maintain the flow over the link. The most beautiful feature of the Software Defined Network is the programmability of the control plane. 3.2 NEED FOR CENTRALIZED CONTROL IN DATA CENTERS The new data centers require frameworks that give more control, adaptability and versatility than at any other time. You require more than server administration. We require a framework that decreases our Mean-Time-To-Repair (MTTR) while utilizing the current T assets you as of now have. In the event that we have to oversee server farms and branch workplaces at whatever time from anyplace, there is an answer. An incorporated administration approach without-of-band ability can expand server farm control, permit us to accomplish more with less, and give remote access from anyplace on the planet. Today's backing minded system managers need day and night access to servers 365 days a year - both at the rack and in remote areas. Luckily, with brought together administration (counting both in-band and out-of-band availability), data centers has never been so close. The use of central controller to manage flows gives us IP availability and security models in your system. Unified administration programming gives you a chance to control a machine that can be associated with servers, serial gadgets, even power dissemination units and ecological checking. With brought together administration, we can control different servers and system gadgets regardless of where they are found. Virtual media is a later expansion. It expands head effectiveness by permitting us to lead a vast host of errands remotely, for example, record exchanges. Whether we require access at the rack, in the Network Operating Center (NOC), or from our portable workstation at home, there are arrangements accessible that will give uncommon, secure get to and help you demonstrate a brisk profit for your venture. 3.3 POX CONTROLLER Definition - “POX is an open source development platform for Python-based software-defined networking (SDN) control applications, such as OpenFlow SDN controllers. It has a simple OpenFlow Data plane implementation. It can be tested using OF Test framework. To connect data ports of POX switch implementation to OF Test Linux virtual Ethernet Interfaces (VETH) are used. And a default controller port 6633 is used for Openflow connection.” Explanation - POX controller is made of main components like additional python programs that are invoked as soon as the POX controller is started in the terminal from command line. All these components implement the functionality of the network in SDN. Running POX controller is similar to running the openflow SDN controller on the network topology. POX controller works on mininet. Mininet is a network emulator that is utilized for emulating Software Defined Network. All the servers and the switches are to be downloaded in the mini-net in the form of the in-built packages. Mini-net has a openflow controller and software in-built in it. Mini-net can be downloaded on all the Operating systems like Ubuntu, Linux/Unix, etc. Steps to emulate POX controller with a user built topology in mininet. • Step 1 – In the terminal 1, we follow this step and run out topology to start the network. we start with Mininet on our Operating System. Then creating a topology for SDN using any one control software like open Day-Light or Floodlight. Commands- a) sudo mn - This command starts the network. b) sudo python our_topology.py – This compiles and starts the network topology built by the user. Hence the topology is emulated in one terminal and now we know that this a SDN topology so we have to give a controller to the topology so that it can learn the whole network and start building the routing tables to finalize the best paths and start communicating. • Step 2 - In second terminal, we have to assign the controller to the emulated topology and make the SDN more efficient resulting for a better performance. We open terminal 2 and run the POX controller installed in Mininet. Commands – a) cd pox – To locate the appropriate directory where pox controller is stored. b) ./pox.py forwarding.L2_learning – This initiates the POX controller. c) sudo python our_topology.py – This compiles and starts the network topology built by the user. Hence the topology is emulated in one terminal and now we know that the controller is initiated. d) Sudo mn --controller = remote, ip=10.0.0.12, port=6633 – This indicates Mininet that a remote controller is active and this is to be used on our topology. As soon as it starts we see that it is listening for switch connection on the port number 6633 (this can be any port number above 1024). Now the network is on and it is controlled by POX controller which means that all the access points and forwarding device will learn about the neighbors and their neighboring connections. Eventually all the devices and hosts in the network will learn about whole topology and figure out the best paths, successive best paths with the help of individually maintained forwarding table. This network is now ready to route the traffic to and fro in the topology. The connectivity and reachability in the network can be checked by ping command. Command pingall is used to ping all the hosts or the nodes connected in the topology and checks the network layer status of the host as well as the reachability to the host. We can ping the nodes individually as well using the command – h1 ping h2. What this command does is, it pings the host h2 from h1 and hence the reachability between host h1 and h2 is tested. If you capture the packets using any network 2 4 monitoring toll like Wireshark or GNS3 and on close observation, we can notice the path that these ping packets follow. This derived path is the best path that the topology feels between host h1 and host h2. When we stop the POX controller we still notice the ping packets on the command prompt and this is because these ping packets are cached in the memory and they will go on until the hard timer goes out. 3.4 OPEN VSWITCH Definition - “Open vSwitch is a production quality, multilayer virtual switch designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols. In addition, it is designed to support distribution across multiple physical servers similar to VMware's virtual Network distributed vswitch” Explanation - Open vSwitch is basically

18**an open source implementation of the** switch **that** is based **on**

OpenFlow protocol and it is to be used as virtual switch in the virtual networks. OpenFlow particulars are focused at Layer 2 and layer 3 usefulness. The most recent networking movement is to empower a switch with Layer 4 to Layer 7 administrations like proxies, Load balancers, Firewalls, IPSec and so on Which makes the center boxes repetitive in the designed network. Hence, we propose an approach to broaden the most normally used Open vSwitch to Layer 4 to Layer 7 administration mindful OpenFlow switch. 3.5 IMPLEMENTING ECMP IN SDN There can be such scenarios where multiple best paths are derived with same metric cost for obtaining the MAC address in particular. In the implementation of ECMP in SDN, the controller defines and implements the rules in all the openflow switches in the case of multiple number of best paths identified in the forwarding tables to use all the paths instead of just one best path for that particular MAC address. This is where we need to educate the controller with the rules that are of conjunction on the destination MAC address. POX controller, Flood light controller or Open Daylight controller can be used for this purpose and the methodology for educating the controller and making them understand the rules are different for all of them. Other measure which needs to be taken care of is that all the switches in the underlying data plane should be supporting openflow protocol and such switches are normally known as open vSwitches. Details for implementation of ECMP in SDN. • Each switch in the network is represented as GraphNode object. • All the switches maintain the forwarding table with the entries of minimum cost paths for deriving the MAC address of the destination. Which needs Dijkstra's algorithm to be slightly modified for maintaining this sort of forwarding tables. As soon as we derive a new best path (with low cost metric than the last best path), the last forwarding table which was maintained by the switch is discarded. The switch builds and maintains a new forwarding table and starts with the first entry with the best path that it just learned. If we now discover any new path with same metric and cost, will be kept it in the table and this makes the table with multiple best paths. This table of paths with all the collection of rules in all switches are hard-state. This means that they are cleared explicitly and these do not timeout. • As soon as a new host or switching device is added or subtracted from the topology, all the rules which were defined in the network prior will be discarded and all the rules are recomputed in all the switches. These rules are not yet installed in the whole topology. Only the nodes which are inspected by the controller are installed as of now. In the next step we show the exact procedure of how these rules are installed in each and every Graph Node inside the topology. • Firstly, when any packet reaches the switch and realizes that there are no rules defined on the node as of now. This packet if forwarded to the controller as an initial step. Secondly, the controller then checks for the number of best paths that the switch has for any particular destined MAC address. If there are more than one best paths available for the destination MAC address, then the controller selects the path with the least used port. This rules is then matched at the destination and the source MAC address. This does not mean that all the packets with different destination MAC address will be routed from this path but it surely means that the packets for this same pair of source and destination MAC address is to be routed from this path. As a final step to define rules on the entire topology, the controller has to forward the packets to next hops on behalf of this considered switch. • All the nodes have to keep track of counts for all of their ports which are to be used in deriving the rules. To balance the load evenly on entire network, it is important that the controller decides the least used port and allots the traffic to that port. This is most useful when we have a highly busy source which communicates with many destination MAC addresses. To avoid the conditions like using the path between two hosts supposedly, h1 and h2 which is better but we still have a better path which is less used and gives a good performance between these two hosts. Hence the basic rule in the controller to be defined must be only to choose the best path between two hosts and allot the traffic on that link. • The final and most important thing to be taken care in entire process is that the rules never timeout. The controller will clear all the old rules defined in the network and it will re- define all the rules again considering a single switch at a time. CHAPTER 4 GOAL OF PROJECT AND TOOLS 4.1 GOAL OF PROJECT

22**The basic idea of our** project **is to** analyze multi **path**

in data center by running the ECMP protocol. To achieve these goals, we conducted two experiments and depending upon the results we evaluated the outcome. The fist experiment involves analysis of ECMP in data center topology. In this scenario we run ECMP Algorithm in the data center network and observe the behavior of ECMP in fat tree topology. ECMP is a multi-path routing protocol. If there are two or more paths with equal cost path it performs load balancing on all these equal cost path. In the second experiment we perform the same experiment but in an SDN based environment. SDN here provides the use of central controller which manages the flow forwarding based on the bandwidth utilization of core links. The controller here acts as the brain of the network. The controller can perform dynamic scheduling based on the bandwidth requirement of the core links. The controller used here is POX controller. Based on the two experiment we evaluate the results depending on the bandwidth requirements. We use IPERF to generate traffic on the links and based on the bandwidth utilization evaluate the results. 4.2 TOOLS MININET - Mininet can be said as networks emulator, or maybe all the more definitely a system copying arrangement framework.

6**It runs an accumulation of end-hosts, switches,** switches, **and** interfaces **on a solitary Linux** bit. **It utilizes lightweight virtualization to make a** solitary framework **resemble a complete system, running the same part, framework, and client code.**

2**A Mininet host acts** basically **like a** certified **machine; you can ssh into it** (if **you start up sshd and** augmentation **the framework to your host) and run subjective activities (checking anything that is presented on the basic Linux** structure.) **The undertakings you run can send** packages through **what** seems **like** a veritable **Ethernet interface, with a given association** speed **and** delay. Packs **get** readied **by what takes after a** honest to goodness **Ethernet switch, switch, or** middlebox, **with a given measure of queueing.** Right **when two** activities, **like an iperf client and server, pass on through Mininet, the** planned **execution should** coordinate **that of two (slower)** nearby **machines. Essentially, Mininet's virtual hosts, switches, associations, and controllers are the** real deal – **they are** simply **made using programming instead of gear – and** by and large their behavior resemble

7**discrete hardware** segments. **It is** regularly **possible to** make **a Mininet** framework **that resembles a** gear framework, **or a hardware** framework **that** takes after **a Mininet** mastermind, **and to run the same** matched **code and applications on either** organize. **4.**

2.1 SPANNING TREE PROTOCOL (STP) Definition – "STP convention makes a loop free topology amongst switches, furthermore keeps TV issues that outcome from circled systems. The protocol makes a tree amongst a system of interlinked switches and makes the briefest way from the root switch to all the switches in the system, in this manner making the best way for availability to every host." Explanation - The basic idea of using Spanning tree protocol is to break loops. When there are two wires between two switches there is a possibility of loop creation. Loop creation is identified by broadcast storm, continuous flickering of LEDs and MAC data base instability. It blocks a port to break the loop SPT basically elects root bridge to avoid loops. One of all the switches is elected as Root Bridge and all others are elected as Non root Bridge. The root bridge is elected based on BPDU exchange. BPDUs are bridge protocol data units. When there are two or more BPDUs heard on any port there is loop creation. BPDUs contain information about the cost to reach thee Root Bridge. The election of Root Bridge happens

24**based on bridge ID which** is **priority + MAC address.**

The lowest priority

10**is selected as the root bridge** and if priority **is** same than **the**

election goes on MAC address. Lower MAC ID is more preferable. After the root bridge election, he advertises the cost to reach the root bridge as 0 to other switches in BPDUs. In the same other switches learn the path to reach the root bridge.

14**If there are two** path **with same cost** than **the** path **with the lowest** port **ID is** preferred. The **port**

directly connected to the root bridge are root port. The port of Root Bridge are designated ports. Designated ports are the forwarding ports of the root bridge and root ports are the ports from where cost to reach the root bridge is the lowest. Similarly, the comparing the lowest MAC of two switches these ports are decided as designed and root ports. Switch which has the lowest MAC his port is selected as designated and other is selected as root port. After this there remains a port whose cost to reach the root bridge is the highest is blocked and looping is avoided. It is referred as non-designated port. This port is not shutdown it is receiving BPDUs but not replying as it is just blocked port. In this way loops are avoided. The only drawback of STP is it takes 50 seconds for port to turn up. There are 4 states learning, listening, forwarding and blocking. 4.2.2 OPEN VSWITCH – OVS - COMMANDS The principle segments of this conveyance

3**are: 'ovs-vswitchd' - a daemon that** executes **the switch,** alongside **a** buddy **Linux** bit **module for** stream **based** exchanging. **'ovsdb-server' - a lightweight database server that ovs-vswitchd** questions **to** get **its** setup. **'ovs-dpctl' - a** device **for** arranging **the switch** bit **module.**

3**'ovs-vsctl'** - an **utility for** questioning **and** redesigning **the** arrangement **of ovs-vswitchd. 'ovs-appctl'** - an **utility that sends** orders **to running Open vSwitch daemons. Open vSwitch** additionally gives **some** apparatuses: **'ovs-ofctl'** - an **utility for** questioning **and controlling OpenFlow switches and controllers. 'ovs-pki'** - an **utility for** making **and** dealing with **the** general population **key** framework **for OpenFlow switches. 'ovs-testcontroller' - a** straightforward **OpenFlow controller that** might **be** helpful **for testing.**

3**A patch to tcpdump that** empowers **it to parse OpenFlow messages.** 4 **.2.** 3 MININET CONTROLLER **The**

controller in mini-net is based on open flow protocol. There are no stream areas. For every incoming packet it analyzes the flows and gathers data such as Source and Destination MAC, port number and VALN information. Considering

1**this information, it** makes **the** stream **table** segments **until it has** gotten **information from all the switches in the**

framework. 4.3 MININET TOPOLOGY CREATION Figure 9 :- creation of Topology in Mininet (1) Figure 10 :- creation of Topology in Mininet (2) Figure 11 :- creation of Topology in Mininet (3) Figure 12 :- creation of Topology in Mininet (4) 4.4 WORKING OF FAT –TREE ARCHITECTURE CODE 4.4.1 CODE FLOW OF FIRST EXPERIMENT AND SECOND EXPERIMENT After the execution of the program, the interpreter checks the main function and checks the users. If, the condition of user being a root user is true then the function to create the topology is called. At the beginning of the program a variable class is assigned in the create topology function. • A class Multi Topo is created in which list is created for core, aggregate and edge layer switches and hots. The list is defined in the following way in the function self.<component List>( ). • The arguments to the functions are passed as number of switches and host and are called which than creates the number of switches and host as defined. The values of number of host and switches to be created are passes as self.<component>( ) . • The switches and the host created on calling the function gets appended in the host and switch list created earlier. The above task is achieved by creating a function. self.addSwitch() and self.addHost() are used as the key componets to create the switch and host and append them to list. • Then links are created between switches and host by using self.addlink(). • net.addController() Function is used to create a SDN controller and IP and port number are assigned to the controller. The IP address assigned is 127.0.0.1 and port number is 6633. • net.start() is used to start the controller and activate the core, edge and aggregate layers. • Then we start the spanning tree by giving the command “ovs-vsctl set Bridge %s stp_enable=true (%s is the name of the switch)”. • To start the command line in the mini-net after creating the topology we used CLI() which provides an interface to talk to the nodes. 4.4.2 THE FLOW CHART FOR THE FIRST EXPERIMENT In the first experiment we first create a fat tee topology in mininet. The topology consists of three layers core layer at the top followed by the aggregate and edge layer. The edge and core layers are connected to each other. All the host in the topology are connected to the edge layer. In our scenario each edge layer is connected to 2 host. Then we add links between the switches and host. The links are created

15**between core and** aggregate **switches and** aggregate **switches**

are

15**connected to edge switches. Each edge** switches than connect **to**

two hosts. After defining links in the topology we run STP. Then to perform multi-path routing we run the ECMP algorithm. ECMP defines the path each host takes to reach other host. Each path passes through the core switch. To test the connectivity between switches and the host we run ping and to generate the traffic on links and test the bandwidth between the links we use IPerf. Figure 13 :- Flow Chart for Experiment 1 4.4.3 THE FLOW CHART FOR THE SECOND EXPERIMENT In the second experiment the topology remains the same. The same fat tree topology created is used for the second the experiment. The only difference is that the core switches are connected to a central controller. The central controller here manages the flows based on bandwidth utilization of core links. The controller we used here is POX controller. The controller here dynamically manages flows. The SDN based environment solves the drawback of normal fat tree topology. When the bandwidth utilization of the core links exceeds the threshold the controller here dynamically adjust the flows. It directs the exceeded the flows to select a path which is used less. Thus the use of controller overcomes the drawback of ECMP in fat tree topology. Figure 14 :- Flow Chart for Experiment 2 CHAPTER 5 PERFORMANCE EVALUATION RESULTS 5.1 SIMULATION AND EVALUATION Figure 15 :- Final topology for DCN To compare the performance of our two experiments we select host destination pairs to different edge switches and measure the bandwidth utilization. For every host there are four equal costs paths as shown in the figure there are four equal cost path between s1 and s9. To perform the evaluation results we transmit traffic between pair's h1-h9, h1-h10, h2-h9, h2-h10, h3-h9, h3-h10, h4-h9 and h4-h10. To compare the performance of different experiment we calculate the bandwidth utilization at the core switches. To generate the traffic on this flows we use Iperf . Some host are set as server and clients and traffic is sent between. The basic idea is to calculate the bandwidth utilization on core links evaluate the performance. 5.2 TRAFFIC DESIGN AND MEASUREMENT 5.2.1 IPERF - Iperf traffic generation tool IPERF is an Internet Performance working group which is generally used to create and a lot different types of traffic in the network topology. We are using IPERF to generate UDP traffic as to start with. The main reason for using IPERF is to actively measure the maximum achieved bandwidth on the Data Centre Network which we have built. IPERF also helps in the tuning of different sort of parameters relevant to that of Timing, Buffer handling and protocols. The most useful results that IPERF provides is that every test reports the parameters like loss, Bandwidth, etc. Some of the IPERF versions for generating traffic are backwards compatible and some are not. For TCP and SCTP, IPERF results consists of the the supporting TCP window size through the buffers of sockets. Secondly the results consist of measured Bandwidth and returns the size of MSS/MTU. Whereas, UDP packets are created and the streams are created out of those packets of particular bandwidth and then we can measure different aspects like Packet loss, Delay jitter and whether it is capable of Multicasting. IPERF traffic allows the server and the client to have multiple simultaneously ongoing connections. It also supports multiprocessing in the server that is, it can handle multiple clients one after one and the server never stop until is commanded to do so. How do we define TCP or UDP traffic in IPERF – There are several options available for playing with the IPERF traffic generation. Some of them which are used in this research project are explained below. • iperf –s [options] - Runs the TCP traffic in Server mode with fix sized streams after handshake is done. • iperf –c server [options] - Runs the TCP traffic in Client mode, connecting IPERF server • iperf –u –s [options] - Runs UDP packet streams on the Server mode. • Iperf –u –c server [options] - Runs the UDP traffic in Client mode, connecting IPERF server which is activated on host.

1**A typical Iperf Command is as follows**• “Mininet> iperf <Source Host Name> <Destination Host Name>• *** Iperf: testing TCP bandwidth between <Source Host Name> and <Destination Host Name>• *** Results: ['1.69 Gbits/sec', '1.70 Gbits/sec']” **Figure** 16 :- **Results**

of IPERF 5.2.2 PINGALL AND PINGALLFULL Pingall command is generally used to ping all the hosts, switches and other network devices connected in the network to check the connectivity between all of them. There might be the case sometime that some host are reachable from some hosts and others just have some problem finding the path or less bandwidth or any other reason from millions of them. We want to ignore the hustle of trial and error method to ping every host from host 1 and again all the other hosts from host 2 and so on, for all the number of hosts. pingall is the best solution for such kind of situations and it can be encroached on the whole topology at once. The result for pingall command shows many information if you look at it closely and you will find, If the pingall command is successful. • We obtain a list of the following analysis as Host_name, IP_address, number of messages sent, number of messages received back from the host, Round Trip Time (RTT) in milliseconds, etc. If the pingall command is unsuccessful. • When this command did not run successfully that means one or more of the nodes in the network is not reachable and the output for this command on the terminal will display an error message. All the error messages must be about different failure points in the network. o 100 % packet loss – If we see this message on prompt then it can mean, ‣ Host in the network is down. ‣ Host is active but the packets are denied at the host. ‣ The network itself is down. ‣ Probably, wrong host was pinged. o Packets are rejected – this says that the packets are reached ta the host and are also read but they are rejected at the host. o Packets did not reach the host – There can be a problem with the connecting link, hence the packets are sent out proper and the receiving host is all set for accepting it but the network fails to deliver the packets at the host. Output for both PINGALL and PINGALLFULL Mininet>pingall Figure 17 :- pingall output on mininet cli Mininet> pingallfull Figure 18 :- pingallfull output on Mininet cli 5.2.3 TRACEROUTE Tracerouteis the direct command and does not requireany optionsor parameters. It can be definedas a “computer network diagnostic tool for displaying the route (path) and measuring transit delays of packets across an Internet Protocol (IP) network. The history of the route is recorded as the round-trip times of the packets received from each successive host (remote node) in the route (path); the sum of the mean times in each hop is a measure of the total time spent to establish the connection. traceroute proceeds unless all (three) sent packets are lost more than twice, then the connection is lost and the route cannot be evaluated. Ping, on the other hand, only computes the final round-trip times from the destination point.” How do traceroute work – It sends the UDP packets in sequence by default to the destination host. But we can also alter it with TCP or ICMP Echo Request. This scenario is for Linux but it is a little different in windows, traceroute uses ICMP Echo Request preferably in place of UDP. But again, you can manually assign with TCP or UDP. The metric for traceroute is hop limit which is also known as

10**time-to live (TTL) value** which shows **the** behavior **of the**

intermediate routers between host 1and host 2. Every time the traceroute packet arrives at the access point in the network, it will reduce the TTL value by one and route it forwardtowards the destination. In this case, as soon as the TTL value for any packet

23**reaches zero** that **packet is discarded and** the **error message**

is sent back to the same route from which it was coming.Common TTL value in Windows is 128 and in Unix-based OS, it is 64. Usually to start with the packet forwarding, any OS starts with the TTL value as 1 and gradually increasing it to the respective value according to the OS. These packets are dropped at first router in the starting. Second packet will be dropped at second router. This is how we find out the TTL between the two hosts and the whole route between them is traced. “Mininet> Xterm <node name> Xterm # traceroute <destination IP>”

1**Figure** 19 :- **Initiating** traceroute from **Host 0**

5.3 ALGORITHM EVALUATION To evaluate the results, we run two experiments in data center networks based on different environment and compare the results based on bandwidth utilization of core links. In the First experiment ECMP is used in a fat tree topology and results are evaluated and in the second experiment SDN based environment is created and results of both the experiment are compared. Traffic is passed on the flows in host destination pairs using Iperf is used to generate the traffic and bandwidth is evaluated on the links. In the second experiment a threshold is set and when the traffic exceed the threshold controller dynamically selects the flows. 5.3.1 FLOW CHART OF TWO EXPERIMENT Figure 20 :- Flowchart for both the Experiments 5.3.2 FIRST EXPERIMENT The possible bandwidth utilization is calculated as shown in TABLE 1. To evaluate the results we calculate the bandwidth utilization of eight flows. The table and graph shows the bandwidth utilization of flows in fat tree topology when ECMP is used. DATA FLOW 4001 - 4001 - 4002 - 4002 - 4003 - 4003 - 4004 - 4004- 4009 4010 4009 4010 4009 4010 4009 4010 BANDWIDTH 0.4 0.2 0.4 0.2 0.4 0.1 0.4 0.1 UTILIZATOIN ECMP bandwidth utilization 0.45 0.4 0.4 0.4 0.4 0.4 0.35 0.3 0.25 0.2 0.2 0.2 0.15 0.1 0.1 0.1 0.05 0 4001 - 4009 4001 - 4010 4002 - 4009 4002 - 4010 4003 - 4009 4003 - 4010 4004 - 4009 4004 - 4010 Bandwidth Utilization Figure 21 :- Table for possible bandwidth utilization of each access flow. 5.3.2 SECOND EXPERIMENT DATA FLOW DATA FLOW BANDWIDT UTILIZAION 4001 - 2001 4002 – 2002 0.4 2001 – 1001 2002 – 1001 0.4 1001 – 2005 1001 – 2005 0.4 + 0.4 = 0.8 2005 - 4009 2005 - 4010 0.4 Figure 22 :- Table for possible bandwidth utilization in SDN based ECMP. From the above graph and table it is concluded that the two flows 4001 - 4009 and 4002 – 4010 are forwarded through the same core switch 1001. So from 1001 core switch both the path are forwarded through the same path from switch 1001 – 2005. So the possibility is that bandwidth utilization of that core link from 1001 – 2005 exceeds the set threshold. The controller than activates and directs the flow to less utilized path. 5.3.3 FLOW CHART OF SECOND EXPERIMENT The flow above states that first a fat tree topology is created in mini-net and ECMP algorithm is started. Flows are forwarded as per ECMP and bandwidth on these flows is recorded using the iperf tool. When the mini-net topology is executed at the same time on another terminal a pox controller is started. The results are evaluated based on the bandwidth utilization of core links. When threshold set exceeds the controller is activated and direct the flows to less utilized paths. If not than normal flow forwarding continues. Figure 23 :- Flowchart for second Experiment (SDN based ECMP). CHAPTER 6 CONCLUSION CONCLUSION - The limitations of data center topology are overcome by fat tree with multipath routing with increased bandwidth and fault tolerance. To efficiently utilize bandwidth and traffic load we suggest a multi-path routing scheme with efficient load balancing capabilities. As a practical solution we suggest a use of centralized scheduler or a controller which manages the network information and forwards flows based on bandwidth utilization of core links. The controller dynamically select flows and makes optimized load balancing decisions when adding new flows to the network. With the use of ECMP in data center networks throughput of topology can be increased but the problem with ECMP is lack of flexibility and dynamic scheduling when selecting flows on equal cost paths. With the increase in the development of SDN we propose a solution which will outperform the issues of ECMP. The work proposed shows that the central controller will dynamically adjust flows when the bandwidth utilization of core links exceeds the threshold which increases the throughput and network performance. This also proves the fact that SDN based environment is suitable for data center networks. Thus it proves that with help of centralized controller

17**it is possible to** improve **the** throughput **of data center networks.**

Experiment results show that our proposed work with the used SDN controller consistently outperforms ECMP in fat tree topology. REFERENCES 1. Open Networking Organization, https://www.opennetworking.org 2. An Introduction to Software Defined Networking, https://www.youtube.com/watch?v=H_3Lk6XbWw0&spfreload=10 3. Are We Ready for SDN? Implementation Challenges for Software-Defined Networks by Sakir Sezer, Sandra Scott-Hayward, and PushPinder Kaur Chouhan, CSIT, Queen's University Belfast Barbara Fraser and David Lake, Cisco Systems Jim Finnegan and Niel Viljoen, Netronome Marc Miller and Navneet Rao, Tabula http://ieeexplore.ieee.org.libaccess.sjlibrary.org/stamp/stamp.jsp?tp=&arnumber=6553676 4. Are We Ready for SDN? Implementation Challenges for Software-Defined Networks by Sakir Sezer, Sandra Scott-Hayward, and PushPinder Kaur Chouhan, CSIT, Queen's University Belfast Barbara Fraser and David Lake, Cisco Systems Jim Finnegan and Niel Viljoen, Netronome Marc Miller and Navneet Rao, Tabula http://ieeexplore.ieee.org.libaccess.sjlibrary.org/stamp/stamp.jsp?tp=&arnumber=6553676 5. Software Defined Network – Architectures: 2014 International Conference on Parallel, Distributed and Grid Computing; Nishtha Department of Computer ScienceHimachal Pradesh University, ShimlaShimla, India-171005, http://ieeexplore.ieee.org.libaccess.sjlibrary.org/stamp/stamp.jsp?tp=&arnumber=7030788 6. Cloud Computing Impact on Data Centers, http://www.techrepublic.com/blog/the- enterprisecloud/how-cloud-computing-will-impact-the-on-premise-data-center/ 7. Cloud Computing Wikipedia, https://en.wikipedia.org/wiki/Cloud_computing 8. Data Center Wikipedia, https://en.wikipedia.org/wiki/Data_center 9. 5 Reasons why a modern data center is required for digital world, http://www.gartner.com/newsroom/id/3029231 10. Do You need a Data Center, http://www.emersonnetworkpower.com/documentation/enus/solutions/ciotopics/documents/execut ive briefs/executivebrief_do you need a new dat a center_fa.pdf 11. Cloud Computing and Evolution of Data Centers, http://www.techrepublic.com/blog/theenterprise-cloud/cloud-computing-and-the-evolution-of-the-data-center/ 12. Cloud Versus Data Centers, http://www.businessnewsdaily.com/4982-cloud-vs-data-center.html 13. Scale out networking in the data Center, http://www.computer.org/csdl/mags/mi/2010/04/mmi2010040029-abs.html 14. Camcube, Rethinking the Data Center Cluster, http://www.cs.vu.nl/~kielmann/hpdc-trends- 2012/costa.pdf 15. Multi-tiered Network Architecture, https://www.digitalspyders.com/us/network-architecture.html 16. Symbiotic Routing in Future Data Centers, http://david.choffnes.com/classes/cs4700sp15/papers/sigcomm10-camcube.pdf 17. A Cost Comparison of Data Center Network Architectures, http://homes.cs.washington.edu/~arvind/papers/datacenter_comparison.pdf 18. A Comparative Analysis of Data Center Network Architectures, http://www.seas.gwu.edu/~guruv/iccngn14.pdf 19. Flattened Butterfly: A Cost Efficient Topology for High-Radix Networks, http://www.cs.berkeley.edu/~kubitron/cs258/handouts/papers/ISCA_FBFLY.pdf 20. Camcube: A Novel data Center, http://trilogyproject.org/fileadmin/publications/Presentations/2010_07_02_Multipath_transport/ro wstoncamcube.pdf 21. OpenFlow Table Patterns, https://www.opennetworking.org/images/stories/downloads/sdnresources/onf-specifications/openflow/OpenFlow Table Type Patterns v1.0.pdf 22. OpenFlow Switch Specification, https://www.opennetworking.org/images/stories/downloads/sdn-resources/onfspecifications/openflow/openflow-spec-v1.4.0.pdf 23. OpenFlow Wikipedia, https://openflow.stanford.edu/display/ONL/POX+Wiki 24. Chapter 4: SDN Controller 'Software Defined Networks An Authoritative Review of Network Programmability Technologies" by Thomas D. Nadeau, Ken Gray. 25. L3 Learning in POX Controller, https://haryachyy.wordpress.com/2014/06/03/learningpoxopenflow-controller-imitating-l3/ 26. A Cost Comparison of Data Center Network Architectures, http://homes.cs.washington.edu/~arvind/papers/datacenter_comparison.pdf 27. Cisco-Fundamentals of Spanning Tree Protocol, http://www.ctwebfactory.com/ccna/html/b3.html 28. Fat-Tree Architecture Wikipedia, https://en.wikipedia.org/wiki/Fat_tree 29. Mininet Simulation, http://www.brianlinkletter.com/mininet-test-drive/ 30. Spanning Tree Protocol Wikipedia, https://en.wikipedia.org/wiki/Spanning_Tree_Protocol 31. Software Defined Networks (EWSDN), 2013 Second European Workshop on, Issue Date: 10-11 Oct. 2013, Written by: Teixeira, J.; Antichi, G.; Adami, D.; Del Chiaro, A.; Giordano, S.; Santos, A. 32. M. Al-Fares, A. Loukissas, and A. Vahdat, ``A scalable, commodity data center network architecture,'' in Proc. ACM SIGCOMM, 2008, pp. 63_74. 33. Hailong Zhang School of Information Engineering, Communication University of China, Beijing, China Xiao Guo ; Jinyao Yan ; Bo Liu ; Qianjun Shuai - SDN Based ECMP Algorithm for data center networks. 34. Yi-Chi Lei ,Kuochen Wang and Yi- Huai - Hsu ,Department of Computer Science, National Chiao Tung University, “Multipath Routing in SDN based Datacenter “, 2015 35. Software Defined Networks (EWSDN), 2013 Second European Workshop on, Issue Date: 10-11 Oct. 2013, Written by: Teixeira, J.Antichi, G.Adami, D.Del Chiaro, A.Giordano, S.; Santos,A. 36. M. Al-Fares, A. Loukissas, and A. Vahdat, ``A scalable, commodity data center network architecture,'' in Proc. ACM SIGCOMM, 2008, pp. 63_74. 37. Hailong Zhang School of Information Engineering, Communication University of China, Beijing, China Xiao Guo ; Jinyao Yan ; Bo Liu ; Qianjun Shuai - SDN Based ECMPAlgorithm fordatacenternetworks. 1 2 3 4 5 9 0 1 1 1 5 1 6 1 7 1 8 1 9 1 0 2 1 2 2 2 3 2 5 2 6 2 7 2 8 2 9 2 0 3 1 3 2 3 3 3 4 3 5 3 6 3 7 3 8 3 9 3 0 4 1 4 2 4 3 4 4 4 5 4 6 4 7 4 8 4 9 4 0 5 1 5 2 5 3 5