International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 8, No.6 (Nov-2019)

http://dx.doi.org/10.12785/ijcds/080610

Performance Evaluation of Data Center Network Topologies via NS-2 Simulations

Ahmed Faeq Abdulhameed[1] and Rana Ejaz Ahmed[1]

[1]*College of Engineering, American University of Sharjah, Sharjah, UAE*

Received 05 Jul. 2019, Revised 20 Aug. 2019, Accepted 20 Oct. 2019, Published 1 Nov. 2019

Abstract: A datacenter provides the core computing and storage elements for cloud computing paradigm. The Datacenter network (DCN) topology defines the structure in which servers and the networking devices are interconnected within a data center. This paper presents the results of a comparative simulation study for four well-known DCN topologies, basic tree, Google fat tree, Facebook fat tree and Dcell, using a NS2 simulation environment. The traffic patterns used in the simulations are one-to-one, one-to-all and the all-to-all. The simulation results for these four topologies are also compared under various parameters changes including packet size and the TCP window size. The simulation results capture the performance metrics which are the average packet delay and the throughput. The simulation study shows that for less than 20 servers, the Dcell topology has smaller latency and higher throughput compared to the other topologies, while the Facebook fat tree topology performs better when the number of servers in the data center is large.

Keywords: Data center network, network topology, NS2, basic tree, Google fat tree, Facebook fat tree, Dcell.

1. Introduction

Cloud computing has lately emerged as a computing environment which enables access to shared pools of flexible computing resources and higher-level services which can be easily provisioned with slight management effort. Most of the recent cloud computing infrastructures are constructed on top of up-to-date data centers. It integrates the Infrastructure-as-a-Service (IaaS), the Software-as-a-Service (SaaS) and the Platform-as-a-Service (PaaS), and provides those services as utilities, so the end-users are charged by the quantity they used. The data center network topology (DCN) represents how different network and storage components within it are interconnected. Many giant information technology corporations have developed their own designs for DCN topologies, such as Facebook, Google, etc. [1-4]. The design of DCN topology is typically based on the requirements of the owner, which means the nature of applications that will be expected to run, format of the files, the way of storing its data, etc. In this paper, a number of popular data center network topologies are simulated in similar traffic situations using the NS2 simulator. The performance of those topologies is compared based on the throughput and the average delay. Several parameters for example packet size, traffic type, TCP window parameters are altered to observe their effects on the monitored performance metrics. The major contribution of this paper is the comparative study of well-known datacenter topologies via NS-2 simulations, considering the realistic TCP traffic among servers.

This paper is organized as follows: In Section II, we will present the background of the topic and give a brief literature review of the datacenters. Section III talks about the performance evaluation methods. In Section IV the outcomes of the simulations will be discussed and analyzed, and the conclusions are presented in Section V.

2. Background and Literature Review

A data center normally consists of thousands of networking devices (switches, routers), servers and physical links (such as optical fiber, cables, etc.). A data center network topology (DCN) describes the structure in which the networking devices and the servers are interconnected within it. The accurate choice of DCN topology is so important as it affects the total performance of the applications that are running in the data center.

A. DCN Topologies

There are numerous data center network topologies used in real data centers and their taxonomy is shown in Figure 1. New topologies have been suggested because of the continuous expansion of the data centers and their associated services, and to provision ever-increasing data volumes that are needed to be processed through the data

E-mail: b00057473@alumni.aus.edu; rahmed@aus.edu

centers [5]. In our project, we focused on only four DCN topologies: Basic tree, Facebook Fat tree, Google Fat tree, and DCell.

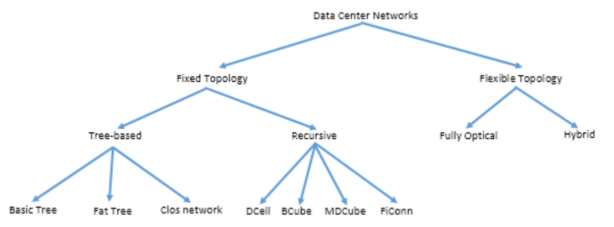

Figure 1. Taxonomy of Data Centre Topologies [6].

Basic tree topology is the beginning of all the fat tree topologies. It merely consists of three layers of network switches. It begins with the core switch, which is usually connected to aggregate switches after that those aggregate switches are connected to all the edge switches [6]. The quantity of switches grows at each lower level. The data center servers are connected to the edge switches, as displayed in Figure 2. The basic tree topology is not very common in the big data centers as it has various single points of failures [7], which is if a certain link fails, we may lose the connectivity to so many connected servers. However, this matter was fixed in the fat tree topology.

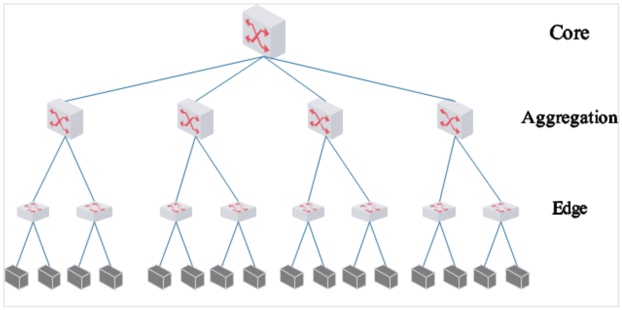

Figure 2. Basic Tree Topology [6].

The Google fat tree topology, in which, the switches of different levels are connected to each other with several links [8], thus, it can reduce the likelihood of failure of a particular point, as shown in the Figure 3 [9]. The connections of this topology start by the core switches level, which holds more than one core switch. Those core switches are then linked to the next level of switches which is a combination of the aggregate and the edge switches. In this layer each two aggregate switches are linked to another two edge switches, creating a unit that is called the "pod" [8]. All pods are connected to all the core switches via the higher level switches (the aggregate switches) and then linked to the servers via the lower level switches (the edge switches). As the Figure 3 depicts, there are several additional links in the network that can provide more than a single link for communication and also can reduce the chances of the network failure whenever a single link fails.

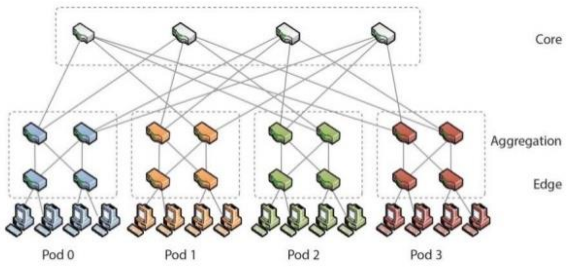

Figure 3. Google Fat Tree Topology [9].

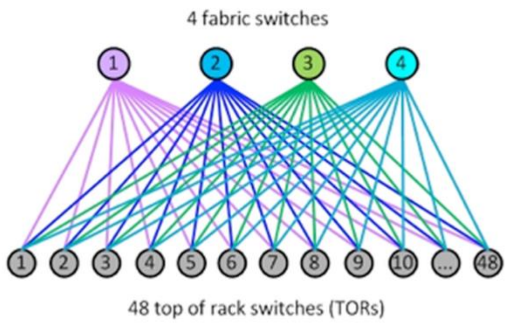

Figure 4. Facebook Fat Tree Topology [10].

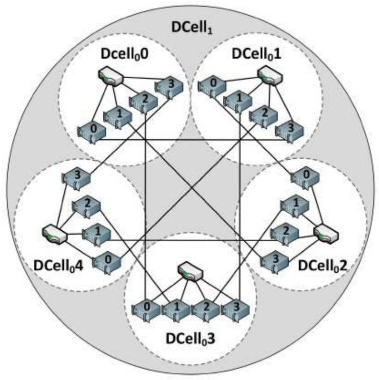

Figure 5. Dcell$_1$ made of 5 Dcell$_0$ units [9].

The Facebook fat tree topology, which is shown in the Figure 4, is newer topology, it consists of the fabric switches and the TOR (top of the rack) switches [10], and both of these two types of switches represent the pods. In each pod, the TOR switch is linked to four different fabric switches. In this topology we can see the smallest diameter (the shortest route between the farthest two nodes in the network) as compared to the other tree type topologies discussed in this study.

One the other hand, The Dcell topology is a recursive data center network topology that consists of smaller units [11]. As shown in Figure 5, each of the basic units are consisted of a single switch and "n" number of servers. This basic "smallest" unit is known as Dcell$_0$. The higher-level unit, $Dcell_1$, is the grouping of (n+1) $Dcell_0$ units that are linked by adding additional NICs (Network interface cards) in every server and each one of those NICs is connected to another server in another $Dcell_0$ unit [11]. Consequently, if the $Dcell_0$ has 4 (n =4) servers, at that time the $Dcell_1$ will be having 20 servers and $Dcell_2$ will be having 420 servers and so on.

B. Evaluation Metrics

The assessment of data center network topologies is mostly based on the following factors:

- **Network Diameter.** Is the shortest path between the farthest two nodes in the network. In another words, it is the total number of hops that the packet has to pass in order to reach to its final destination. I general, the smaller the diameter, the better (lower) the latency. So the diameter can grow logarithmically when there is an increase of the amount of servers [6].
- **Degree of the server.** Is the average number of NIC or network ports that are present in a single server. For all the tree type topologies, It is one but for the recursive type of topologies like the Dcell, it can be more than one of [6].
- **Number of switches.** We can say that it is obvious that the Dcell topology uses a smaller amount of switches as compared to the tree type topologies since it uses the servers that are with multiple NICs to execute the required switching. Therefore, every cell in Dcell topology has only a single switch. While on the other hand, the basic tree topology has three layers of switches, the core, the aggregate and also the edge. The core switches are less in quantity and linked to the aggregate switches which are larger in quantity. The aggregate switches are linked to the edge switches which typically are the highest in quantity. The edge switches are linked directly to the servers. For The Google tree topology, it also starts with the core switches which are linked to a higher amount of aggregate switches. The aggregate switches are linked to the same amount of edge switches, creating a unit identified as the "pod". The Facebook fat tree has only two levels of switches, the fabric level and the TOR level (top of the rack level) where each one of the fabric level switches is connected to all TOR switches.
- **Number of wires.** The number of wires does not illustrate the cost of the wires, but only indicate the complication of the wiring. In fact, the cost of the wiring is based on the total number of wires, the length of those wires, and the kind of those wires. The Dcell topology uses extra wires compared to the other tree type since each server uses more than one wire subject to the number of NICs in every server. The fat tree uses added wires compared to the basic tree, as the switches have more than one route for a particular pair of servers.
- **Number of servers.** While all the metrics mentioned earlier are based on the same amount of servers, the number of servers itself is a measure of the scalability of the network. For example, the addition of servers to the tree type topologies requires additional switches, while the addition of servers to the Dcell topology requires additional units/cells that has a single switch but it also requires additional NICs to the servers of the existing basic cells.

C. Literature Review

Several present research works suggested and assessed new data center topologies. In [7], the writers have done a comparative study of the fat tree topology and the Dcell topology and have simulated the topologies using the NS3 network simulator. Their concentration is on the averaged throughput and latency of the studied network based on the total number of contributed servers. Their simulation shows that at a low quantity of servers, the Dcell topology has better throughput and smaller latency; however, when the amount of servers increases further than a certain limit, then the outcomes are changed. In this research, the performance of Dcell is degrading as related to the fat tree when the amount of servers exceeds 20 servers.

In [9], the researchers utilized the mininet network simulator to build and mimic four DCN topologies, which are the Google fat tree Dcell, the Facebook fat tree and Bcube. Their study mimics the performance of these topologies under a number of traffic forms. They also study the recovery from certain kinds of failures which may occur in the real traffic. Their study demonstrates that the Facebook fat tree has the top performance results with respect to throughput and latency.

A good survey for a number of DCN topologies is presented in [2]. The researchers used both of gem5 and mininet simulators to mimic three famed data center topologies, which are Google fat tree, Facebook fat tree and the Dcell. Their simulation showed that the Facebook topology is better in latency but it is unstable for a large amount of hosts; on the other hand, the Google fat tree is the best among others in terms of throughput.

The DCN architecture HyScaleII [12] was proposed as an improvement of HyScale [13]. HyScaleII is a switch-centric type high performance hybrid optical-network DCN architecture that has most of the desired properties of a data center, for example, high scalability, high bisection width, low diameter, low network complexity and fault-tolerance. An efficient and simple routing scheme, called HySII routing, that exploits the structural specifications of HyScaleII is also presented. It is found that HyScaleII has a lower packet loss with a higher average aggregate throughput compared to HyScale by an average of 50% to 13.8%.

In [14], a simulation using NS3 simulator tested the performance of several DCN architectures in various realistic scenarios. The results of the simulation show that the fat-tree based topology performs better as compared to the DCell DCN architecture with respect to the average network latency and throughput.

In [15], the writers suggested a new DCN design framework, known as "REWIRE", to build networks by using an optimization algorithm. The algorithm figures a network with highest bisection bandwidth and least end-to-end latency while meeting user-defined requirements and accurately modeling the anticipated cost of the structure. The assessment of REWIRE is performed on a wide range of factors and it is found that its designed network have up to 100-500% extra bisection bandwidth and lesser end-to-end network latency compared to the equivalent-cost DCNs.

In [16], the researchers have implemented and simulated some DCN models which were the legacy DCN architecture, the switch-based architecture and the hybrid models, and then compared their effectiveness by finding the network throughput and the average packet delay. Their presented analysis may be used as a background benchmarking study for research on the simulation and implementation of the DCN topologies and customized addressing protocols for the large-scale data centers. They have performed detailed simulations under several network traffic patterns to assure the strengths and inadequacies of those different DCN architectures.

A simple and scalable architecture, called MatrixDCN, is proposed in [17]. MatrixDCN is an approximate non-blocking DCN, in which the switches and the servers are arranged in rows and columns that form a matrix structure. A MatrixDCN network may accommodate up to hundreds of thousands of servers without having bandwidth bottlenecks. In addition, the physical topology of the MatrixDCN network can be designed recursively with its logical topology, which helps to lessen the complexity of the management and the maintenance of the data centers.

In [18], "SlickFlow", a robust source routing approach is presented. 'SlickFlow" is implemented with OpenFlow and it allows fast failure recovery via combining source routing with alternative path information that is carried in the packet header. A primary path, as well as alternative paths, are compactly encoded as a series of segments that are written in packet header fields. Under the existence of failures along the primary path, the packets can be rerouted to the alternative paths by the switches themselves without the need for the controller to be involved. SlickFlow was evaluated on a prototype implementation that is based on Open vSwitch and proved its effectiveness in mininet emulator scenarios for fat-tree, Dcell, and Bcube topologies.

Zhang et al. [21] characterized the complexity of lifecycle management of datacenter topologies, and introduced some new metrics in that regard. However, they did not make comparison for existing, well-known topologies.

3. PERFORMANCE EVALUATION OF DCN TOPOLOGIES

A. Simulation Environment

NS2 (Network Simulator 2) is a discrete event simulator [19]. It is used to simulate different DCN topologies in this project. The simulator, NS2, is designed to target networking researches. It provides significant support for simulation of routing, Transmission Control Protocol (TCP), and multicast protocols over the wireless and wired networks.

In our simulations, we executed three traffic patterns: one-to-one, one-to-all, and all-to-all. Starting with the one-to-one traffic model, one server is sending simulated packets of data to another randomly selected server in the DCN topology. On the other hand, in the one-to-all traffic, one server transmits packets to all other servers, whereas in the all-to-all, all the servers are transmitting packets to each other. These three traffic models represent the real-life traffic situations in the data center topologies.

B. Performance Metrics

The main metrics of interest in the simulation are the throughput and the average packet delay for a given DCN topology.

- **Latency** is the packet's average delay that a packet experiences within the given DCN from the source to the destination servers. It is supposed that all traffic streams are TCP-based. For all the kinds of traffic patterns, the latency is measured by calculating the absolute difference between the sending and receiving times of a packet with the same ID number .After iterating this measuring process on all the sent packets throughout the simulation run, we measure the average delay time for the concerned topology under the same set of constraints. The simulation process then is repeated with changing in the parameters, for example TCP packet (segment) size and the TCP window size. The TCP window size denotes the maximum number of TCP packets that to be sent together at one-go without the need to wait for an acknowledgment, and it helps in executing flow control [20]. There are numerous algorithms used in the simulation to control the size of the TCP window.

- **Throughput** denotes the quantity of data bytes that are received fully by the intended receiver during a certain time period. We typically calculate it by finding the whole number of received packets at the anticipated receiver multiplied by the packet size in bytes and then divide by the total simulation time period. Similarly, for the one-to-one traffic, the said formula gives us the exact throughput. Though, for the one-to-all traffic pattern, we must divide the total amount of received packets by the total number of receivers (which is the total number of servers minus one). Likewise, for the all-to-all traffic pattern, we calculate for each server the one-to-all throughput, and after that we sum such throughputs from all the servers and then divide by the total amount of servers. We replicate this process after changing the parameters one by one, which are the TCP packet (segment) size and the TCP window size.

C. Simulation Parameters

For every simulation run, the simulation interval is set to 50 seconds. For all the simulated topologies, the hardware parts are defined below:

- **Basic tree.** This topology contains 13 switches and 20 servers. Additionally, it has 35 links that link those components, a single core switch is linked to five aggregate switches that are linked to 10 edge switches. Every server has an individual NIC. All links bandwidths are fixed to 10 Mbps and the its delay is 10 ms.

- **Google fat tree.** This DCN topology is made up of 25 switches plus 20 servers, and every server has a single NIC. All of these components are connected by 60 wires or links, and all the switches are linked to more than one link to guarantee additional path in case one link fails. That's why the amount of links here is nearly the double of the same-sized basic tree topology. Each of the five main (core) switches is linked to all the pods of the aggregate and edge level switches. Every pod consists of four switches: two aggregation plus two edge switches. Each aggregation switch is linked to an edge switch. The architecture ensures the availability of additional routes and avoids the single failure point. All links bandwidths are set to 10 Mbps and the links delay is 10 ms.

- **Facebook fat tree.** This topology is an improvement of the Fat tree topology. It has 14 switches and 20 servers. The links that are connecting the components of it are 60 wires or links. Every fabric switch is linked to all ten TOR (top of the rack) switches. Every one of the TOR switches is linked to the servers by an individual link only. All the bandwidths of the links are set to 10 Mbps and the links delay is 10 ms.

- **Dcell topology.** This topology differs from all other topologies as it is a server-centric topology wherein each server has more than an individual NIC cards. The topology we made, $Dcell_1$, contains 5 switches and 20 servers. Every switch is linked to 4 servers via one wire or link for everyone, forming $Dcell_0$. Five of $Dcell_0$ cells are created, then everyone is connected to the other 4 cells using the server-to-server links, and then these 5 cells will make the $Dcell_1$, with a total amount of links equals 30, as shown in Figure 5. All the bandwidths of the links were set to 10 Mbps and the connection links delay is 10 ms.

D. Protocols Used

In our study, various communication protocols are used. For sending/receiving the data, we use the transmission control protocol (TCP) to manage the communication and handshaking process of the network hosts. The data transfer is managed by the application layer file transfer protocol (FTP), which lets us to increase/decrease the size of the data packets. For the routing protocol inside the network, we used the distance vector protocol that is used to choose the best route if several paths that are available and to avoid the loops. Those protocols are used in all the simulation tryouts without any changes, excluding the window size and the FTP file size which was controlled by a command line in the code of the simulation.

4. Results And Analysis

A. Results

Figures 6 and 7 illustrate the outcomes of the latency and throughput, respectively, for the scenario of the default congestion window size of 20 and 1 Kbyte TCP segment size for various topologies and different traffic kinds. The size of the set of data segments that are sent in one go depends on the sender's configured window size and the advertised receiver's window size, that depends on the network business (congestion), the buffer size of the sender and the receiver, and the TCP selected congestion control mechanism that is Tahoe in our simulation.

Figure 6, illustrates that for the one-to-one traffic, both of the Basic tree and Google fat tree have nearly the same average delay because of the likeness in route length and no other traffic is affecting the communication. The average delay in the Dcell is a little bit fewer than it in the above-mentioned DCN topologies because of the fact that the average route length in Dcell is smaller as compared to the Basic and the Google fat tree. The Facebook fat tree shows the lowermost average delay because of its very small route length between the sender and the receiver.

In Figure 6, for one-to-all traffic, we can realize that the average delays are much more than the same for the one-to-one traffic and that is due to the congestion that is produced in the network during the data give-and-take process. We can still realize the likeness between the basic and Google fat tree and that is due to the same causes mentioned above that the average path lengths are equal between the senders and receivers. The Facebook fat tree has also enlarged delay due to the multiple receiver's traffic pattern but still showing the smallest delay compared to the others. The average delay in the Dcell topology is greater than others because of its distinctive structure, in which the receivers have different route lengths and the routing of the packets is going through the servers and the switches as well.

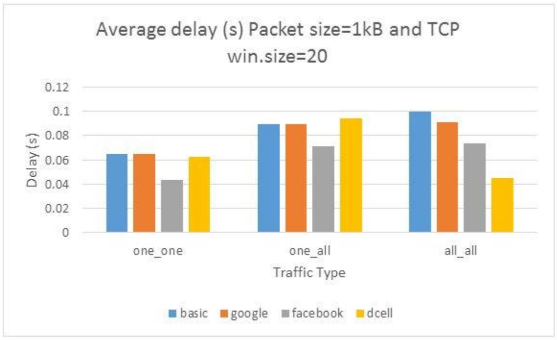

Figure 6. Average Packet Delay in (s) for TCP win. 20 and Packet size 1 kB

In Figure 6, for the all-to-all traffic, which is the closest to the real-life traffic scenarios, the Dcell is the only one with decreasing average delay while the other topologies have higher delays due to higher levels of traffic congestion. However, still the Facebook is better than the Google and that is due to lower diameter and the availability of multiple paths.

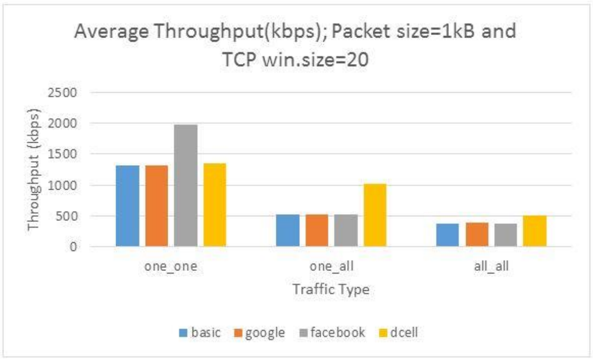

Figure 7. Average Throughput in (kbps) for TCP win. 20 and Packet size 1 kB

In Figure 7, the average throughput values are higher for the one-to-one traffic compared to the other patterns, and this is mainly due to the fact that the network is lightly congested. The highest throughput among the topologies is for the Facebook topology because it has the smallest diameter. For the one-to-all traffic, the average throughput is less than that of the one-to-one traffic, because of some lost packets during the congestion. For the same reasons, the average throughput for the all-to-all traffic is less than that of the one-to-all traffic pattern because of the greater amount of packets that are exchanging in the network, and the greater possibility of packets being lost and retransmitted. We also notice that, for both the one-to-all and all-to-all traffic patterns, the Dcell topology has the highest throughput and that is for the same reason mentioned above for the lowest latency values.

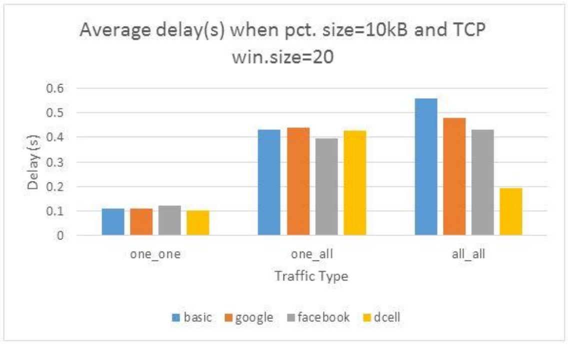

Figure 8. Average Packet Delay in (s) for TCP win. 20 and Packet size 10 kB

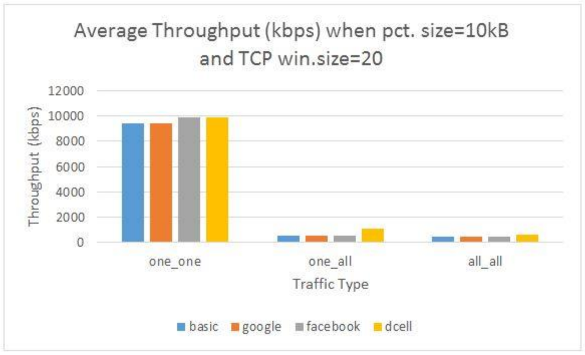

Figure 9. Average Throughput in (kbps) for TCP win. 20 and Packet size 10 kB

Figures 8 and 9 show the simulation results when the packet size changes to 10 Kbyte. As expected, the one-to-one traffic has the smallest average delay compared to the other traffic types, and the Dcell topology has the smallest delay among them. For the one-to-all traffic, the overall delay is higher as compared to the one-to-one traffic, which is due to the packets being lost and retransmitted because of the congestion. The Facebook topology shows the lowest delay followed by Dcell, then Google and the basic tree. The explanation of trends for average throughput and average delay for different topologies is the same as applicable for 1 Kbyte packet.

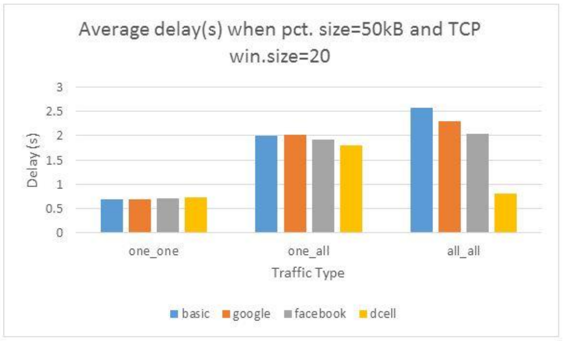

Figure 10. Average Packet Delay in (s) for TCP win. 20 and Packet size 50 kB

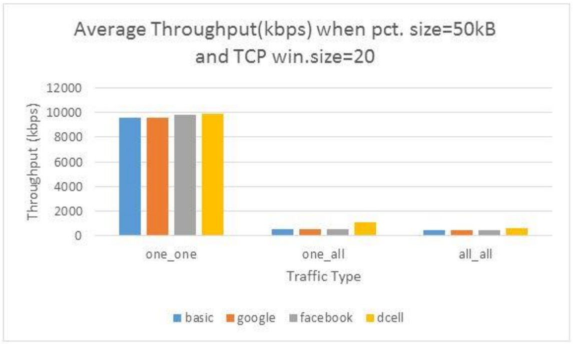

Figure 11. Average Throughput in (kbps) for TCP win. 20 and Packet size 50 kB

Figures 10 and 11 show the simulation results for 50 Kbyte packet size, and we observe similar trends.

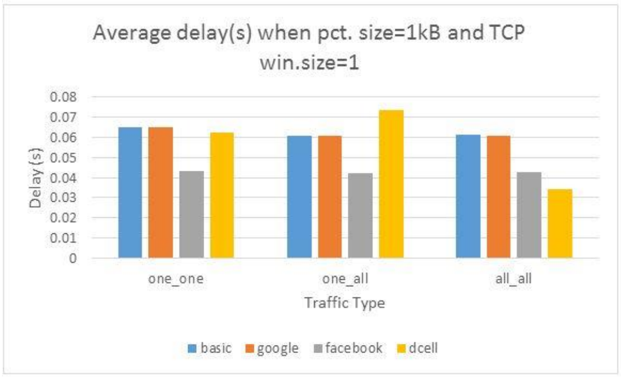

Figure 12. Average Packet Delay in (s) for TCP win. 1 and Packet size 1 kB

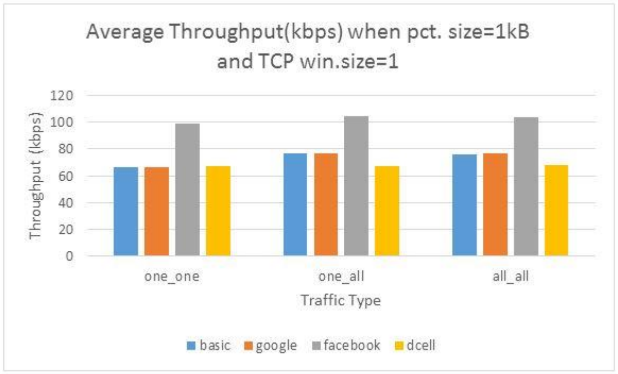

Figure 13. Average Throughput in (kbps) for TCP win. 1 and Packet size 1 kB

Figures 12 and 13 show the results when the TCP window size is changed from 20 to 1. For the average delay, we can see that the results are somehow closer to each other for all the traffic patterns, and that is because of the small window size, which is 1. This means that the sender can send only one packet to the receiver and wait for an acknowledgment before sending the next one. We can see that the results for the one-to-one traffic are logical and as per the network diameter. The Facebook has the smallest diameter and hence has the smallest delay. For the one-to-all traffic, the delay in Dcell is higher than the others and that is because of the higher average network diameter for all the hosts. For the all-to-all traffic, the delay in Dcell decreases for the reason of the focused traffic to the nearest hosts. For the average throughput results, we can see that no matter what the traffic type is, the results are almost the same, and that is because of the limited window size which prevents the advantage of sending multiple packets simultaneously. A minor increase in the throughput is observed for the one-to-all traffic and a little bit more for the all-to-all traffic, and that is because more packets are sent to/from different hosts, which lead to extra packets delivery.

We changed the packet size to 10 Kbyte for the same window size of 1 and the Figures 14 and 15 show the results for latency and throughput, respectively. For the one-to-one traffic, the results are as expected; bigger packet size causes more transmission delays and hence higher average delays as compared to the test cases where smaller packet sizes are used. For the throughput results, the Facebook is the best for the one-to-one traffic, but the Dcell is much better for the one-to-all and all-to-all traffic.

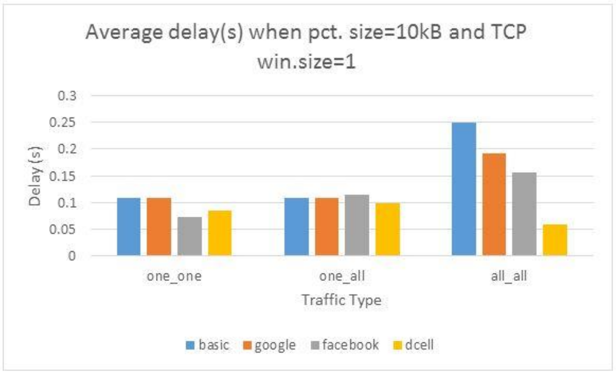

Figure 14. Average Packet Delay in (s) for TCP win. 1 and Packet size 10 kB

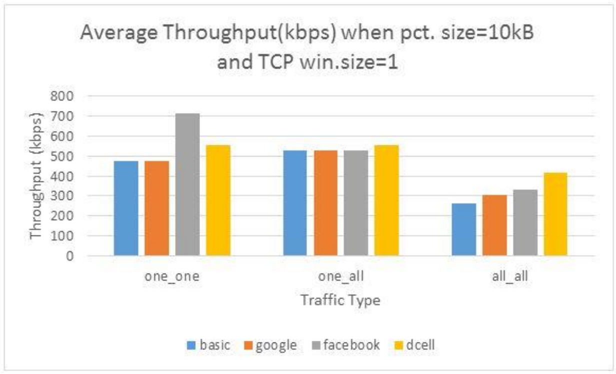

Figure 15. Average Throughput in (kbps) for TCP win. 1 and Packet size 10 kB

Figures 16 and 17 show the simulation results after changing the packet size to 50 Kbytes for the average delay and throughput, respectively, and similar trends are observed.

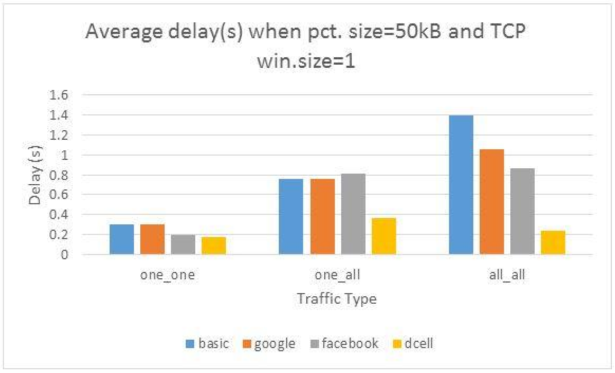

Figure16. Average Packet Delay in (s) for TCP win. 1 and Packet size 50 kB

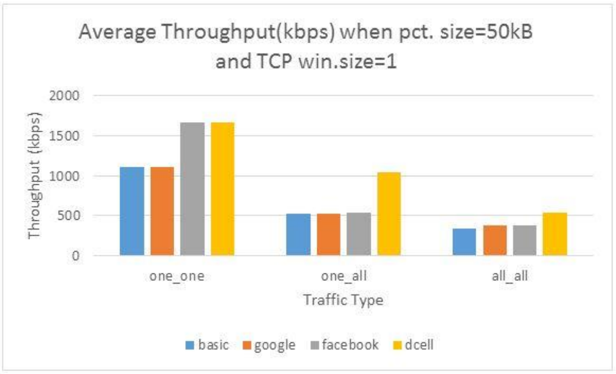

Figure 17. Average Throughput in (kbps) for TCP win. 1 and Packet size 50 kB

Figures 18 and 19 show the results after increasing the TCP window size to 100, and similar trends in the results are observed for different traffic types and topologies.

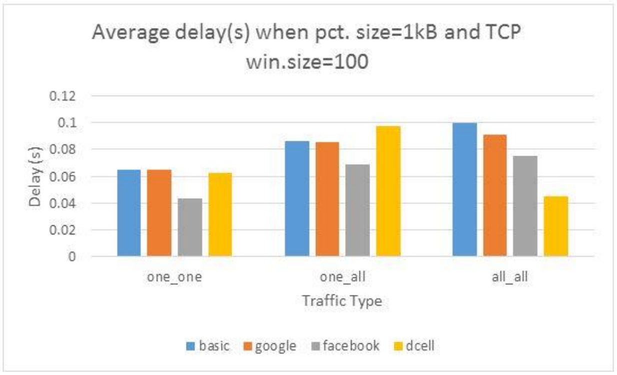

Figure 18. Average Packet Delay in (s) for TCP win. 100 and Packet size 1 kB

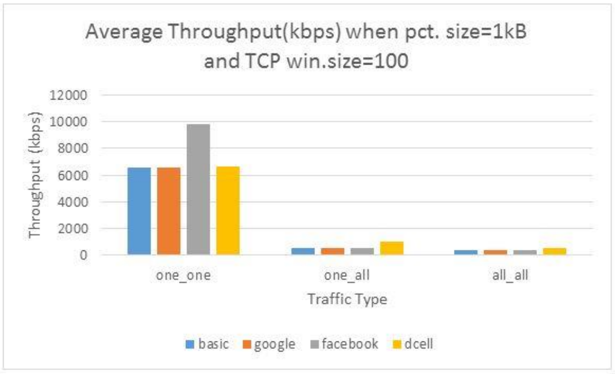

Figure 19. Average Throughput in (kbps) for TCP win. 100 and Packet size 1 kB

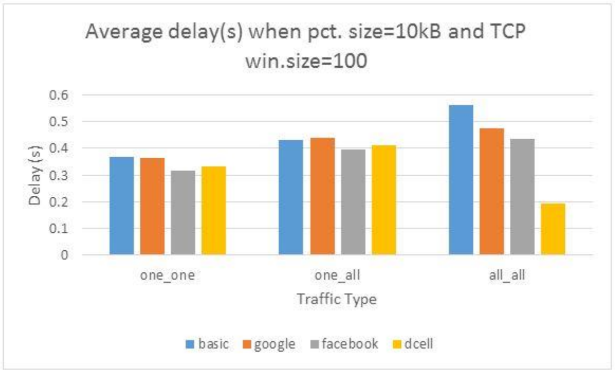

Figure 20. Average Packet Delay in (s) for TCP win. 100 and Packet size 10 kB

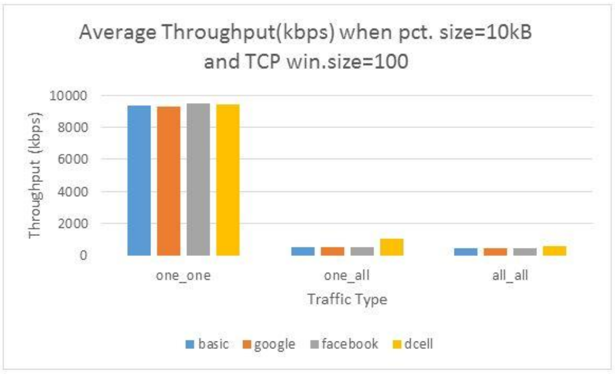

Figure 21. Average Throughput in (kbps) for TCP win. 100 and Packet size 10 kB

Figures 20 through 23 show the simulation results for the other TCP packet sizes of 10kB and 50kB, respectively. Similar trends in the results are observed for different traffic types and topologies.

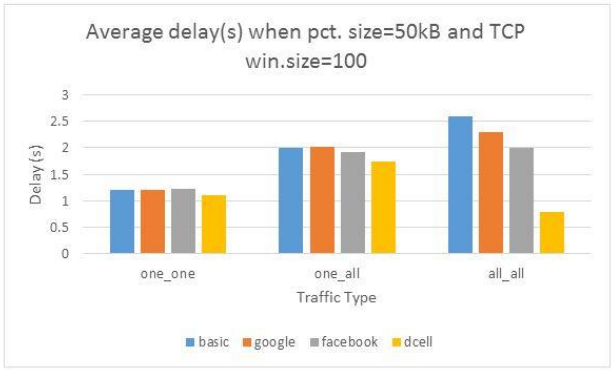

Figure 22. Average Packet Delay in (s) for TCP win. 100 and Packet size 50 kB

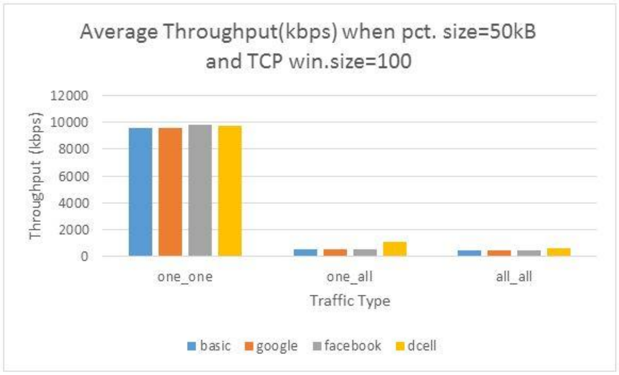

Figure 23. Average Throughput in (kbps) for TCP win. 100 and Packet size 50 kB

B. *Analysis*

For the tree type topologies, the basic tree has the lowest throughput and the highest average delay compared to the other tree type topologies. This is due to the fact that the other two tree-based topologies (Google and Facebook) have extra links in between the switches which provide extra paths for the traffic to be forwarded. We also noticed that the Facebook topology is much better than the Google fat tree because it has a smaller diameter, and hence less delay and higher throughput. The Dcell topology results depend on the traffic type and also the packet size. Dcell shows higher throughput compared to the Facebook topology in the all-to-all and one-to-all traffic patterns. The throughput in Dcell is lower than Facebook topology in the one-to-one traffic pattern. This is mainly because of the extra links and NICs that the Dcell servers have, which allow them to send the packets to more than one destination at the same time which enhances the average delay of that topology. In addition to that, the average diameter of the Dcell is relatively small, because it has many short paths to the other servers. In our case, each server has one link to one server from another cell, and two links to the servers in the same cell and more links to the other servers.

It should also be mentioned that when we increase the packet size, the average packet delay always increases, because the higher the amount of sent data, the higher the required delivery time. We can see that for the 1 Kbyte packet, the required time was in between 0.04 and 0.1 seconds. But for the big packets like 50 Kbytes, the average delay was in between 0.2 and 2.6 seconds. On the other hand, the average throughput increases with the packet size increase. For the 1 Kbyte packet, the average throughput is in between 65 to 2000 kbps; while for the 50 Kbyte packet, the average throughput is in between 333 to 10000 kbps.

The change in the TCP window size also affects the results of the delay and throughput. In general, the increase in the TCP window size results in an increase in the delay for the big size packets, while the delay for smaller packets gets minor change only. For the 50 Kbytes packets, the average delay for the TCP window of size 1 is in between 0.17 and 1.4 seconds while for the TCP window of size 100 is in between 0.8 and 2.6 seconds. For the average throughput, it increases with the TCP window size increase.

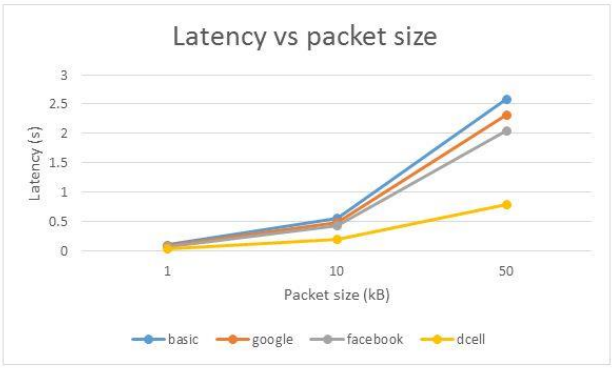

Figure 24. Latency vs packet size, when TCP win. Size = 20

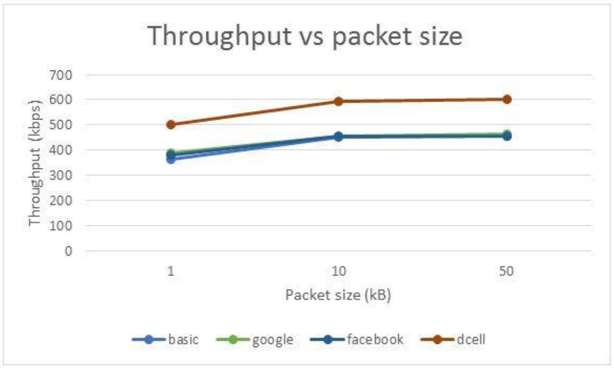

Figure 25. Throughput vs packet size, when TCP win. Size = 20

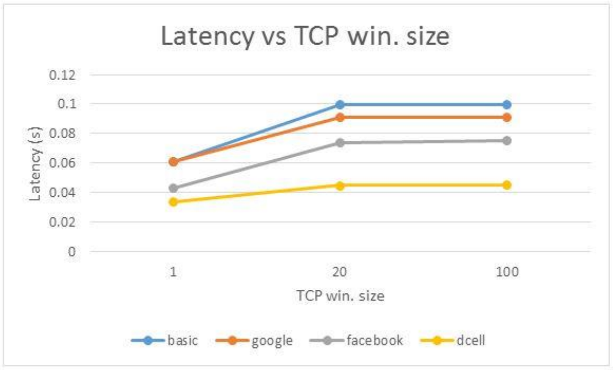

Figure 26. Latency vs TCP win. size, when packet size = 1kB

Therefore, we can see that the average throughput is in between 66 and 1600 kbps for the TCP window of 1. While for the TCP window size of 20 and 100, the average throughput is in between 380 and 10000 kbps. The Figures 24 through 27, show the summery of all the results.

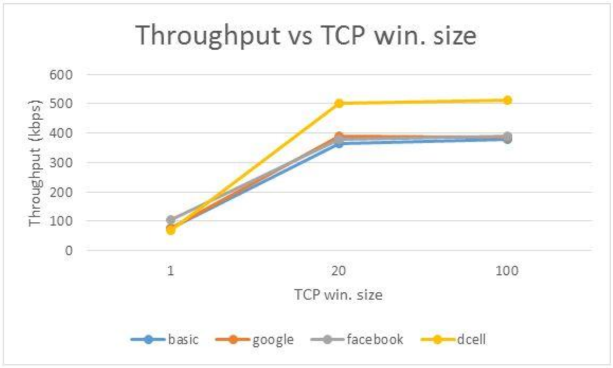

Figure 27. Throughput vs TCP win. size, when packet size = 1kB

5. CONCLUSION

We have presented the simulation results for four well-known Data Center Network topologies, which are the basic tree, Google fat tree, Facebook fat tree, and Dcell. Using the NS2 simulator, we have simulated three traffic patterns on each DCN topology. The traffic patterns were the one-to-one, one-to-all and the all-to-all. We can conclude that if the desired data center is required to serve a small number of servers that may increase significantly in a short time, we can suggest the Dcell topology because of its exceptional scalability compared to others, but if the data center is a fixed one with a large number of servers then the Facebook fat tree will be suitable because the Dcell efficiency reduces with the large numbers of servers.

REFERENCES

[1] Q. Zhang, L. Cheng and R. Boutaba, "Cloud computing: state-of-the-art and research challenges", Journal of Internet Services and Applications, vol. 1, no. 1, pp. 7-18, 2010.

[2] B. Hay, K. Nance and M. Bishop, "Storm Clouds Rising: Security Challenges for IaaS Cloud Computing," 2011 44th Hawaii International Conference on System Sciences, pp. 1-7, 2011.

[3] L. Barroso, J. Clidaras and U. Hölzle, "The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines," Second edition, Synthesis Lectures on Computer Architecture, vol. 8, no. 3, pp. 1-154, 2013.

[4] D. Bruneo, "A Stochastic Model to Investigate Data Center Performance and QoS in IaaS Cloud Computing Systems", IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 3, pp. 560-569, 2014.

[5] L. Brain, M. Aman and T. Niharika, "A Survey and Evaluation of Data Center Network Topologies", eprint arXiv:1605.01701, 2016.

[6] Y. Liu, J. Muppala, M. Veeraraghavan, D. Lin and M. Hamdi, Data center networks: Topologies, architectures and fault-tolerance characteristics. Springer Science & Business Media, 2013.

[7] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya and A. Vahdat, "PortLand", ACM SIGCOMM Computer Communication Review, vol. 39, no. 4, pp. 39, 2009.

[8] M. Al-Fares, A. Loukissas and A. Vahdat, "A scalable, commodity data center network architecture", ACM SIGCOMM Computer Communication Review, vol. 38, no. 4, 2008.

[9] H. M. Helal and R. E. Ahmed, "Performance evaluation of datacenter network topologies with link failures," 2017 7th International Conference on Modeling, Simulation, and Applied Optimization (ICMSAO), Sharjah, pp. 1-5, 2017.

[10] "Introducing data center fabric, the next-generation Facebook data center network", Facebook code, 2017. [Online]. Available: https://code.facebook.com/posts/360346274145943/. Accessed 21 November 2017.

[11] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang and S. Lu, "Dcell", ACM SIGCOMM Computer Communication Review, vol. 38, no. 4, pp. 75, 2008.

[12] S. Saha, J. S. Deogun and L. Xu, "HyScaleII: A high performance hybrid optical network architecture for data centers," 2012 35th IEEE Sarnoff Symposium, Newark, NJ, pp. 1-5, 2012.

[13] S. Saha, J. S. Deogun and L. Xu, "HyScale: A hybrid optical network based scalable, switch-centric architecture for data centers," 2012 IEEE International Conference on Communications (ICC), Ottawa, ON, pp. 2934-2938, 2012.

[14] K. Bilal, S. Khan, J. Kolodziej, L. Zhang, K. Hayat, S. A. Madani, M. Nasro, L. Wang and D. Chen, "A Comparative Study of Data Center Network Architectures.", ECMS, Koblenz, Germany, pp. 526-532, 2012.

[15] A. R. Curtis, T. Carpenter, M. Elsheikh, A. López-Ortiz and S. Keshav, "REWIRE: An optimization-based framework for unstructured data center network design," 2012 Proceedings IEEE INFOCOM, Orlando, FL, pp. 1116-1124, 2012.

[16] K. Bilal, S. Khan, L. Zhang, H. Li, K. Hayat, S. Madani, N. Min-Allah, L. Wang, D. Chen, M. Iqbal, C. Xu and A. Zomaya, "Quantitative comparisons of the state-of-the-art data center architectures", Concurrency and Computation: Practice and Experience, vol. 25, no. 12, pp. 1771-1783, 2012.

[17] Y. Sun, M. Chen, L. Peng, M. Hassan and A. Alelaiwi, "MatrixDCN: a high performance network architecture for large-scale cloud data centers", Wireless Communications and Mobile Computing, vol. 16, no. 8, pp. 942-959, 2015.

[18] R. M. Ramos, M. Martinello and C. Esteve Rothenberg, "SlickFlow: Resilient source routing in Data Center Networks unlocked by OpenFlow," 38th Annual IEEE Conference on Local Computer Networks, Sydney, NSW, pp. 606-613, 2013.

[19] T. Issariyakul and E. Hossain. Introduction to network simulator NS2. Springer, 2011

[20] K. Fall and S. Floyd, "Simulation-based comparisons of Tahoe, Reno and SACK TCP", ACM SIGCOMM Computer Communication Review, vol. 26, no. 3, pp. 5-21,1996.

[21] M. Zhang, R. Mysore, S. Supittayapornpong, and R. Govindan, "Understanding Lifecycle Management Complexity of Datacenter Topologies", 16th USENIX Symposium on Networked Systems Design and Implementation (NSDI '19), February 26–28, 2019, Boston, MA, USA

Ahmed Faeq Abdulhameed had received his B.Sc. degree in Computer Engineering from the University of Mosul, Iraq in 2009. He received his master's degree in Computer Engineering in the American University of Sharjah, UAE. His research interests are in networking, mobile application development, building automation systems and traffic control systems. He worked as a SCADA/TCS/ELV/BMS Engineer in (Saif Bin Darwish Company) in Abu Dhabi, UAE since 2011 till 2019.

Dr. Rana Ejaz Ahmed has a PhD in Electrical Engineering from Duke University, USA, currently working in the American University of Sharjah, UAE. He had worked as a faculty member at Lakehead University, Canada, and at King Saud University, Saudi Arabia. He was a visiting faculty member at the University of Ottawa during academic year 2007-08 while on sabbatical leave. He also worked at Research In Motion (RIM) and SpaceBridge in Canada in the areas of software testing, ATM switch testing and software quality assurance. His current research interests are in the areas of computer networking, telecommunications and computer architecture.