SDN Implementation in Data Center Network

Tariq Emad Ali [1], Mohammed A. Abdala [2], and Ameer Hussein Morad[1]
[1] University of Baghdad / Al-Khwarizmi College of Eng. / Dept. of Information and Communication Eng., Baghdad, Iraq
[2] Al-Hussain University College, Karbala, Iraq
Email: tariqemad@kecbu.uobaghdad.edu.iq; mohammedalmushdany@yahoo.com; ameer@kecbu.uobaghdad.edu.iq

***Abstract*—**Current requirements in the development of Information and Communication Technologies (ICT) are guiding new challenges to the future Internet, for which universal availability, large bandwidth utilization, and dynamic management are critical. However, traditional methods, where a manual configuration of branded devices is used, are complex and fallible. Recently, the development of Software-Defined Networking (SDN) as an optimal solution, has used to manage additional effective configuration, enhanced performance, and more flexibility to deal with huge network designs. It decoupling the control plane from the data plane and submitting a standard interface (OpenFlow) between the control plane and the networking devices (data plane). The software used to maintain the implementation of SDN completely. Thus, a programmatic command is controlling the performance of network devices.

***Index Terms*—**OpenDayLight; DCN; SDN; API; OpenFlow

I. Introduction

The environment of computer networks is generally consisting of ~~a~~ several networking devices such as routers, switches, and middle boxes (i.e., firewall). The networking devices are embedded with executed protocols to deal with many network events and applications. Network engineering are manually configure high policies into low-level configuration commands while adjusting to changing network conditions. These tasks are very intricate and frequently achieved with access to very limited tools. As a result, network management and performance modification are extremely difficult and therefore fallible [1]. Traditional Data Center Network (DCN) is considered as another challenging issue to the network operators and researchers due to the growing number of clients and bandwidth utilized that guide to many implementation limitations. Traditional DCN has two main complications they are limited by the physical infrastructure, which is devoted to a specific purpose, and a restricted amount of traffic handling ability. Therefore, new devices require high efforts to install and monitor .Secondly, the usage of some network resources is decreasing because of static routing mechanisms. This is a consequence of a rapid growth in the number of applications, websites and storage capacity. To overcome these restrictions, Software-Defined Networking (SDN) based on OpenFlow Data Center network architecture is developed to enhance the performance matrices and engineering traffic such as load balancing function. To reduce network congestions, the load balancing function hand out the traffic over the connected servers [2]. Consequently, the current networking devices, where the control plane and forwarding plane are coupling, are inappropriate for dynamic figuring and storage requirements. Thus, SDN is developed to simplify network management and change the concept of traditional networks by separating the control plane from forwarding plane~~s~~ [3]. The main goal is to allow software developers to count on network resources as easily as they do on storage and computing resources. In SDN, the network intelligence is the control plane which is logically centralized in software-based controllers. Also, network devices, depicted in the data plane, are used for packet forwarding devices that can be programmed by an open interface (e.g., CES [4], OpenFlow (OF) [5], etc.).

In this paper, SDN technology is presented and used to design and implement a data center network. Also, SDN is compared with traditional data center network in order to illustrate the efficiency of implementing SDN technology in data center networks. Section II defines SDN with OpenFlow protocol. Section III illustrates the network design architecture. An overview of the SDN controller is provided in section IV. Section V shows SDN data center design. The simulation results and evaluation of comparison between the two networks are represented in section VI. Finally, section VII presents the final conclusion.

II. Software Defined Network with Openflow Protocol

The introduction of SDN gets the interest of many researchers in the networking fields since it offers a method for programming the current networks and simplifying the design and management.

The tight coupling between the network's controlling plane and forwarding plane makes the design and management processes of the networks a complex task [1].

In SDN, the control software, that considered as the brain of the entire construction, is termed the SDN OpenFlow-Controller, which has the entire of the network and is responsible for the decision making [6]. Moreover, hardware devices, such as switches or routers, are

Manuscript received May 2, 2018; revised February 23, 2019.
doi:10.12720/jcm.14.3.223-228

implemented to manage the forwarding process of the packets to the destination depending on the commands approaching from the controller. The decoupling of the controlling layer from the forwarding layer simplifies the network evolution. In addition, several protocols and network applications applied over the network without interfering the network traffic [7].

The OF protocol is an open and standard protocol that describes the process of ruling and configuring the control plane in an SDN network by a centralized controller. Various complicated switching and routing protocols are handling the data, which is stored in Mac table and Routing tables. The forwarding plane in the traditional networks is developed based on these tables. OF protocol offers standardized and centralized rules that can control all the flow tables. Each OF network consists of one or more OF switches; each switch has one or further flow tables. Each port and flow table in the OF switch is linked with many counters which collect many events that the switch must handle. Fig. (1) Shows the OF flow table entry.

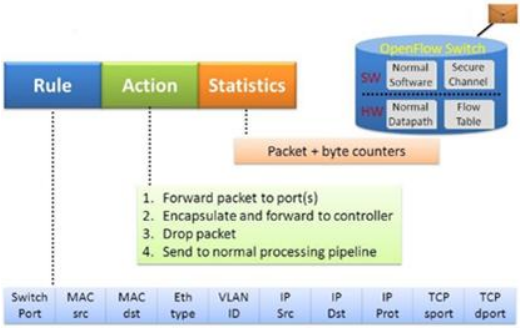

Fig. 1. OpenFlow table entry

III. Network Design Architecture

Present networks, where both the forwarding and the controlling planes are existing in the same networking device, are designed to satisfy the necessary requirements of industry, service providers, end users, and organizations. Recently, though, the existing network standards have been operated properly, but with the contemporary virtualized world, it would be challenging for current networks to encounter the new crucial requirements. Information technology (IT) departments of large enterprises are seeking to virtualize most of their services because of the restricted budget [8]. The restrictions of the traditional network architectures are:

A. Scalability Problem Issue

The traffic engineer is dynamic and random in today's virtualized data centers. Yahoo!, Google and Facebook suffer from several challenges associated with scalability as the scope of the end-user applications were expanded. This expansion guide to a rapid change in the network space that refers to scalability issues [9].

B. Vendors Dependability Issue

Datacenters and Internet service providers (ISPs) are continuously seeking to implement new networking services to achieve varying requirements. In addition, the capability to modify the network to the required environments had been limited by the absence of open standardized interfaces between different vendors. Such a problem between customer requirements and network abilities had severely affected the industry [8].

C. Configuration Issue

The configuration in the traditional network is a time consuming and is prone to errors. Prior to using the tools create updates; administrators are adding a device to the network and configuring multiple devices manually (e.g. routers, switches, and firewalls). Updates are occurring in many configuration settings such as Virtual local area networks (VLANs), Access control list (ACLs), and Quilt of Service (QoS). This method of configuration entails complex task for the administrator to add consistent policies [10].

D. The Complexity of Management Issue

The network management is costly and intricate. The predominant cause is that the management plane depends on the forwarding plane and, hence, the management of the network device interface are diverse [11].

The decoupling of the network architecture plane by SDN to forwarding plane (e.g., router/switch) and control plane are readily allowed to monitor, manage, and control a network from central software control [12]. The objective of SDN is to provide an open interface allowing the development of a specific software. The software controls the connectivity among network resources and the flow of network traffic, with possible modification achieved in the network [13]. Fig. 2 illustrates the SDN architecture that consists basically of three basic layers; application, control, and infrastructure layer.

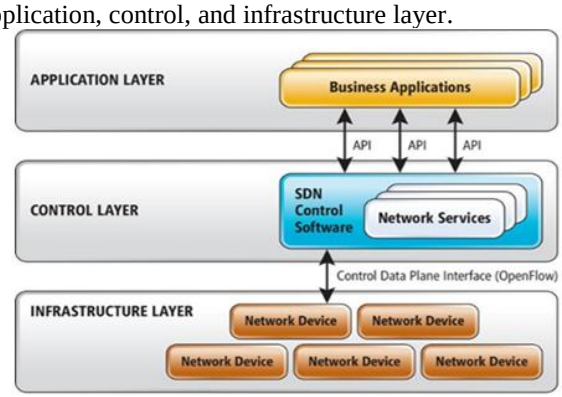

Fig. 2. Basic SDN architecture

SDN network provides two categories of Application Programmable Interface (API). The Southbound API is used to achieve communication between the controlling software and the network forwarding elements. OF protocol, is the major common example of Southbound API [9]. The Northbound API exist between the application and the controlling planes in order to easily supply and configure network elements. Northbound API provide various types of services such as the virtual private network (VPNs), Service-level agreement (SLAs),

and Traffic Engineering (TE) [14]. Fig. 3 shows the difference between traditional network and SDN network [15].

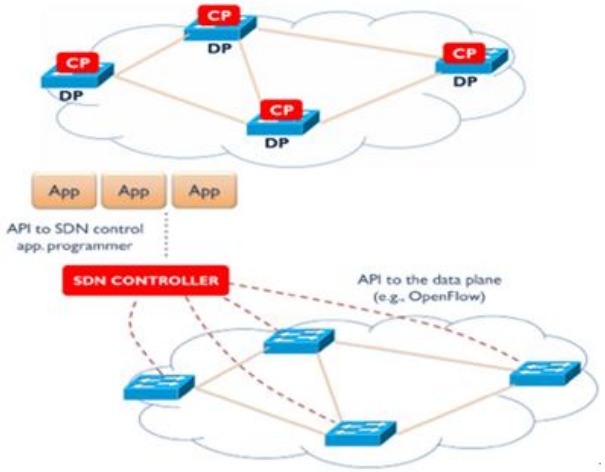

Fig. 3. Traditional and SDN networks

IV. SDN Controller

The logically centralized software-based controllers and the variety of different controller choices and their recent applications were highlighted in section I. This has raised the importance of gauging controllers and selecting the appropriate one. The distributed controller is a significant feature in the SDN network [16]. One of the advantages of a distributed controller is faultlessness. Multiple controllers in a network could create communication with a switch given that one controller has the primary role and the others have the secondary role [17]. Multiple controllers enhance the reliability, as the switch in OF mode is exempt from the status of the links with the controller. This will allow a recovery from any deficiency. In this paper, OpenDaylight Controller [18], shown in Fig. 4, is chosen for SDN implementation in DCN. This controller has the specification of supporting Python 2.7, OpenFlow (1.0 – 1.3), and connects with Open Virtual Switch (OVS) [19]. In addition, the Mininet is used as a tool for network emulation to create a network of virtual switches, hosts, and links on one Linux kernel [20].

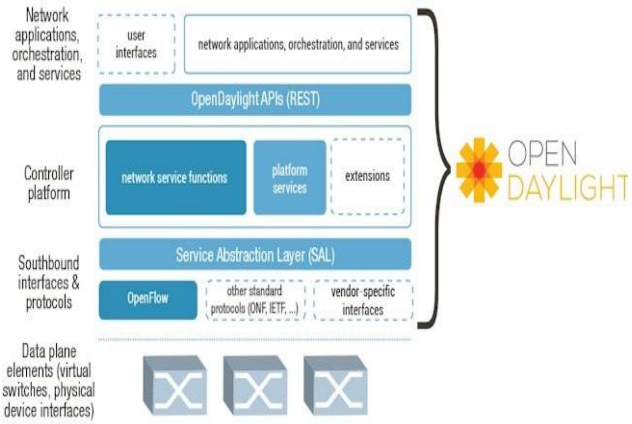

Fig. 4. OpenDaylight network controller

V. SDN Data Center Network Design

The design of SDN-DCN network is consists of several OpenFlow switches, hosts and OpenDayLight controllers. The OpenDayLight controllers, shown in Fig. 5, are directly connected to the network, which is responsible for configuring the switches to forward the traffic as a layer two (L2) switch. This controller allows the traffic forwarding between all of the switches exists in the network that is connected to it. The intention behind using this controller is to decouple the control plane from all the switches and to configure the switches to be a forwarding device by switching the flow tables between them.

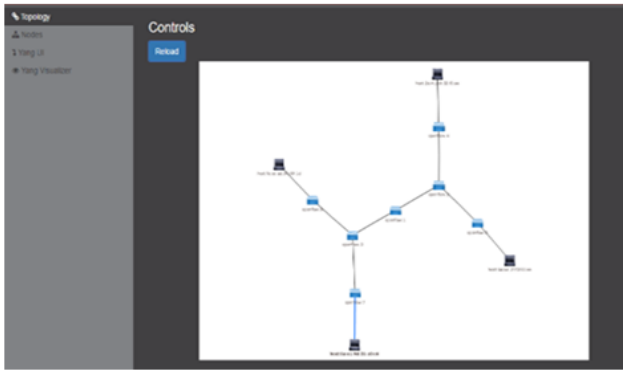

Fig. 5. OpenDaylight controller's graphical user interface

VI. Data Center Network Implementation Scenarios

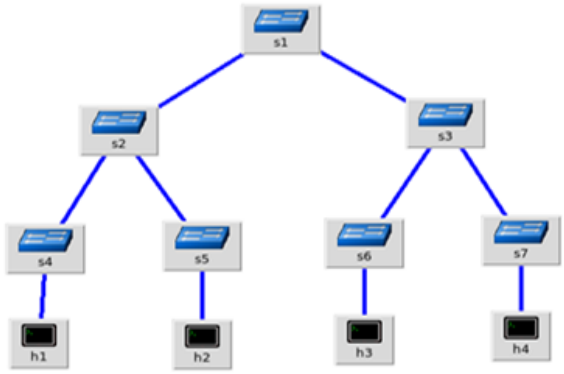

Fig. 6. Traditional network

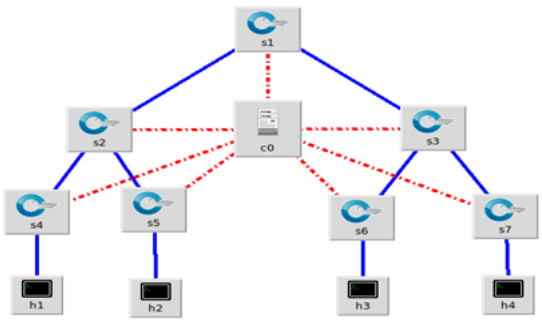

Fig. 7. SDN network

To implement and understand configurable networks and to achieve the DCN, it is important to have an

environment of numerous specifications. Two DCN network scenarios were implemented namely: traditional DCN and SDN- DCN. Each network was designed as a tree datacenter topology. The switches in the traditional network are working individually without a controller. Additionally, in the SDN network, OpenFlow-enabled switches are used. These switches were connected to an OpenDaylight controller (C0). Figs. (6 & 7) shows the traditional and SDN networks respectively.

The specifications of controllers are shown in Table (I).

TABLE I: OPEN DAYLIGHT CONTROLLER SPECIFICATION

Parameter	Controller
IP address	192.168.1.100
Port Number	6633
CLI	Enabled
OpenFlow version	1.3
Protocol	TCP
Connected Switch	OVS

Finally, the switch runs in the root by default. So, running a command on the "switch" is similar to running it from a regular terminal. In addition, each host has its own IP address which was defined before the emulation starts [21].

VII. SIMULATION RESULTS AND EVALUATION

In this section, Traditional DCN and SDN-DCN scenarios were investigated. Also, both scenarios were compared to observe the size and throughput of transmission delay of the SDN-based network compared with the traditional network.

A. Delay Measurement

With reference to Fig. 6-7, host one (h1) from each pod in both SDN and the traditional network had been selected to achieve the delay test. In this test, the aim behind this selection is to use all the network pods with varying connectivity distances to inspect the influence of these distances on the results. The hosts (h2, h3, and h4) are elected to perform the destination IP address of (h1). The delay of the transmission is measured for both traditional and SDN based network by four dissimilar number of packets as shown in Fig. 8, in which the delay in the traditional network is higher than in the SDN network. Since each device has a control plan and data plan in the traditional network, then each device make processing for each packet. This will increase the time reaching packet to a destination. While in the SDN network there is a controller device that has the control plan that prepares the processing and takes the decision for the packet route. Whereas the other devices in the SDN network have a data plan that will only make forwarding the packet depending on the decision that emerges from the control device, this will decrease the time reaching packet to the destination, since there is one control device in the network. The number of packets had increased and the response time changed, the SDN network produces identical performance due to the use of a similar controller.

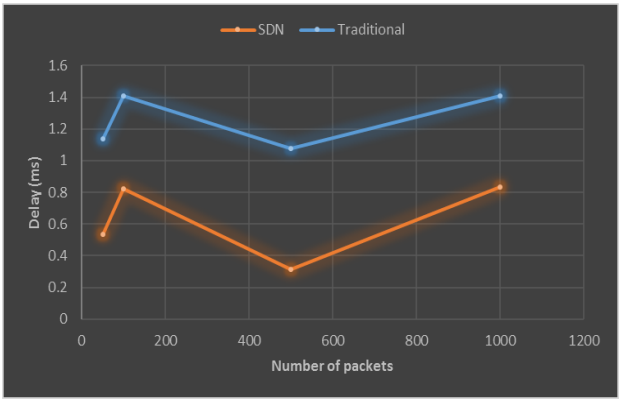

Fig. 8. Delay in SDN and traditional DCNs

B. Throughput Measurements

The second performance metric is the throughput. The way of measuring the throughput depends on varying the parameters of the TCP/UDP protocol (i.e. window size and buffer length). The measurement of the throughput is divided into two parts, one is varying of TCP connection and the other is varying of UDP connection. The throughput is measured for both traditional and SDN networks. The TCP throughput is restricted by the round time trip of the link and the window size. The default window size value equals 85.3 Kbyte. Fig. 9 shows the TCP throughput in both SDN and traditional networks with varying the TCP window size. The throughput in SDN network is higher than the traditional network since the network throughput is defined as the amount of data transmitted from source node to destination node in a given time period. Throughput is typically measured in bits per second (bps). Analytically, it can be defined as the ratio of maximum receiver bandwidth to round-trip time (delay) between nodes:

$$\text{Throughput} = \frac{\text{maximum receiver bandwidth}}{\text{round} - \text{trip time}}$$

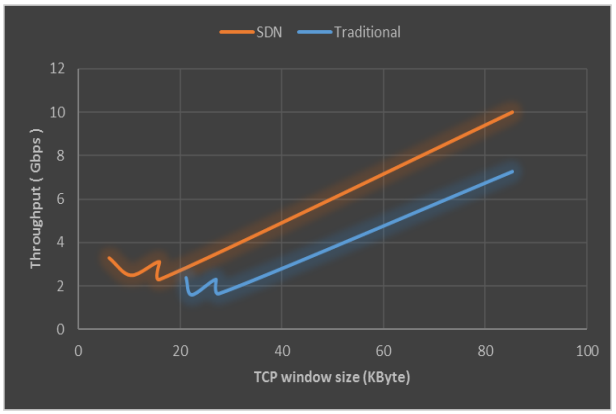

Fig. 9. Throughput using TCP in SDN and Traditional DCNs

The UDP throughput in both SDN and traditional network with varying the UDP buffer length is shown in Fig. 10 (its default value is 1470 Byte). It has been observed that the throughput in the SDN network is higher than the traditional network for the same reason explained for the TCP connection except for the difference in the methods of connection between the TCP and UDP.

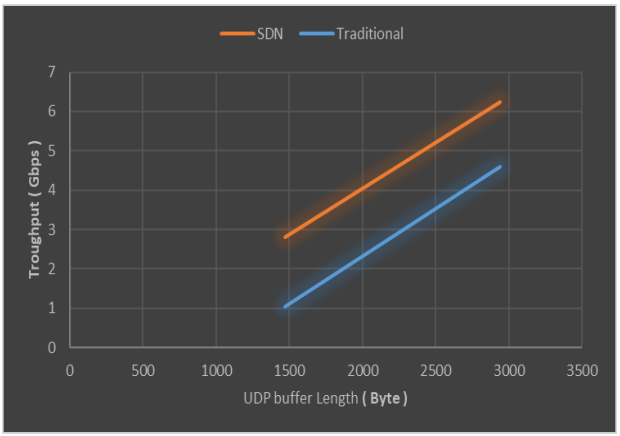

Fig. 10. Throughput using UDP in SDN and traditional DCNs

VIII. Conclusion

SDN is a network technology that provides an optimal solution to the today's rapidly growing network requirements. This paper addresses the extra flexibility of SDN technology in attaining superior network performance when compared with the traditional network. This can be achieved without the need of adding new devices or perform manual configuration for all devices. Moreover, the SDN technology is effective with an additional switch or router linked to OpenFlow protocol enabled feature. Successful feasibility tests were undertaken. In comparison with the traditional network, the SDN showed an enhanced delay and throughput through the use of OpenFlow protocol with a centralized controller in SDN network represented by OpenDayLight. In addition, the SDN platform simplified the use of controller device to configure a module that makes the switches as a forwarding device. Although this work is applied to small tree DCN, however, it can be easily expanded and applied to larger DCN.

References

[1] N. Feamster, J. Rexford, and E. Zegura, "The road to SDN: An intellectual history of programmable networks," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 2, pp. 87-98, April 2014.

[2] S. Mahapatra and X. Yuan, "Load balancing mechanisms in data center networks," in *Proc. 7th International Conference & Expo on Emerging Technologies for a Smarter World*, September. 2010.

[3] X. Foukas, M. Marina, and K. Kontovasilis, *Software Defined Mobile Networks (SDMN): Beyond LTE Network Architecture*, 1st Edition, WIELY Publisher, Finland, March 2015.

[4] B. Lantz, "Mininet and open vSwitch," Open vSwitch Fall Conference, Open Networking Laboratory. The Open vSwitch 2015 Fall Conference was held at the San Jose Doubletree on November 16 and 17, 2015.

[5] Jisha and Radhika, "Software defined networking: A new frontier in networking," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 5, no. 7, pp. 1263-1266, July 2015.

[6] V. Shukla, *Introduction to Software Defined Networking-OpenFlow & VxLAN*, 1st Edition, CreateSpace Independent Publishing Platform, June 2013.

[7] D. Kreutz, F. Ramos, P. Verissimo, C. Esteve, S. Azodolmolky, and S. Uhlig, "Software-Defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14-76, January 2015.

[8] A. Sonba and H. Abdelkarim, "Performance Comparison of the State of Art openFlow Controllers," Master thesis in Computer Network Engineering, Halmasted University, December 2014.

[9] H. Kim and N. Feamster, "Improving network management with software defined networking," *IEEE Communications Magazine*, vol. 51, no. 2 , February 2013.

[10] I. P. Knowledge, *Traditional vs Software Defined Networking*, 2014, last accessed 4th Sept. 2016.

[11] H. Ballani and P. Francis, "CONMan: Taking the complexity out of network management," in *Proc. SIGCOMM Workshop on Internet Network Management*, ACM, Cornell University, New York, Sept. 2006.

[12] R. Khondoker, A. Zaalouk, R. Marx, and K. Bayarou, "Feature-Based comparison and selection of Software Defined Networking (SDN) Controllers," Fraunhofer Institute for Secure Information Technology Rheinstr., Darmstadt, Germany, June 2014.

[13] D. Sharma, "Training Report on Software Defined Networking," thesis in Electronics & Communication Engineering, Florida International University, Florida USA, Sept. 2016.

[14] F. Nick, J. Rexford, and E. Zegura. "The road to SDN: an intellectual history of programmable networks." ACM SIGCOMM Computer Communication Review 44, No. 2, pp. 87–98, 2014.

[15] T. Emad , A.Morad and M.Abdala, "Load balance in data center SDN networks," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 5, pp. 3086-3092, October 2018

[16] P. Goransson and C. Black, *Software Defined Networks: A Comprehensive Approach*, 1st Edition, Morgan Kaufmann Publishers, June 2014.

[17] M. Basheer and E. Bassam, "Survey of software components to emulate OpenFlow protocol as an SDN implementation," *American Journal of Software Engineering and Applications*, vol. 3, no. 6, pp. 74-82, Dec. 2014.

[18] P. Robb, *OpenDaylight – State of Development*, March 5th 2015, lst accessed Sept. 2016.

[19] Open Networking Foundation, "OpenFlow Switch Specification version 1.0.0", ONF TS-01, Dec. 2009.

[20] C. Logbottom. (May 2012). Data Center Design: Using Engineered Racks, Pods, and Containers. [Online]. Available: http://www.computerweekly.com/tip/Data-centre-design-Using-Engineered racks-pods-and-containers.
[21] Mininet Team. Mininet Walkthrough. (2016). [Online]. Available: http://www.mininet.org/walkthrough/#interact-with-hosts-and-switche.

Mr. Tariq Emad Ali has B.Sc and M.Sc. in Electronic and Communication Engineering, College of Engineering, Baghdad University. He is Assistant lecturer at the University of Baghdad, Al-Khwarizmi Collage of Engineering, Information and Communication Engineering Department. He has 6 published scientific & technical papers including IEEE explorer scoops index. Mr. Tareq Emad has 11 years of academic & practical's and consulting experience in Networking & Communication. He currently teaches & conducts research programs in the areas of software computer networks, soft computing, intelligent agents, Ad Hoc networks, Wireless Sensor Networks, Routing Protocols and Security of VANETs, Smart Antenna in MANETs and WiMAX Networks, SDN Networks, IOT, IOE, IOV and others.

Dr. Mohammed A. Abdala is Associate Professor and Head of Electronic and Communication Engineering, University of AL Hussain .He is a senior member of IEEE, Iraqi Engineers Union and Federation of Arab Engineers. He is active in works that led to the establishment of several engineering departments & colleges in Iraq. He published more than 37 scientific & technical papers. Dr. Abdala has 31 years of academic & consulting experience in Microelectronics & Advanced VLSI. He currently teaches & conducts research programs in the areas of software engineering, image processing, and computer.

Ameer H. Morad: He received the Ph.D. Degree (1998) in Signals and Information Processing from Institute of Information Science at Beijing Jiao Tong University, China. He is interested in Image and video processing, pattern recognition, Computer Vision, Cryptography, Computer Networking and Discrete Event Simulation. Currently, he is Deputy Dean at Al-Khwarizmi College of Eng., University of Baghdad, Iraq.