ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

A Review: Performance Measurements of Transport Protocols

A. M. Anisul Huq

Faculty Member, Dept of Computer Science, American International University – Bangladesh (AIUB), Dhaka, Bangladesh.

Abstract: Seamless integration of wireless devices into the global Internet still poses a formidable challenge for the telecommunication industry. The main obstacle here is that, TCP was designed for wire-line networks and hence it interprets difficulties/errors in radio transmission differently. The objective of this paper is to review the measured performances of TCP in various operating conditions. The measurement results demonstrate the impact of different operating paradigms (e.g. WAN, WLAN etc.) on the performance of TCP.

Keywords: round trip time (RTT), bandwidth-delay product, Multiple Access with Collision Avoidance for Wireless (MACAW), Carrier sense multiple access (CSMA), bit error rates (BER).

I. INTRODUCTION

The transport layer deals with the quality of service issues of reliability, flow control, and error correction. The most important protocol operating in this layer is Transmission Control Protocol (TCP). TCP is a connection-oriented protocol that provides excellent and flexible ways to create reliable, well-flowing, low-error network communications. It controls the dialogue between source and destination while packaging application layer information into data units called segments. The term "connection-oriented" does not mean that a circuit exists between the communicating computers. However, it does mean that layer 4 segments (in the OSI model, "transport" is the 4th layer) travel back and forth between two hosts to acknowledge the connection exists logically for some period.

Other transport layer protocols include UDP, RTP, SIP and SCTP. User Datagram Protocol (UDP) is a lightweight protocol that allows applications to directly use of the unreliable datagram service provided by the underlying IP layer. Because of its lightweight, UDP is somewhat faster. This protocol is commonly utilized for applications that use simple query/response transactions (e.g. Domain Name System) or applications that support real-time communications (e.g. Voice over IP and online gaming). Other transport layer protocols include RTP, SIP and SCTP. Real-Time Transport Protocol (RTP) carries audio and video data over the internet during video conferences. Session Initiation Protocol (SIP) is used to set up and tear down such conversations. Stream Control Transmission Protocol (SCTP) on the other hand, is an extension of TCP, capable of handling several data streams in parallel. It is another protocol used for voice transfer over the Internet.

In this paper, we have confided our discussions only to the issues related to the performance measurements of TCP. The main reason for this approach is, TCP alone supports the transfer of over 90% of all traffic across the Internet [1]. Hence, by measuring the performance of TCP alone, we can roughly identify the drawbacks in the transport layer and subsequently design ways to mitigate those challenges. Also, in our opinion, transport protocols such as, UDP, RTP and SIP are designed and utilized for different purposes. In other words, they do not have many common grounds. Therefore, they should not be compared with each other just on basis of performance measurement.

Coming back to TCP, our performance measurement and analysis will include different operating paradigms. The rest of the review paper is organized as follows. Section II looks at TCP's performance on PCs. In section III, we focus on TCP's performance in the conventional Wide Area Network (WAN). Subsequently, in sections IV and V, we inspect TCP's performance issues in Wireless LANs (WLAN) and Mobile Networks respectively. Section VI tries to draw a conclusion by suggesting ways in which TCP's performance can be improved in a wireless environment.

II. TCP PERFORMANCE MEASUREMENTS ON PCS

Back in 1992, *Huang et al.* [5] did a performance measurement of the processing overhead of TCP on personal computers (PCs). Their measurement environment consisted of two PCs that were connected via an Ethernet running TCP/IP protocol suite for communication. However, for the purposes of their experiment, *Huang et al.* [5] devised their own version of TCP/ IP, which they claim, was quite similar to the standard version. The only improvement they made, was to rewrite the checksum computation of TCP with assembly language instead of C. The PCs used in the system had 80386-33 CPUs with 64 KB cache memory. In all measurements, the transactions contained 20 MB of data with a maximum segment size of 1024 bytes and they did not allow any other stations to send data on the Ethernet while the measurements were taken. For avoiding extra delay, an ACK was sent by the receiver immediately after a packet was received. The authors took detailed measurements of the overheads caused by checksum computation and acknowledgement processing.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

Huang et al. [5] figured out the reasons why checksum computation is slower on PC architecture. Mainly because, the high-low byte order of an integer (16 bits) stored in a PC's memory is different from that defined in TCP/IP. Hence, a reordering has to be done for each integer before checksum can be computed. Additionally, the 1's complement addition instruction is not present in PCs (at least in 80286 and 80386 CPUs). Therefore, we have to use 2's complement addition first and convert it to 1's complement addition.

After measuring the overhead, *Huang et al.* [5] showed that, TCP checksum computation takes up almost 25% of the overall time. It should be noted that, the checksum computation here, was written in Assembly language, which is much faster than a high-level language such as C. In their experiment, a file transfer was considered which results in a higher percentage of checksum computation overhead. Based on their findings, *Huang et al.* [5] suggested that TCP checksum computation should be implemented at the hardware level. They claimed that, it would reduce the overhead by 50%.

Sending and receiving acknowledgement (ACK) for each packet is another source of CPU overhead. *Huang et al.* [5] shows that ACK processing is responsible for roughly 10% of all overhead, which is not negligible in high-speed network [5]. They proposed that, one ACK should be sent for every two packets, which in turn will half the overhead.

III. TCP PERFORMANCE MEASUREMENTS ON WANS

TCP, as a transport protocol, has limitations when it is operating over long distance. This can cause many applications to perform poorly. Hence, TCP performs sufficiently over short-distance LANs; but lacks while transmitting over Wide Area Networks (WANs). In the following subsection, we explore the challenges of TCP performance over WANs and suggest ways to mitigate those challenges.

A. Latency & Small TCP Window Size

Latency refers to the round trip time (RTT) for a packet to traverse from a sender to the receiver [4]. According to [6], TCP experiences significant throughput loss as link latency increases (Fig. 1). For a T3 link (BW = 45 Mbps), the TCP throughput starts out at the available line rate for lower latencies. At higher latencies, the throughput begins to degrade rapidly. According to [7], at a latency of 100 ms, TCP can only utilize 10% of the link's available bandwidth. WAN links that cross the United States have average latency times ranging from 75 ms to 100 ms. In global networks, RTT regularly exceeds 250 ms. Latency on satellite links can be as high as 430 ms [3]. These intervals might look quite small, but inadvertently they will wreak havoc on application's performance over a WAN because of the interactive nature of TCP (i.e. ACKs).

Eradicating latency is impossible; simply because, its physics. Data will take some time (even when transmitted at the speed of light!) to travel long distances and there will be delays with store-and-forward hops across routers [4].

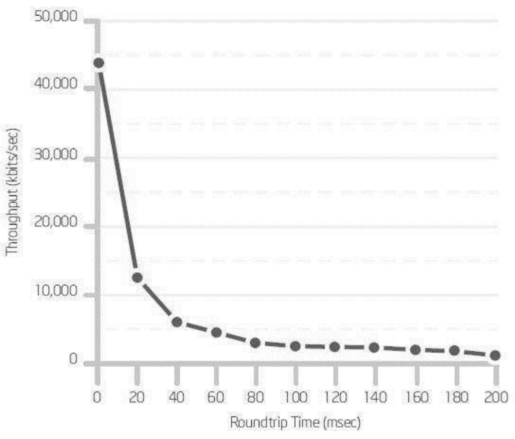

Fig. 1 Impact of line latency on throughput. Single TCP connection's throughput with 32K window size [6].

What we need to do, is to understand and reduce the impact latency has on applications. Bandwidth-delay product gives an indication on how latency will affect an application's performance. The bandwidth-delay product can be calculated using the following formula:

$$\text{Bandwidth-delay product} = \text{bandwidth} * \text{RTT}$$

Here, bandwidth (BW) refers to the link's BW and latency is measured in round trip time or RTT [3]. In order to fully understand the effect of bandwidth-delay product, let us consider a T-1 link running across the United States:

$$1.544 \text{ Mbps} * 90 \text{ ms RTT} = 138{,}960 \text{ bits} = 17370 \text{ bytes} = 17.3 \text{ KB}$$

Now we need to compare this bandwidth-delay product with the host's TCP window size [3]. We know that, window size is the amount of data allowed to be outstanding at any given point of time by the transporting software. Most standard TCP implementations are limited to 65 KB windows [4]. Here we see that the bandwidth-delay product is less than the window size making bandwidth the limiting factor. If bandwidth-delay product somehow becomes greater than window, then latency is the limiting factor; which happens when we consider a cross-country OC3 optical link. Here, the link's BW is 155 Mbps and RTT is 60 ms, making the bandwidth-delay product 1,163 KB. For a DS3 satellite connection the bandwidth-delay product goes up to 3,038 KB (here the link's BW is 45 Mbps and RTT is 60 ms). Even with enhanced TCP versions capable of using up to 512 KB or larger windows, there remains a very big gap between the bandwidth-delay product and the window utilized. This ultimately results in large amount of "dead air" and inefficient bandwidth utilization. This can seriously degrade the performance of real time applications [2][3][4].

To alleviate the problem of limited window size, [7] suggests to "turn-on" the TCP extensions specified in RFC1323 [15]. This feature is now supported by most operating systems (OS). Here, the TCP window scaling option allows us to utilize TCP window sizes of up to 1,073,741,823 bytes, which is sufficient until speed reaches 1Tb/s (with RTT of 100ms).

IV. TCP PERFORMANCE MEASUREMENTS ON WIRELESSLANS (WLANS)

It is a well-known fact that, networks with wireless links suffer from significant packet losses caused by bit errors and handoffs. TCP responds to all such losses by invoking congestion control and avoidance algorithms, which ultimately degrades the end-to-end performance of WLANs. To improve TCP's performance, simulation based investigations took place.

Using simulation, *Gerla et al.* [13] addressed the problem of TCP data packets colliding with TCP ACKs over a wireless multi-hop network. They argued that, this problem can partly be solved by using link level ACKs in conjunction with TCP error and window control. Their findings indicate that, embedding the ACKs in the MAC layer protocol, MACAW (Multiple Access with Collision Avoidance for Wireless) provides the best throughput. They have compared CSMA (Carrier sense multiple access) with MACAW using GloMoSim [14] as the simulation platform and for simplicity, only single TCP connection was used.

In the first set of experiments (table I shows the result), the single TCP connection covered a variable number of hops, from 1 to 7. The TCP window (W) was kept fixed at 1460 Bytes (i.e., W = 1 packet). The comparative results for CSMA and MACAW throughputs were reported as a function of number of hops (H) in Table 1. The throughput is inversely proportional to the hop distance. CSMA throughput is much higher than MACAW, mostly because the latter has additional control frames (e.g. Request to RTS or RRTS).

TABLE I: THROUGHPUT (KBIT/S), SINGLE TCP CONNECTION, VARIABLE NUMBER OF HOPS, W = 1460B (1 PACKET) [13]

Number of Hops (H)	CSMA	MACAW
1	1838.4	971.7
2	921.3	485.8
3	614.8	323.9
4	461.4	242.9
5	369.2	194.3
6	307.7	161.9
7	263.4	138.8

In the second set of experiments (table II shows the result), they allowed the TCP window to grow up to 32KB. As the window is increased, multiple packets and multiple ACKs will travel on the path in opposite directions, creating interference and collisions. This was not the case with W = 1 packet. Because, there were no contention. Single packet and ACK took turns while using the channel. In table II, the simulation results show that CSMA throughput collapses when number of hopes (H) is greater than 2. Hidden terminal losses, which become substantial for longer paths, cause the loss of TCP ACKs and subsequent throughput degradation. On the other hand, MACAW does far better than CSMA with larger TCP windows. Comparing table 1 and table 2, it is evident that, throughput was consistently higher when window size was 1460B. Note that, the throughput tends to become constant as hop distances grow.

TABLE II: THROUGHPUT (KBIT/S), SINGLE TCP CONNECTION, VARIABLE NUMBER OF HOPS, W = 32KB [13]

Number of Hops (H)	CSMA	MACAW
1	1791.2	888.8
2	439.5	612.3
3	0.5	411.2
4	0.5	364.8
5	0.5	335.0
6	0.5	324.9
7	0.5	311.7

MACAW performs well due to the fact that, it acknowledges every frame sent and performs local retransmissions if an acknowledgement for a frame is not received.

From the above results, *Gerla et al.* [13] concluded that if link level ACK is absent (as is the case of CSMA), using larger window sizes (i.e. W > 1) or TCP connections covering multiple hops can prove to be a fatal mistake. On the other hand, using larger window, can be very effective on multiple hops when the link level ACK is present (as with MACAW).

In contrast to the above approach, *Rathke et al.* [10] took performance measurements in a real environment without making any assumptions about the wireless link. In our opinion, such an approach is very important in understanding TCP's behavior over WLANs.

For the purposes of this paper, we will only consider a part from the experiment conducted by *Rathke et al.* [10]. They have used ARLAN [12] as a wireless LAN with reliable MAC service (i.e. MAC of this WLAN includes some sort of error control mechanism). *Rathke et al.* [10] argues that, ARLAN is quite similar to 802.11. Hence, results obtained from using ARLAN can give a proper indication of what would happen in an 802.11 compatible WLAN. The WLAN used in their experiment had a 2 Mb/s bit rate and Carrier sense multiple access with collision avoidance (CSMA/CA) was the medium access mechanism. Direct-sequence spread spectrum (DSSS) operating at 2.4 GHz was the modulation technique. As shown in Fig. 2, the experimental environment consisted of a wireless end system that communicated with a fixed host via a base-station. The base-station and the fixed host were connected via a 10 Mb/s Ethernet. As stated earlier, WLAN technology used here is ARLAN [12]. The end-systems used here are PCs with Intel Pentium processor of 133 MHz and the TCP is implemented for Linux version 2.0.27.

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

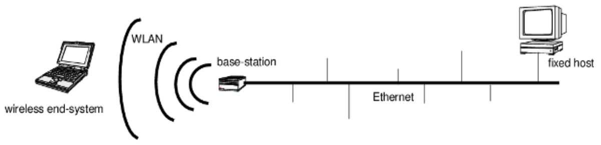

Fig. 2 General measurement configuration [10]

The measurement environment (Fig. 3) was treated as a single radio cell covering an area of 40x50 square meters that had many workstations in it. The base-station was positioned in room no.123.

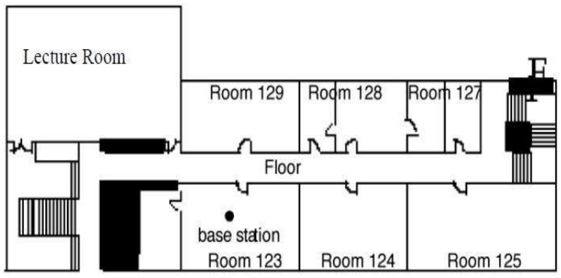

Fig. 3 Measurement environment [10].

The results from this experiment reiterate the fact that, WLANs have many location dependent characteristics and hence the mean TCP-throughput gained in such an environment also strongly relies upon the location. Using FTP as the application, the mean TCP-throughput inside ARLAN [12] varied between 0 KB/s and 120 KB/s.

Roughly, the locations inside the radio-cell were classified into three categories. A location was marked as "good" if it had a mean throughput greater than 100 KB/s (mainly rooms 123 to 125). Then there were the "bad" locations where mean throughput varies between 30 KB/s and 100 KB/s. (mainly rooms 127 to 129).
Lastly, the "unacceptable" locations, with mean throughput less than 30 KB/s. However, the results from this experiment had a twist! *Rathke et al.* [10] found that, "bad" positions were located all over the radio cell and in some cases near the base-station. They even found a "bad" position located just 2 meters from the base-station. One might argue to put another base-station in one of the "bad" position rooms (i.e. room 127 to 129). Unless different radio channels are used, this measure will cause overlapping of radio waves.
Therefore, *Rathke et al.* [10] came to the conclusion that, even with increased number of base-stations, the problem of mean TCP throughput dependence upon location would still remain.

V. TCP PERFORMANCE MEASUREMENTS OVER MOBILE NETWORKS

Conventional TCP was not designed with mobile networks in mind. Therefore, TCP performs poorly when compared with its performance in the traditional wired network. One of the main reasons for this poor performance is that, wired links have a much lower bit error rates (BER) than wireless links. When TCP encounters packet losses due to such bit errors, it triggers congestion control procedures (even if the packet losses are not due to congestion). These procedures cause a significant reduction in throughput and severely degrade network performance. *Eckhardt et al.* [16] showed that, BER of wireless links could reach up to 10^{-5}.

Elaarag et al. [17] ran a simulation using Network Simulator [NS] to study the effect of BER on Tahoe TCP. They used the network topology shown in Fig. 4.

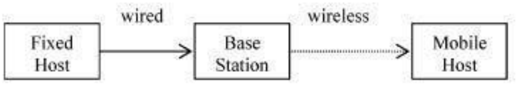

Fig. 4 Network topology [17].

It was a typical mobile network composed of a wired backbone part and a wireless part. The wireless part is geographically divided into cells, where each cell has a base station (BS) in it to provide an end-point connection to the roaming mobiles. The wired link had a BW of 1.5 Mb and a delay of 10 ms, while the wireless link had a BW of 0.8 Mb and a delay of 100 ms. Bulk data were transferred using ftp from the fixed host (FH) to the mobile host (MH). BS transfers the data between FH and MH. As in real environment, the wireless link suffered from high BER.

The Tahoe TCP window size was set to 50 while the slow-start threshold **(**ssthresh**)** was set to 32. They chose the packet sizes to be 1024 bytes and ACK size to be 40 bytes. *Elaarag et al.* [17] used DropTail queues (i.e. a packet is put onto the queue if the queue is shorter than its maximum size) at BS and MH. They chose the queue sizes to be equal to 50. They ran the simulation for 1000 seconds and the results are shown in table 3. Table 3 shows the performance of TCP with BER of 10^{-5} and 10^{-6}. Here, throughput is the number of packets MH receives per second. Success probability (an indication of network utilization) was calculated as the amount of useful data received by MH (sent from FH). Transfer time is the time needed to transfer 5000 packets (in seconds).

TABLE III: EFFECT OF BER ON PERFORMANCE OF TCP [17].

	BER = 10^{-5}	**BER =10^{-6}**
Throughput (pkts/sec)	39.439	87.455
Success Probability	0.9892	0.999
Transfer time	123.847	58.032

From Table III, it is evident that, BER difference by only one order of magnitude can significantly affect TCP's performance. Compared to the BER at 10^{-5}, TCP throughput is almost doubled when BER is reduced to 10^{-6}. Higher BER causes the sender to have smaller windows, which ultimately results in low throughput. The success probability also increases with lower BER. Because lower BER means that, fewer packets or ACKs get corrupted and retransmitted. This results in better network utilization. With Compared to the BER at 10^{-5}, TCP was able to send 5000 packets in almost half the time when BER was reduced to 10^{-6}.

Using (Wireless Transmission Control Protocol) WTCP [18], in the base station between the FH and MH, can lessen the effect of BER. The best feature of WTCP is that, it does not try to replace TCP. Instead, WTCP works with it to enhance TCP's performance in the wireless part of the mobile network. WTCP detects wireless-related problems (e.g. high BER) by using timeouts and duplicate

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
Vol. 4, Issue 8, August 2015

acknowledgments. WTCP then attempts to mitigate the problem by retransmitting a lost segment only once, until it receives an acknowledgment from the MH. Any other lost segments will have to wait in the WTCP's buffer until the first ACK has arrived. In order to prevent TCP from going into its congestion avoidance mode, WTCP hides the time spent by the packets at the WTCP proxy (i.e. BS), so that the RTT estimation is not affected.

VI. CONCLUSION

The focus of this paper has been on TCP's performance measurements in different operating environments. The measurements confirm that, TCP alone is not sufficient to maintain QoS in wireless and mobile environments. Researchers have suggested the use of MAC layer ACKs along with TCP to provide protection against collisions. In our opinion, the idea of using WTCP [18] looks to be the most promising. Because *Sinha et al.* [18] has successfully shown that, WTCP improves the performance by 20%-200% when compared to TCP algorithms such as New Reno, Vegas, and Snoop.

The only drawback that we notice in these performance measurements is that, most of them were done using simulators. Apart from the one by *Rathke et al.* [10], all of the data presented in this review paper were acquired using artificial assumptions about the operating environment. In our opinion, researchers need to carry out more surveys and take measurements from the real world (i.e. the Internet). Only then, we may get a clear picture about the challenges faced by TCP and subsequently find ways to mitigate those drawbacks.

REFERENCES

[1] Geoff Huston, "TCP Performance", The Internet Protocol Journal, vol. 3, no. 2, July. 2000.
[2] Steve Thompson. (2006) The cold hard truth about TCP/IP performance over the WAN. [Online]. Available: http://cncmachinesinfo.blogspot.com/2006/11/cold-hard-truth-about-tcpip.html.
[3] Juniper Networks, Inc. (2005). Accelerating Application Performance across the WAN [White Paper]. Retrived from: http://www-935.ibm.com/services/in/gts/pdf/accelerating_application_performance_acros_the_wan_whitepaper.pdf.
[4] Behrouz, A. Forouzan and Sophia Chung Fegan, TCP/IP protocol suite, 3rd ed., NY: McGraw Hill, 2006.
[5] Jau Hsiung Huang, Chi-Wen Chen, "On performance measurements of TCP/IP and its device driver", in Proc. 17th Conference on Local Computer Networks, pp. 568 – 575, 1992, doi: 10.1109/LCN.1992.228142.
[6] Intel Corporation. (May, 2006). Optimizing WAN Performance for the Global Enterprise [White Paper]. Retrived from: http://book.itep.ru/depository/security/intrusions/optimizing-wan-performance.pdf
[7] Stanislav Shalunov. (2008) TCP over WAN Performance Tuning and Troubleshooting. [Online]. Available: https://johnsofteng.wordpress.com/2008/12/18/tcp-over-wan-performance-tuning-and-troubleshooting/.
[8] Mark Lewis. (2010) Its (still) not just about bandwidth. [Online]. Available: http://www.riverbed.com/blogs/-its-still-not-just-about-bandwidth-.html.
[9] Stanislav Shalunov, Richard Carlson, "Detecting Duplex Mismatch on Ethernet", in Proc. 6th International Workshop on Passive and Active Network Measurement, pp.135-148, 2005.
[10] B. Rathke, M. Schlager, and A. Wolisz, "Systematic measurement of tcp performance over wireless lans," in Technical Report, Technical University Berlin TKN01BR98, 1998.
[11] H Balakrishnan, V. Padmanabhan, S. Seshan, R. H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links", IEEE/ACM Transactions on Networking, vol. 5, Dec 1997, pp 756 - 769, doi: 10.1109/90.650137.
[12] Aironet PCM/CIA Adapter Card. Available at: http://www.aironet.com. [Accessed: 3rd December 2010].
[13] Mario Gerla, Ken Tang, Rajive Bagrodia, "TCP Performance in Wireless Multi-hop Networks", in Proc. Second IEEE Workshop Mobile Computer Systems and Applications, (WMCSA), 1999.
[14] X. Zeng, R. Bagrodia and M. Gerla, "GloMoSim: a Library for the Parallel Simulation of Large-scale Wireless Networks", in Proc. 12th Workshop Parallel and Distributed Simulations, pp. 154–161, 1998.
[15] V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", International Journal of Computer Networks & Communications (IJCNC), vol.2, no.2, March, 2010.
[16] David Eckhardt , Peter Steenkiste, "Measurement and Analysis of the Error Characteristics of an In-Building Wireless Network", in Proc. Applications, technologies, architectures, and protocols for computer communications (SIGCOMM), 1996.
[17] Hala Elaarag, "Improving TCP performance over mobile networks", ACM Computing Surveys (CSUR), vol. 34, issue. 3, September, 2002.
[18] Prasun Sinha , Thyagarajan Nandagopal , Raghupathy Sivakumar , Vaduvur Bharghavan, "WTCP: A Reliable Transport Protocol forWirelessWide-Area Networks", in Proc. of ACM Mobicom '99, 1999.

BIOGRAPHY

A. M. Anisul Huq received his M. Sc. (Tech.) degree in Mobile Computing - Services and Security from Aalto University, Helsinki, Finland. He obtained his B. Sc. (Engg.) in CSE degree from Shah Jalal University of Science & Technology, Bangladesh. He is currently working as a faculty member in the department of CS, American International University – Bangladesh (AIUB). He has previously worked for Aalto University and Nokia Solutions & Networks (NSN), Finland. His research interests are in TCP/IP performance, Routing scalability issues, Energy consumption in Mobile devices during Video Streaming, Green Data Center and P2P Networks.