TCP Performance in 2.5G/3G Networks, Autumn 2002

PETER SALIN
Helsinki University of Technology, Espoo

As the services of 2.5G and 3G wireless networks progressively are made available to public users, the amount of mobile clients accessing the Internet using TCP/IP is expected to increase drastically in the near future. TCP is however originally constructed for use in wired networks, which characteristics differ significantly from those of wireless networks. This paper presents and evaluates different techniques for optimizing TCP for 2.5G/3G wireless networks.

1. INTRODUCTION

The success of the circuit-switched 2G systems initiated an exponential growth in the number of wireless networks during the last decade, covering most populated areas of the industrialized world. Now, these networks will form the backbone of the new packet-switched 2.5G and 3G networks, extending the scope of the TCP/IP suite to include the wireless environment as well.

The characteristics of 2.5G/3G network links, however, differ considerably from those of links in wired networks, where the TCP/IP protocol suite traditionally operates. There are other solutions than TCP for transporting data in wireless networks, e.g. implementing a completely new transport protocol layered over IP or UDP. However, given the amount of knowledge possessed and research performed relating to the TCP protocol, it is hard to reject TCP when it comes to wireless environments. Although, there are arguments both for and against the suitability of TCP in wireless networks [19], the vast majority of solutions proposed and research carried out in this area involve the TCP protocol.

This paper presents some of the main techniques invented for improving the performance of TCP in wireless networks. The operation of each selected technique is described, the benefits as well as possible drawbacks are presented and if available, the opinions of other researchers regarding the technique are expressed.

The rest of the paper is organized as follows. Section 2 presents the main characteristics of wireless networks. Section 3 presents problems arising when using original TCP [22] in such networks. Section 4 describes a number of techniques for optimizing TCP performance in wireless networks. Finally, section 5 summarizes and presents the conclusion of this paper.

2. CHARACTERISTICS OF 2.5G/3G WIRELESS NETWORK LINKS

This section presents the characteristics of wireless links, the intention being to describe how these links varies from the links in wired networks. As the next section describes problems arising when using original TCP in wireless networks, the two sections combined should provide an understanding of why the TCP protocol needs optimizing to better fit the wireless environment.

The section is divided into four subsections, each presenting a particular 2.5G/3G network characteristic.

2.1 Data rate

The data rate of 2.5G systems is expected to lie around 10-20 kbps uplink and 10-40 kbps downlink, whereas the first 3G systems should have data rates at approximately 64 kbps uplink and 384 kbps downlink [11]. Obviously these rates are low when compared to modern wired networks. The data rate of a single connection over 2.5G/3G networks is expected to fluctuate as multiple users compete for the provided bandwidth. Due to the limited maximum bandwidth, this will affect data rates more than in wired networks. Both intra- and intersystem handovers can also cause quick changes to data rates, a phenomenon that can't be found in wired networks.

2.2 Packet loss rate

In traditional wired networks, congestion is the main reason for packets being dropped before reaching their destination. Packets are rarely dropped due to bit errors. Because of the nature of wireless transportation, the bit error rate (BER) is considerably higher in wireless networks, making bit errors a common reason for packet losses. In order to reduce end-to-end packet losses 2.5G/3G networks apply link-level error correction techniques. This drastically reduces end-to-end packet losses, but introduces more delay jitter (see section 2.4).

Author's contact information: jsalin@cc.hut.fi

Published in *Internet Protocols for Mobile Computing - Seminar on Internetworking, Autumn 2002*, by Helsinki University of Technology, Telecommunications Software and Multimedia Laboratory. The complete publication can be found at http://www.tml.hut.fi/Studies/T-110.551/2002/papers/December/index.html. Permission to make digital/hard copy of all or part of this material without fee for personal or classroom use provided that the copies are not made or distributed for profit or commercial advantage, a copyright/server notice, the title of the publication and the article, and its date appear.
ISBN 951-22-6272-X - ISSN 1455-9749 - HUT - TML - TML-C9

2.3 Delay

The delays in 2.5G/3G wireless networks can be several magnitudes greater than the corresponding delays in traditional wired networks. The delays in 2.5G/3G networks primarily stem from the extensive processing performed at the physical and link-layer elements of these networks as well as from the delays induced by radio access network transmissions.

2.4 Delay jitter

Along with the characteristics presented above, the round trip time variations, or delay jitter, have a noticeably different behavior compared to the corresponding jitter in wired networks. One cause for the greater fluctuation in round trip times in 2.5G/2G networks is the handovers that occur in these networks. As mentioned in section 2.2, link-layer error correction also increases delay jitter. There are several additional causes for variations in round trip times, e.g. network congestion or physical objects, such as tunnels, temporarily blocking radio access.

3. PROBLEMS ARISING FROM USING ORIGINAL TCP IN 2.5G/3G NETWORKS

This section presents the problems arising from using original TCP in the 2.5G/3G network environment. Along with the link characteristics presented in the previous section, this section aims to explain why optimizations to TCP are needed in order to improve its performance in 2.5G/3G networks.

As mentioned earlier, TCP originally was designed for wired networks, therfore its operation is based on optimizing data transportation over links in these networks. Section 2 showed that the characteristics of links in 2.5G/3G networks differ considerably from those of the wired networks. The main problem with using original TCP in 2.5G/3G networks is the following: When a timeout occurs, original TCP assumes that the problem is caused by congestion in the network, since almost all packet losses in wired networks are caused by congestion. As a consequence TCP acts accordingly, which means backing off and therefore reduce the congestion. However, in 2.5G/3G networks, congestion cannot be assumed to be the cause of TCP timeouts, e.g. handovers, high delay jitter or bandwidth oscillations can be the source of the problem as well, resulting in TCP adapting to changed conditions too slowly [19]. Next, we will consider each of the characteristics of wireless links introduced in the previous section and investigate how original TCP adapts to them.

The value of the *data rate* does not itself create any problems, but combined with the long delays of wireless networks the size of the TCP window needs to be large enough. Especially in 3G networks the value of the bandwidth-delay product (BDP) may be so high that a window larger than the maximum of 64 KB is needed to reach optimal performance. The quick changes in data rates caused by e.g. handovers can also cause non-optimal adaptation to the new rate by TCP.

Link-layer error correction schemes decrease the *packet loss rate* to a level not introducing problems for original TCP. However, the additional delay jitter produced as a side effect of these schemes can again leave TCP unable to adapt optimally to the changing circumstances.

The long *delays* in 2.5G/3G networks means that it takes more time for TCP to adapt to changes, but this is normal behavior, the real problem is the *delay jitter* causing unnecessary timeouts and retransmissions. Severe delay jitter can cause the throughput of TCP connections to decrease significantly.

4. TCP OPTIMIZATIONS FOR 2.5G/3G WIRELESS NETWORKS

This section presents techniques that can be used to optimize the performance of TCP over 2.5G/3G networks. The operation of each technique is described; advantages and possible drawbacks are presented. For some techniques the general opinion of researchers investigating the technique is presented.

The techniques are grouped into three categories, presented in separate subsections of this section. The first group contains techniques striving to improve TCP performance with optimizations below the transport layer, i.e. modifications are not done to the TCP protocol itself. The second group contains techniques modifying the actual TCP protocol in order for it to better adapt to the demands of 2.5G/3G networks. Finally, the third group holds other techniques for optimizing TCP for wireless networks, usually including an intermediate node.

4.1 Optimizations below the transport layer

This subsection contains optimizations made in layers below the transport level aiming at improving the performance of TCP in wireless networks.

4.1.1 *Link layer error correction.* Link layer error correction is part of the 2.5G/3G specifications, making it available throughout 2.5G/3G networks. A combination of ARQ (Automatic Repeat reQuest) and FEC (Forward Error Correction) is used to improve the end-to-end reliability of these networks. Several different ARQ/FEC schemes exist, different schemes might be used at separate locations. It should be noted that although ARQ/FEC improves the bit error rate considerably in 2.5G/3G networks, it does not guarantee reliable delivery, thus TCP is still needed to accomplish completely reliable end-to-end transmissions.

The advantage gained by using link layer error correction is obvious. Without link layer error correction the bit error rate on end-to-end connections would be considerably higher, introducing extensive amounts of TCP retransmissions. Since original TCP assumes that the cause of packet losses is congestion and backs off when encountering timeouts, packets lost in wireless networks due to bit errors could cause the TCP connection to starve. Error correction at the link layer reduces the bit error rates to the levels of wired networks, therefore removing the problem of TCP starvation. The disadvantage of link layer correction is delay jitter introduced by link layer retransmissions when packets are lost.

The advantages of link layer error correction easily outweighs the disadvantages, the technique not only improves the performance of TCP but the overall performance of 2.5G/3G networks as well. Consequently, link layer error correction is included in the 2.5G/3G specifications.

4.1.2 *Path MTU discovery.* The path maximum transfer unit (PMTU) is the size of the largest packet able to travel from sender to receiver over an IP network in one whole piece. Sending packets smaller than the PMTU imply a more header information per sent data byte, sending packets larger than the PMTU causes fragmentation overhead as well as retransmission of several fragments when if only one fragment is lost.

Path MTU Discovery [18] is a technique for finding out the PMTU, adapting the packet size to the PMTU in order to maximize data throughput. In order to find out the PMTU, packets are sent with progressively increasing sizes. Each IP packet has its 'Don't Fragment' -bit set, in which case routers encountering packets larger than they can send in one piece should discard the packet and send an ICMP message back to the sender, who then knows the PMTU. The advantage of Path MTU Discovery is the optimal header/data rate, enabling better data throughput. Problems arise e.g. when routers do not send ICMP messages back to the sender correctly (blackhole routers), solutions to this problem and others exist [16] and are widely used in PMTU Discovery implementations.

The Path MTU Discovery technique is in several papers recommended for use in wireless networks [19; 11; 7]. Studies indicating that the PMTU necessarily isn't the optimal packet size in wireless networks also exist [3], but further research is still needed.

4.1.3 *Compression.* By compressing TCP/IP packets, the same amount of data can be transferred using fewer bytes, thus improving throughput. Two kinds of compression exists; header compression and payload compression.

Header compression reduces header overhead by compressing the header and sending only changed parts of the header. The initial Van Jacobson header compression [12], does not perform well in environments where packet losses occur frequently since each lost packet causes the loss of all packets in the current TCP window. RFC2507 [8] addresses and fixes this particular problem, but neither RFC1144 nor RFC2507 compression performs well when the TCP Timestamp Option [14] is used, since constantly changing timestamp causes the header not to be compressed. The IETF Working Group ROHC (RObust Header Compression) works on developing header compression schemes performing well in wireless networks as well. Dawkin et al. [7] among others recommend using RFC2507 compression when not using Timestamp Options, using RFC1144 compression is only recommended when using reliable links.

Compressing the IP payload can reduce the amount of data to be transmitted over the connection. The only drawback of IP payload compression is the processing overhead introduced. Since increasing amounts of data is compressed already at the application layer, payload compression cannot further compress the data, hence extra processing overhead is sometimes introduced. However, this overhead is rather insignificant and IP payload compression is generally recommended in networks with low data rates, although no remarkable gains are to be expected.

4.2 TCP protocol optimizations

This subsection contains techniques for optimizing the TCP performance in 2.5G/3G networks by direct modifications to the TCP protocol.

4.2.1 *Initial window size.* The initial TCP window size specifies how many segments the initiator of a TCP connection sends before waiting for acknowledgements. Original TCP uses an initial window size of one segment. Using a larger initial window size reduces idle times in the beginning of transmissions, speeding up transmissions of small data amounts such as most email and webpage transmissions. Packet losses and poor performance in congested networks are possible disadvantages of larger initial window sizes [1]. Most recommendations suggest using an initial window of at least 2 segments, studies indicate that sizes of up to 4 segments using the algorithm specified in RFC2414 would be beneficial to use [21].

4.2.2 *Window scale option.* The maximum window size of TCP is limited to 64 KB by the specification of original TCP. In networks with high delays and sufficient bandwidth, i.e. the bandwidth delay product (BDP) is high enough, the amount of data traversing the network simultaneously when the throughput is optimal, might exceed this value. In 2.5G networks this will not be the case, but 3G network bandwidths along with long delays might in the future require larger window sizes in order to reach their maximum throughput.

The TCP window scale option [13] provides a solution to this problem. Upon initiation of the TCP connection a window scale factor is sent with the SYN segment. If the receiver supports window scaling, it will send its own scale factor in response. Both parts need to support window scaling in order for the technique to be used. Using the window scaling option the new maximum size of the TCP window is 2^{30} bits, more than 1 GB.

Although the window scaling option probably won't be needed in 2.5G/3G networks in the near future, the technique can already be taken into use as the cost of its use is minimal. When window scaling is not used, not a single extra bit is sent over the network, and when used it is included only in the two connection-initiating SYN-segments. Having no real disadvantages, the technique is in fact already included in most of the modern TCP implementations, as it is beneficial for all TCP connections with high BDP, e.g. networks where data rates reach gigabit values.

4.2.3 *Selective acknowledgements.* Original TCP acknowledgements contain the number of the next segment expected to arrive. In case one or more segment is lost, the receiver will include the number of the first lost segment in the acknowledgement sent upon reception of new packets. If the segments received after the lost segments, fit in the recipients window they will be accepted. The sender will, however, not know which packets arrived after the lost packet, and will either have to retransmit just one segment and then wait for the acknowledgement to find out which segment to send next, or retransmit several segments that actually might have reached the recipient already.

Selective acknowledgements [17] provide a solution to this problem. The technique is based on the TCP option mechanism. When a recipient of TCP data notice that segments have been lost (or possibly delayed), a normal acknowledgement is sent. Included in the TCP options of the acknowledgement is the selective acknowledgement. The selective acknowledgement contains the sequence numbers at the left and right edges of the blocks of segments missing. The maximum amount of missing blocks that can be specified is four (three, if the timestamp option is used). Using the selective acknowledgements received, the sender is often able to retransmit all missing segments at once.

Selective acknowledgements improve performance in environments where losses of bulks of segments are possible. In 2.5G/3G networks this phenomenon is possible, e.g. handovers or temporal loss of network connectivity caused by blocking objects can at times lead to the loss of several segments at once. Especially 3G networks, having higher bandwidth, are exposed to the problem.

Although the selective acknowledgements are known to improve TCP performance only slightly in wireless networks, it has no serious disadvantages, except for the complexity involved in the implementation. In consequence, several papers recommend using selective acknowledgements [11; 19; 20].

4.2.4 *Timestamps option.* Original TCP uses one round trip time (RTT) sample per window of data to update its retransmission timer. In traditional wired networks, with short delays and small variations in round trip times, this method is sufficient, but in 2.5G/3G networks, where delays are long and considerable variations in round trip times occur frequently, the described scheme might not be sufficient enough.

TCP timestamps option [14] attempts to improve RTT estimations in order to make TCP respond more quickly to fluctuations in RTTs. Using TCP options, a timestamp is incorporated into each sent TCP packet. Upon reception of a packet, the receiver includes the timestamp contained the packet in the corresponding acknowledgement, i.e. the receiver effectively echoes the received timestamp back to the sender. The sender can then calculate the RTT by comparing timestamp values of received acknowledgements to the current time. Since timestamps are included with each sent packet, the parties are able to update their retransmission timers more frequently than when using original TCP, which should reduce the amount of unnecessary retransmissions. Both parties are required to support the timestamps option, otherwise it cannot be used.

There are some disadvantages involved in using the timestamps option. First, extra header overhead is created as 12 additional bytes are included in each sent packet. The additional bytes do hardly affect the data rate of high speed connections, but might have a small effect on slow speed networks, such as 2.5G networks. Second, using the timestamps option effectively makes header compression useless, as described in section 4.1.3. The ROHC (RObust Heading Compression) IETF Working Group is working on compression schemes enabling the use of the timestamps option.

Despite the disadvantages mentioned, research performed indicates that using the timestamps option is beneficial [10; 27] at least in networks with the characteristics of 2.5G/3G networks. Therefore its use is generally recommended when header compression is disabled.

4.2.5 *Explicit congestion notification.* No actions are taken upon congestion before packets have been lost in original TCP. If preventive actions would be taken before any packets are lost, the amount of TCP retransmissions could be reduced and overall TCP performance would increase. Explicit congestion notification (ECN) [23] tries to accomplish this.

Routers employ active queue management techniques in order to reduce the risk of congestion in the network. Random early discard (RED) is one such technique, randomly discarding packets arriving in burst in order to prevent congestion. Explicit congestion notification is an extension to RED, causing the routers to mark some of the IP packets instead of dropping them. TCP can then exploit these packets in order to prevent retransmissions. When a receiver of TCP data receives a marked packet, i.e. the 'Congestion Experienced' bit is set; this information is echoed to the sender using the reserved bits in the flag field of the TCP header. The sender reacts to the congestion notification by triggering the same congestion avoidance algorithm it uses when timeouts occurs, i.e. backing off for a while. This enables TCP connections to back off before packets are lost avoiding the need to retransmit packets normally lost before the congestion was noticed.

The greatest difficulty in applying this technique is the need of intermediate routers in the network to support ECN in order to achieve improved performance over end-to-end TCP connections. Otherwise ECN has no major disadvantages, the additional processing needed is insignificant and absolutely no extra traffic is generated. Researchers have found ECN to be useful [25], especially in low bandwidth networks with higher than normal probability of congestion.

4.3 Other optimizations

This subsection contains two techniques for optimizing TCP performance in 2.5G/3G networks using intermediate nodes. Numerous different implementations of both split TCP and performance enhancing proxies exist, here we try to explain the main principles of the techniques.

4.3.1 *Split TCP.* The endpoint of the majority of TCP connections initiated in 2.5G/3G networks will reside in the wired network; usually the target is a server of some content, e.g. webpages, multimedia or email. This implies connections will consist of two very contrasting parts. One part will traverse the wireless network, with long delay, low bandwidth, high delay jitter etc., as the other part will traverse the wired network, which links have completely different attributes.

Split TCP techniques strive to improve TCP operation by splitting connections in two, one TCP wireless network connection and another wired network connection. The connections are joined by an intermediate node located at the border between the wired and the wireless network. The advantage of this technique is obvious, the performance of the two connections can be optimized for their environment, i.e. techniques optimizing TCP in wireless networks can be used on one connection while other optimizations can be used on the other. The major drawback with split TCP is the fact that TCP end-to-end semantics are broken. This breaks e.g. the IPSec end-to-end security. In addition, if the intermediate node becomes unavailable, the connection breaks, whereas it normally only would have been rerouted over another path. Handovers in wireless networks could further complicate the operation of split TCP [19].

Several techniques using split TCP have been developed including I-TCP [2], Mowgli [15] and MTCP [26]. Due to the amount of complications originating from using split TCP, its use is currently not recommended by many, but in case those difficulties can be solved, there is a lot to gain using split TCP.

4.3.2 *Performance enhancing proxies.* By performance enhancing proxy (PEP) is usually meant any software residing in an intermediate node, aiming to improve the speed at which requested data arrives at the destination. Some PEPs try to optimize throughput by addressing the transport layer, while others direct their attention at the application layer (application level proxies), we will concentrate on the first alternative as application level proxies usually do not address TCP related issues.

PEPs can use a wide range of different mechanisms for improving TCP performance. For example, a PEP can locally acknowledge packets from the sender, cache them and then locally handle retransmissions occurring when packets are lost between the PEP and the destination, thus enabling faster loss recovery. Other mechanisms that can be used by PEPs include tunneling of messages, packet compression, split TCP, etc. [5].

Numerous PEP implementations exist, such as SNOOP [4], WTCP [24] and M-TCP [6]. Most of the implementations are relatively new and not thoroughly tested yet, improved TCP performance can be expected if research proves some of the implementations suitable for 2.5G/3G environments.

5. CONCLUSION

The introduction of 2.5G/3G capable mobile devices is expected to trigger an exponential growth in the amount of packet switched data traversing the 2.5G/3G networks. The most popular transport protocol of wired networks, TCP, was developed for use in wired networks and does therefore not perform optimally in 2.5G/3G networks. This paper presented some of several techniques aiming at improving the performance of TCP in environments similar to those of 2.5G/3G environments.

First, we presented the differences between wired and wireless networks, the increased delay and delay jitter being the major difference. Also the data rate and packet loss rate deviations where discussed. Then, the problems arising from using original TCP in 2.5G/3G environments were listed. Several of the problems stem from the fact that TCP interprets timeouts as congestion in the network. Finally, techniques for improving the performance of TCP were presented. Some techniques aim at improving TCP performance by modifications in layers below the transport layer, while others attempt to optimize TCP by direct modifications to the protocol. A few techniques using intermediate nodes to improve TCP performance were also introduced.

Most of the presented techniques could show advantages that easily outweighed the corresponding disadvantages. Consequently, the majority of the techniques presented in this paper are standardized and some of them, e.g. window scale option, selective acknowledgements and timestamps option, are already implemented in many current TCP stacks. In addition, link layer error correction was considered of such importance that 2.5G/3G specificators decide to include it directly in the specification. Further research is needed before split TCP and performance enhancing proxies can be deployed.

In conclusion, the presented techniques certainly improve the performance of TCP in 2.5G/3G networks. Further improvement is still needed, but research in this area is very extensive and new solutions are bound to arrive unceasingly. As of now, it seems quite likely that TCP will establish itself as the primary transport protocol in the 2.5G/3G wireless environment as well.

REFERENCES

[1] Allman, M., Floyd, S., and Partridge, C., Increasing TCP's Initial Window, IETF RFC 2414, September 1998.
URL: http://www.ietf.org/rfc/rfc2414.txt

[2] Bakre, A.,and Badrinath, B.R., I-TCP: Indirect TCP for Mobile Hosts *15th International Conference on Distributed Computing Systems*, May 1995.

[3] Bakshi, S.,Krishna, P.,Vaidya, N.,and Pradhan, D., Improving Performance of TCP over Wireless Networks. *17th International Conference on Distributed Computing Systems*, 1997,pp. 365-373.

[4] Balakrishnan, H.,Seshan, S.,Amir, E.,and Katz, R., Improving TCP/IP Performance over Wireless Networks *Proc. 1st ACM Conf on Mobile Computing and Networking (Mobicom)*, November 1995.

[5] Border, J.,Kojo, M.,Griner, J.,Montenegro, G.,and Shelby, Z., Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations, IETF RFC 3135, June 2001.
URL: http://www.ietf.org/rfc/rfc3135.txt

[6] Brown, K.,and Singh, S., M-TCP: TCP for Mobile Cellular Networks *ACM Computer Communications Review Vol. 27(5)*, May 1997.

[7] Dawkins, S.,Montenegro, G.,Kojo, M., and Magret, V., End-to-end Performance Implications of Slow Links. IETF RFC 3150, July 2001.
URL: http://www.ietf.org/rfc/rfc3150.txt

[8] Degermark, M., Nordgren, B., and Pink, S., IP Header Compression, IETF RFC 2507, February 1999.
URL: http://www.ietf.org/rfc/rfc2507.txt

[9] Elaraag, H., Improving TCP Performance over Mobile Networks. *ACM Computing Surveys, Vol 34, No. 3*, September 2002, pp. 357-374.

[10] Gurtov, A., Making TCP Robust Against Delay Spikes. *University of Helsinki, Department of Computer Science, Series of Publications C, No. C-2001-53*, November 2001.

[11] Inamura, H.,Montenegro, G.,Ludwig, R., Gurtov, A., and Khafizov, F., TCP over Second (2.5G) and Third (3G) Generation Wireless Networks. Work in progress, IETF Internet Draft, 4.11.2002 [referred to 15.11.2002].
URL: http://www.ietf.org/internet-drafts/draft-ietf-pilc-2.5g3g-11.txt

[12] Jacobson, V., Compressing TCP/IP Headers for Low-Speed Serial Links, IETF RFC 1144, February 1990.
URL: http://www.ietf.org/rfc/rfc1144.txt

[13] Jacobson, V., and Braden, R., TCP Extensions for Long-Delay Paths, IETF RFC 1072, October 1988.
URL: http://www.ietf.org/rfc/rfc1072.txt

[14] Jacobson, V.,Braden, R., and Borman, D., TCP Extensions for High Performance, IETF RFC 1323, May 1992.
URL: http://www.ietf.org/rfc/rfc1323.txt

[15] Kojo, M.,Raatikainen, K.,and Alanko, T., Connecting Mobile Workstations to the Internet over a Digital Cellular Telephone Network. *Proc. Workshop on Mobile and Wireless Information Systems (MOBIDATA)*, November 1994, Available at: http://www.cs.helsinki.fi/research/mowgli/. Revised version published in Mobile Computing, pp. 253-270, Kluwer, 1996.

[16] Lahey, K., TCP Problems with Path MTU Discovery, IETF RFC 2923, September 2000.
URL: http://www.ietf.org/rfc/rfc2923.txt

[17] Mathis, M.,Mahdavi, J.,Floyd, S., and Romanow, A., TCP Selective Acknowledgement Options, IETF RFC 2018, October 1996.
URL: http://www.ietf.org/rfc/rfc2018.txt

[18] Mogul, J., and Deering, S., Path MTU Discovery, IETF RFC 1191, November 1990.
URL: http://www.ietf.org/rfc/rfc1191.txt

[19] Montenegro, G.,Dawkins, S.,Kojo, M.,Magret, V., and Vaidya, N., Long Thin Networks. IETF RFC 2757, January 2000.
URL: http://www.ietf.org/rfc/rfc2757.txt

[20] Montenegro, S.,Kojo, M.,Magret, V., and Vaidya, N., End-to-end Performance Implications of Links with Errors. IETF RFC 3155, August 2001.
URL: http://www.ietf.org/rfc/rfc3155.txt

[21] Poduri, K., and Nichols, K., Simulation Studies of Increased Initial TCP Window Size, IETF RFC 2415, September 1998.
URL: http://www.ietf.org/rfc/rfc2415.txt

[22] Postel, J., Tranmission Control Protocol, IETF RFC 793, September 1981.
URL: http://www.ietf.org/rfc/rfc793.txt

[23] Ramakrishnan, K., Floyd, S., and Black, D., The Addition of Explicit Congestion Notification (ECN) to IP. IETF RFC 3168, September 2001.
URL: http://www.ietf.org/rfc/rfc3168.txt

[24] Ratnam, K.,and Matta, I., WTCP: An Efficient Transmission Control Protocol for Networks with Wireless Links *Technical Report NU-CCS-97-11, Northeastern University*, July 1997.

[25] Salim Hadi, J., and Ahmed, U., Performance Evaluation of Explicit Congestion Notification (ECN) in IP Networks. IETF RFC 2884, July 2000.
URL: http://www.ietf.org/rfc/rfc2884.txt

[26] Yavatkar, R.,and Bhagawat, N., Improving End-to-End Performance of TCP over Mobile Internetworks, *Proc. Workshop on Mobile Computing Systems and Applications*, IEEE Computer Society Press, Los Alamitos, California, 1994.

[27] Yavuz, M., and Khafizov, F., TCP over Wireless Links with Variable Bandwidth. *Proc. of IEEE Vehicular Technology Conference*, September 2002.