Volume 2, Issue 4, April 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

TCP Performance Issues and Related Enhancement Schemes over Wireless Network Environment

Purvang Dalal
EC Department, D.D.University, India
pur_dalal@yahoo.com

Dr. K.S. Dasgupta
IIST, Trivendrum, India
ksd@iist.ac.in

Abstract—**Transmission Control Protocol (TCP), an important transport layer communication protocol, is typically tuned to perform well in traditional wired networks, where Bit Error Rate (BER) is low and congestion is the primary cause of packet loss. The emergence of various mobile access networks has prompted researchers to look for suitable modifications to TCP so as to make it survive in the wireless era as well. This article provides both, the problems that arise when TCP is used over wireless links and an in-depth survey of various TCP enhancements which are specifically targeted for these problems. The problem is first characterized by the behaviour of wireless links and the aspects of TCP operation that affect performance. Here the objective is to review the performance issues of TCP variations, when employed in the wireless networks. Finally, this document explores a categorized analysis of different existing solutions comparatively, as we all know that it is difficult to create a "one size fits all" TCP for wireless networks.**

Keywords—**TCP Congestion Control , Loss Differentiation Schemes , Intermittent Connectivity , Handoff , Bit Error Rate , Wireless Networks , Retransmission Timeout , Round Trip Time , Medium Access Control.**

I. INTRODUCTION

TCP [RFC793] is the most widely accepted reliable, connection-oriented, full-duplex, byte stream transport level protocol that is in use today. It is basically developed for the data communication over reliable network environments. There are millions of network applications that have already been built on top of TCP and will continue to be in the foreseeable future. With the invention of handheld devices and laptops, new environments such as mobile telephony and Wireless LANs are now becoming ubiquitous. As a result of these [1,2,4], there is a significant increase in the number of combined wired and wireless networks. We could expect significant usage of the commonly used TCP/IP protocol stack in these systems, considering massive amount of popular network applications deployed on the top of TCP. One considerable problem that has emerged and received attention in research is the performance of the TCP over a mobile, wireless link. Optimizing TCP performance over these networks would have a broad and major impact on the user perceived data application performance. However, it is important to improve its performance without any modification to the application interface provided by TCP on fixed hosts (FHs), as this is the only way by which mobile devices communicating on wireless links can seamlessly integrate with the rest of the wired Internet.

The main reason for this decrease in performance is the packet losses in wireless networks; caused frequently by several factors other than congestion such as noisy channels or fading radio signals, interference, host mobility and disconnection due to limited coverage ([1]-[3]). The current TCP mechanisms cannot distinguish between congestion and non-congestion packet losses, and therefore make unnecessary reduction in the transmission rate and cause severe performance degradation ([1],[3]-[5]).

The rest of the article is structured as follows; Section II briefly describes general TCP behaviour & recovery mechanism, where as in Section III, the problem with TCP over wireless network is characterized. Section IV deals with the schemes made to resolve the problem. At the end we conclude the article by providing comparison between all TCP enhancement schemes, proposed by the researchers and scope for further enhancement.

II. TCP BEHAVIOUR AND LOSS RECOVERY

One of the reasons for the widespread usage of TCP over the Internet (as well as intranets and extranets) is its inbuilt flow and congestion control algorithms and its end-to-end reliability. TCP assumes that packet losses are caused by congestion in a network and applies congestion control techniques in order to give the network a chance to recover. We briefly introduce them below for the sake of continuity.

A. TCP Congestion Control

The implementation of the congestion control scheme is intertwined with TCP's window based flow control scheme through the use of two sender-side state variables congestion window (cwnd) and the slow start threshold (ssthresh) both of which are measured in bytes. A TCP sender is never allowed to have more bytes outstanding than the minimum of the advertised window (by the receiver) and the congestion window. So, a TCP sender's load on the network is limited by flow control imposed by the receiver and implicitly by the congestion in the network [6]. TCP uses the sliding window

mechanism, as illustrated in fig.1 below; to accomplish reliable, in-order delivery and flow/congestion control.

The basis of TCP congestion control lies in the following algorithms: slow start, congestion avoidance, fast-retransmit and fast recovery. TCP-Tahoe implements the slow-start, congestion avoidance and fast-retransmit algorithms. TCP-Reno implementation modified the TCP sender logic to include the fast recovery algorithm [7].

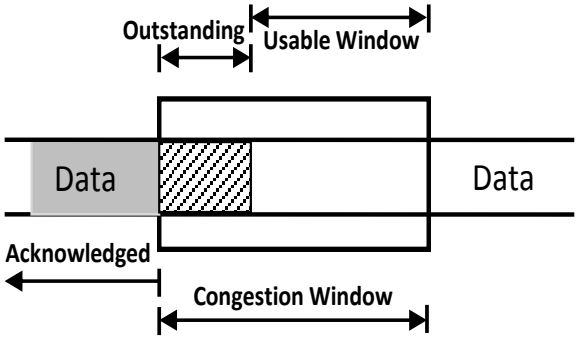

Fig. 1 TCP Window Management

B. TCP's Response to Packet Loss

A TCP packet, after transmission is determined to have been lost either if no acknowledgement (ACK) is received within the Retransmission Timeout (RTO) period or if multiple duplicate ACKs (DupACK) arrive for the packet prior to the one that was lost. TCP continually measures how long the acknowledgments take to return. It maintains a running average of this delay (Round Trip Time - RTT) and an estimate of the expected deviation in delay from the average (Delay Variation). These RTT measurements are collected and the RTO is set to the sum of the smoothed RTT (or approximate average) and four times its mean deviation. When a lost packet is determined by expiration of the RTO, TCP initiates an exponential back-off of the RTO and enters the slow start and congestion avoidance mode. The exponential back-off of the RTO involves doubling its value with successive failure of packet retransmissions. Then actions are taken so as to evade network congestion with the help of reduced transmission rate. It is important to note here that multiple lost packets will cause the slow start threshold to be repeatedly reduced, and thus the congestion avoidance mode will dominate and the packet transmission rate will grow very slowly. This can lead to degradations in throughput.

RTO estimation by TCP is also crucial for effective control of TCP's transmission rate, which in turn assures effective utilization of resources. If the RTO is excessive, retransmission will be unnecessarily delayed, resulting in slow recovery of network operation. If the RTO is too short, unnecessary retransmissions will occur and effective throughput will be decreased.

On the other hand, if a packet is determined to be lost by the reception of DupACKs, TCP begins fast retransmit and fast recovery. In this case, TCP is responding to what it believes is not congestion, so it does not wait for the RTO to expire before retransmitting the packet. The fast recovery algorithm then involves skipping over the slow start phase to avoid excessively decreasing the transmission rate. Instead, the congestion window is halved and congestion avoidance mode begins. Fast retransmit and fast recovery were introduced as an enhancement to the traditional congestion control, and they are helpful in retaining good performance after losses occur on a wireless link.

III. WIRELESS CHARACTERISTICS & RELATED TCP ISSUES

A wireless network is considered as a lossy network [3, 7] due to a) probable interference from other wireless stations present in the neighbourhood, which are operating in the same frequency band b) channel fading due to mobility of users and c) frequent disconnection of wireless links due to power crises and/or excessive mobility. In general, when compared to a wired link, a wireless link is more unreliable, with rapidly changing characteristics, and intermittent connectivity. This nature of the wireless channel means that some degradation to performance is inescapable. However, TCP's response to the wireless channel causes unnecessary additional degradation ([8]-[12]). As TCP's behaviour is dominated by congestion control, non-congestion packet losses are incorrectly interpreted by TCP as network congestion [7] and the subsequent actions taken to recover from the losses using standard congestion control mechanism results in suboptimal performance (refer to fig.2). (Although back off some times helps the network to recover). Since TCP's approach to error detection is based on mechanisms that only confirm that a packet is missing; the nature of the error is not detected and hence does not determine alternative recovery strategies.

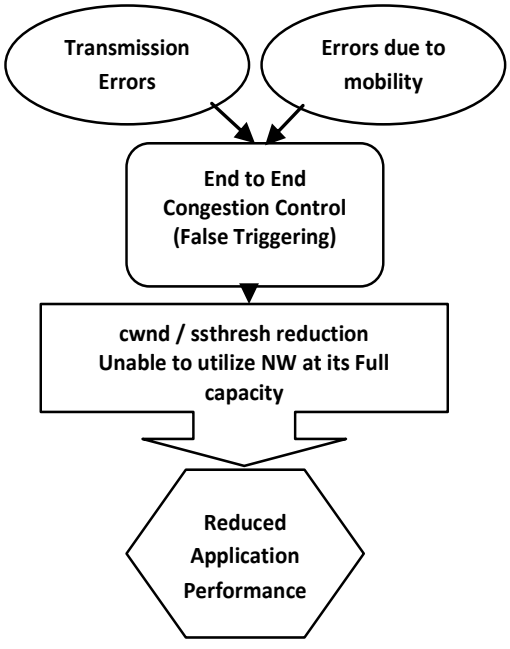

Fig. 2 False Triggering of TCP Congestion Control

In addition to above, the Medium Access Control (MAC) layer coordination function has the responsibility to retransmit the lost packet for a specified number of times for a packet loss. Unlike wired networks, when an 802.11 packet is lost, the 802.11 MAC layer is required

to retransmit the packet repeatedly [13][14], delaying the delivery of data packets. Sometime this delay may exceed the round trip time of the TCP endpoints. The acknowledgement delayed beyond the currently estimated value of RTT results into false packet loss detection. This will cause futile TCP retransmissions, responsible for pulling down TCP's efficiency.

All above factors cumulatively prevent TCP to operate over the network to its full capacity. It is observed that over 60% of network bandwidth remains unutilized [15] because of such reasons, while operating TCP over wireless networks. This degrades quality of service of the applications built on the top of TCP. This has lead to a lot of research to mitigate the performance problem of TCP related to the wireless environment.

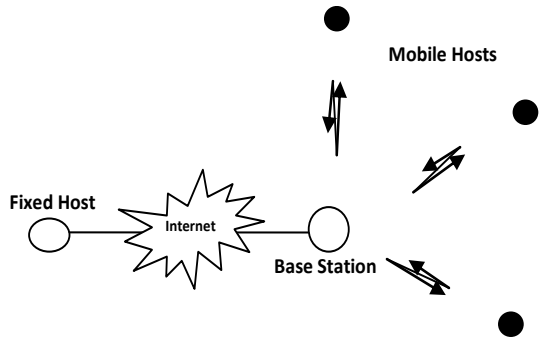

Fig. 3 Wireless Network Scenario for Simulation

To illustrate the effect of above issues on TCP performance, we ran simulations in NS-2 [16]. We used the network topology shown in Fig. 3, in which all the wired links have a bandwidth of 10 Mbps and propagation delay of 20 ms, whereas wireless links have a bandwidth of 2 Mbps and a propagation delay of 5 ms.

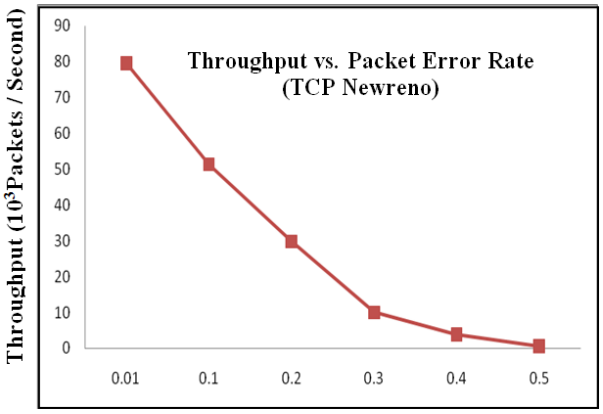

Fig. 4 TCP throughput with PER variations

Fig. 4 shows the throughput performance of TCP Newreno version versus the packet error rate in wireless links. It can be noted that as the BER is decreased by 10 percent, throughput almost doubles. Higher BER, along with mobility and limited spectrum, results in a smaller window which in turn results in very low throughput. We have noticed that at BER = .02, TCP was able to send about 29,000 packets on an average, whereas at BER = .5 it was able to send only 519 packets on an average in 400 s.

IV. PROPOSED SOLUTIONS

In order to enhance its performance, TCP has been modified several times to improve its performance and, as a result, several important TCP versions have emerged, such as TCP Tahoe [7], TCP Reno [7], TCP Vegas [17], TCP New Reno [18], and TCP SACK [19]. However all these mechanisms and various versions do not work in the same manner when called to work in dissimilar wireless environments, such as satellite networks, last-hop wireless networks and mobile ad hoc networks. Significant efforts have been made in researching and developing techniques that would enhance TCP performance in the wireless portion of heterogeneous networks.

The available solutions for improving TCP performance over wireless environments are broadly categorized into four categories: a) Split Connection Schemes b) Explicit Notification Schemes c) End to End Estimation Schemes and d) Local recovery Schemes. In many cases, a cross-layer approach is taken where one layer must be aware of characteristics of another layer.

A. Split Connection Schemes

In this approach (solutions such as I-TCP [20], M-TCP [21], METP [22], etc.), an attempt is made to separate congestion control and flow control functionalities on wired links from that of wireless links.

Indirect TCP (I-TCP) splits an end-to-end connection between Mobile Host(MH) and a host on the fixed network (FH) into two separate connections one between the MH and its mobile support router (MSR) at the Base Station (BS) over the wireless medium and another between the MSR and the FH over the fixed network. This approach takes care of the fact that MHs have limited resources (e.g., memory, power supply, disk space, etc.) and moves much of the networking task to the BS from the MH. Thus, it allows the MH to run a simple transport protocol to communicate with the BS, which, in turn, uses an entire TCP/IP stack to communicate with the FH. All the packets intended for the MH are now received, buffered, and acknowledged by the BS. When the MH moves to another cell, the state of the connection along with buffered packets are transferred to the new BS (Indirection). This ensures that the MH can communicate while moving between cells. This splitting along with indirection resulted into the additional benefits:

- A separate transport protocol for the wireless link can support notification of events such as disconnections, moves, available bandwidth, etc to the higher layers that can be used by link aware and location aware mobile applications.
- The FH is completely unaware of the indirection and is not affected by the handoff.

Connection splitting certainly violates the end-to-end semantic of TCP Acknowledgment, which has a serious

implication from an applications' point of view. It may so happen that, after acknowledging a bulk of TCP packets and before transmitting to the MH, the BS crashes and loses all the packets. In this case, even if the BS recovers quickly, there is no way for the BS to transmit these packets to the MH [40]. These packets are permanently lost. Another problem with split connection involves handoff latency when a MH moves between cells. The old BS then needs to transfer all the states of the connection to the new BS. This state information can be fairly large and may cause significant delay [23]. Last but not the least is the question of scalability. The BS may be overwhelmed if it has to serve a large number of MHs. Thus, there is a need to develop smart buffering techniques at the BS[23]. On the other side, M-TCP is designed to work well in the presence of frequent disconnections and low bit-rate wireless links, while maintaining of end-to-end semantics. However it suffers from other drawbacks mentioned above.

B. Explicit Notification Schemes

In these schemes (solutions such as ECN [24], ELN [25], EBSN [26], ELN-ACK [27], etc.), the sender explicitly notified by the next router about the cause of packet loss, primarily to discriminate between congestion and wireless losses. By using this information, TCP is made aware of the fact that some of the losses occurred are not due to congestion and thus prevents TCP from invoking the congestion control algorithm each time it detects a loss. The intuition behind this approach is that the router knows better when congestion occurs and when it disappears, or the state of the link it is controlling.

Explicit Congestion Notification (ECN) is an extension proposed to Random Early Detection (RED). RED is an active queue management mechanism in routers, which detects congestion before the queue overflows and provides an indication of this congestion to the end nodes. A RED router signals incipient congestion to TCP by dropping packets probabilistically before the queue runs out of buffer space. Upon receipt of congestion notification, the TCP receiver informs the sender (in the subsequent acknowledgement) about incipient congestion, which in turn will trigger the congestion avoidance algorithm at the sender. On downside of this scheme, ECN requires support from both the router as well as the end hosts, i.e., the end host TCP stack needs to be modified.

On the other side, Explicit Bad State Notification (EBSN) proposes a mechanism to update the TCP timer at the source to prevent source form decreasing its congestion window, provided there is on congestion. EBSN would cause the previous timeouts to be cancelled and new timeouts put in place, based on existing estimate of round trip time and variance. Thus, the new timeout value is identical to the previous one. The EBSN approach does not interfere with actual round trip time or variance estimates, and at the same time prevents unnecessary and harmful timeouts from occurring. This prevents timeouts for packets that had already been put on the network before the wireless link encountered the bad state. Explicit Loss Notification (ELN) adds an ELN option to TCP *acks*. When a packet is dropped on the wireless networks, future cumulative acknowledgements corresponding to the lost packet are marked to identify that a non-congestion related loss has occurred. Upon receiving this information along with duplicate acknowledgements, the sender may perform retransmissions without invoking congestion- control procedures. Mobility Awareness Incorporated as TCP Enhancement (MAITE) is a scheme that uses Link layer messages to inform TCP of high BER and disconnection conditions.

However, these schemes may fail when the path between the FH and the MH is changed frequently. These schemes are also not useful whenever IPSec is enabled at the BS [28], preventing BS to modify TCP header.

C. End to End Connection Schemes

The Most of the schemes proposed for optimizing Transport Layer (TCP) over heterogeneous networks needs intermediaries such as mobility support station for flow control, due to which the end to end semantics of TCP are not maintained and problems like degradation in throughput are experienced. In some cases, even the word "reliable" attached with TCP is itself challenged [1]. The idea behind this scheme (solutions such as Freeze TCP [29], TCP-Probing [30], TCPW [31], JTCP [32], TCPW-A [33], TCP Veno [34], etc.) is that, instead of depending on intermediate nodes, the actual protocol is modified.

The end-to-end approaches do not require much modification to the existing protocols. These schemes do not depend on any special support from intermediate routers. The approach preferred by majority of the researchers, is to decouple congestion control from loss recovery of non congestion losses. In order to differentiate between packet losses, many researchers have proposed solutions based on either End to End Loss Differentiation Scheme (LDA) [35] or Cross Layer Loss Differentiation Schemes [36]. These schemes are developed to solve one of the following problems:

- Dealing with frequent and long disconnection (Freeze-TCP)
- Saving battery energy (TCP-Probing)
- Discriminating between congestion and wireless losses (TCPW, TCP-Veno, JTCP, and TCPW-A).

Freeze-TCP eliminates the effect of disconnection by allowing the sender to freeze the retransmission timers when the receiver advertises a zero window, thereby preventing sender timeout and invocation of congestion control procedures. The problem with Freeze-TCP is that the network stack needs to be aware of the host's mobility. The receiver must predict the impending disconnection within one RTT; otherwise, the performance will be same as standard TCP. TCP-Probing is well suited for battery powered MHs. It saves battery energy by reducing the amount of data transferred. It measures the congestion risk level and transmits only when the network is in good condition. It cannot prevent

the sender from entering Slow Start if the packets are lost due to high BER or disconnection and handoff.

Rest of the protocols tries to calculate the available share of the bandwidth and use this estimate to compute congestion window and slow start threshold. These protocols attempt to select a slow start threshold and congestion window which are consistence with the effective bandwidth used at the time congestion is experienced. These protocols (except JTCP) invoke the congestion control procedure several times if multiple losses occur in a window in the same RTT. If multiple losses occur in a window in the same RTT, JTCP treats it as one congestion event and reduces congestion window only once. Although these protocols continuously measure the bandwidth availability, they (except TCPW-A) do not allow the sender to speed up if extra bandwidth is available. One of the beauties of TCPW-A is that it employs an adaptive, end-to-end bandwidth estimation algorithm and allows the sender to speed up if extra bandwidth is available. The protocols fail when they estimate incorrect bandwidth, as they use RTT to estimate available share of bandwidth and RTT in wireless links is unpredictable

However, in attempt to recover quickly from the losses these schemes may trigger busty TCP transmissions; leading to the issues related to fairness and friendliness, with the competing traffic on the network.

D. Local Recovery Schemes

In this approach (solutions such as Snoop [37], DDA [38], SACK-Aware Snoop [39], SNACK-New Snoop (SNACK-NS) [40], etc.), reliable link-level protocols are implemented on the wireless link which perform local retransmissions to improve the reliability of communication independent of the higher-level protocols. The traditional Internet approach is to delegate issues such as congestion and error control to higher (end to end) layers, so as to avoid imposing the corresponding recovery overhead on all applications. While this is adequate for reliable wired links, in error prone wireless links local (link layer) error recovery can be faster and more adaptable to the link characteristics.

To minimize the effect of BER, techniques, such as forward error correction (FEC) for error control and automatic repeat request (ARQ) or a hybrid of the two; have been suggested. Error correction techniques deal with providing enough redundant information with each packet to enable the receiver not only to detect the error but also to correct it without requiring retransmission. On the other hand, ARQ techniques enable the receiver only to detect the error. It cannot correct the error and so request the transmitter to retransmit the packet locally; thereby limiting the response of TCP mostly to congestion losses. An example is the AIRMAIL protocol. It utilizes intermediate routers at BSs to minimize the effect of high BER of a wireless link by hiding the link from FH. The BS stores all the unacknowledged packets and retransmits when a packet drop is detected. This approach maintains a virtual end-to-end TCP semantic by allowing the MH to generate an ACK for the received packets, which is justified for performance reason. Since propagation delay of the wireless last hop is considerably shorter than end-to-end delay, this approach has immediate knowledge about the packet drops and can thus respond more quickly than higher layers.

Link-layer protocols traditionally work independently of the transport layer, and this can directly cause problems for TCP. Some ARQ schemes retransmit packets out of order, which can cause duplicate TCP ACKs, unnecessary invocation of fast retransmissions, and thus degraded throughput. The timeout value for local (link level) retransmissions is of major concern. Interaction between the link level retransmission timeouts and the transport-level timeouts for TCP can lead to degraded performance if care is not taken while selecting the timeout values. To resolve this problem, it is suggested in [43] that the link layer protocols be given knowledge of TCP. This could allow the link layer to block duplicate ACKs from TCP and avoid retransmissions being initiated by both layers. A TCP aware link layer protocol investigated in [43] is shown to give 10-30% higher throughput than one that works without knowledge of TCP. Although this approach enhances TCP performance, there is a serious problem regarding its usability. As network security is taken more and more seriously, encryption is likely to be adapted widely, and an approach which depends on the BS is bound to fail when the traffic is encrypted [44]. For instance, IPSec is becoming an integral part of IPv6. In such cases, the whole IP payload is encrypted, so that the intermediate nodes may not even know that the traffic being carried in the payload is TCP.

V. CONCLUSIONS

It is extremely difficult to do a comprehensive comparison of the schemes because each aims to solve a different problem (e.g., high BER, handoff, frequent disconnection, battery constraint, etc.) of the wireless link. We believe that these two features a) end-to-end TCP semantic and b) encrypted TCP payload must not be violated in any TCP enhancement scheme. A careful scrutiny of the protocols indicates that, if we are to design an efficient TCP for last-hop wireless networks, the desirable properties of the protocol must include the detection of the frequent disconnection and handoff events. The protocol must require minimum changes to existing protocols, both for simplicity and compatibility reasons. An approach that calls for modification to the protocol endpoint on an MH is preferable to one that requires modification to the routers and endpoint on a stationary host.

In Table 1 below; we have summarized TCP Schemes in terms of their ability to handle various wireless network issues.

TABLE I
COMPARISON OF TCP ENHANCEMENT SCHEMES

Wireless Issues	Split Connection	End To End Connection	Local Recovery	Explicit Notification
Mobility	Supported at the cost of high network latency	Not Supported	Not Supported	Not Supported

LDA	Supported	Supported	Supported	Supported
Encrypted Traffic	Not Handled	Handled	Not Handled	Handled
Scalability	Not Ensured	Supported	Not Ensured	Supported
Deployment	Easy	Difficult	Easy	Difficult
End to End Semantics	Not Maintained	Maintained	Maintained	Not Maintained

Despite extensive research, it may be noted that there is no standardized solution available for the above stated problems. To our knowledge, certain issues as listed below have remained unresolved.

- In spite of having correct differentiation between losses, the existing schemes are unable to adjust TCP's transmission rate to fill up network to its maximum capacity, before another loss event is experienced.
- These schemes are improving TCP's performance in presence of wireless losses; however, they have failed in protecting TCP's performance in presence of losses due to network congestion.
- The cross layer schemes frequently overwhelm the intermediate router due to its persistent transmissions, creating local congestion. Under this situation their conformity with fairness and friendliness is not always guaranteed.
- As described earlier, delay introduced by 802.11 link layer causes futile TCP retransmissions, which degrade TCP's efficiency and effective bandwidth utilization.
- In a network suffering from frequent disconnection losses, TCP transmissions should be triggered as soon as the link is re-established. However the issue is not attempted so far.

REFERENCES

[1] S. Schmid and R. Wattenhofer, "A TCP with guaranteed performance in networks with dynamic congestion and random wireless losses", In Proceedings of the 2nd Annual International Wireless Internet Conference (WICON'06), August 2006.

[2] S. Mascolo, C. Casetti, M. Gerla, M. Y. Sanadidi, and R. Wang, "TCP Westwood: Bandwidth estimation for enhanced transport over wireless links," Proceedings of the 7th Annual International Conference on Mobile Computing and Networking (ACM SIGMOBILE), pp. 287-297, 2001.

[3] A. Capone, L. Fratta, and F. Martignon, "Enhanced band- wisth estimation algorithms in TCP congestion control scheme," in Proceedings of the IFIP Conference on Network Control and Engineering of QoS, Security and Mobility, pp. 469-480, 2002.

[4] R. Roy, S. Das, A. Ghosh, and A. Mukherjee, "Modified TCP congestion control algorithm for throughput enhancement in wired-cum-wireless networks" In Pro- ceedings of the 18th Annual IEEE International Sympo- sium on Personal, Indoor and Mobile Radio Communi- cations (PIMRC), 2007.

[5] J. Lee, H. Cha, and R. Ha, "A two-phase TCP congestion control for reducing bias over heterogeneous networks," in Proceedings of International Conference on Information Networking, Convergence in Broadband and Mobile Networking, (ICOIN), pp. 9-108, 2005.

[6] P. Karn, C. Partridge, Improving Round-Trip Time Estimates in Reliable Transport Protocols, InProceedings of ACM SIGCOMM 88.

[7] K. Fall and S. Floyd, "Simulation-based Comparisons of Tahoe, Reno, and SACK TCP," Computer Commun. Rev., vol. 26, issue 3, July 1996, pp. 5–21.

[8] "Adaptation and Cross Layer Design in Wireless Networks" CRC Press , Francis and Taylor Group, 2008 Ed.

[9] B. Jothi Mohan and V.C. Ravichandran,"Cross Layer Optimization for congestion control in ADHOC Networks", Information and Technology Journal, 6(2), 283-287, 2007.

[10] P. Dalal, "Adaptive TCP: Enhancing Performance over Heterogeneous Networks". Proc. IEEE ICOICT 2009, February 2009.

[11] S. Floyd and T. Henderson, The New-Reno Modification to TCP's Fast Recovery Algorithm, RFC 2582, Apr. 1999.

[12] R. Caceres and L. Iftode, "Improving the Performance of Reliable Transport Protocol in Mobile Computing Environment," IEEE JSAC, vol.13, issue 5, June 1995, pp. 850–57.

[13] Rui Jiang , Vikram Gupta and Chinya Ravishankar," Interactions between TCP and the IEEE 802.11 MAC protocol", Proc. DARPA Information Survivability Conference and Exposition", DISCEX, IEEE 2003.

[14] W. Ding and A. Jamalipour, "A New Explicit Loss Notification with Acknowledgment for Wireless TCP," PIMRC2001, vol.1, pp. B-65 - B-69 San Diego, CA, Sept 30–Oct 3, 2001. 34 IEEE Communications Surveys & Tutorials • 3rd Quarter 2006

[15] C. Liu and R. Jain, "Approaches of Wireless TCP Enhancement and a New Proposal Based On Congestion Coherence," Proc. 36th Annu. Hawaii Int'l. Conf., Jan. 2003, pp. 307–16.

[16] Network Simulator NS-2 : http:// www.isi.edu/nsnam /ns

[17] L. Brakmo and L. Peterson, "TCP Vegas: End to End Congestion Avoidance on A Global Internet," IEEE JSAC, vol. 13, no. 8, Oct. 1995, pp. 1465–80.

[18] S. Floyd and T. Henderson, The New-Reno Modification to TCP's Fast Recovery Algorithm, RFC 2582, Apr. 1999.

[19] S. Floyd, "Issues of TCP with SACK," Tech. Report, Mar. 1996.

[20] A. Bakre and B. Badrinath, "I-TCP: Indirect TCP for Mobile Hosts," Proc. IEEE ICDCS '95, 1995, pp. 136–43.

[21] K. Brown, S.Singh, "M-TCP: TCP for Mobile Cellular Networks," ACM Comp. Commun. Rev., vol. 27, no. 5, 1997, pp. 756–769.

[22] Wang and S. K. Tripathi, "Mobile-End Transport Protocol: An Alternative to TCP/IP Over Wireless Links," IEEE INFOCOM, vol. 3, Mar. 1998, pp.1046–53.

[23] A. A. Hanbali, E. Altman, and P. Nain, "A Survey of TCP Over Ad Hoc Networks," IEEE Commun. Surveys and Tutorials, 3rd Quarter 2005, pp. 22–36.

[24] K. Ramakrishnan and S. Floyd, "A Proposal to Add Explicit Congestion Notification (ECN) to IP," RFC 2481, Jan. 1999.

[25] H. Balakrishnan et al., "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," ACM/IEEE Trans. Net., vol. 5, no. 6, Dec. 1997, pp. 756–769.

[26] B. S. Bakshi et al., "Improving Performance of TCP over Wireless Networks," ICDCS, 1997, pp. 365–73.

[27] W. Ding and A. Jamalipour, "A New Explicit Loss Notification with Acknowledgment for Wireless TCP," PIMRC2001, vol.1, pp. B-65 - B-69 San Diego, CA, Sept 30–Oct 3, 2001. 34 IEEE Communications Surveys & Tutorials • 3rd Quarter 2006

[28] Chonggang Wang and Kazem Sohraby, Mahmoud Daneshmand, and Yueming Hu, "A Survey of Transport Protocols for Wireless Sensor Networks", IEEE Network • May/June 2006

[29] T. Goff et al., "Freeze-TCP: A True End-to-End TCP Enhancement Mechanism for Mobile Environments," IEEE INFOCOM, vol. 3, Apr. 2000, pp. 1537–45.

[30] V. Tsaoussidis and H. Badr, "TCP-Probing: Towards an Error Control Schema With Energy and Throughput Performance Gains," Proc. 8th IEEE Int'l. Conf. Network Protocols, 2000, pp. 12–21.

[31] M. Gerla et al., "TCP Westwood: Congestion Window Control Using Bandwidth Estimation," IEEE GLOBECOM, vol. 3, San Antonio, Texas, USA, Nov. 25–29, 2001, pp. 1698–702.

[32] E. H. K. Wu and M. Z. Chen, "JTCP: Jitter-Based TCP for Heterogeneous Wireless Networks," IEEE JSAC, vol. 22, no. 4, May 2004, pp. 757–66.

[33] R. Wang et al.," TCP With Sender-Side Intelligence to Handle Dynamic, Large, Leaky Pipes," IEEE JSAC, vol. 23, no. 2, Feb. 2005.

[34] C. P. Fu and S. C. Liew, "TCP Veno: TCP Enhancement for Transmission Over Wireless Access Networks," IEEE JSAC, vol. 21, issue 2, Feb. 2003, pp. 216–28.

[35] Chang-Hyeon Lim, Ju-wook Jank , "An Adaptive End to End Loss Differentiation Scheme for TCP over Wired/Wireless Networks" IJCSN, Vol. 7 No.3 , March 2007.

[36] "Adaptation and Cross Layer Design in Wireless Networks" CRC Press , Francis and Taylor Group, 2008 Ed.

[37] H. Balakrishnan, S. Seshan, and R. H. Katz, "Improving Reliable Transport and Handoff Performance in Cellular Wireless Networks," ACM Wireless Networks, vol. 1, no. 4, Nov. 1995, pp. 469–481.

[38] N. Vaidya and M. Mehta, "Delayed Duplicate Acknowledgments: A TCP-Unaware Approach to Improve Performance of TCP over Wireless," Texas A&M University, Tech. Report 99- 003, Feb. 1999.

[39] S. Vangala and M. Labrador, "The TCP SACK-Aware-Snoop Protocol for TCP over Wireless Networks," IEEE VTC, Orlando, FL, vol. 4, Oct. 2003, pp. 2624–283.

[40] F. Sun and V. L. Soung C. Liew, "Design of SNACK Mechanism for Wireless TCP with New Snoop," IEEE WCNC, vol. 5, no. 1, Mar. 2004 pp. 1046–51.

