International Journal of Data Science and Analysis
2017; 3(6): 69-76
http://www.sciencepublishinggroup.com/j/ijdsa
doi: 10.11648/j.ijdsa.20170306.12
ISSN: 2575-1883 (Print); ISSN: 2575-1891 (Online)

Review Article

Piggyback Scheme over TCP in Very High Speed Wireless LANs: Review

Ali Ahmad Milad*, Saad Mohamed Lafi, Mustafa Almahdi Algaet

Department of Computer, Faculty of Education, Elmergib University, Alkhums, Libya

Email address:

Ali106014@yahoo.com (A. A. Milad)

*Corresponding author

To cite this article:

Ali Ahmad Milad, Saad Mohamed Lafi, Mustafa Almahdi Algaet. Piggyback Scheme over TCP in Very High Speed Wireless LANs: Review. *International Journal of Data Science and Analysis*. Vol. 3, No. 6, 2017, pp. 69-76. doi: 10.11648/j.ijdsa.20170306.12

Received: September 5, 2017; **Accepted**: October 8, 2017; **Published**: November 23, 2017

Abstract: When the data packets transferred in duplex directions from part A to part B, and the data arrives at B, instead to sending a control frame from B to A, receiver B waits until the network layer at B send the next packet to A, and the Acknowledgment is attached in data frame from B to A using the field of Acknowledgment in the data frame header. so the acknowledgment got a free ride in the data frame, this technique known as piggybacking. One of the most advantages for piggyback scheme is improving the efficiency, which is reducing the overhead and increase system throughput. In this paper, we want to provide an overview of a research progress in piggyback scheme over wireless LANs. The research contributions are organised and summarized and it highlights the piggyback schemes that need to be investigated via high speed wireless LANs.

Keywords: Piggyback Scheme, Wireless LANs, IEEE802.11, TCP

1. Introduction

TCP is a generally used protocol on the Internet designed for reliable data transfer, when the receiving side has to acknowledge some received packets, it can be wait for the next data packet to be sent and put the acknowledgement information in the header, so does not have to send a separate TCP ACK packet. Piggyback mechanism reduces overhead and improves the performance [1], [2], but still need some improvements to get good system. In wireless LANs based on IEEE 802.11 becomes increase everywhere to support many applications using TCP, UDP, HDTV, and VOIP. In very high speed wireless LANs the physical layer rate may reach up to 600Mbps to get high efficiency at MAC layer. The idea for that when increasing the physical rate causes increasing transmission at MAC link and then cause increasing the overhead [3]. See figure 1. In IEEE802.11 the throughput does not scale well with increasing the physical rate. However, in IEEE802.11n the throughput achieves 100Mbps at MAC layer.

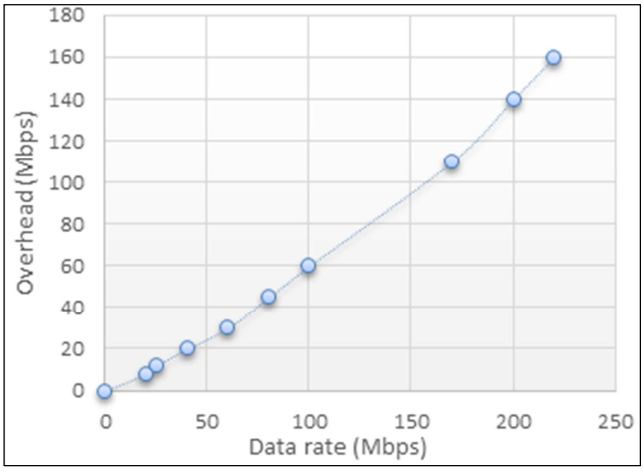

Figure 1. *Increase Overhead with Increase Data Rate.*

To improve the performance widely, we need to figure out the main problem at MAC inefficiency, see figure 2. The theoretic throughput higher limit and a theoretic delay lower limit exist for the IEEE 802.11 protocols. The existence of

such limits shows that by increasing the data rate without reducing overhead [4].

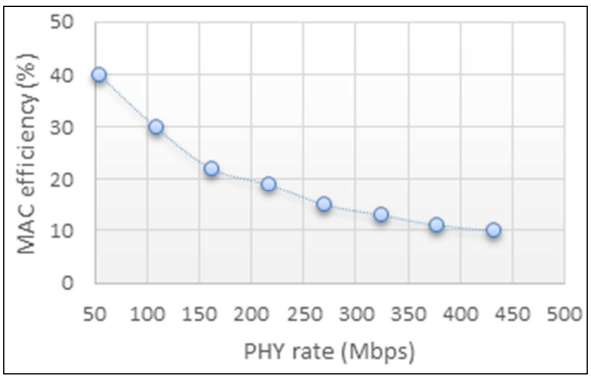

Figure 2. *MAC Inefficiency in Ideal Case.*

2. Background on Piggyback Scheme over TCP

In this section, we will discuss in exact details an overview of piggyback and its specific details and then we will overview the TCP and its functions, then we will overview of high speed wireless networks. Finally, we will discuss the challenges of TCP in high speed wireless LANs.

2.1. Overview of Piggyback Scheme

When the receiver station has frame to the sender station and allow sending data frame with ACK to the sender this process called piggyback scheme: so the acknowledgment gets free ride in the data frame and takes few bits, and this is a distinct ACK. So each frame needs ACK header and data frame (checksum). So the piggyback depends on the receiver, because fewer frames sent mean fewer frames arrival and this depends on how the receiver is organized. However, the piggyback scheme introduced some complication cases such as: how long the data link layer should wait until the packet which to piggyback the ACK? And we know the link layer wait for a time period, if the data link layer wait longer than sender timeout period the frame will be retransmitted, so the data link layer must wait for fixed time like Ad hoc scheme such as number of milliseconds. On the other hand one of the benefits of piggyback, when the receiver has a frame to send to the sender so the piggyback frame does not need to rivalry the channel again, because does not need to be in the front of the queue but the nearest frame to the destination at the sender [1], in same reference [1] shows the overhead with/without piggyback mechanism, if the receiver has a frame to send to the sender after receiving a frame, it needs to complete the channel again by at least a CTS frame time, an RTS frame time, two SIFS times, a DIFS time, and a random backoff. Otherwise, if the receiver can piggyback a frame to the sender with the acknowledgment, then the sender send ACK to acknowledge the piggybacked frame after SIFS time and the overhead is reduced already. See figure 3. The data frame can piggyback a control frame to increase the channel efficiency in a wireless networks: such as IEEE 802. 11 WLAN. However, the piggyback scheme may cause the decrease the channel efficiency and the increase of the frame transmission delay for other stations if has the low transmission rate and the control frame presents the global control information such as: the channel reservation time. So the piggyback problem as the low physical transmission rate, and evaluate the effect of this problem with respect to the average frame transmission delay and the channel utilization. So the authors propose the delay-based piggyback scheme to mitigate the piggyback problem [5], [6]. And they found that the piggyback decrease the channel efficiency and increase the frame transmission delay even if the presence of one station with low physical transmission rate.

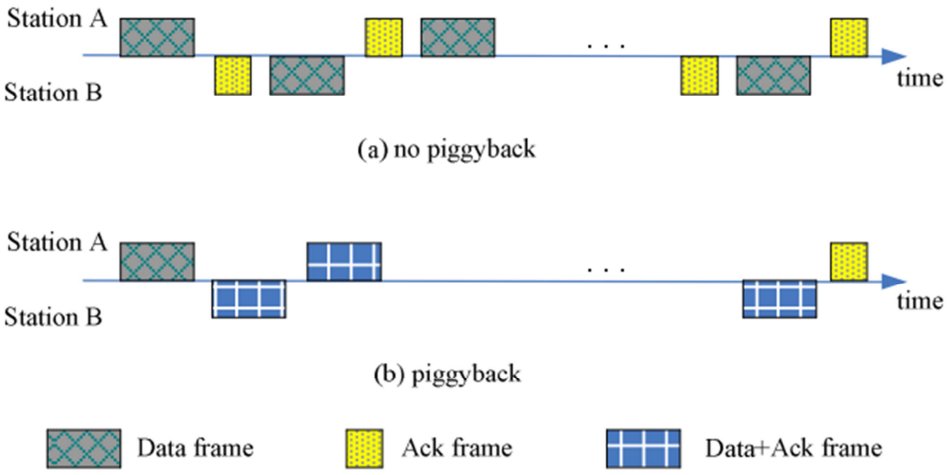

Figure 3. *The Overhead with/Without Piggyback Scheme.*

Tsern-Huei Lee et al [7] investigated that in piggyback scheme when the data frame transmitted there are two cases for the station. If the frame corrupted so whole the process will be restarts, or the data frame will be received successfully, showed also that the wireless station turn on piggyback scheme when the packet is less than 1100bytes, in additional the throughput is very low because the overhead, high data rate is better to piggyback, because the time missed in retransmitting the "data+ACK" frames is reduced. The throughput improvement by piggyback in the best case is

about 40%. Rastin Pries et al [8] the performance of piggyback requests is already evaluated, so the ratio of piggyback requests and the influence on the delay was Insistence for different traffic, with the large number of users the performance increased by using the piggyback requests, also the web traffic model showed that large number of bandwidths can piggybacked on previous packets. Hyun-Jin Lee et al [9] defined the piggyback problem that when the control frame is piggybacked the channel efficiency is decreased and in the same time the station has low physical rate, also showed that and evaluate the channel utilization with and without the piggyback frame, utilization of channel means the ratio of the total frame transmission time to the super frame length. So they solved the piggyback problem as the low physical rate by calculate the delay of piggyback scheme and propose the delay base piggyback scheme and found the piggyback decrease the channel efficiency and increase the frame transmission delay even in one station physical rate. However, when the physical transmission rate increases. The channel efficiency and the delay efficiency are increased also. Therefore, if piggyback practice well the channel efficiency increased and delay transmission increased. But the proposed algorithm decreases the average frame transmission delay and the channel utilization about 24% and 25%, respectively if there is one station which has low physical transmission rate. Jin Soo Park et al [10] showed that the piggyback mechanism reduces the delay for uplink and downlink packets, and the packets loss probability for uplink traffic and downlink traffic for the case of backoff method and piggyback method. For downlink data packets does not occur for piggyback method, and the loss packets of downlink data packet for the backoff is not small, also the piggyback method reduces energy consumption significantly and there is no losses of downlink data packets. Hyun-Jin Lee and Jae-Hyun Kim [11] investigated that many types of QoS data frames and their related usage rules to increase the channel efficiency. A CF-boll used to grant the channel to QSTA and piggybacked in QoS data frame to increase the channel efficiency. However, the channel efficiency may decreased by CF-boll piggyback problem when QSTA associated in QBSS use the low physical rate. The CF-boll piggyback scheme is various between 24 and 36Mbps depending on the traffic load. Jianhua He et al [12] proposed to study the impacts of channel access, bandwidth and piggyback scheme on the performance, and it is observed that the bandwidth utilization can be improved if the bandwidth for random channel access can be properly configured according to the channel access parameters, piggyback scheme and the traffic of network, Piggyback requests can be used to improve the bandwidth efficiency, but it is possible to increase the delay of channel access, The bandwidth efficiency with piggyback requests can be saturated more quickly with the increased number of SSs than that without piggyback requests [2].

2.2. Over View of TCP

Most of the performance issues in TCP/IP networks arise from various interactions between the TCP engine and the surrounding communication environment. In this part of review paper we are going to mention about TCP protocol fundamentals. TCP is a complex protocol. To understand the performance of TCP we have to mention its basic operations, in this section we are going to explain some of key features of TCP [13].

2.2.1. TCP Services

TCP provides several useful services to its applications. And we are going to describe briefly these services.

Connection oriented services. TCP is a connection-oriented protocol. Before two application processes can start sending data to each other, they must establish a TCP connection between them. If multiple application processes are running on a given IP host, each process is identified by a unique port number in that host, so each of them can establish a separate TCP connection. Each TCP connection is identified by a 4 tuple, source IP address, source TCP port number, destination IP address and destination TCP port number. The connection is terminated upon complication of the communication session [14].

Streaming Service. TCP provides a streaming service to its application. Once a TCP connection is established between two a sender and receiver, the sender writes a stream of bytes or characters into the connection and the receiver reads these bytes out of the connection. The stream oriented abstraction is visible only to the applications; the TCP layer operates on a packet mode. The sending TCP accumulates a certain amount of application bytes, forms a packet called a TCP segment, and sends the segment to receiver. The receiving TCP extracts application bytes from the segment, orders them if necessary and delivers them as a stream of bytes to the appropriate receiving application process.

Full Duplex Service. TCP is a full duplex protocol supporting data flow in both directions. That is mean once a TCP connection has been established between two applications, either process can send data to the other over the same connection at the same time.

Reliable Service. TCP guarantees delivery of every single byte, in order, without any duplication. To achieve reordering of any out of order arrival and to eliminate any duplicate delivery, the receiving TCP buffers the incoming data before delivering them to the application process. To guarantee the delivery of data, TCP uses the acknowledgment mechanism to check if the transmitted data have been received correctly by the receiver. If the underlying communication channel is noisy and error-prone, several retransmission of the same segment may be necessary for correct delivery of data to the receiver. For most data applications, such as file transfer and the World Wide Web, the reliability feature is extremely important as the applications do not have to worry about the lost or disordered data.

End to End Semantic. The reliability of TCP is based on the end-to-end semantic. Acknowledgments are generated only by the receiving TCP and only after the data are received correctly by the receiver. Therefore, when a TCP

sender receives an ACK, it is guaranteed that the data have reached the receiver safely. It is this end-to-end semantic that provides the ultimate reliability at the TCP layer. The end to end semantic would be violated if any intermediate node (not the TCP destination) generates ACKs on behalf of the destination.

2.2.2. Acknowledgement Mechanism

TCP provides an acknowledgment to confirm correct delivery from the receiver, some of TCP acknowledgments are:

Cumulative Acknowledgment. Each Acknowledgment is a confirmation that all bytes up to the ACK number have been received correctly.

Acknowledgment only segment and piggybacking. The ACK is indicated through an ACK field in TCP header. So, to acknowledge correctly received bytes, the receiver can also create ACK segment and this segment carries only the header containing the ACK number [15]. Or it can send the ACK in data segment. When an ACK travels in a data segment, this process call piggybacking [16].

Delayed Acknowledgment. TCP receiver has choice of either generating an ACK and receives a segment or delaying the ACK for a while. If delaying the ACK the receiver able to acknowledge two segments at one time and reduces the ACK traffic. However, delaying an ACK for long time cause a timeout and retransmission at the sender. The receiver should not delay ACK more than 500ms.

Duplicate ACK. When the segment gets lost in traffic, and the following segment arrives safely at the receiver. The receiver possible to receive data with a sequence number beyond the expected range. In this case the receiver buffers the incoming bytes and regenerates the ACK for the bytes received so far in sequence. Regeneration of the same ACK number causes duplicate ACK phenomenon. Because the sender receives the same ACK more the once. In the original case of TCP, the sender ignores the duplicate ACK [17], [18].

2.3. Overview of High Speed Wireless LANs

Wireless local area networks (WLANs) are becoming more popular and increasingly important. The IEEE 802.11 WLANs is accepted as a matching technology to high-speed IEEE802.3 (Ethernet) for portable and mobile devices. The reason for such success is that it keeps increasing data transmission rates while maintaining a relatively low price. The IEEE 802.11, 802.11b, and 802.11a/g specifications provide up to 2 Mb/s, 11 Mb/s, and 54 Mb/s data rates [19], [4]. Moreover, the IEEE 802.11Working Group is following IEEE 802.11n, a modification for higher throughput and higher speed enhancements. Different from the goal of IEEE 802.11a/.11b/.11g/.11e, for example, to provide higher speed data rates with different physical layer (PHY) specifications, IEEE 802.11n aims at higher throughput instead of higher data rates with PHY and medium access control (MAC) enhancements [20]. In IEEE802.11b [21], a physical layer standard for WLANs in 2.4 GHz radio band is specified. Three channels are supported and the maximum rate is 11 Mbps per channel. In IEEE802.11a [22], a physical layer standard for WLANs is specified. However, the radio band is limited in 5GHz, which means that 802.11a is not compatible with 802.11b/g. There are eight channels and the maximum physical layer rate is 54 Mbps per channel. In IEEE802.11g [23], a new physical layer standard for WLANs in the 2.4 GHz radio band is defined. There are three channels available in 802.11g, with a maximum rate of 54 Mbps per channel. The 802.11g standard supports OFDM (orthogonal frequency division multiplexing) modulation. For backward compatibility with.11b however, it also supports CCK (complementary code keying) modulation. The aim of IEEE802.11 technology is supported the multimedia applications such as DVD and HDTV 9.8Mbps and 20Mbps respectively. IEEE 802.11e [24] is a MAC layer extension to the 802.11 standard for Quality of Service (QoS) provision.

2.4. TCP Performance Issues over Wireless Links

In this section we discuss the TCP performance issues raised by the high transmission error rates in wireless links. We explain the fundamental problem caused by transmission errors.

2.4.1. Inappropriate Reduction of Congestion Window

Transmission errors are the primary source of performance problems for TCP applications in wireless network. While a few errors per packet may be corrected by low level FEC codes, more errors may lead to packet corruption. Corrupted packets are discarded without being handed over to TCP, which assumes that these that these packets were lost. Because TCP takes packet loss as sign of network congestion, it reacts to these losses by reducing its congestion window. In most cases, wireless transmission errors are not related to network congestion, thus, these inappropriate reductions of the congestion window lead to unnecessary throughput losses for TCP applications. The resulting throughput degradation can be very severs, depends on factors such as the distance between the sender and the receiver and the bandwidth of the communication path.

2.4.2. Throughput Loss in WLANs

The WaveLAN suffers from frame error rate (FER) of 1.55% with clustered losses when transmitting 1400 byte frames over an 85 foot distance [22]. Reducing the frame size by 300bytes halves the measured FER but causes framing overhead to consume a larger fraction of the bandwidth. In shared medium WLANs, forward TCP traffic (data) contends with reverse traffic (acknowledgments). In the waveLAN, this can lead to undetected collisions that significantly increase the FER visible to higher layers [25]. File transfer tests over a WaveLAN with a nominal bandwidth of 1.6Mbps achieved a throughput of only 1.25 Mbps [26]. This 22% throughput reduction caused by a FER of only 1.55% is caused by the frequent invocations of congestion control mechanism which repeatedly reduce TCP's transmission rate. If errors were uniformly distributed rather than clustered, throughput would increase to 1.51 Mbps. This is consistent

with other experiments showing that TCP performs worse with clustered losses, as multiple losses within the same transmission window may cause TCP to resort to (slow) timeout initiated recovery [27]. To illustrate the direction of TCP performance caused by wireless errors, table 1 shows TCP throughput over LAN path, consisting of a single WLAN, versus a WAN path, consisting of a single WLAN plus 15 wired links [28]. We show the throughput in the absence of any losses, the actual throughput achieved when the WLAN suffers from independent frame losses at a FER of 2.3% for 1400-byte frames, and the percentage of the nominal bandwidth that was achieved. In the WAN case, this percentage is half of that in the LAN case. Because TCP recovers from errors via end to end retransmission, recovery is shower in high delay paths. Table 2 shows the nominal bandwidth and actual TCP throughput measured over a single link path, using either an IEEE 802.11 WLAN or an IEEE 802.11b WLAN. The percentages here show that the high-speed link after each loss, it takes longer to reach the peak throughput supported by higher speed links.

Table 1. *TCP Throughput over LAN and WAN Connections.*

Connection	Nominal Bandwidth	Actual TCP Throughput	% Achieved
LAN	1.5 Mbps	0.70 Mbps	46.66
WAN	1.35 Mbps	0.31 Mbps	22.96

Table 2. *TCP Throughput over IEEE 802.11 LAN Connections.*

LAN Type	Nominal Bandwidth	Actual TCP Throughput	% Achieved
IEEE 802.11	2 Mbps	0.98 Mbps	49
IEEE 802.11b	11 Mbps	4.3 Mbps	39.1

2.4.3. Throughput Loss in Cellular Communication Systems

CC links in transparent (voice) mode suffer from a residual FER of 1% to 2%, after low level error recovery, despite their short frames [29]. For example, a full rate IS-95 link would segment a 1400 byte IP datagram into 68 frames. Assuming independent frame errors, the probability of a successful packet transmission is 50.49% at a FER of 1%. Frame errors are bursty than bit errors, because multiple frames are interleaved before transmission. Although this process reduces the loss rate and randomizes frame errors, thus avoiding audible speech degradation, it considerably increases processing delay because of interleaving before transmission and deinterleaving after reception. If we reduce the size of IP datagrams to reduce the packet loss probability, user data throughput also decreases because of the higher TCP/IP header overhead. TCP/IP header compression may be used over slow CC links, shrinking TCP/IP headers to 3 to 5 bytes [30]. Header compression, however, may adversely interact with TCP error recovery and link layer resets, leading to a loss of synchronization between the compressor and the decompressor, thus causing entire windows of TCP data to be dropped [31]. Although the RLP used in the nontransparent mode of GSM usually manages to recover from wireless losses before TCP timers expire, it exhibits high and widely and varying RTT values. Measurements using ping over a GSM network in San Francisco showed that 95% of the RTT values were around 600ms with a standard deviation equivalent to 20ms [32]. Our measurements with ping over GSM networks in Oulu, Helsinki, and Berlin produced similar results but with higher standard deviations. Large file transfer experiments, however, reveal that RTT can be occasionally much higher with real applications over operational networks, reaching values of up to 12 seconds. Increasing the size of the TCP Maximum Transfer Unit (MTU) not only reduces TCP/IP header overhead, thus improving bulk transfer throughput, but also increases the response time of interactive applications. For example: transmission of a 1500 byte IP datagram over GSM takes around 1.25seconds, which is unacceptable for interactive applications. Measurements over operational GSM networks show that TCP throughput is optimized for a MTU size of approximately 700 bytes (690 bytes in [33].

3. Approaches to Improve TCP Performance in Very High Speed WLANs

In this paragraph we are going to present some approaches have been proposed in the literature to improve TCP in wireless networks.

3.1. Splitting TCP Connections

Because end to end retransmissions are slow over longer paths, TCP connections can be split at the wireless gateways connected to both wireless and wired links. In this manner, when packets are corrupted by transmission errors over wireless links, they can be retransmitted over wireless part of the path only. Indirect TCP [34], also known as I-TCP, is a TCP enhancement scheme based on the split approach. In this scheme, a software agent at the wireless gateway intercepts TCP connection establishment message and transparently decomposes the end to end connection into separate TCP connections for the wired and wireless parts of the path. The agent bridges these connections by forwarding TCP packets between the two. The connection over the wireless part has a lower delay, leading to faster TCP retransmission, while the connection over the wired part remains unaware of wireless losses. TCP can also be replaced over the wireless part of the path by another transport protocol, providing improved error recovery [35]. The main drawback of the split approach is that it violates end to end

TCP semantics, because an acknowledgment originating from the wireless gateway may reach the sender before the corresponding data packet reaches its destination. If the gateway crashes after the acknowledgement has been returned to the sender, but before the data packet has reached its destination safely. Another issue with split schemes is that wireless gateways face significant overhead as packets must undergo TCP processing twice.

3.2. Snooping TCP at Base Stations

The snoop TCP scheme has the same objective as split TCP, that is, to confine retransmissions over the wireless part of the path only. This is achieved by snooping inside TCP connections so as to transparently retransmit corrupted packets without breaking end to end TCP semantics [36]. In this scheme a snoop agent maintains state for each TCP connection traversing the wireless gateway. TCP data packets sent from the wired to the wireless host are cached locally, until TCP acknowledgment from the wireless host verify that they were received. When duplicate acknowledgments arrive, indicating that the packet was lost, the packet is retransmitted by the agent from its local cache. The duplicate acknowledgments are then suppressed, that is, they are not propagated to the wired host, to avoid triggering end to end TCP transmissions and congestion control. The agent also uses local timers to detect losses when duplicate acknowledgments themselves are lost. The agent snoop inside TCP packet headers to gather the state information it needs to avoid generating its own control messages.

Snoop outperforms split TCP scheme without violating TCP semantics, because TCP itself remains unmodified. It also avoids conflicting local and TCP retransmissions [37] by suppressing duplicate TCP acknowledgments whenever it performs local error recovery. With Snoop, however, only the direction of transfer from the wired to the wireless host benefits from local error recovery, as the TCP receiver is implicitly expected to be located next to the wireless gateway. This is because Snoop relies on TCP acknowledgments to detect whether a packet was received or lost, which are returned very fast when the agent and the TCP receiver are on either side of the wireless link. As a result, snoop is most profitable when nearly all data flow from the wired toward the wireless host.

3.3. Notifying the Causes of Packet Loss

As we discussed earlier, the main reason for degraded TCP performance over wireless links is that TCP cannot determine whether a packet was lost because of transmission errors or because of congestion. Explicit Loss Notification (ELN) [38] is a scheme that enables TCP to distinguish between corruption and congestion induced losses, thus allowing the sender to properly react in each case. Whenever an agent at the wireless gateway, such as the Snoop agent, detects a noncongestion related loss, it sets an ELN bit in subsequent TCP headers and propagates it to the receiver, which echoes it back to the sender. The agent uses queue length information to heuristically distinguish congestion from wireless errors. When receiving an ELN notification, the TCP sender at the wireless host retransmits the lost packet without invoking congestion control. If a significant amount of data originates from the wireless host, as in interactive applications, the ELN scheme can considerably improve performance. This scheme works well in conjunction with Snoop TCP, because both schemes are required to perform local retransmission in both directions over the wireless link. However, because lost packets can only be retransmitted after a round trip time has elapsed when an acknowledgment with the ELN bit set is returned, error recovery is slow compared to Snoop TCP. Although ELN is applicable to most topologies; it requires modifications to the transport layers of remote wired hosts, in addition to the agents at wireless gateways.

3.4. Adding Selective Acknowledgments to TCP

When multiple packets are lost in the same transmission window, the sender can only infer the first packet that was lost from the duplicate acknowledgments returned. After retransmitting the lost packet, the sender must wait for new duplicate acknowledgments to be returned to detect the next lost packet. As a result, TCP can only recover from a single loss per RTT [27]. Wireless links may frequently corrupt multiple packets per window, leading to high error recovery delays, especially over high delay paths. The Selective Acknowledgment (SACK) option for TCP allows each acknowledgment to specify, in addition to the last packet received in sequence, up to three contiguous blocks of data that have been received beyond this packet [39]. The sender can, thus, infer which packets have been lost after the last packet acknowledged and retransmit them without waiting for additional duplicate acknowledgments. TCP with the SACK option may be used either end to end or only over the wireless part of a split TCP connection, significantly improving throughput in both cases. In the end to end case recovery remains quit slow over high delay paths, because the SACK option cannot speed up individual retransmissions.

3.5. Aggregation with Fragment Retransmission (AFR)

The traditional retransmission scheme, a whole frame is retransmitted even if only one bit is lost. This raises the question of whether it is possible to retransmit only the erroneous parts of a frame, if properly designed, such partial retransmission could be expected to improve performance. In AFR scheme, multiple packets are aggregated into and transmitted in a single large frame. If errors happen during the transmission, only the corrupted fragments of the large frame are retransmitted [3], [40].

4. Summary and Comparison of Enhancement Schemes

To assess the TCP enhancement schemes discussed here,

we must consider the following factors:

1. End to end semantic. A reliable transport protocol must provide true end to end semantics, that is, acknowledgments must absolutely certify that data packets have reached their destination safely. It is. Therefore, crucial for an enhancement scheme to preserve the end to end semantic of TCP [38].
2. IP payload access. Schemes that require the wireless gateway to access the payload of IP datagrams violate the layering principle. Furthermore, when IPSEC is used for secure communications, the IP payload is encrypted by the end hosts, thus, it is not visible to intermediate nodes.
3. Wireless gateway overhead. While TCP enhancement schemes may require cooperation from wireless gateway, they should keep the corresponding overhead to a minimum. Schemes that require state maintenance for each TCP connection do not scale well for large networks.
4. Ease of deployment. Schemes that require modifications to existing infrastructure, for example, wired servers and wireless gateways, are not easy to deploy [39].

Table 3. *Comparison of TCP Enhancements for Wireless Links.*

Enhancement scheme	End to end Semantic	IP Payload Access	Wireless Gateway Overhead	Ease of Deployment
Split TCP	NO	Yes	High	Not easy
Snoop TCP	Yes	Yes	High	Not easy
ELN	Yes	Yes	Low	Not easy
SACK	Yes	No	None	Easy

Table 3 compares the TCP enhancements discussed previously against these factors. With the exception of the SACK option, which is already being deployed but only provides minor improvements, no enhancement scores well in all areas. This implies that we must always make a trade-off between these factors. Which factor should take precedence over the rest depends on the networking scenario. For example, if a TCP enhancement solution is needed for a private financial transaction network, then modifications in existing infrastructure may not be serious because of high reliability requirements.

5. Conclusion

The piggyback scheme is one of the best techniques to improve the performance of TCP such as: reduce the overhead and increase the system throughput, this paper provides a huge survey, which summarize the problems in piggyback scheme over TCP in wireless networks, one of the main problems is the channel efficiency is decreased and in the same time the station has low physical rate. TCP enhancements have been made in the literature to improve the throughput system and losses. In summary, the existing piggyback scheme is not sufficient to provide increase the channel efficiency and decrease of the frame transmission delay. Therefore, designing a suitable piggyback scheme to improve the performance of TCP over very high speed wireless LANs still a task of future work.

References

[1] Y. Xiao, “IEEE 802.11 performance enhancement via concatenation and piggyback mechanisms,” *Wireless Communications, IEEE Transactions on,* vol. 4, no. 5, pp. 2182-2192, 2005.

[2] A. Ahmad Milad, M. Noh, Z. Azri, A. S. Shibghatullah, S. Sahib, R. Ahmad, and M. A. Algaet, “TRANSMISSION CONTROL PROTOCOL PERFORMANCE COMPARISON USING PIGGYBACK SCHEME IN WLANS,” *Journal of Computer Science*, pp. 1-5, 2013.

[3] T. Li, Q. Ni, D. Malone, D. Leith, Y. Xiao, and T. Turletti, “Aggregation with fragment retransmission for very high-speed WLANs,” *IEEE/ACM Transactions on Networking (TON),* vol. 17, no. 2, pp. 591-604, 2009.

[4] Y. Xiao, and J. Rosdahl, “Throughput and delay limits of IEEE 802.11,” *Communications Letters, IEEE,* vol. 6, no. 8, pp. 355-357, 2002.

[5] H.-J. Lee, J.-H. Kim, and S.-H. Cho, "A delay-based piggyback scheme in IEEE 802.11." pp. 447-451.

[6] A. A. Milad, M. Noh, Z. Azri, A. S. Shibghatullah, and M. A. Algaet, “Reverse Direction Transmission in Wireless Networks: Review,” *Middle-East Journal of Scientific Research*, pp. 767-778, 2013.

[7] T.-H. Lee, Y.-W. Kuo, Y.-W. Huang, and Y.-H. Liu, "To Piggyback or Not to Piggyback Acknowledgments?." pp. 1-5.

[8] R. Pries, D. Staehle, and D. Marsico, "Performance evaluation of piggyback requests in IEEE 802.16." pp. 1892-1896.

[9] H.-J. Lee, J.-H. Kim, and S. Cho, "A novel piggyback selection scheme in IEEE 802.11 e HCCA." pp. 4529-4534.

[10] J. S. Park, T. O. Kim, K. J. Kim, and B. D. Choi, "Performance Analysis of IEEE 802.15. 4 Non-Beacon Mode Where Downlink Data Packets Are Transmitted by Piggyback Method." pp. 1-6.

[11] H.-J. Lee, and J.-H. Kim, "A optimal CF-poll piggyback scheme in IEEE 802.11 e HCCA." pp. 6 pp.-1959.

[12] J. He, K. Yang, K. Guild, and H.-H. Chen, “On bandwidth request mechanism with piggyback in fixed IEEE 802.16 networks,” *Wireless Communications, IEEE Transactions on,* vol. 7, no. 12, pp. 5238-5243, 2008.

[13] K. R. Fall, and W. R. Stevens, *TCP/IP illustrated, volume 1: The protocols*: addison-Wesley, 2011.

[14] “IEEE Standard for Broadband over Power Line Networks: Medium Access Control and Physical Layer Specifications,” *IEEE Std 1901-2010*, pp. 1-1586, 2010.

[15] A. D. Rathnayaka, and V. M. Potdar, "Wireless sensor network transport protocol: A critical review," *Journal of Network and Computer Applications,* vol. 36, no. 1, pp. 134-146, 2013.

[16] A. Ahmad Milad, M. Noh, Z. Azri, A. S. Shibghatullah, A. Mustapha, and M. A. Algaet, "Reverse direction transmission using single data frame and multi data frames to improve the performance of MAC layer based on IEEE 802.11 n," *Science International-Lahore*, pp. 1861-1864, 2014.

[17] M. Hassan, and R. Jain, *High performance TCP/IP networking*: Prentice Hall, 2003.

[18] A. A. Milad, B. M. Noh, Z. Azri, A. S. Shibghatullah, and M. A. Algaet, "Design a novel reverse direction transmission using piggyback and piggyback with block ACK to improving the performance of MAC layer based on very high speed wireless lans." pp. 263-266.

[19] I. W. Group, "IEEE Standard for Information Technology–Telecommunications and Information Exchange between Systems–Local and Metropolitan Area Networks–Specific Requirements–Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments," *IEEE Std,* vol. 802, pp. 1-51, 2010.

[20] Y. Xiao, "IEEE 802.11 n: enhancements for higher throughput in wireless LANs," *Wireless Communications, IEEE,* vol. 12, no. 6, pp. 82-91, 2005.

[21] J. P. Pavon, and S. Choi, "Link adaptation strategy for IEEE 802.11 WLAN via received signal strength measurement." pp. 1108-1113.

[22] J. Maurer, T. Fugen, and W. Wiesbeck, "Physical layer simulations of IEEE802. 11a for vehicle-to-vehicle communications." pp. 1849-1853.

[23] A. V. Barbosa, M. F. Caetano, and J. Bordim, "The theoretical maximum throughput calculation for the IEEE 802.11 g standard," *International Journal of Computer Science and Network Security,* vol. 11, no. 4, pp. 136-143, 2011.

[24] J. Zhu, and A. O. Fapojuwo, "A new call admission control method for providing desired throughput and delay performance in IEEE802. 11e wireless LANs," *IEEE Transactions on Wireless Communications,* vol. 6, no. 2, 2007.

[25] G. T. Nguyen, R. H. Katz, B. Noble, and M. Satyanarayanan, "A trace-based approach for modeling wireless channel behavior." pp. 597-604.

[26] G. Xylomenos, and G. C. Polyzos, "TCP and UDP performance over a wireless LAN." pp. 439-446.

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

[28] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless links," *IEEE/ACM transactions on networking,* vol. 5, no. 6, pp. 756-769, 1997.

[29] P. Karn, "The Qualcomm CDMA Digital Cellular System."

[30] V. Jacobson, "Compressing TCP/IP headers for low-speed serial links," 1990.

[31] R. Ludwig, B. Rathonyi, A. Konrad, K. Oden, and A. Joseph, "Multi-layer tracing of TCP over a reliable wireless link." pp. 144-154.

[32] R. Ludwig, and B. Rathonyi, "Link layer enhancements for TCP/IP over GSM." pp. 415-422.

[33] R. Ludwig, A. Konrad, A. D. Joseph, and R. H. Katz, "Optimizing the end-to-end performance of reliable flows over wireless links," *Wireless Networks,* vol. 8, no. 2/3, pp. 289-299, 2002.

[34] A. V. Bakre, and B. Badrinath, "Implementation and performance evaluation of indirect TCP," *IEEE Transactions on computers,* vol. 46, no. 3, pp. 260-278, 1997.

[35] R. Yavatkar, and N. Bhagawat, "Improving end-to-end performance of TCP over mobile internetworks." pp. 146-152.

[36] H. Balakrishnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," *Wireless Networks,* vol. 1, no. 4, pp. 469-481, 1995.

[37] A. DeSimone, M. C. Chuah, and O.-C. Yue, "Throughput performance of transport-layer protocols over wireless LANs." pp. 542-549.

[38] H. Balakrishnan, and R. H. Katz, "Explicit loss notification and wireless web performance." pp. 1-5.

[39] M. Mathis, J. Mahdavi, S. Floyd, and A. Romanow, *TCP selective acknowledgment options,* 2070-1721, 1996.

[40] A. A. Milad, Z. A. B. M. Noh, A. S. Shibghatullah, M. A. Algaet, and A. Mustapha, "Design a New Bidirectional Transmission Protocol to Improve the Performance of MAC Layer Based on Very High Speed WLANs," *Journal of Computer Science,* vol. 11, no. 5, pp. 707, 2015.