17

Cross-Layer Error Recovery Optimization in WiFi Networks

Dzmitry Kliazovich, Nadhir Ben Halima, and Fabrizio Granelli

DIT, University of Trento, Via Sommarive 14, I-38050 Trento, Italy
klezovicdit.unitn.it, nadhirdit.unitn.it, granelli@dit.unitn.it

Summary. This paper presents a novel approach for cross-layer error control optimization in WiFi networks. The focus is on the reduction of the overhead deriving from the duplicate ARQ strategies employed at the link and transport layers. The proposed solution, called ARQ proxy, substitutes the transmission of a transport layer acknowledgement with a short request sent at the link layer. Specifically, TCP ACKs are generated based on in-transit traffic analysis and stored at the Access Point. Such TCP ACKs are released towards TCP sender upon a request from the mobile node, encapsulated into link layer acknowledgement frame. TCP ACK identification is computed at the Access Point as well as at the mobile node in a distributed way. ARQ proxy improves TCP throughput in the range of 25–100% depending on the TCP/IP datagram size used by the connection. Additional performance improvement is obtained due to RTT reduction and higher tolerance to wireless link errors.[1,2]

1 Introduction

Wireless Local Area Networks (WLAN) represented by IEEE 802.11 standard, often referred to as WiFi, provide mobile access to networks and services – omitting the requirement for a cable (and fixed) infrastructure, thus enabling fast and cost-effective network organization, deployment and maintenance.

As a drawback, the capacity offered by wireless links is relatively low as compared to wired networks. Such capacity limitations derive from the very physical nature of the wireless medium, characterized by limited bandwidth, time-varying behavior, interference, etc.

In particular, Bit Error Rate (BER) on wireless links ranges from 10^{-3} to 10^{-1} as opposed to 10^{-6} to 10^{-8} in wired links [1]. This difference of several

[1] This work is supported by the Italian Ministry for University and Research (MIUR) under grant "Wireless 802.16 Multi-antenna Mesh Networks (WOMEN)".

[2] The ARQ Proxy approach is currently patent-pending under EP 07425087.9 "Cross-Layer Error Recovery Optimization for 3G LTE Systems".

orders of magnitude results in poor performance of Transmission Control Protocol (TCP) [2] which accounts for over 95% of Internet traffic [3]. The reason for that is in TCP congestion control mechanism which treats all packet losses as congestion related and halves the outgoing rate for every loss detected.

In order to counteract such variation of error rates, IEEE 802.11 standard employs an Automatic Repeat Request (ARQ) at the link layer. Following a "stop-and-wait" approach, it does not allow the sender proceeding with next frame transmission until positive acknowledgement is received for the previous frame. Lack of positive acknowledgement triggers frame retransmission until a maximum number of retransmissions is exceeded.

However, the link layer is not the only layer which acknowledges packet delivery: TCP reliability is obtained through the utilization of a positive acknowledgement scheme, which specifies TCP receiver to acknowledge data successfully received from the sender. TCP header reserves special fields for enabling it to carry acknowledgement information. As a result, the TCP receiver can produce a TCP acknowledgment (TCP ACK) as standalone packet or, in case of bi-directional data exchange, encapsulate it into outgoing TCP segments.

Considering data transmission over an IEEE 802.11 link using the TCP/IP protocol stack (Fig. 1), whenever a TCP segment is transmitted over the wireless link, the sender first receives an acknowledgement at the link layer. Then, TCP entity at the receiver generates an acknowledgement at the transport layer. This acknowledgement represents ordinary payload data for the link layer, which should be acknowledged by the link layer protocol of the sender node. As a result, a single application data block is acknowledged three times: one at the transport level and two times at the link layer.

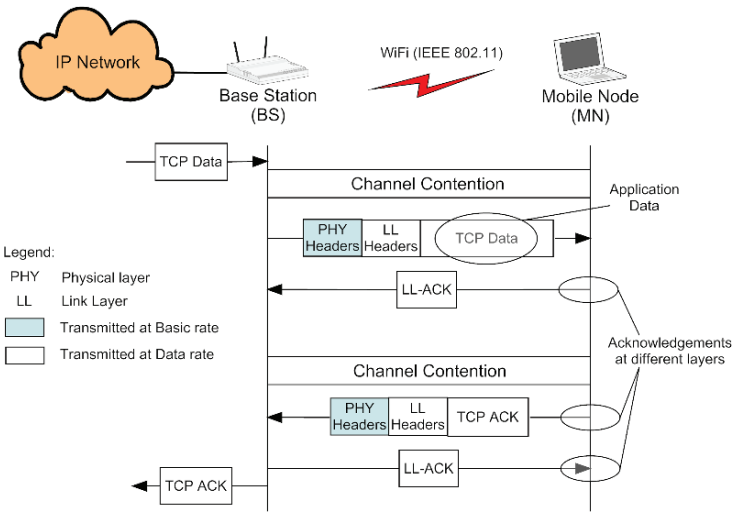


Fig. 1. TCP data packet delivery over IEEE 802.11 wireless link

In this paper, we propose a joint optimization of ARQ schemes operating at the transport and link layers using a cross-layer approach called ARQ proxy. The main idea behind ARQ proxy is to substitute the transmission of TCP ACK packet (including the associated physical and link layers overheads) with a small link layer request which is encapsulated into the link layer acknowledgement frame – which does not require any additional bandwidth resources. As a result, ARQ proxy releases network resources associated with TCP ACK transmission over the shared link, thus allowing the corresponding resources to be utilized for a concurrent data transmission originated at any station located within the cell.

The rest of the paper is organized as follows: Sect. 2 provides design and implementation details of ARQ proxy approach focusing on the infrastructure network scenario; Sect. 3 provides ARQ proxy performance evaluation in terms of TCP throughput and delay performance with the respect to TCP/IP datagram size and wireless link error rate; Sect. 4 concludes the paper with summary, conclusions, as well as directions for future work on the topic.

2 ARQ Proxy

ARQ proxy design is primarily focused on infrastructure network scenario – the most widely deployed WLAN scenario nowadays [7]. Implementation details in single-hop and multi-hop scenarios are discussed afterwards, as they have significant similarities.

The main idea of the proposed approach is to avoid the transmission of standalone TCP ACK packets over the radio channel on the link between the Base Station (BS) and Mobile Node (MN). In order to support this functionality, no changes are needed to the TCP protocol, but new software entities need to be introduced: the ARQ proxy and ARQ client (see Fig. 2).

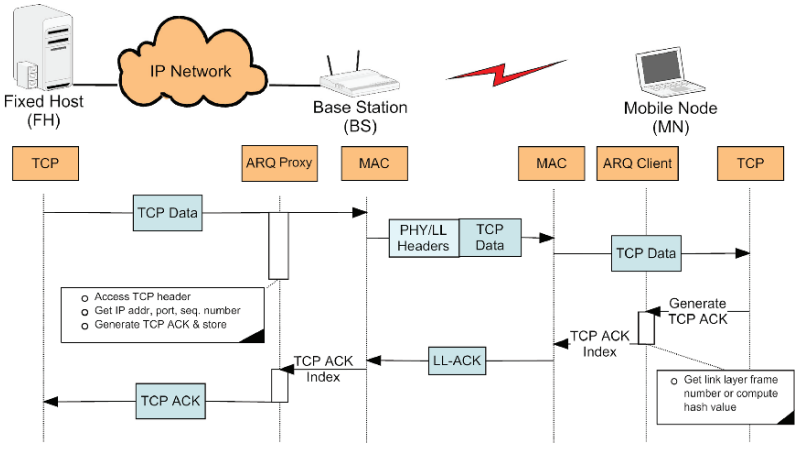


Fig. 2. ARQ proxy and ARQ client functionality

ARQ proxy is a software module located in the protocol stack of the wireless Base Station (BS) or Access Point (AP). Having access to TCP and IP headers of the in-transit traffic, ARQ proxy generates TCP ACK for every TCP data packet destined to MN which confirms successful data reception up to the flow segment carried in this TCP data packet. ARQ proxy does not require any flow-related state information or TCP layer implementation in a conventional sense. Indeed, TCP ACK is generated using a simple memory copy operation applied to the fields (IP addresses, port numbers, and flow sequence numbers) of the received TCP data packet into a previously generated template of TCP ACK.

The fact that no TCP flow state-related information is used in TCP ACK generation process implies the assumption that all the segments of a given TCP flow are successfully received at the destination node. Since this assumption is not always true, TCP ACKs generated by ARQ proxy module are not released to the Fixed Host (FH) immediately, but stored in BS memory until requested by the ARQ client.

Packet Identification. TCP ACKs generated by ARQ proxy should be easily identifiable by ARQ client without direct communication between these parties. There are two alternative approaches that satisfy this property: frame sequence numbers and hash values (see Fig. 3).

The IEEE 802.11 standard specifies that every sender needs to mark outgoing frames with continuously incremented, 12-bit long sequence numbers at the link layer. The reader should note that in case of TCP/IP datagram fragmentation at the link layer frame sequence number remains the same for all the fragments. As a result, ARQ client located at the MN can indirectly identify TCP ACK generated by ARQ proxy, by referring to the frame sequence number added by the BS at the link layer to the TCP data packet used in TCP ACK generation.

An alternate approach for packet identification that can be used in wireless network with no sequence numbers provided at the link layer is the use of hash values. In this way, TCP ACK is associated with a hash value computed by applying a proper hash function to TCP data packet headers for which the TCP ACK is generated. Traditionally, hash functions are used in

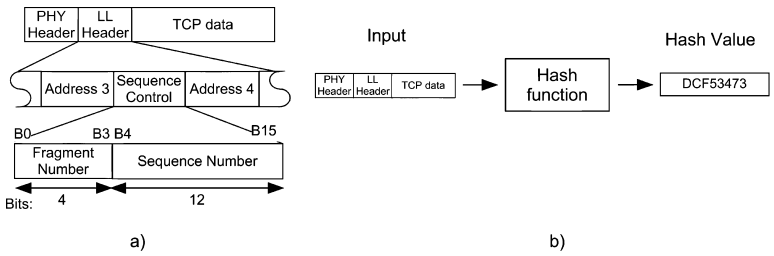


Fig. 3. Packet identification techniques: (**a**) frame sequence numbers and (**b**) hash values

cryptography, data storage and search applications. In networking, the use of hash functions is mostly limited to integrity check, error detection and error correction techniques – commonly performed using Cyclic Redundancy Check (CRC) or MD5 algorithms.

In this paper we limit our choice to frame sequence numbers due to simplicity, while for further details on hash functions the reader is directed to [4].

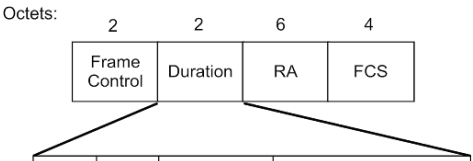
ARQ Client is a software module which logic position is between the link and transport layers of the MN protocol stack. It suppresses all outgoing standalone TCP ACK packets and replaces them with MAC layer requests for the appropriate TCP ACK transmission initiated at ARQ proxy.

In order to do so, whenever a standalone TCP ACK is produced at MN transport layer, a TCP ACK suppression request is scheduled for the transmission at the link layer immediately, while the original TCP ACK packet travels down the protocol stack which involves corresponding processing at each layer, output queuing delay, shared medium access and other procedures.

Whichever comes first to the physical layer (the TCP ACK or the corresponding suppression request) will be transmitted, while the other one cancelled.

TCP ACK suppression request includes identification associated with TCP ACK generated by ARQ proxy. This identification depends on the chosen packet identification technique: a frame sequence number or a hash value. At the link layer, TCP ACK identification is inserted into the next outgoing link layer acknowledgement (LL-ACK) frame. In particular, it is inserted into the reserved portion of the "duration" field of LL-ACK frame (see Fig. 4), which does not require modification of the frame structure specified by IEEE 802.11 standard.

The use of the reserved portion of LL-ACK frame favors incremental deployment of the proposed technique enabling operation in the mixed network environment where nodes which implement ARQ proxy co-exist with those not implementing the proposed approach.



Bit 15	Bit 14	Bit 13-0	Usage
0	0 - 32767		Duration
1	0	0	Fixed value during CFP
1	0	1-4096	TCP ACK index
1	0	4097-16383	Reserved
1	1	0	Reserved
1	1	1-2007	AID in PS-Poll frames
1	1	2008 - 16383	Reserved

Fig. 4. IEEE 802.11 ACK frame with extension capable to carry TCP ACK index

The lack of TCP flow related information at the BS allows ARQ proxy to generate TCP ACK which acknowledges only in-sequence segment delivery. For that reason, in order to maintain TCP error recovery procedure, ARQ client does not request TCP ACK generated at ARQ proxy in the following cases:

- During TCP connection establishment and connection termination phases, which are explicitly marked by SIN and FIN flags in the packet headers. These packets carry initial sequence numbers, maximum window sizes, and other parameters required by connection setup, and cannot be substituted.
- TCP ACK encapsulated into outgoing TCP data packet. In case of bidirectional data transfer and delayed-ACK option enabled, TCP receiver delays TCP ACKs assuming to have outgoing data segment in order to encapsulate TCP ACK using ACK bit and ACK sequence number fields into the packet header. Consequently, with such encapsulation, TCP ACKs do not create any additional overhead, and thus are not a subject for ARQ proxy optimization.
- Duplicate TCP ACKs. Upon out-of-order segment reception, TCP receiver must generate duplicate ACK for the last successfully received in-sequence segment. Due to the lack of TCP state related information at the BS, duplicate ACKs can not be generated by ARQ proxy and should be transmitted by MN.
- TCP ACK advertising exhausted buffer resources at the receiver (reported in *rwnd* field of TCP header). This ensures the receiver is not running out of the buffer space in case not being able to process traffic at the incoming link rate.

TCP ACKs generated at the BS are associated with a lifetime timer at the moment of generation. Upon expiration of this timer, which recommended value should be equal to or greater than TCP timeout, TCP ACK is silently dropped from the buffer. This lifetime technique is designed to clean up resources from TCP ACKs not requested by ARQ client module.

By the level of the achieved performance improvement the proposed technique is similar to LLE-TCP, proposed by the authors in [6]. However, ARQ proxy conceptually extends LLE-TCP by changing the point triggering TCP ACK generation. In fact, in infrastructure network scenario, LLE-TCP base station is completely responsible for TCP ACK generation with no feedback available from the receiver. On the contrary, in ARQ proxy approach, the generation of all TCP ACKs received at TCP sender is triggered by the receiver – following the end-to-end principle of Internet protocol design. Additionally, ARQ proxy avoids the need for storing TCP flow related information at the base station – unloading the hardware and enabling application of the technique in scenarios with high mobility.

Implementation of ARQ proxy approach in single hop ad hoc network scenario is the same as in infrastructure network scenario presented above. The only difference is that ARQ proxy module is located at the mobile sender

node and not at the BS, and TCP ACKs it produces are not routed through the network but immediately directed to the transport layer following ARQ client request sent by wireless receiver at the link layer.

In a multi-hop ad hoc network scenario, ARQ proxy technique can be applied at the last hop of multi-hop connection.

3 Performance Evaluation

In order to analyze the performance of the proposed scheme, the corresponding modules of the ns-2 network simulator (version 2.31) [5] are added supporting ARQ proxy and ARQ client functionality. ARQ proxy module is attached to the BS, while ARQ client is located in MN protocol stack. The configuration of the wireless link between BS and MN follows IEEE 802.11b specification parameters with $11\,\mathrm{Mb\,s^{-1}}$ physical data rate. Parameters of the wired link ($100\,\mathrm{Mb\,s^{-1}}$ 15 ms) model the situation when a mobile user is connecting to an Internet server physically located within the same metropolitan area. The BS ingress buffer is limited to 700 packets, and RTC/CTS exchange is turned off at the MAC layer as the most appropriate configuration widely used in infrastructure network scenario. Obtained results are averaged over 10 runs with different seeds used for random generator initialization.

TCP NewReno is chosen for performance evaluation as the most widespread TCP version in Internet nowadays. However, it is important to underline that ARQ proxy approach is not constrained to any specific TCP implementation.

Connection throughput and Round Trip Time (RTT) are chosen as main performance metrics of TCP flow evaluated against variable TCP/IP datagram size as well as Packet Error Rate (PER) on the wireless link.

Figure 5 shows the throughput level achieved by TCP NewReno for different TCP/IP datagram sizes. The throughput and level of performance

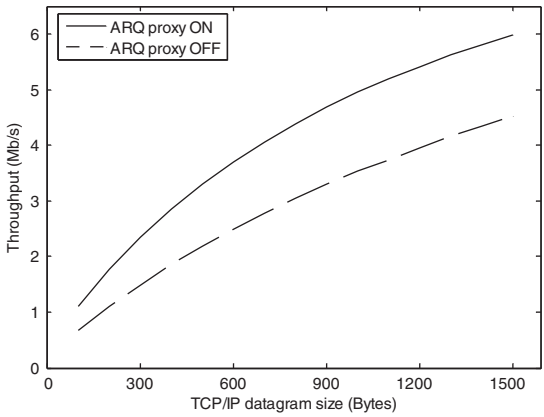


Fig. 5. ARQ proxy TCP throughput comparison

improvement of ARQ proxy approach is reversely proportional to the size of TCP data packet. Indeed, the smaller the packet the larger the resources released from TCP ACK substitution. For the maximum considered TCP/IP datagram size of 1,500 bytes (which corresponds to the Ethernet MTU), ARQ proxy performance improvement is only 25–30%. For small packets (40–200 bytes), it is in the range of 60–70%. However, the general rule is that for TCP data packets which tend to be similar in size to TCP ACK packets the throughput improvement can reach 100%.

Along with throughput performance improvement, ARQ proxy reduces RTT of TCP connection. TCP ACKs generated at the BS by ARQ proxy agent avoid transmission, propagation and queuing delays experienced at the wireless link. As it can be observed in Fig. 6, this delay is typically in the order of several milliseconds for IEEE 802.11b.

RTT reduction leads to TCP flow performance increase due to faster window evolution and faster reaction to packet drops performed by Additive Increase Multiplicative Decrease (AIMD) flow control mechanism [8].

Figure 7 illustrates TCP throughput with variable link error rate and TCP/IP datagram size equal to 1,500 bytes. While the throughput level is linearly decreasing, ARQ proxy performance improvement remains constant and corresponds to around 30% for PERs of up to 0.25. Additionally, by enabling ARQ proxy, TCP NewReno is able to sustain higher PERs (see Fig. 7 for PER > 0.25). This is motivated by the fact that in such scenario no wireless link errors propagate into TCP ACKs generated at the base station.

Summarizing, performance evaluation results validate the proposed approach and confirm ARQ proxy design initiatives. In details, ARQ Proxy provides:

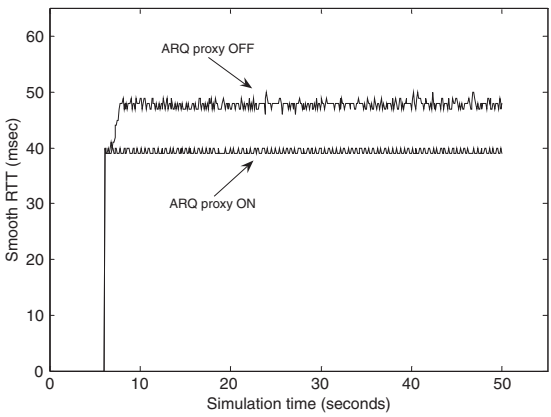


Fig. 6. ARQ proxy Round Trip Time (RTT) reduction

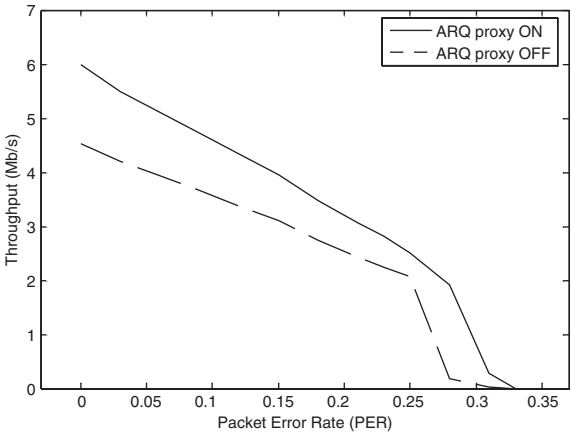


Fig. 7. TCP throughput against wireless link errors

- TCP throughput improvement from 25% to 100%, depending on TCP/IP datagram size
- RTT reduction for TCP ACK delivery over the wireless link (typically several milliseconds for IEEE 802.11b standard)
- Higher tolerance to link errors

4 Conclusions

The paper introduced a novel approach aiming at TCP performance improvement in WLAN networks. The core of the work lies in the substitution of the transmission of TCP ACK packets with a short link layer request sent over the radio link. Specifically, TCP ACKs are generated by an ARQ proxy located at the base station based on in-transit traffic analysis and stored in the buffer until requested by the ARQ client (located at the mobile node). All TCP ACK transmissions are triggered by the mobile end-node, which maintain end-to-end TCP semantics.

Two packet identification methods are considered for the proposed scheme: using frame sequence numbers available from the link layer (link layer dependant) and using hash values (link layer independent). In the latter case, hash values can be obtained by applying a predefined hash function onto the raw packet headers' data.

No TCP flow related information is stored at the BS, enabling applicability of the technique in a scenario with high mobility as well as incremental deployment in already operational networks.

Performance evaluation results demonstrate that ARQ proxy brings TCP throughput improvement in the range of 25–100% (depending on payload size), reduction of RTT of the connection for several milliseconds, as well as higher tolerance to errors on the wireless link.

Ongoing activities on ARQ proxy include application of the presented technique in 3G LTE and WiMAX (IEEE 802.16) environments.

Acknowledgement

The authors would like to thank Simone Redana and Nicola Riato from Siemens (Italy) for their contribution and valuable comments on the ARQ proxy approach.

References

1. W.C.Y. Lee, "Mobile Communications Design Fundamentals," 2nd Ed., Wiley, 1993.
2. J.B. Postel, "Transmission Control Protocol," RFC 793, September 1981.
3. C. Fraleigh, S. Moon, B. Lyles, C. Cotton, M. Khan, D. Moll, R. Rockell, T. Seely and S.C. Diot, "Packet-level traffic measurements from the Sprint IP backbone," IEEE Network, vol. 17(6), pp. 6–16, Nov.-Dec. 2003.
4. A. Partow, "General Purpose Hash Function Algorithms," http://www.partow.net/programming/hashfunctions/
5. NS-2 simulator tool home page, http://www.isi.edu/nsnam/ns/, 2000.
6. D. Kliazovich and F. Granelli, "A Cross-layer scheme for the TCP Performance Improvement in Wireless LANs," IEEE Global Communications Conference, GLOBECOM'04, Dallas, U.S.A., December 2004.
7. "Fixed Wireless, WiMax, and WiFi Market Opportunities, Strategies, and Forecasts, 2005 to 2010," WinterGreen Research, Inc., May 2004.
8. F. Kelly, "Mathematical Modelling of the Internet," B. Engquist and W. Schmid (eds.), "Mathematics Unlimited – 2001 and Beyond", Springer, Berlin Heidelberg Newyork, pp. 685–702, 2001.