RESEARCH ARTICLE OPEN ACCESS

Proposition of an Adaptive Retransmission Timeout for TCP in 802.11 Wireless Environments

Zouhair Bazzal, Abdel Mehsen Ahmad, Ibrahim El Bitar, Mostafa Rizk, Mohamad Raad

School of Engineering - Department of Computer and Communications Engineering Lebanese International University (LIU) Bekaa, Lebanon

ABSTRACT
The Transport Control Protocol (TCP) is used to establish and control a session between two endpoints. The problem is that in 802.11 wireless environments TCP always considers that the packet loss is caused by network congestion. However, in these networks packet loss are usually caused by the high bit error rate, and the wireless link failures. Researchers found out that TCP performance in wireless networks can be highly enhanced as long as it is feasible to identify the packet loss causes; hence appropriate measures can be dynamically applied during an established TCP session in order to adjust the session parameters. This paper proposes an end-to-end adaptive mechanism that allows the TCP session to dynamically adjust the RTO (Retransmission Timeout) of a TCP session; the server will have to adjust the timers based on feedbacks from clients. Feedbacks are piggybacked in the TCP Options header field of the ACK (Acknowledgment) messages. A feedback is an approximation of the time needed by the wireless channel to get the errors fixed. The mechanism has been validated using numerical analysis and simulations, and then compared to the original TCP protocol. Simulation results have shown better performance in terms of number of retransmissions at the server side due to the decrease in the number of timeouts; and thus lowest congestion on the wireless access point.
***Index Terms*:** IEEE 802.11, TCP, RTO, Congestion, DCF.

I. INTRODUCTION

TCP [1] is a protocol used to send data in the form of message units between computers over the Internet. It is currently the most widely used protocol for internet applications that require reliable data transfers over internet. Originally, TCP was designed to work in wired environments, where the congestion is the main cause of losses in packets since the Bit Error Rate (BER) in the wired links is very low. Each TCP segment is identified by a sequence number, and only correctly received in-order segments are acknowledged by the receiver endpoint. The failures are mainly caused by packet loss or out-of-order packets. To avoid these failures, the protocol uses flow and congestion control mechanisms [2].

Most popular Internet applications and protocols are based on TCP [3] for reliable data transfer. However, with the growing deployment of 802.11 wireless networks, it becomes crucial for TCP to provide high level of performance in both wired and wireless environments. Wireless networks suffer from major issues when compared to wired networks. Thus, when the end to end connection is not entirely wired, the TCP congestion control algorithm that depends on a window size must be well designed in order to calibrate the amount of data to be sent without being acknowledged. Currently, the congestion algorithm assume that timeouts are due to congestion on the end to end path, and not by transmission errors that might occur on the wireless segment. Wireless links suffer from high BER, shadowing, fading, and interference. Any packet loss on the wireless segment is considered by TCP as congestion, thus the TCP congestion algorithm running on the sender side will reduce the window size to few segments [2]; this results in poor throughput. In fact, in timeout situations over wireless networks, TCP often makes wrong decisions by reducing the bursts of segments since the window size is not well calibrated.

This paper aims to improve the performance of TCP when the path between two TCP endpoints is not entirely wired. It consists of an extended version of previous works published in [4] [5]. To sum up, we have added the following contributions:

- The proposition of an approach that adapts on the fly the retransmission timeout (RTO) implemented in the current TCP congestion algorithm.
- The use of the Options field of the TCP header in order to carry a feedback that consists of a delay value that allows the TCP sender to adjust the RTO dynamically.
- The extension of the analytical model proposed in [4], [5] in order to estimate the targeted delay value.

Simulation results have shown significant reduction in terms of number of retransmissions and timeouts at the TCP sender. This leads to less congestion on the wireless access points, less power consumption and processing/resources at the TCP endpoints.

The paper is organized as follows. In Section II, we overview the related works. Section III briefly introduces the IEEE 802.11 MAC protocol. Section IV describes the proposed approach. Section V illustrates the performance analysis based on numerical analysis and simulations. Finally, Section VI concludes the paper.

II. RELATED WORKS

Many TCP improvement schemes have been proposed to improve TCP performance over wireless networks. Many researches have dealt with the problems of TCP over wireless networks [6] in order to improve its performance and try to identify the causes of packet loss. The LT-TCP (Loss Tolerant TCP) in [7], [8] uses an adaptive end-to-end ARQ (Automatic repeat request)/FEC (Forward Error Correction) strategy for error correction; this strategy exploits the ECN (Explicit Congestion Notification) bits and works well in wireless networks with high error rates. The drawback is that ECN needs support from intermediate routers and thus the end-to-end concept of TCP is violated. In addition, when an ACK is lost, the retransmission that happens after the timeout at the server cannot be avoided. In [9], a hybrid version of FEC/ARQ for wireless links is implemented. The authors study the bandwidth tradeoff between FEC and TCP and conclude that the TCP performance varies as a function of the amount of FEC. The authors in [10] demonstrate that the overall TCP throughput can be improved by adding an end-to-end FEC to TCP. They use the receiver feedbacks to adjust the FEC parameters at the sender side. The approach proposed in [11] illustrates a new adaptive FEC scheme for TCP video streaming that optimizes the extra bandwidth usage for the redundancy level. All the above researches suffer from the fact that the generation of FEC correcting segments require extra processing activities at the stations.

The Indirect-TCP [12] requires that the connection between sender and receiver is divided into two segments: wired and wireless. The errors resulting within the wireless connection are corrected at the proxy base station & mobile support router and are not transferred to the wired network. This approach still have several drawbacks since it introduces an extra overhead as packets are processed twice: one time between the fixed host and the proxy and another time between the proxy and the mobile host.

WTCP (Wireless TCP) [13] uses a special mechanism in order to distinguish between error losses and congestion losses. WTCP cannot be implemented in a mixed wireless and wired network, and assume that all delay at base station is always due to queuing without taking into consideration the congestion on the wireless hop.

In [14] the authors integrates a Snoop agent between the wired and the wireless network. Upon reception of a packets from the wired network, the Snoop agent buffers it into an internal cache, then transmits it to the mobile host and wait for an ACK from the mobile host. When the ACK arrives, the agent checks the status of the packet, if it is in the cache, then the lost packet is forwarded to the destination without going back through the wired network; otherwise, the agent transmits the ACK to the wired source indicating a congestion issue. The proposed mechanism violates the end-to-end design principle of TCP.

In [15] the approach makes use of the 3 reserved bits in the TCP header. These bits will be used to inform the user about the type of the link over which the connection is established. In addition, the approach uses the SNR (Signal to Noise Ratio) to detect the type of errors. However, this approach faces major problems when the SNR varies drastically.

The objective of TCP-friendly [16] is to control the congestion when streaming real-time applications over wireless networks. The mechanism makes use of ECN bits to detect and distinguish between real congestion and wireless link errors. TCP-friendly improves the bandwidth usage and ensures smooth transmission rates; however, ECN needs special support from intermediate routers and thus the end-to-end concept of TCP is also violated.

So far, to the best of our knowledge and belief, there does not exist a TCP implementation which makes use of the options field of the TCP header to improve the performance of TCP over wireless channels. In our approach, we believe that it is crucial to rely on the 802.11 MAC layer to control the errors on the wireless link. This requires an additional time delay that should be approximated and sent back to the server. This delay can be piggybacked inside the Options field of the TCP ACK messages; thus the TCP server will introduce less timeouts and thus less retransmissions to perform and less congestion on the wireless access point. To do so, the server needs a new mechanism to estimate the RTO. Section IV provides a detailed overview of the proposed approach.

III. IEEE 802.11 MAC PROTOCOL BACKGROUND

Fig. 1 illustrates the DCF (Distribution Coordination Function) protocol of the IEEE 802.11 standard [17] which defines how the medium is

shared among 802.11 mobile nodes. DCF is based on Carrier Sense Medium Access with Collision Avoidance (CSMA/CA) and consists of a basic access method which uses two-way handshaking (DATA/ACK) mechanism. Each node with a packet to transmit has to monitor the channel to determine if another node is transmitting. If the channel is idle for an interval of time that exceeds the Distributed Inter-Frame Space (DIFS), the node proceeds with its transmission. If the channel is sensed as busy, the node defers transmission till the end of the ongoing transmission. The node then generates a random discrete-time backoff interval (illustrated in the next section) for an additional deferral time before transmitting. This backoff is decremented only when the medium is idle for a DIFS again and it is frozen when the medium is sensed busy. Since the backoff interval is chosen randomly, this minimizes collisions during contention among multiple nodes. The node is allowed to transmit when the backoff timer reaches zero.

In addition, to avoid channel capture, a node must wait a random backoff time between two consecutive new packet transmissions, even if the medium is sensed idle in the DIFS time. Along with the Collision Avoidance, the 802.11 introduces a positive acknowledgment (ACK) scheme. All the packets received by a node implementing 802.11 MAC must be acknowledged by the receiving MAC. After receiving a packet, the receiver waits for a brief period, called the Short Inter-Frame Space (SIFS), before it transmits the ACK.

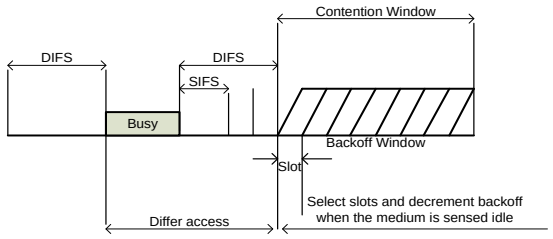

Fig. 1. Basic access mechanism

Within a certain range, hidden nodes can lead to collision with others in the same domain. DCF solved this problem by using the four-way handshaking process "Reservation Based Scheme", in which the node senses the channel and determines its state. Even though the channel is idle, it waits for a DIFS time and a backoff time; then instead of sending the data directly, it sends a RTS packet. After that, the receiver sends an ACK to the node after SIFS time. These RTS/CTS exchange messages shown in Fig. 2, include timing for which all nodes within the range know the time needed for a certain transaction. Therefore, these hidden nodes update their Network Allocation Vector (NAV), and know that they can't send any packet as long as the channel is busy.

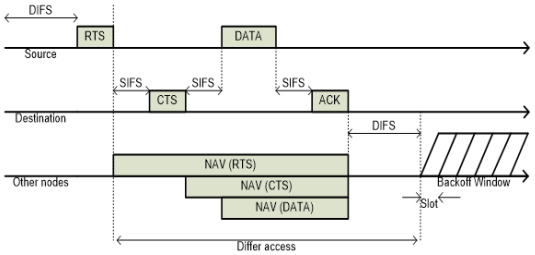

Fig. 2. RTS/CTS mechanism

IV. THE PROPOSED APPROACH

A. Preliminaries and network design

As shown in Fig. 3, the network consists of many nodes which are connected to the access point and are all trying to access the shared medium, which may cause collisions (in addition to the channel problems). Once the node gains the medium, it will send a request to the FTP server in order to download a file. Any loss over the wireless server will be corrected through DCF mechanism. The user is continuously estimating the average time needed to successfully deliver the packets to the access point; this time will be communicated to the server in order to extend its RTO.

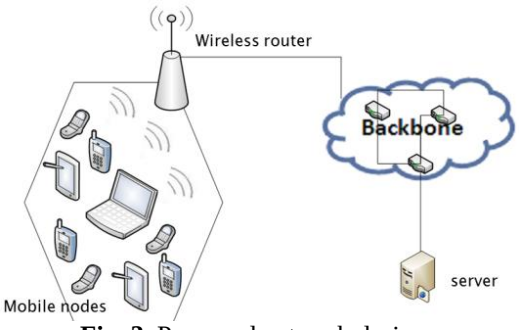

Fig. 3. Proposed network design

B. Modeling the backoff window

It is essential to model the backoff window implemented in DCF MAC layer in order to derive the delay $\overline{D}$ that will serve later to extend the RTO. In 802.11, before any transmission, a node initializes a backoff time which is a random integer distributed over the interval $(0, W-1)$, where W represents the contention window and depends on the number of failed transmissions per packet. For the first transmission attempt, the value is equal to CW_{min} which is called the minimum contention window.

Let p be the probability that the transmitted packet faces a collision in the channel due to two or more nodes transmitting at the same time in the same slot. In this case, after each unsuccessful transmission, the value of W is doubled, up to a maximum value CW_{max} , where $CW_{max} = 2^{m} CW_{min}$, and m represents the number of unsuccessful

attempts for this packet, i.e., the maximum backoff stage. Once W reaches CW_{max}, it keeps this value until it is reset to CW_{min}.

Now, we can derive the general probability that the contention window chosen by a node is equal to W. This probability is given by:

$$P\{Window = W\} = \begin{cases} p^{m-1}(1-p), & \text{for } W = 2^{m-1}CW_{min} \\ p^{m}, & \text{for } W = CW_{max} \end{cases} \quad (1)$$

The approach in [18] derived the collision probability p for the case of a saturated network where a transmitting node always has a queue of packets to send, so each incoming packet is immediately backlogged, i.e. it is preceded by a backoff. Therefore, the derived average backoff window in the saturated case is given by:

$$\frac{W}{2} + p(1-p)\frac{2W}{2} + \cdots + p^{m}(1-p)\frac{2^{m}W}{2} + p^{m+1}\frac{2^{m}W}{2} \quad (2)$$

In our case, we should use the Poisson process to obtain an approximation for the collision probabilities within the non-saturated arrival rates. Considering a wireless cell with N_{C_i} nodes operating in a discrete time where each node could be represented as an M/G/1 queuing system with an infinite storage. The arrival rate of packets is given by λ while the packet service is μ. So as an M/G/1 system, the probability that the packets interface queue is empty could be approximated by the following equation:

$$\pi_0(node) = 1 - \frac{\lambda}{\mu} \quad (3)$$

A packet is backlogged on arrival if the system is not empty, therefore the probability that the cell (N_{C_i} nodes in steady state) is empty is given by:

$$\pi_0(cell) = \left(1 - \frac{\lambda}{\mu}\right)^{N_{C_i}} \quad (4)$$

Then, the backoff window is 0 for any arbitrary packet with probability $\pi_0(cell)$, and it is backlogged with probability $1 - \pi_0(cell)$. Therefore, the average backoff window size for general (non-saturated) arrival rates is given by:

$$\overline{W}_{general\ case} = \left[1 - \left(1 - \frac{\lambda}{\mu}\right)^{N_{C_i}}\right]\left[\frac{1-p-p(2p)^{m}}{1-2p}\frac{W}{2}\right] \quad (5)$$

Considering the fact that the cell contains N_{C_i} nodes and only those with a non-empty queue can actually collide with packets from other nodes. The packet collision probability can be obtained by solving:

$$\mathbf{p = 1 - \left[(1 - \pi_0(node))\left(1 - \frac{1}{\overline{W}_{general\ case}}\right)\right]^{N_{C_i}-1}} =$$

$$\mathbf{1 - \left(\frac{\lambda}{\mu}\right)^{N_{C_i}-1}\left[1 - \frac{2(1-2p)}{(1-p-p(2p)^{m}W)\left(1-\left(1-\frac{\lambda}{\mu}\right)^{N_{C_i}}\right)}\right]^{N_{C_i}-1}} \quad (6)$$

On the other hand, let q be the probability that a node transmits in a randomly chosen slot, the probability p that a transmitted packet faces a collision on the channel in a given slot will be equivalent to the probability that at least one of the $(N_{C_i} - 1)$ remaining nodes transmits in the same time slot. Therefore, the probability p that a collision occurs is given by:

$$p = 1 - (1-q)^{N_{C_i}-1} \quad (7)$$

C. 802.11 Cell throughput analysis

Let S_{C_i} be the saturation throughput inside the wireless cell, defined as the expected time needed to transmit the data payload with respect to idle, collision and header transmission time, during a cycle of frame exchange. We know that:

$$S_{C_i} = \frac{Probability\ \ (success\ \ tx\ in\ a\ slot\ ) \times Payload\ \ \_size}{Duration\ \ of\ a\ cycle\ \ of\ frame\ \ exchange} \quad (8)$$

A cycle of frame exchange consists of several collision cycles and one successful data frame transmission plus header transmission and idle times. Therefore,

$$S_{C_i} = \frac{N_{C_i} q(1-q)^{N_{C_i}-1} \times L(DATA\ )}{Succes\ s_{cycle}\ + Collisio\ n_{cycle}\ + Idle_{\ cycle}} \quad (9)$$

Where

$$\begin{cases} Success_{cycle} = \alpha \times Prob(success\ transmission) \\ Collision_{cycle} = \beta \times Prob(collision) \\ idle_{cycle} = \gamma \times Prob(idle) \end{cases} \quad (10)$$

By substitution, we get the following equation:

$$\begin{cases} Success_{cycle} = \alpha \times N_{C_i} q(1-q)^{N_{C_i}-1} \\ Collision_{cycle} = \beta \times \left[1 - (1-q)^{N_{C_i}} - N_{C_i} q(1-q)^{N_{C_i}-1}\right] \\ idle_{cycle} = \gamma \times (1-q)^{N_{C_i}} \end{cases} \quad (11)$$

α represents the time that the channel is captured with a successful node transmission, β represents the collision duration, i.e., the time that the channel is captured by the node with a collision and γ represents the duration time of a time slot. The value of the parameters α and β differs depending on the access model. Assuming that the packets are data fragment only, that means that there is no fragmentation. Thus, for the basic DCF access mechanism without RTS/CTS:

$$\begin{cases} \alpha = DIFS + \overline{PHY_{hdr}} + \frac{MAC_{\ hdr}\ + DATA}{\vartheta} + SIFS + \frac{ACK}{\vartheta} + 2\varepsilon \\ \beta = DIFS + \overline{PHY_{hdr}} + \frac{MAC_{\ hdr}\ + DATA}{\vartheta} + \varepsilon \end{cases} \quad (12)$$

$\overline{PHY_{hdr}}$ represents the synchronization time between the source node and the destination node, i.e., the

transmission time of the PLCP preamble and PLCP header which defines the physical header of an 802.11 packet. ϑ represents the channel bit rate and ε represents the average propagation delay of any frame on the channel. All these parameters are defined in the IEEE 802.11 standard [17].

By resolving equation (9), we can obtain the relationship between the parameters which will be used in the numerical analysis:

$$S_{C_i} = \frac{N_{C_i} q(1-q)^{N_{C_i}-1} \times L(DATA)}{\alpha N_{C_i} q(1-q)^{N_{C_i}-1} + \beta\left[1-(1-q)^{N_{C_i}} - N_{C_i} q(1-q)^{N_{C_i}-1}\right] + \gamma(1-q)^{N_{C_i}}} \quad (13)$$

D. 802.11 Cell delay analysis

The delay $\overline{D}$ is defined by the average time from the point when a packet becomes head of the node's queue to the point when it is successfully received by the destination, i.e., when a positive acknowledgment is received. We model this delay without taking into account the waiting time in the packets' interface queue before transmitting.

$$\overline{D} = \left(\gamma \overline{W}_{\text{general case}} + \alpha\right) + service\ time \quad (14)$$

α represents the time that the channel is captured with a successful node transmission. The service time is expressed using the average time required for successful packet transmission, thus:

$$\text{service time} = \left(\gamma \overline{W}_{\text{general case}} + \beta + \Delta\right) \overline{N}_{cp} \quad (15)$$

β represents the collision duration, i.e., the time that the channel is captured by the node with a collision and γ represents the duration time of a time slot. The value of the parameters α and β differs depending on the access model.

Δ represents the time that a node has to wait when its packet transmission collides before sensing the channel again. Thus:

$$\Delta = \text{SIFS} + \text{ACK\_Timeout} \quad (16)$$

$\overline{N}_{cp}$ represents the average number of collisions of a packet until it is received successfully. Owing to the assumption of independence at each retransmission, we will be able to calculate $\overline{N}_{cp}$ and approximate the average rate of successful transmission per packet $\overline{N_{sp}}$ which follows a geometric distribution which has an average of $\frac{1}{1-p}$.

Let $\overline{N_{sp}}$ be the average number of node attempts to successfully transmit its packet, i.e., the node has been exposed to $\overline{N}_{sp} - 1$ collisions before to successfully transmit its packet. Therefore:

$$\overline{N}_{cp} = \left(\frac{1}{1-p} - 1\right) \quad (17)$$

After resolving all these equations, we can obtain:

$$\overline{D} = \left(\gamma \overline{W}_{\text{general case}} + \alpha\right) + \frac{p\left(\gamma \overline{W}_{\text{general case}} + \beta + \Delta\right)}{1-p} \quad (18)$$

E. Setting the retransmission timeout (RTO)

The Retransmission Timeout policy of TCP is governed by the rules described in RFC 2988 [19]. Upon reception of an ACK, the RTO is calculated after smoothing out the measured samples, and weighting the most recent RTT (Round Trip Time) variation history.

The standard [19] calculates the RTT as follows:

EstimatedRTT[new] = (1-α)* EstimatedRTT[old] + α*SampleRTT[new]

Typically, the value of α is 0.125. We can clearly notice that the influence of past RTT samples decreases exponentially fast. To set the RTO, the standard uses the EstimatedRTT and a safety margin:

RTO = EstimatedRTT + 4 DevRTT (19)

DevRTT serves to estimate how the SampleRTT deviates from the EstimatedRTT. DevRTT is calculated as follows:

(1-β)DevRTT[old] + β|SampleRTT[new] - EstimatedRTT[old]|

Typically, the value of β is 0.25. This RTO constitutes the subject of interest in the present paper. In the design we extend the RTO in such a way it will include the delay value $\overline{D}$ received in the feedbacks from the clients. Therefore, upon reception of each ACK which piggybacks $\overline{D}$ within the TCP Options header field, we propose to compute the following dynamic RTO:

RTO = EstimatedRTT + 4 DevRTT + $\overline{D}$ (20)

V. PERFORMANCE ANALYSIS

A. Numerical analysis results

Table I summarizes the parameters used for the simulation. Moreover, the maximum throughput for the channel has been assumed equal to 1 Mbps; and the number of mobile nodes "n" inside the 802.11 network was varied between 10 and 50 in order to study the impact on the whole performance.

A numerical study has been conducted in this section. Theoretical throughput and delay will be presented while varying the number of mobile nodes (cell size) with respect to the parameters specified in IEEE 802.11b specifications [17] which define the MAC and physical layers parameters. These parameters are given in table I.

Table I. Simulation Parameters

PARAMETER	VALUE
MAC Header	272 bits
PHY Header	192 bits
Synchronization time	192 μsec
ACK Length	112 bits
Data Transmission Rate ϑ	1 Mbps
Propagation Delay (ε)	1 μsec
ACK_Timeout	212 μsec

SIFS	10 μsec
DIFS	50 μsec
Time Slot γ	20 μsec
CW_{min}	32
CW_{max}	1024

In Fig. 4, we notice that, as long as the window size "W" is less than or equal to 500, the throughput decreases as the number of active users within the cell increases beyond this value of window size. Better throughput performance is shown in case of large number of nodes. On the other hand, we neglect the probability of having large values of W in case of small number of nodes, since the probability of collisions in this case is very small in comparison to the case of large number of nodes.

Fig. 5 shows that the delay $\overline{D}$ is large for small values of W and large number of nodes. Large W may, in fact, increase the delay; because for small W, the amount of idle slot times per packet transmission is very low. This value becomes significant only when W gets larger and the number of nodes is small. When the number of nodes is large, the throughput of large windows are very close, but the delay decreases severely for smaller contention windows.

Fig. 6 shows that the number of transmissions per packet significantly increases as the window size decreases. This effect is more significant for large number of users.

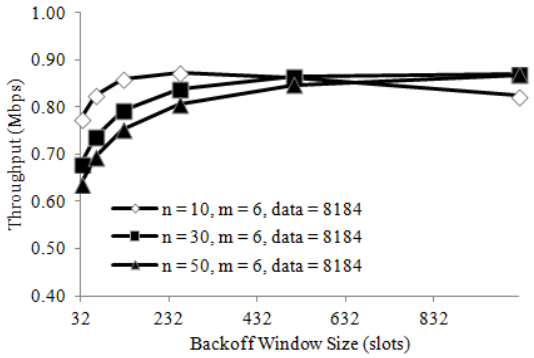

Fig. 4. Throughput versus back-off window size

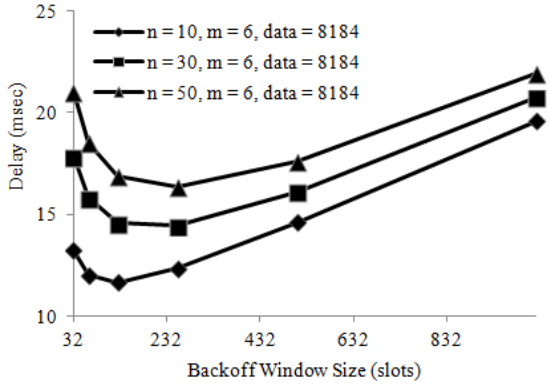

Fig. 5. Delay versus back-off window size

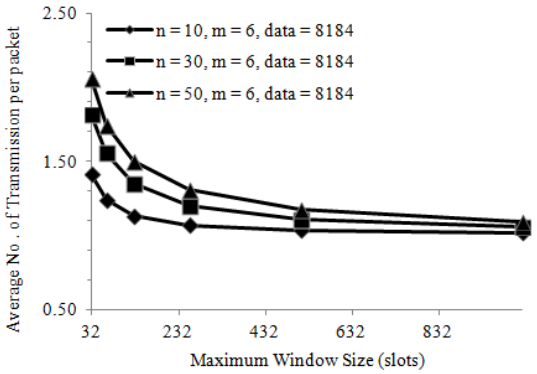

Fig. 6. Average no. of transmission per packet versus window size

B. Simulation results

In order to validate the proposed approach, we opted for an FTP service to simulate bulk TCP data. The RTO was computed as shown in equation (20). The FTP traffic was configured as follows: The Inter Request Time (seconds) was set to 10 seconds which defines the amount of time between file transfers, the file size was set to 50000 bytes, and the type of service was set to Best Effort. During the simulation, we also varied the number of active nodes between 10 and 50 users.

Fig. 7 shows the average traffic sent (bytes/sec) before and after modification in TCP, the blue curve represents the modified TCP that is based on the new RTO mechanism. With this amount of used traffic, we noticed an enhancement in terms of the global FTP traffic sent. This is due to a decrease in the number of timeouts at the FTP server. Therefore, the traffic is flowing faster between the server and the access point.

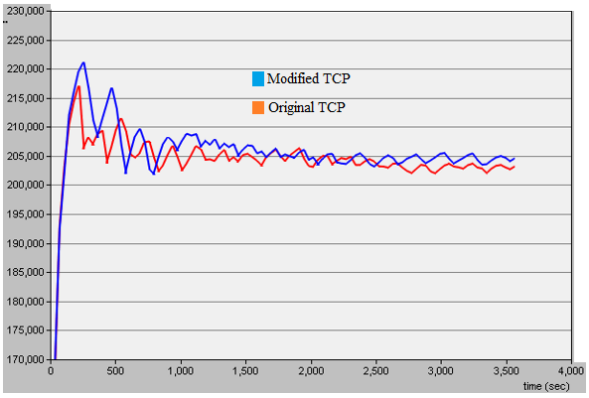

Fig. 7. FTP traffic sent at the FTP server (bytes/sec)

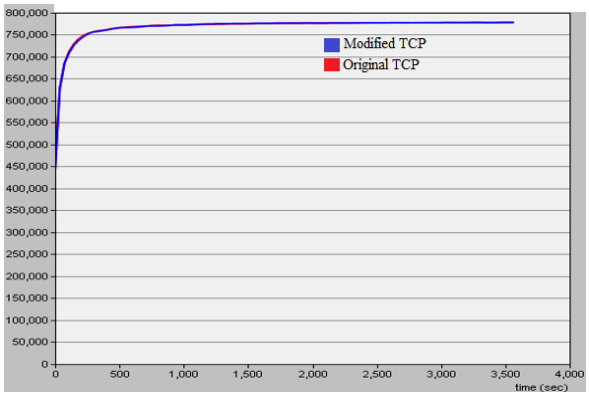

Fig. 8. Average LAN throughput

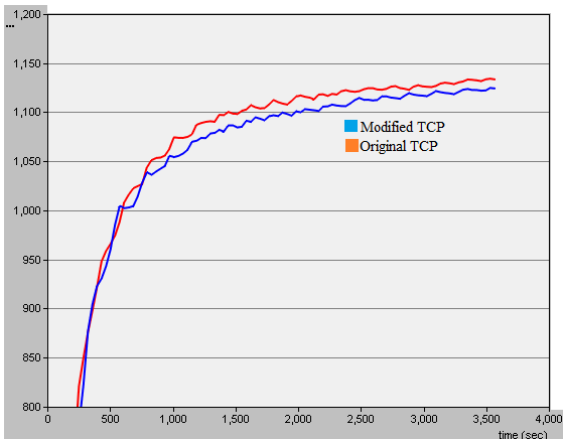

Fig. 9. TCP retransmission count at the FTP server

Fig. 8 shows the average throughout (bits/sec) inside the wireless LAN (Local Area Network). We observe that the modifications did not affect the Wireless LAN throughput. This is due to the fact that under saturation the wireless LAN has reached a steady state. The results shown in this figure also validates the numerical analysis shown in Fig. 4.

Fig. 9 illustrates that the average TCP retransmission count at the server side has also been enhanced compared to the original TCP. The number of retransmissions has decreased, hence the number of timeouts is also decreased which will lead to an enhancement in terms of throughput (Fig. 7).

VI. CONCLUSION AND FUTURE WORK

With a growing deployment of wireless networks, it is crucial to support TCP-based applications in both wired and wireless environments. Since failures in 802.11 networks can be corrected by the 802.11 MAC layer, it becomes crucial to propose a new mechanism for setting the TCP retransmission timeout (RTO). In this paper, we have modeled the time needed by the 802.11 network to successfully deliver a packet; this time was piggybacked in the TCP Options header field and sent to the TCP server in the form of feedbacks that will serve in adjusting the RTO dynamically. Simulation results have demonstrated significant performance improvement in terms of overall TCP traffic sent, and retransmissions at the TCP server. We believe that this enhancement will contribute in reducing the level of congestion at the access point since duplicate TCP messages will be avoided on the path between the server and the wireless LAN.

REFERENCES

[1] Information Sciences Institute, University of Southern California, "Transmission Control Protocol", RFC 793, September 1981.

[2] M. Allman and V. Paxson, "TCP Congestion Control", RFC 5681, September 2009.

[3] Wan G.Zang and Ljiljana Trajkovic, "TCP Packet Control for Wireless Networks," *WM-195 NSERC GRANT*, p. 8, October 2005.

[4] Bazzal Zouhair, Nahas M., Raad M., Ghareeb M., and Haj-Ali A. "Improving the Performance of the Transport Control Protocol over 802.11 Wireless Networks". The 25th International Conference on Microelectronics (ICM 2013) - Circuits and Systems. Beirut, Lebanon. December 2013.

[5] Bazzal Zouhair, Kadoch, M., Gagnon, F., Bennani, M.; "Optimizing the Performance of the Generated Clusters in Mobile Ad hoc Networks"; The 11th World Multi-Conference on Systemics, Cybernetics and Informatics WMSCI 2007. Orlando, Florida, USA, July 2007.

[6] A. Pattanayak, "TCP in wireless and mobile networks: challenges and solutions", technical report, March 2013

[7] Subramanian, V., Kalyanaraman, S. and Ramakrishnan, K. K., "An End-to-End Transport protocol for Extreme Wireless Network Environments", Military Communications Conference, MILCOM 2006, pp. 1-7, 2006.

[8] Y. Sharma, K. Ramakrishnan, K. Kar, and S. Kalyanaraman, "Complementing tcp congestion control with forward error correction," Networking 2009, pp. 378-391, 2009.

[9] C. Barakat and A. Al Fawal, "Analysis of link-level hybrid fec/arq-sr for wireless links and long-lived tcp traffic," Petformance Evaluation, vol. 57, no. 4, pp. 453-476, 2004.

[10] L. Baldantoni, H. Lundqvist, and G. Karlsson, "Adaptive end-to-end fec for improving tcp performance over wireless links," 2004 IEEE International Conference on Communications (IEEE Cat. No.04CH37577), 2004.

[11] T. Tsugawa, N. Fujita, T. Hama, H. Shimonishi, and T. Murase, "Tcp-afec: An

adaptive fec code control for end-to-end bandwidth guarantee," Packet Video 2007, 2007.

[12] B.R. Badrinath. Ajay Bakre, "I-TCP: Indirect TCP for Mobile Hosts," Department of Computer Science, Rutgers University, Piscataway, Report 1994.

[13] T. Nandagopal, N. Venkitaraman, V. Bharghavan, P. Sinha, "WTCP: A Reliable Transport Protocol," Wireless Networks, pp. 301-316, August 2002.

[14] Jack Chow, and Ljiljana Trajkovic Chi Ho Ng, "Performance Evaluation of TCP over WLAN 802.11 with the Snoop," School of Engineering Science, Simon Fraser University, Vancouver, Report 1997.

[15] Youssef Bassil, "TCP Congestion Control Scheme for Wireless Networks based on TCP Reserved Feild and SNR Ratio," International Journal of Research and Reviews in Information Sciences (URRIS), p. 7, June 2012.

[16] H.-J. Byun and J.-T. Lim H.-J. Lee, "TCP-friendly Congestion Control for streaming real-time applications over wireless networks," The Instituation of Engineering and Technology, vol. 2, pp. 159-163, January 2008.

[17] "IEEE Standard for Wireless Lan Meduim Access Control (Mac) and Physical Layer (PHY) Specifications," IEEE, 1999.

[18] K. Chua Y. Tay, "A Capacity analysis for the IEEE 802.11 Mac protocol," Wireless Networks, vol. 7, no. 2, pp. 159-171, 2001.

[19] V. Paxson and M. Allman. Computing TCP's Retransmission Timer, RFC 6298, June 2011.