Journal of Computer Science 9 (8): 967-971, 2013
ISSN: 1549-3636
© 2013 Science Publications
doi:10.3844/jcssp.2013.967.971 Published Online 9 (8) 2013 (http://www.thescipub.com/jcs.toc)

TRANSMISSION CONTROL PROTOCOL PERFORMANCE COMPARISON USING PIGGYBACK SCHEME IN WLANS

Ali Ahmad Milad, Zul Azri Bin Muhamad Noh,
Abdul Samad Shibghatullah, Shahrin Sahib, Rabiah Ahmad and Mustafa Almahdi Algaet

Department of System and Computer Communication,
Faculty of Information and Communication Technology,
Universiti Teknikal Malaysia Melaka, 76100, Melaka, Malaysia

Received 2013-01-23, Revised 2013-04-29; Accepted 2013-07-02

ABSTRACT

The main problem at wireless networks is the overhead at MAC layer; when the data physical rate is increasing it causes increasing the overhead and decreasing at the MAC efficiency. In this study we study the performance comparison of TCP protocol in WLANs with and without using piggyback. The study of results concerning of implemented both mechanisms in NS2 simulator and find out the good performance from this comparison. Based on the results from our experiments show that the Piggyback scheme is one of the efficient ways to reduce the overhead at MAC wireless networks.

Keywords: Wireless Networks, TCP, Piggyback

1. INTRODUCTION

It is evident that the majority of the issues concerned with the performance of the TCP/IP networks are resulted from a variety of interaction patterns between the TCP engine and the surrounding communication environment. Thus, this section reviews information about the TCP protocol fundamentals. Since TCP is defined as a complex protocol, it is important that its performance should be well understood and so, this section highlights its basic operations and explains some of key features of TCP (Dobbins *et al*., 2010). TCP provides many services to its applications. They are as follows.

1.1. Connection Oriented Services

TCP is defined as protocol which is oriented by a connection. It is important that establishing a TCP connection between two application processes is accomplished before they start sending data to each other. However, in a case when these processes for multiple applications operate on a particular IP host, identification of each process is usually recognized by a unique port number in that host. Therefore, each process is able to make up its own separate TCP connection. Thus, a four TCP types are, source IP address, source TCP port number, destination IP address and destination TCP port number are all together are used in identifying each separate TCP connection relevant to that process. The connection is terminated upon complication of the communication session. Streaming service is pointed out that the other service provided by TCP is a streaming service which is offered or supplied to its application (Dafalla and Hammoshi, 2012). As soon as establishment of a TCP connection is accomplished between two sides; a sender and receiver, the sender carries out the process of writing a stream of bytes or characters into the connection and at the same time, the receiver performs the function of reading bytes out of the connection. The stream-oriented abstraction is seen exclusively by the applications and the operation of the TCP layer is found on a packet mode. The sending TCP performs these three functions; accumulating a certain amount of application bytes, forming a packet named a TCP segment and sending the segment to the receiver. For the functions performed by the receiving TCP, they include extracting the application bytes from the

Corresponding Author: Ali Ahmad Milad, Department of Computer System and Communication, Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, 76100, Melaka, Malaysia segment, ordering or organizing them if needed and delivering them as a stream of bytes to the suitable receiving application process.

1.2. Full Duplex Service

TCP is recognized or perceived as a full duplex protocol which plays a role in supporting the data flow in both directions. This implies that the process of sending data from one direct to another can be done by either process once establishing a TCP connection between two applications is achieved.

1.3. Reliable Service

This type of service is realized through the guaranteed delivery of every single byte, in order, without being duplicated as performed by TCP. However, in cases when such bytes arrive out of order and their delivery is duplicated, the incoming data delivers to the application process with TCP buffers process in order to achieve re-ordering of out of order arrival and to eliminate any duplicate delivery respectively. For achieving guaranteed delivery of the data, TCP utilizes the acknowledgment mechanism in order to ensure that the receiver has received the transmitted data correctly. If the underlying communication channel is noisy and error-prone, it is stated that the same segment can be transmitted several times as to ensure or guarantee a correct delivery of the transmitted data to the receiver. Thus, to sum up, this particular reliability characteristic has its extreme importance especially in the majority of data applications such as file transfer. This is because having that feature; such applications do not need to worry about the lost or disordered data (Podlesny and Williamson, 2010).

1.4. End-to-End Semantic

According to this feature, it is pointed out that the reliability of TCP mainly relies on the end-to-end semantic, which means that generating the acknowledgments is made exclusively by the receiving TCP and only after reception of the correct data has been acknowledged or notified by the receiver. Therefore, when an ACK is received by the TCP sender, safe reception of the data by receiver is totally guaranteed. Thus, such an end-to-end semantic feature is said to be a distinguished feature providing or making the TCP layer ultimately reliable. However, it is important to be noted that it can be subjected to violation especially when the ACKs are generated by any intermediate node (not the TCP destination) on behalf of the destination (Luo *et al*., 2010).

1.5. Overview of Piggyback Scheme

Piggyback scheme has been addressed in many researchers before (Lee *et al*., 2010; 2007a; 2007b) and the main idea for piggyback is when the receiver station has a frame to the sender station and allows sending the data frame with ACK to the sender, this process is called a piggyback scheme. So the acknowledgment gets a free ride in the data frame and takes few bits and this is a distinct ACK. Therefore, each frame requires an ACK header and data frame (checksum). This means that the piggyback merely relies on the receiver since the fewer the frames are sent, the fewer the frames that arrive and this is dependent the way the receiver is organized. However, the emergence of the piggyback scheme has posed some cases of complexity. For example, the question of how long the data link layer is supposed to wait until the packet is transmitted to piggyback the ACK is still a posing a complex case. As far as we know, the link layer usually waits for a certain period of time and if this waiting period consumed by the data link layer is longer than the sender timeout period, it is expected that the frame will be retransmitted. Therefore, it is assumed that 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 the piggyback is that the piggyback frame does not need to rivalry the channel again in a case when the receiver has a frame to send to the sender. This is because it does not need to be in the front of the queue but the nearest frame to the destination at the sender and the same researcher shows the overhead with/without piggyback mechanism in case when the frame belong to the receiver is sent to the sender after receiving a frame as completion of the channel. After this a CTS frame time, an RTS frame time, two SIFS times, a DIFS time and a random back off is required to be done by it. Otherwise, in a case when the frame is possible to be piggybacked by the receiver to the sender along with the acknowledgment, the ACK is sent by the sender as a way of acknowledging the piggybacked frame after reducing the SIFS time and the overhead has been already accomplished Xiao (2005) **Fig. 1**.

1.6. Without Piggyback Scheme

When the traditional transmission accrue, means the data transmission from node A and only the receiver at B received the data and replay ACK this called DCF distributed coordination function mechanism.

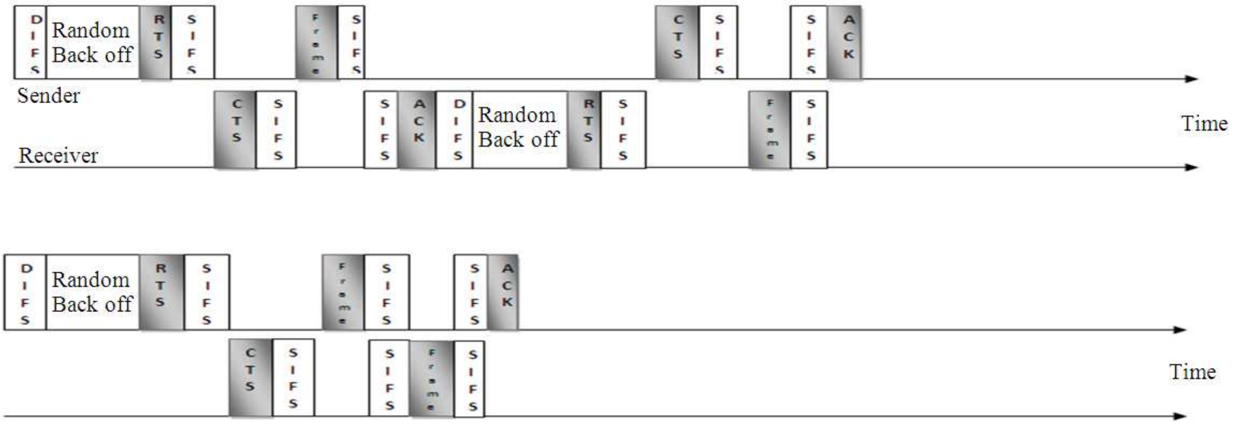
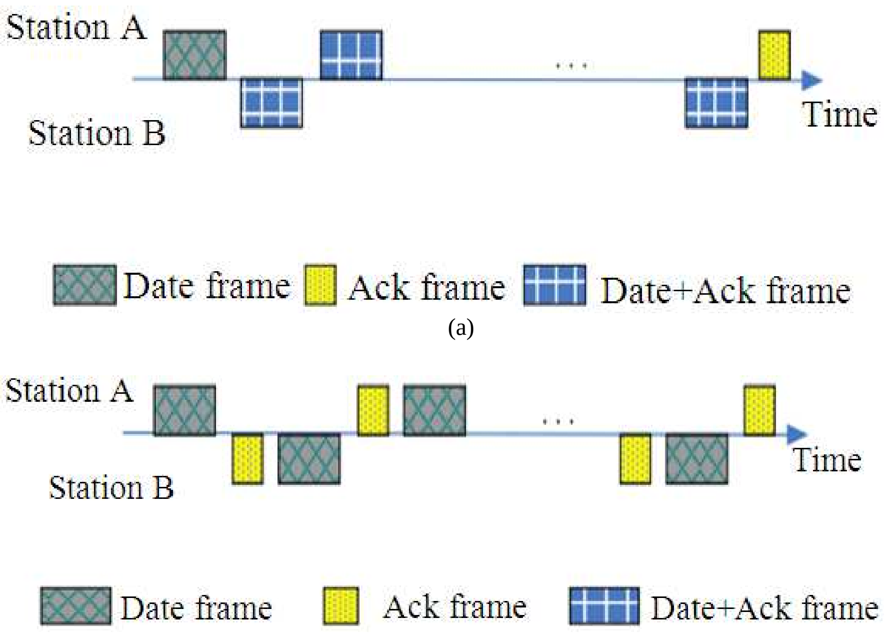


Fig. 1. The overhead with/without piggyback scheme



(a)

(b)

Fig. 2. (a) Piggyback mechanism (b) Without piggyback mechanism

So when there is no data ready to send from the receiver to sender by piggybacking the ACK the DCF will performed by default Xiao (2005).

2. MATERIALS AND METHODS

In our simulation using NS2, we have two stations A and B transmitted the data in reverse direction based wireless network using TCP traffic. Reverse direction means attach the ACK in data frame (piggyback), as shown in **Fig. 2a**.

In **Fig. 2b** showed the normal transmission between A, B stations without piggyback, when station A send the data to station B, station A waiting a period of time it is not more than 200 ms to receive ACK, if the time is up the station A will be retransmission the data frame again and this is a type of the overhead at wireless networks.

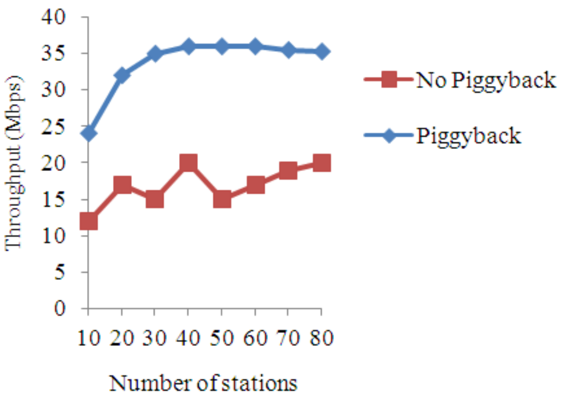


Fig. 3. Simulation result compared the throughput with number stations using TCP traffic

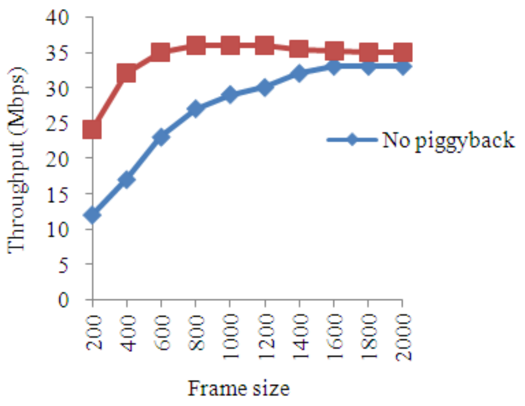


Fig. 4. Compared the throughput with the frame size

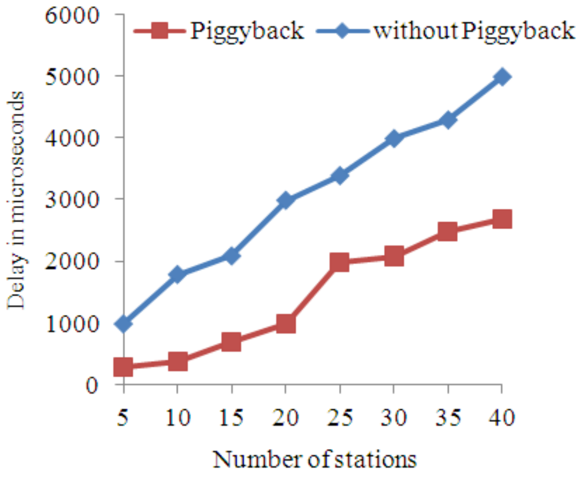


Fig. 5. Compared delay with number of stations

Table 1. Parameters used in our simulation

Type of parameters	TCP
Basic rate	54 Mbps
Data rate	150 Mbps
Number of stations	2
Frame size	Varied 200, 400, 600
ACK frame size	36 bytes

3. RESULTS

In our simulation we used TCP traffic and the parameters have listed in **Table 1**. The Important feature of TCP includes the fact that traffic is elastic and so achieved throughput is related to network capacity.

In **Fig. 3**, described the throughput performance using with/ without piggyback compared with number of stations, we can see the piggyback mechanism achieves high throughput better than without piggyback.

In **Fig. 4** we showed the performance throughput compared with the frame size and piggyback scheme achieved high performance.

Based on the simulation result in **Fig. 5**, we concluded that the delay with piggyback scheme is better than without piggyback.

4. DISCUSSION

Both mechanisms piggyback and without piggyback have been implemented and executed using NS 2.27 and the main idea of piggyback scheme is sending data from sender to receiver and the receiver has a frame to send data directly to the sender with the ACK, so the ACK got free ride and takes few bits at the header of the frame and without piggyback mechanism is sending the data from sender to the receiver and the receiver replay only the ACK without any data. So, based on the parameters those mention in **Table 1**, the performance of piggyback mechanism for both throughput and delay is better than without piggyback mechanism.

5. CONCLUSION

In this research paper, two types of mechanisms have been compared using TCP based on WLANs namely, piggyback and without piggyback were implemented and compared their performance throughput and delay by sending TCP traffic. The implementation was in NS version 2.27 simulator. In the end, the results acquired from simulation were the throughput and delay performance of piggyback scheme is better than without piggyback scheme.

6. ACKNOWLEDGEMENT

This research was funded by the Ministry of Higher Education Malaysia and Universiti Teknikal Malaysia Melaka under Long Term Research Grant LRGS/2011/FTMK/TK01/R0002.

7. REFERENCES

Dafalla, Z.I. and M. Hammoshi, 2012. Comparative Performance Evaluation of Different TCP Derivatives over MANET. Athens Institute for Education and Research ATINER.

Dobbins, K., T.A. Grant, D.J. Ruffen, L. Kane and T. Len *et al*., 2010. Distributed connection-oriented services for switched communication networks. Google Patents.

Podlesny, M. and C. Williamson, 2010. Providing fairness between TCP NewReno and TCP Vegas with RD network services. Proceedings of the 18th International Workshop on Quality of Service, Jun. 16-18, IEEE Xplore Press, Beijing, pp: 1-9. DOI: 10.1109/IWQoS.2010.5542752

Lee, H.J., J.H. Kim and S. Cho, 2007a. A novel piggyback selection scheme in IEEE 802.11e HCCA. Proceedings of the IEEE International Conference on Communications, Jun. 24-28, IEEE Xplore Press, Glasgow, pp: 4529-4534. DOI: 10.1109/ICC.2007.748

Lee, H.J., J.H. Kim and S.H. Cho, 2007b. A delay-based piggyback scheme in IEEE 802.11. Proceedings of the IEEE Wireless Communications and Networking Conference, Mar. 11-15, IEEE Xplore Press, Kowloon, pp: 447-451. DOI: 10.1109/WCNC.2007.87

Lee, T.H., Y.W. Kuo, Y.W. Huang and Y.H. Liu, 2010. To Piggyback or not to piggyback acknowledgments? Proceedings of the IEEE 71th Vehicular Technology Conference, May 16-19, IEEE Xplore Press, Taipei, pp: 1-5. DOI: 10.1109/VETECS.2010.5493905

Luo, H., S. CI, D. Wu and H. Tang, 2010. End-to-end optimized TCP-friendly rate control for real-time video streaming over wireless multi-hop networks. J. Visual Commun. Image Represent., 21: 98-106. DOI: 10.1016/j.jvcir.2009.06.006

Xiao, Y., 2005. IEEE 802.11 performance enhancement via concatenation and piggyback mechanisms. IEEE Trans. Wireless Commun., 4: 2182-2192. DOI: 10.1109/TWC.2005.853875