International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.3, No.10, October 2014 DOI:10.15693/ijaist/2014.v3i10.102-108

# Study the Behavior of One-Way TCP

Dr. Saad Talib Hasson
Professor, Dean of Almusyab College For Engineering And Technology
Babylon University, Babylon, Iraq

Ahmed Jawad Kadhim
Ministry of Education, Iraq
General Directorate for Education in Qadisiyah Ibn al Nafis School for Boys

***Abstract*— The one-way TCP versions are differ with respect to their congestion control and segment loss recovery techniques. The congestion control used to determine the window size according to the bandwidth available on the network. This paper uses simulation to compare the one-way TCP versions that are *TAHOE, RENO, NERRENO, VEGAS, FACK, SACK, LINUX* according to throughput, dropped packets, normalize routing load (NRL), average end to end delay , packet delivery fraction (PDF), and average jitter in MANET to study the behavior of these types in order to know the best one than others. This paper applies the simulation using NS-2 with all types of sink that work with TCP like *TCPSink, TCPSink/DelAck, TCPSink/Sack1, and TCPSink/Sack1/DelAck.***

***TAHOE, RENO, NERRENO, VEGAS, FACK, SACK, LINUX.***

## I. INTRODUCTION

TCP transmission control protocol has two common features that are reliability and connection oriented [1]. The reliability performed by using the acknowledgement. When the packet reaches to the receiver, it is return an ACK packet to the sender. If the sender not receive an ACK packet from the receiver in certain period of time, the TCP sender will assume that the packet is lost and will retransmit this packet again [2]. There are two types of TCP that are one-way TCP and two-way TCP. One-way TCP include *Tahoe, Reno, NewReno, Vegas, Fack, Sack, Linus.* Two-way TCP include *FullTCP* only [3]. In Tahoe, the congestion control has three algorithms that are slow-start, congestion avoidance and fast retransmission. *Reno* adds the fast recovery algorithm to the algorithms available in Tahoe. *NewReno* like *Reno* but it is modify the recovery operation to deal with multiple packets instead of one in single transmission window [4]. Sack is faster than *NewReno* in the recovery operation from multiple losses [5]. *Vegas* present new technique for congestion detecting during congestion avoidance and slow-start. Also, there is modification of transmission strategy [6]. *Fack* is *Reno* TCP with forward acknowledgement. In *Linux*, the congestion control of TCP is imported from kernel of linux. The TCP sender must peer with a “TCP sink” objects like *TCPSink, TCPSink/DelAck, TCPSink/Sack1,* and *TCPSink/Sack1/DelAck* work as receiver. For more information about TCP sink see [3].

## II. RELATED WORK

Muhammad Ijaz and et al. at 2009 studied the performance of AODV, DSR and TORA with different TCP types (Tahoe, Reno and NewReno) in terms of throughput, delay and congestion window. From the results, they concluded that AODV is the best routing protocol for MANET [4].

Tayade Mandakini and et al. compared Tahoe, Reno, New Reno, Sack, Westwood, and Vegas in MANET based on congestion avoidance, fast recovery, slow start, fast retransmission, selective acknowledgment and congestion control. From the results, they found that the network effected on the TCP variant behavior. These TCP variant has solution for the problems of the network [7].

Ahmed jawad kadhim at 2013 modify NS-2 by adding TCP variants to it. From the results, he found the network simulator NS-2 becomes more efficient to help the searchers to study the behavior of the TCP types as a sender and receiver [8].

## III. THE SIMULATION

NS-2 was used to implement MANET environment for 10 different runs in order to collect the values of the performance metrics for each version of TCP. Table (1) represents the parameters of suggested MANET.

Table1: The parameters of MANET

| Parameter | Value |
|---|---|
| The network simulator | NS-34 |
| The number of nodes | 20 |
| Routing protocol | AODV |
| Simulation time | 80 |
| Simulation area | 1000 m * 1000m |
| Pause time | 4 s |
| The speed of nodes | 10 m/s |

International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.3, No.10, October 2014 DOI:10.15693/ijaist/2014.v3i10.102-108

| | |
|---|---|
| Max connection | 5 |
| TCP version as sender | Tahoe, Reno, Newreno, Sack1, Fack, Vegas, and Linux. |
| TCP as receiver | TCPSink, TCPSink/DelAck, TCPSink/Sack1, and TCPSink/Sack1/DelAck. |
| Propagation model | Two ray round. |
| MAC | 802.11 |
| Antenna | Omni Antenna |
| Traffic type | FTP |
| Packets size | 512 bytes /second |
| Transition rate | 4 packets/second |
| Mobility model | Random way point model |
| Speed type | Uniform |
| Pause time type | Uniform |

## IV. The Results

The results of the performance metrics of the MANET for each version of TCP are obtained by running the simulation 10 times and then find the average of it.

Figure (1), Figure (2), Figure (3), Figure (4), Figure (5), and Figure (6) represent the throughput, the dropped packets, PDF, NRL, average end-to-end delay, and average jitter respectively of the all TCP versions as sender and *TCPSink* as receiver.

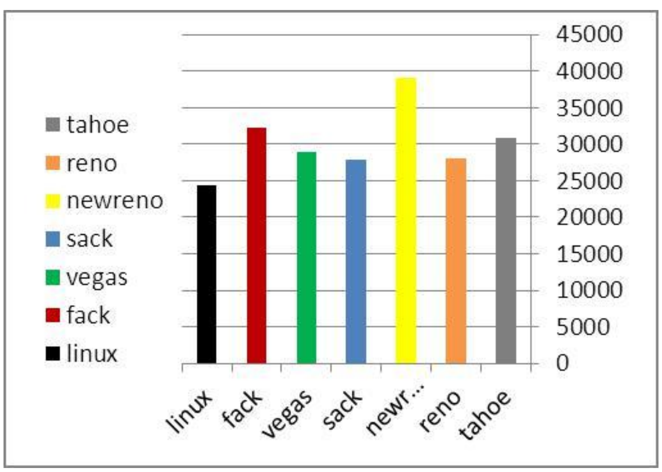


Figure 1: The throughput of all TCP versions with *TCPSink* as receiver

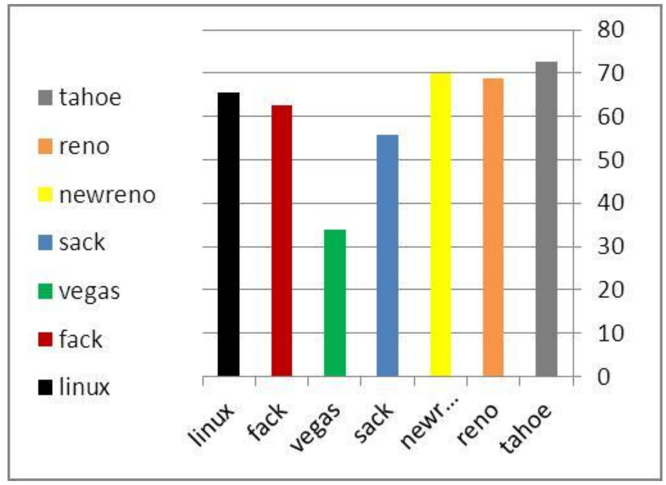


Figure 2: Dropped packets of all TCP versions with *TCPSink* as receiver

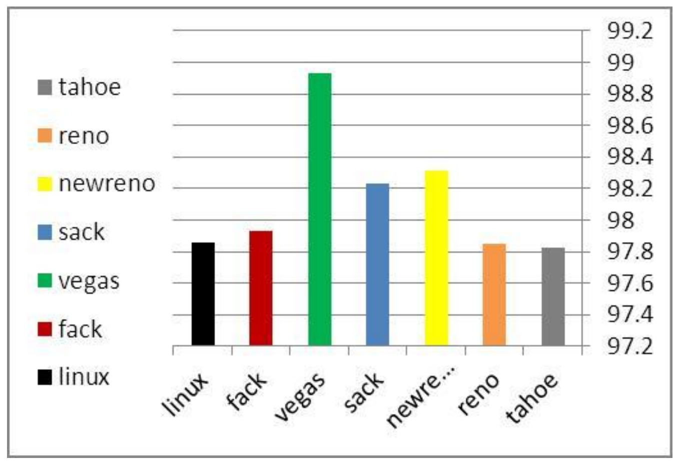


Figure 3: PDF of all TCP versions with *TCPSink* as receiver

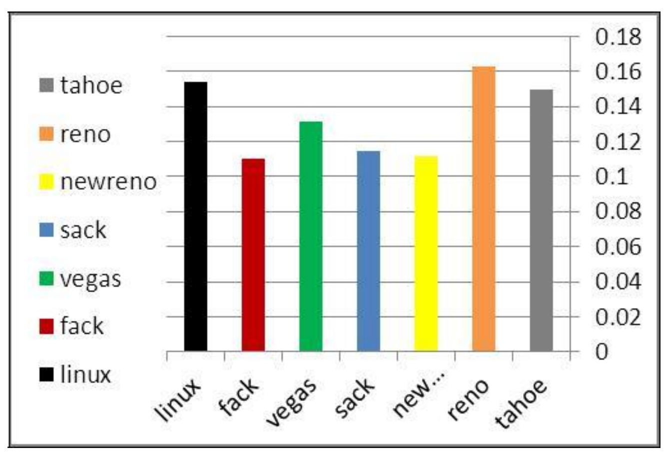


Figure 4: NRL of all TCP versions with *TCPSink* as receiver

International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.3, No.10, October 2014 DOI:10.15693/ijaist/2014.v3i10.102-108

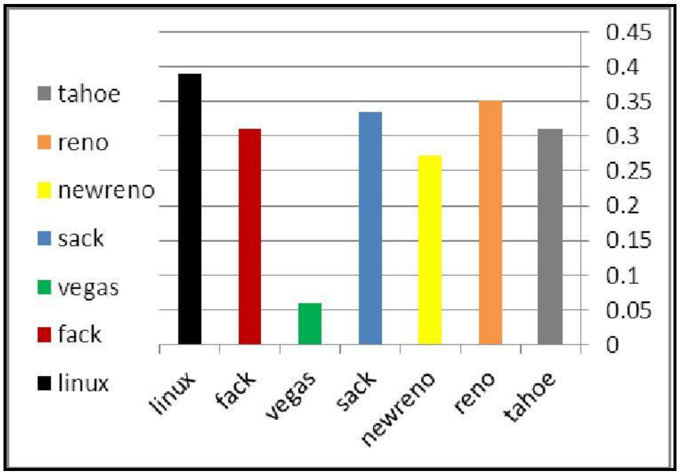


Figure 5: Average E2E delay of all TCP versions with *TCPSink* as receiver

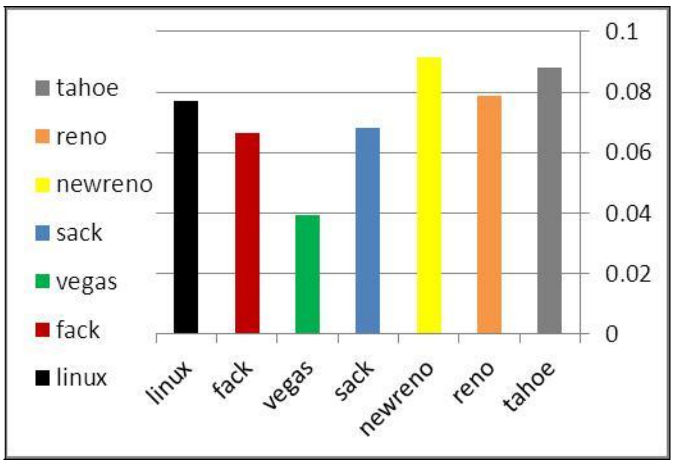


Figure 6: Average jitter of all TCP versions with *TCPSink* as receiver

Figure (7), Figure (8), Figure (9), Figure (10), Figure (11), and Figure (12) represent the throughput, the dropped packets, PDF, NRL, average end-to-end delay, and average jitter respectively of the all TCP versions as sender and *TCPSink/DelAck* as receiver.

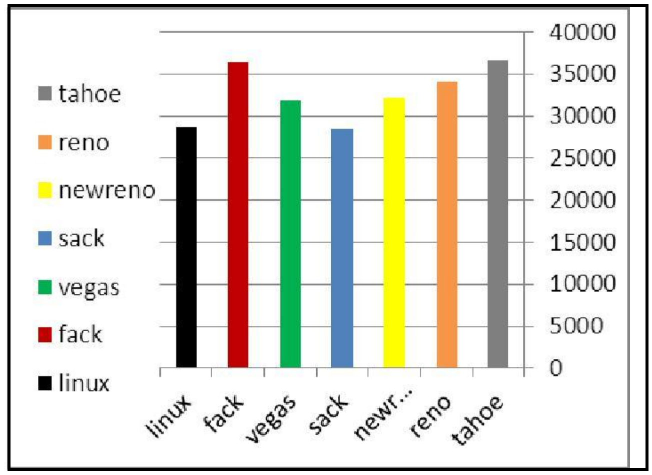


Figure 7: The throughput of all TCP versions with *TCPSink/DelAck* as receiver

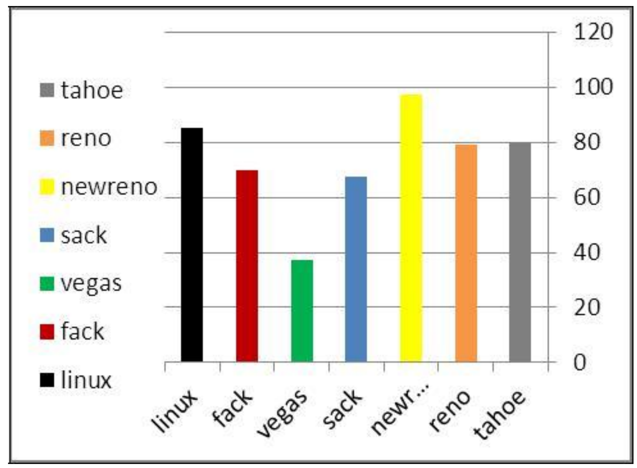


Figure 8: Dropped packets of all TCP versions with *TCPSink/DelAck* as receiver

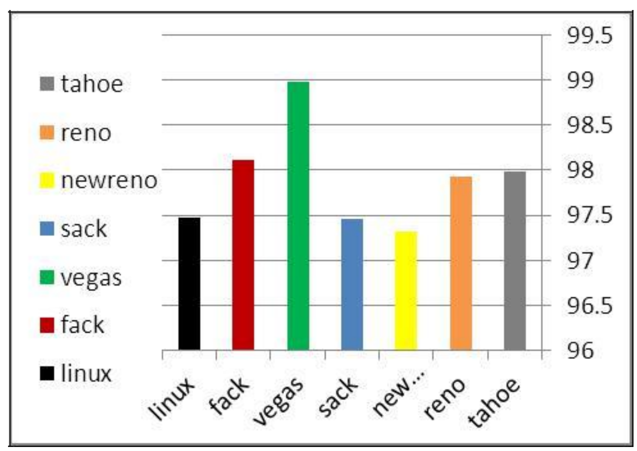


Figure 9: PDF of all TCP versions with *TCPSink/DelAck* as receiver

International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.3, No.10, October 2014 DOI:10.15693/ijaist/2014.v3i10.102-108

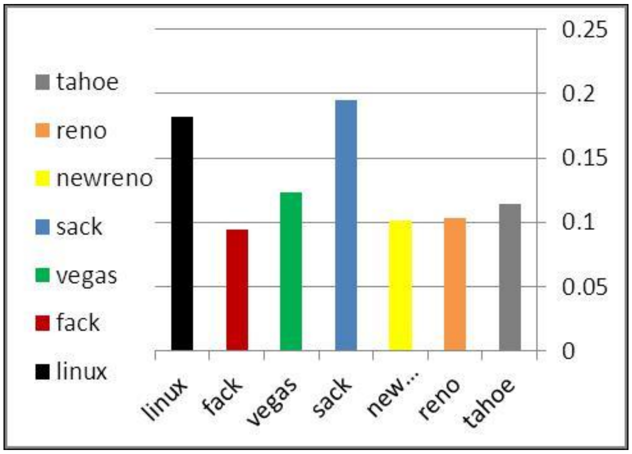


Figure 10: NRL of all TCP versions with *TCPSink/DelAck* as receiver

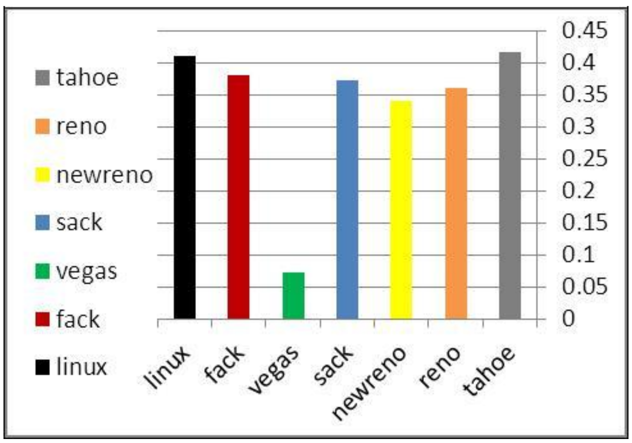


Figure 11: Average E2E delay of all TCP versions with *TCPSink/DelAck* as receiver

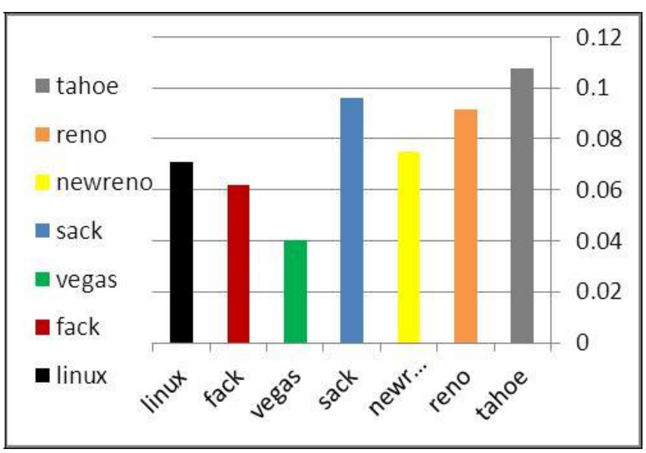


Figure 12: Average jitter of all TCP versions with *TCPSink/DelAck*as receiver

Figure (13), Figure (14), Figure (15), Figure (16), Figure (17), and Figure (18) represent the throughput, the dropped packets, PDF, NRL, average end-to-end delay, and average jitter respectively of the all TCP versions as sender and *TCPSink/Sack1* as receiver.

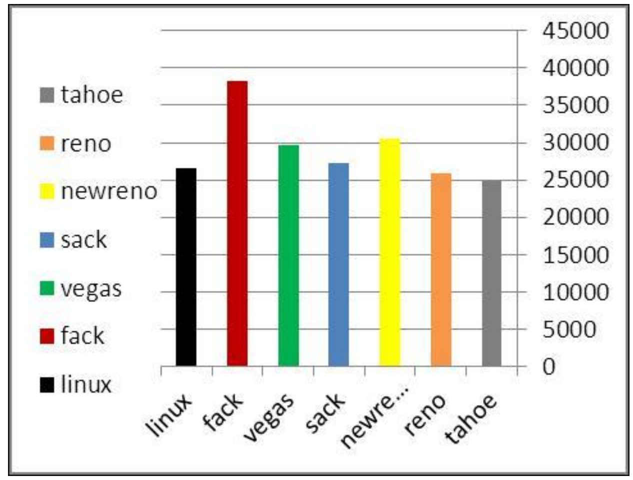


Figure 13: The throughput of all TCP versions with *TCPSink/Sack1*as receiver

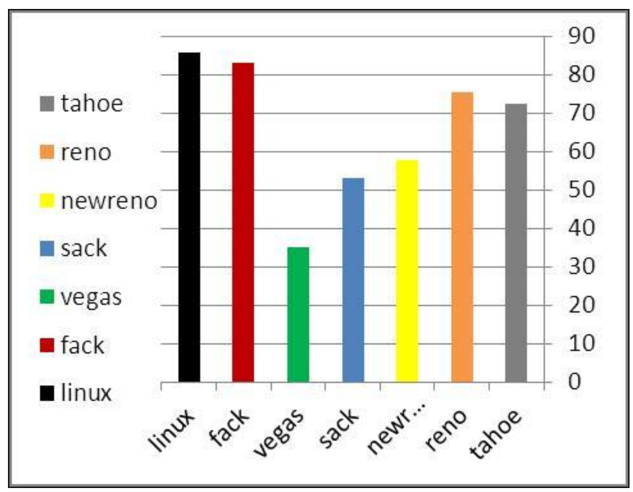


Figure 14: Dropped packets of all TCP versions with *TCPSink/Sack1*as receiver

International Journal of Advanced Information Science and Technology (IJAIST) ISSN: 2319:2682
Vol.3, No.10, October 2014 DOI:10.15693/ijaist/2014.v3i10.102-108

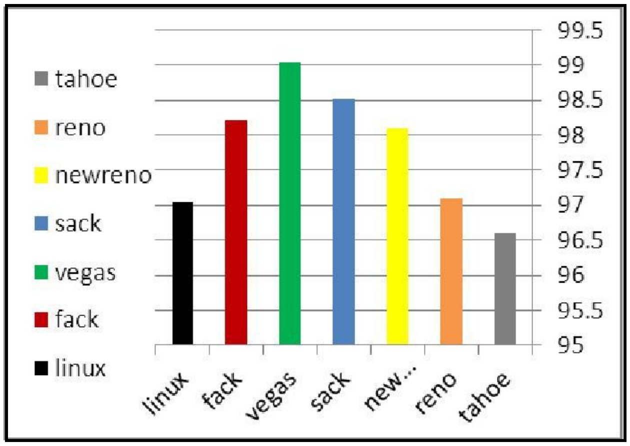


Figure 15: PDF of all TCP versions with *TCPSink/Sack1*as receiver

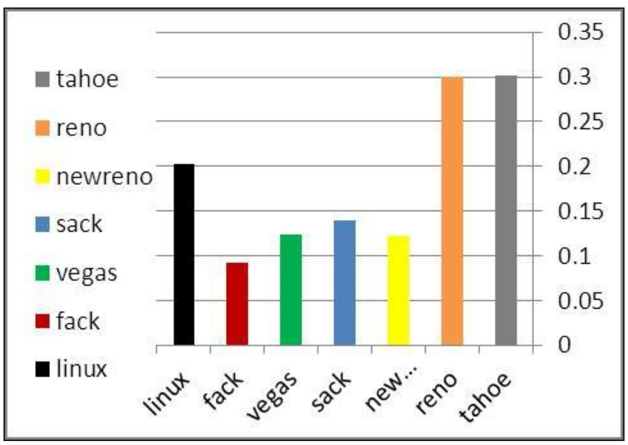


Figure 16: NRL of all TCP versions with *TCPSink/Sack1*as receiver

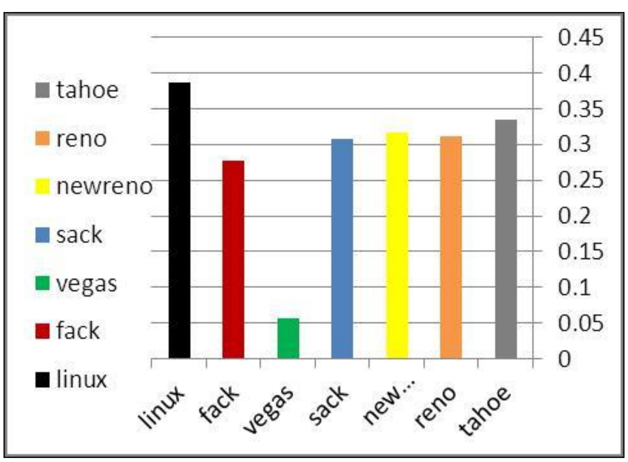


Figure 17: Average E2E delay of all TCP versions with *TCPSink/Sack1*as receiver

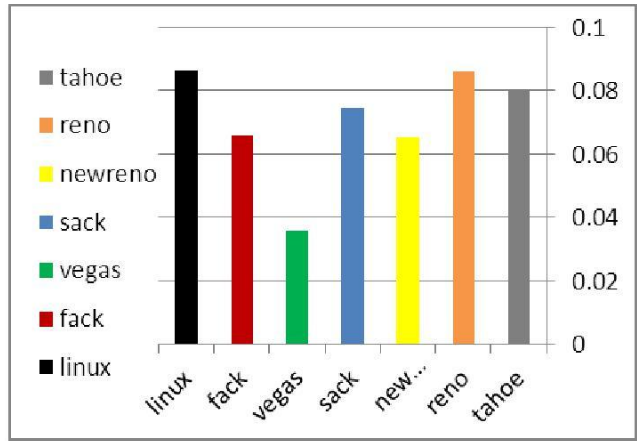


Figure 18: Average jitter of all TCP versions with *TCPSink/Sack1*as receiver

Figure (19), Figure (20), Figure (21), Figure (22), Figure (23), and Figure (24) represent the throughput, the dropped packets, PDF, NRL, average end-to-end delay, and average jitter respectively of the all TCP versions as sender and *TCPSink/Sack1/DelAck* as receiver.

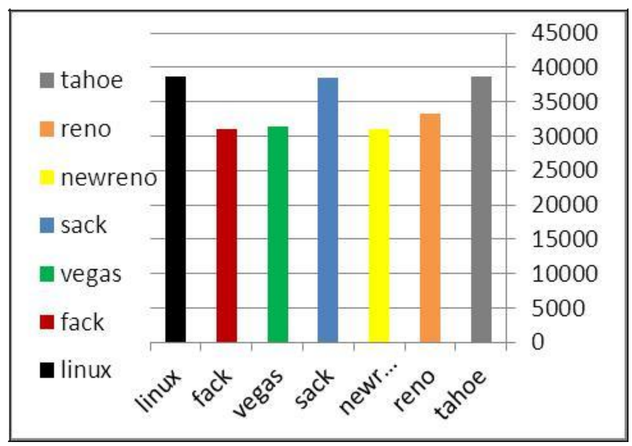


Figure 19: The throughput of all TCP versions with *TCPSink/Sack1/DelAck* as receiver

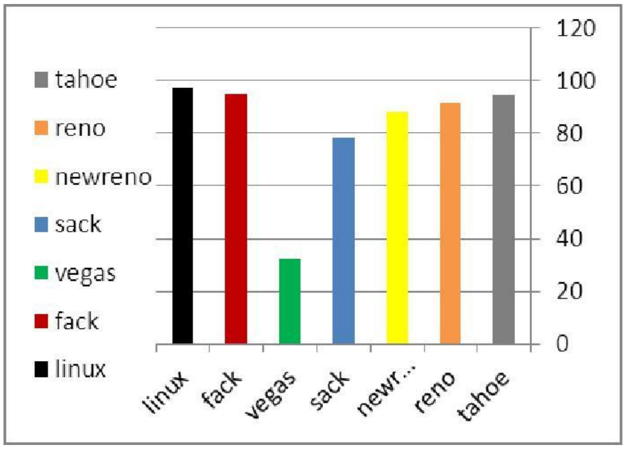


Figure 20: Dropped packets of all TCP versions with *TCPSink/Sack1/DelAck*as receiver

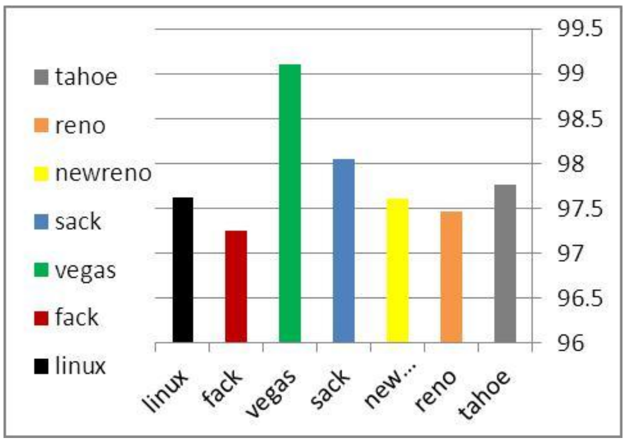


Figure 21: PDF of all TCP versions with *TCPSink/Sack1/DelAck* as receiver

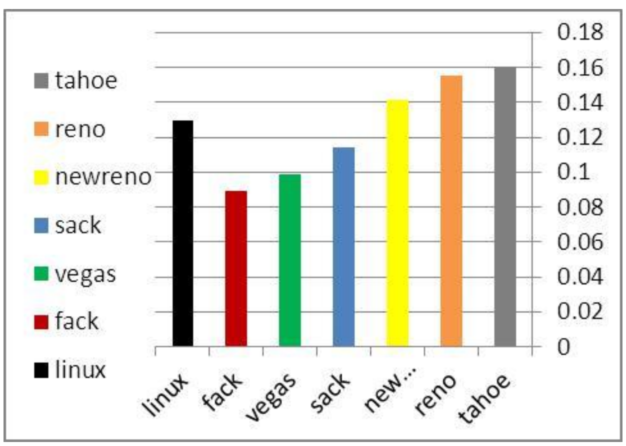


Figure 22: NRL of all TCP versions with *TCPSink/Sack1/DelAck*as receiver

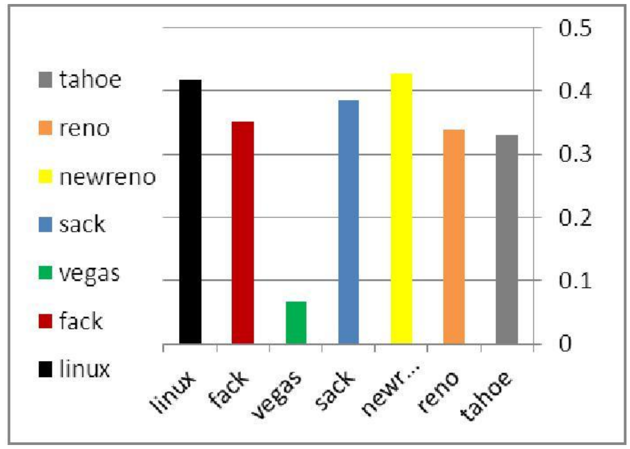


Figure 23: Average E2E2delay of all TCP versions with *TCPSink/Sack1/DelAck* as receiver

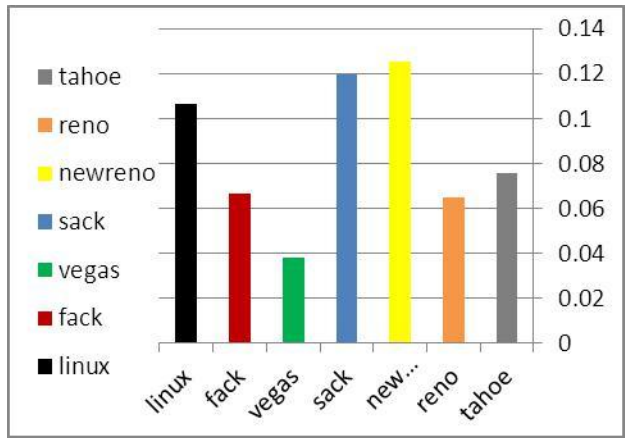


Figure 24: Average jitter of all TCP versions with *TCPSink/Sack1/DelAck*as receiver

## V. Conclusion

In MANET, the results explained that the *Vegas-TCP* version is the best in the terms of dropped packets, PDF, average end-to-end delay, and average jitter with *TCPSink*, *TCPSink/DelAck*,,TCPSink/Sack1,and *TCPSink/Sack1/DelAck* as receiver. While the *Fack-TCP* version is suitable for all above versions as receiver in term of NRL. In the case of the throughput, *Newreno-TCP* is the best version in the *TCPSink* as receiver, *Tahoe-TCP* is the best for the receiver type *TCPSink/DelAck*, *Fack-TCP* version is suitable for *TCPSink/Sack1*, and *Linux-TCP* is better for *TCPSink/Sack1/DelAck* as receiver.

## References

[1]. P.Periyasamy and E.Karthikeyan, "PERFORMANCE EVALUATION OF AOMDV PROTOCOL BASED ON VARIOUS SCENARIO AND TRAFFIC PATTERNS", International Journal of

Computer Science, Engineering and Applications (IJCSEA) Vol.1, No.6, December 2011.

[2]. Teerawat Issariyakul and Ekram Hossain, "Introduction to Network Simulator NS2", springer, ISBN: 978-0-387-71759-3, 2009.

[3]. Kevin Fall and KannanVaradhan, "The ns Manual (formerly ns Notes and Documentation)", 2010.

[4]. Muhammad Ijaz and Adrian Popescu, “TRASMISSION CONTROL PROTOCOL (TCP) PERFORMANCE EVALUATION IN MANET”, BLEKINGE INSTITUTE OF TECHNOLOGY, 2009.

[5]. Nadim Parvez, Anirban Mahanti, and Carey Williamson, An Analytic Throughput Model for TCP NewReno, IEEE/ACM Transactions on Networking, Vol. 18, No. 2, April 2010.

[6]. L.S. Brakmo and L.L. Peterson. TCP Vegas: End to End Congestion Avoidance on a global Internet. IEEE Journal on Selected Areas in Communications, 13(8):1465–1480,Oct 1995.

[7]. Tayade Mandakini, Sharma Sanjeev, “Review of Different TCP Variant in AD-hoc Networks”, International Journal of Engineering Science and Technology (IJEST), Chennai, 2011, pp.1906-1913.

[8]. AHMED JAWAD KADHIM, "Adding TCP-Variants to NS-2", Oriental Journal of Computer Science and Technology, Volume 06 No. 03 Page No. 363-368 Sep 2013.

## Authors Profile

**AHMED JAWAD KADHIM AL SHAIBANY** received his B.S. Degree in Computer Science from the Sciences College, Al-qadisyiah University, Iraq, 2007. M.S. degree in Computer Science from Sciences College, Babylon University, Iraq, 2012.