PERFORMANCE ANALYSIS OF TCP IN WIRELESS MULTI-HOP ENVIRONMENT

M. Shivaranjani[1] | A. Kameswaran[2]

[1](ECE Department, M.P.Nachimuthu M.Jaganthan Engineering College, Erode, India, ranjani093@gmail.com)
[2](ECE Department, M.P.Nachimuthu M.Jaganthan Engineering College, Erode, India, kameswaranme@gmail.com)

Abstract**—***Multi-hop technologies are recently gaining momentum due to its integration with wireless internet, Transmission Control Protocol (TCP), a consistent self-clocking transport protocol plays a decisive role in an Internet's performance as it carries huge quantum of internet data traffic globally. TCP's performance over multi-hop wireless networks remains a challenging concern because of its characteristics, such as mobility, sudden route breakage, link failure and high bit error rate. The foremost aim of this paper is to experimentally analyze and appraise the micro level behavior of different standard TCP variants which are installed in Windows and Linux kernel under multi-hop wireless environment. The micro level analysis of standard TCP's gives a stronger platform in designing a resilient TCP that adapts for highly dynamic multi-hop wireless environment.*

Keywords—Transmission Control Protocol (TCP), multi-hop wireless network, slow start, congestion window and window size

1. INTRODUCTION

Multi-hop wireless networks [2-6] are standalone distributed infrastructure less wireless networks that are interconnected through the relay of wireless nodes. Each node in the multihop network works in an autonomous fashion where every mobile node acts as a router and plays different role such as sender, receiver and relay nodes. A wireless node in a multi - hop network can communicate with other node beyond its transmission range by choosing yet other nodes as relay nodes which acts as routers. All the nodes in the network have the same capacity and there is no need of any access point for data transmission. In some special case, nodes in a multi - hop network acts as a gateway, which helps in connecting all other nodes in the network to communicate with external nodes via the internet.

Transmission Control Protocol [1] affords a reliable end-to-end data transmission over wired networks. TCP performs well on wired networks by adapting to the delay and packet losses caused by congestion. As the bit error rate (BER) is very low or negligible in wired networks, traditional TCP assumes that packet losses are only due to network congestion. In order to control the network congestion, TCP reduces its window size according to available bandwidth. In contrast, packet losses in wireless networks may arise due to error-prone wireless channels, MAC layer contention and route breakage which are not related to congestion. However, the past analysis [14-17] exposes that the performance of TCP severely degraded under multi-hop wireless environment due to its poor assumptions.

The prime intend of this experimental simulation analysis is to investigate the micro level behavior of exiting standard TCP variants implemented in Windows and Linux kernel under dynamically varying multi-hop wireless conditions. The remaining part of the paper is edited and arranged in the following way, Section II deals with the rich literature into the congestion control algorithms of standard TCP variants which are installed in Windows and Linux kernel (BIC, Cubic, NewReno, Compound, Vegas, Veno and Westwood), Section III gives an in-depth microscopic experimental analysis of these standard TCP variants under four different wireless scenario patterns of multi-hop wireless networks and Section IV draw the concluding remarks derived from the simulation experimental analysis.

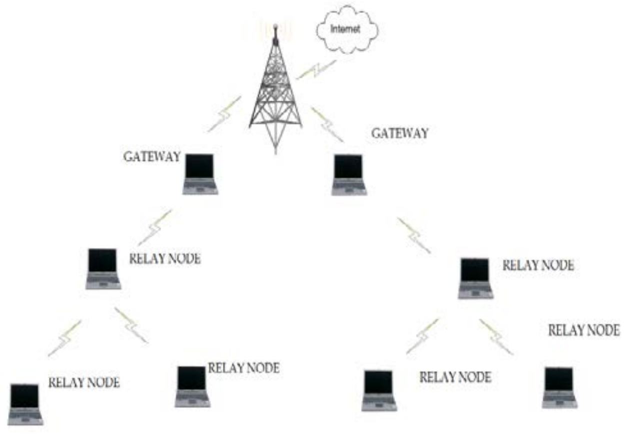


Fig. 1. Wireless multi-hop networks

2. RELATED WORK

This section gives a deeper insight into the congestion control algorithms of the following standard TCP variants such as BIC, Cubic, NewReno, Compound, Vegas, Veno and Westwood which are installed in Windows and Linux kernel and perceiving how they differ from each other.

A. New Reno

TCP New Reno [13] is an improved version of RENO, which is capable of detecting multiple segment loss. In slow start mode, the window size is increased exponentially for every received acknowledgement (ACK). New Reno identifies network congestion by packet loss due to Timeout and 3 duplicate ACK. When an ACK is not received within a specified time, New Reno assumes a packet drop is of network congestion and sets a slow start threshold (ssthresh) value as half of the current window size and kick off slow start. In the next RTT, the window size is increased exponentially up to ssthresh and then increased linearly above the threshold.

However, if the receiver didn't receive the expected packet, it will send duplicate ACK. When the sender continuously receives 3 duplicate ACK, it will enter into fast retransmit phase and retransmit the unacknowledged packet in the same RTT. In the next RTT, New Reno moves to a fast recovery mode where the window size is decreased by half and increased linearly.

B. BIC

Binary Increase Congestion Control (BIC) TCP [12] is designed to achieve RTT fairness, TCP friendliness, scalability and convergence. TCP BIC uses two algorithms to control window size called additive increase and binary search increase. When the congestion window is large, additive increase with a large increment is used to ensure RTT fairness and scalability. When the congestion window is small, the binary search increase is used to ensure TCP friendliness.

For every ACK received, if the cwnd is lesser than W_{max} , cwnd is increased by the difference between W_{max} and cwnd of half times. If cwnd is greater than W_{max} , cwnd is increased by the difference between cwnd and W_{max}. Else S_{min} is set to cwnd and cwnd is set to S_{max}. Where S_{min} and S_{max} minimum and maximum increase factor. Cwnd should be incremented within the limit. cwnd should increase at least by S_{min} for ACK received. When there is congestion, it checks the cwnd. If the cwnd is less than W_{max} , W_{max} value is changed as cwnd * (1 – ß / 2). If cwnd is greater than W_{max} , W_{max} is set to cwnd. After that, cwnd value is decremented by the factor of 1 – ß.

C. Cubic

In order to lessen the complexity of TCP BIC, TCP CUBIC [10] was introduced. As the name indicates, window growth is a function of the cube which shape function is similar to BIC but very less complexity. For every ACK received cwnd is modified as

$$cwnd = C(t - K)^3 + W_{max} \tag{1}$$

Where t is elapsed time from the last window reduction, K is set as $\sqrt[3]{W_{max}\beta/C}$, W_{max} is the highest window size β is the growth factor and C is a scaling factor. cwnd is maintained greater than the below value in order to maintain growth rate same as standard TCP in short RTT networks.

$$cwnd = \beta \quad W_{max} + 3 \quad \frac{1 \quad \beta}{1 + \beta} \quad \frac{t}{RTT} \tag{2}$$

When there is congestion due to timeout or by 3 duplicate ACK, the value of K is restructured and then W_{max} value is updated as follows.

$$K = \sqrt[3]{W_{max}\,\beta/C} \tag{3}$$

$$W_{max} = \beta * W_{max} \tag{4}$$

D. Compound

Compound TCP [11] is a sender-side TCP enhancement designed for high-speed and long distance networks. Compound TCP is a amalgamation of delay and loss-based congestion control to achieve RTT fairness, TCP fairness, efficiency and network stability. For a single flow, there exists two components cwnd and dwnd. cwnd is loss based component, which acts like cwnd in TCP Reno and dwnd is delay based component. Sender's sending window (win) size is based on these two values. *cwnd* is increased by one for every ACK received and decreased half upon a packet loss event.

The dwnd is restructured based on delay like TCP Vegas to detect early congestion in the network pathway. It calculates Actual Throughput, which is the ratio of cwnd to RTT and Expected Throughput, which is the ratio of cwnd to Base RTT (minimum of all measured RTT). Difference (DIFF) between Expected and Actual Throughput is calculated.

$$\text{Actual} = \text{win} / \text{RTT}$$
$$\text{Expected} = \text{win} / \text{baseRTT}$$
$$\text{Diff} = (\text{Expected} - \text{Actual}) * \text{Base RTT}$$

Congestion is detected when the number of packets in a queue is larger than a threshold γ. If diff < γ, the link is assumed as under-utilized otherwise, the link is assumed as congested. The dwnd is updated as below

$$\text{dwnd}(t+1) = \begin{cases} \text{dwnd}(t) + (\alpha * \text{win}(t)^k - 1) & \text{if diff} < \Upsilon \\ \text{dwnd}(t) - \zeta * \text{diffif diff} \geq \Upsilon & \\ \text{win}(t) * (1-\beta) - \text{cwnd}/2 & \text{if loss is detected} \end{cases} \tag{5}$$

E. Vegas

TCP Vegas [9] is a Transmission Control Protocol (TCP) which increases throughput and decrease packet loss. Network situation is measured by the RTT (Round Trip Time). The cwnd (Congestion Window) is increased or decreased by comparing the expected RTT and actual RTT.

Slow Start Mechanism tries to find out the current window size without incurring losses. It calculates Actual Throughput, which is the ratio of cwnd to RTT and Expected Throughput, which is the ratio of cwnd to Base RTT (minimum of all measured RTT). Difference (DIFF) between Expected and Actual Throughput is calculated. When the DIFF is less than γ, cwnd is increased exponentially that is for every ACK received in an RTT, cwnd is increased by 1. When DIFF is increased than γ due to queue buildup in Slow Start, Vegas goes to congestion avoidance phase.

When congestion occurs, Vegas calculates the product of DIFF and Base RTT, if the product is less than α, cwnd is increased linearly that is increased one for each RTT. If the product is greater than β, cwnd is decreased linearly. Otherwise, cwnd is maintained as constant. If it receives a duplicate ACK, it will resend the segment without waiting the 3 duplicate ACK. This reduces the time to detect the segment loss.

F. Veno

TCP Veno [8] modifies the sender side protocol of Reno and uses the same receiver side protocol of Reno. Veno calculates Actual Throughput and Expected Throughput as Vegas. It calculates the difference between Expected and Actual Throughput. The importance of TCP Veno is to use the measurement N not as a way to adjust the window size proactively, but as an indication of whether the connection is in a congested state. If N < β, when a packet loss is detected, Veno will assume the loss is due to random bit errors. Otherwise, Veno will assume the loss is due to congestion.

$$N = Actual * (RTT - BaseRTT) = Diff * BaseRTT \quad (6)$$

When cwnd is below ssthresh, cwnd is adjusted by Slow Start Algorithm. When cwnd is above ssthresh, cwnd is increased one for a each RTT. If the link is not utilized fully, for every ACK received cwnd is increased by 1. If the link is fully utilized, for every other ACK received cwnd is increased by 1 for each RTT.

When there is a segment missing at the receiver, Veno retransmit the packet and set the ssthresh as half of cwnd and the next cwnd as ssthresh +3. If a duplicate ACK receives increment the cwnd by 1 and if ACK is received set the cwnd as ssthresh.

G. Westwood

Westwood [7] is a sender side modification of the TCP algorithm. TCP Westwood attempts to select slow start threshold and congestion window which are consistent with the bandwidth used at the time of congestion experience. It uses available bandwidth estimation (ABE) algorithm, where the Sender side bandwidth is estimated by measuring and low pass filtering of the rate of returning ACKs.

Congestion may be measured by 3 duplicate ACK and time out. If 3 duplicate ACK is received, ssthresh is set as the product of Bandwidth and RTT divided to Maximum Segment Size (MSS) and cwnd is set as ssthresh. If the timeout expires, ssthresh is calculated as previous one but cwnd is set to 1 here instead of ssthresh.

When congestion is occurred due to timeout or 3 duplicate ACK, cwnd is decreased to 1 in timeout case and cwnd is decreased to ssthresh in 3 duplicate ACK case. After the decrement in window size, cwnd is exponentially increased until ssthresh in timeout condition. cwnd is linearly increased that is increased one for each RTT from ssthresh for both the cases. This phase is called Additive Increase Algorithm.

3. Simulation Analysis

The simulation experiments in this paper are conducted using the network simulator NS-2 [18]. Random Way Point mobility model for wireless environment condition was constructed using Bonnmotion [19] mobility scenario generator. The simulation runs with standardized TCP variants which are installed in Windows and Linux kernel like BIC, CUBIC, Compound, NewReno, Vegas, Veno and Westwood with traffic type of FTP. The finer parameter details of the simulation are given in the table 1.

TABLE I SIMULATION PARAMETERS

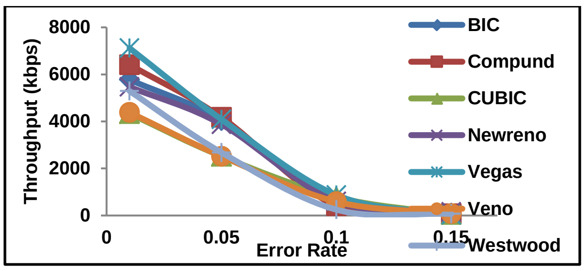
Simulation Parameter	
MAC	IEEE 802.11 b
Bandwidth	11 Mbit/s
Frequency	2.4 GHz
TCP data Packet Size	512 bytes
Window Size	128
Routing Protocol	AODV
Propagation Model	Shadowing
Simulation Area	850 m x850 m
Simulation Time	600.0 Sec

Four performance metrics such as throughput, well put, delay and dropped packets were chosen as per by RFC 5166 [20] to measure the effectiveness of the standardized TCP variants. The analysis was carried out by varying the parameter like error rate, mobility, no of nodes and no of traffic connections. A total of 112 simulation experiments were conducted under the multi-hop wireless conditions.

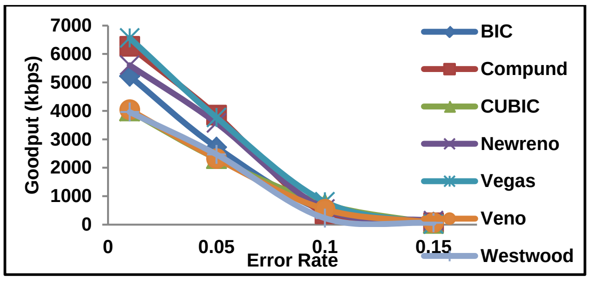
A. Error rate vs Metrics

The first analysis is to measure the performance of the standard TCP variants under different error rates. For more realistic simulation, mobility is varied from 1m/s to 5m/s. The graph (fig 2) shows that the average throughput degrades to 98.63%, average goodput degrades to 98%, packet drop increases to 89.55% and end to end delay increases to 72.03%, when the error rate varies from minimum to maximum condition.

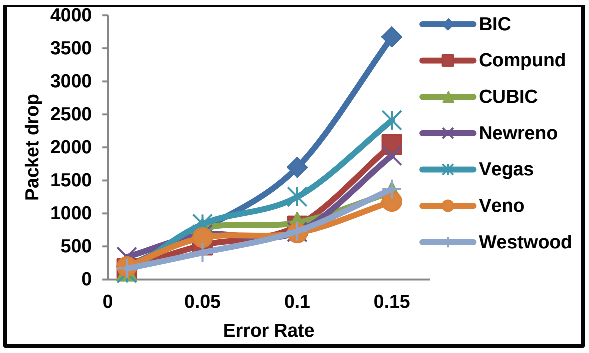
Higher wireless error rate introduces severe packet drop in the wireless network. This is because TCP fails to distinguish between packet drop due to congestion from packet drop due to wireless errors and falsely reducing the sending rate. Higher error rates leads to frequent initiation of slow start and thereby decreasing the performance of TCP.



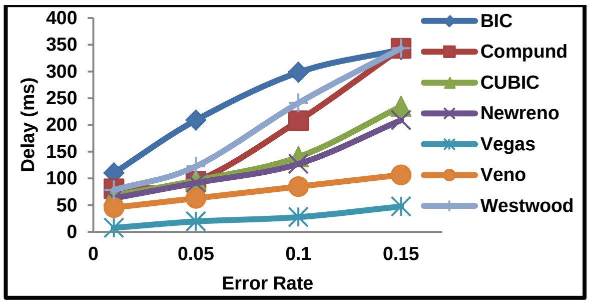
2(a)



2(b)

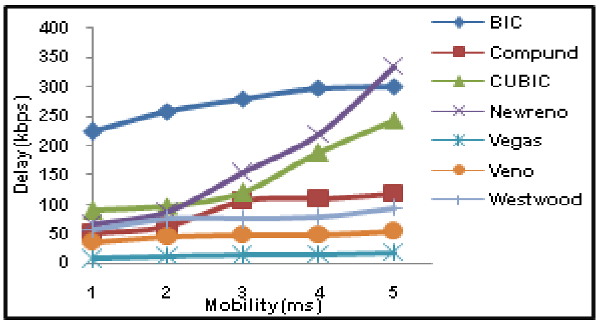


2(c)



2(d)

Fig. 2(a) Error rate vs Throughput, (b) Error rate vs Goodput, (c) Error rate vs Packet dropped and (d) Error rate vs Delay.

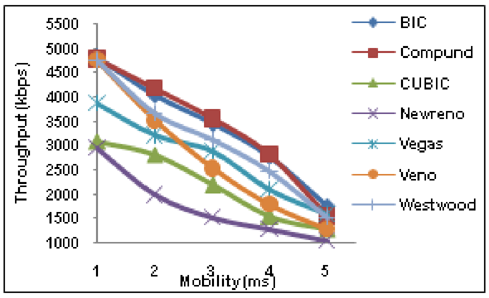


3(d)

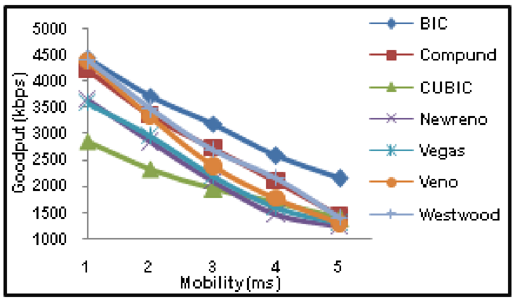
Fig. 3(a) Mobility vs. Throughput, (b) Mobility vs Goodput, (c) Mobility vs. Packet dropped and (d) Mobility vs Delay

B. *Mobility vs Metrics*

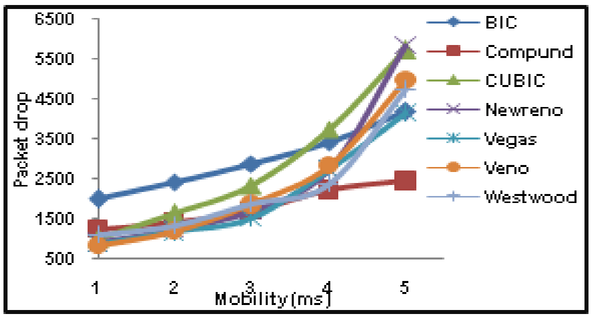
The second analysis is to measure the performance of TCP variants under different mobility conditions. The simulation is done with varying mobility from 1m/s to 5m/s. The graph (fig. 3) clearly shows that the average throughput degrades to 49.38%, average goodput degrades to 51.32%, packet drop increases to 59.93% and end to end delay increases 43.37%, when the wireless mobility varies from minimum to maximum condition.



3(a)



3(b)

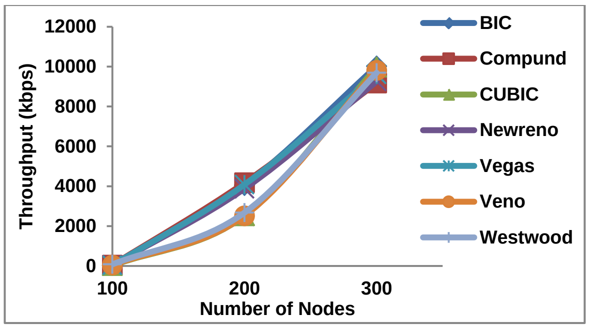


3(c)

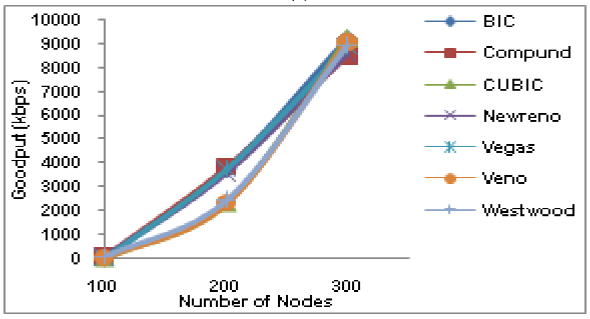
It is observed that the performance of TCP degrades at higher mobility conditions. At higher mobility conditions, frequent link breakage occurs, this leads to increase the packet drop and RTT. The congestion control algorithm of standard TCP variants misunderstands the packet drop due to mobility as packet drop because of congestion. This will initiate slow start most frequently, which leads to the least utilization of available bandwidth and poor performance of TCP.

C. *No of nodes vs Metric*

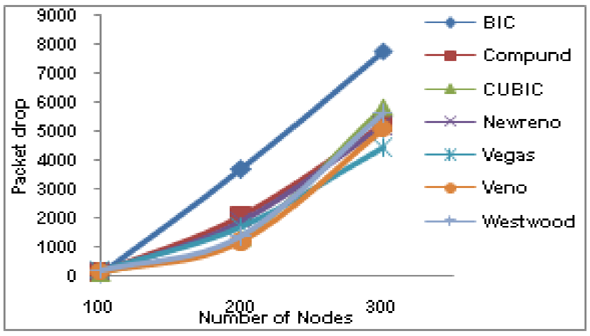
The third analysis model is made to measure the performance of TCP under sparse and dense wireless environment. For the simulation the number of nodes varies from 100 to 300. From the graph (fig. 4), the average throughput increases to 99.56%, average goodput increases to 99%, packet drop degrades to 96.99% and end to end delay decreases to 63.86%, when the environment changes from sparse to dense condition.



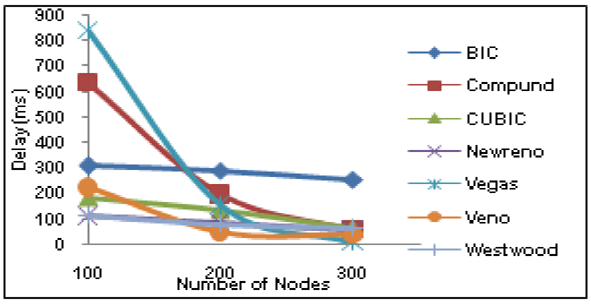
4(a)



4(b)



4(c)



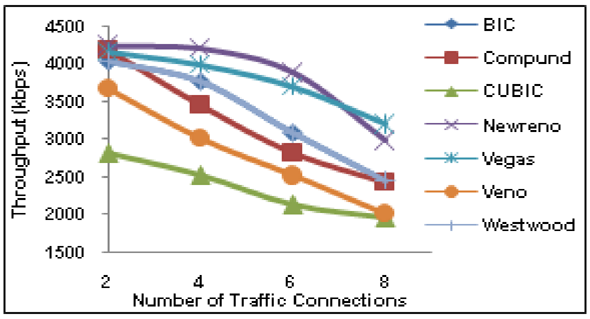
4(d)

Fig. 4(a) No of nodes vs Throughput, (b) No of nodes vs Goodput, (c) No of nodes vs Packet dropped and (d) No of nodes vs Delay

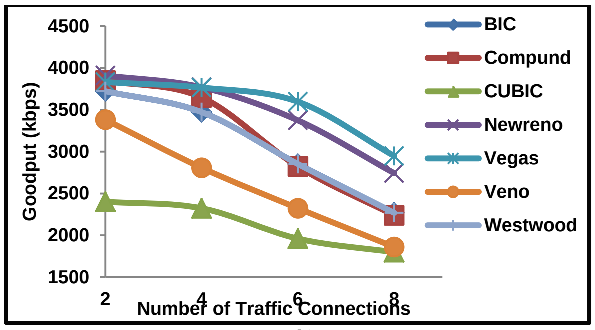
From the simulation, it is observed that TCP performance is better in the dense environment than the sparse environment. In sparse condition, frequent link failure leads to performance degradation of TCP. In contrast, TCP performance is better because of improved connectivity due to a larger number of nodes in dense environments.

D. *No of traffic connections vs Metrics*

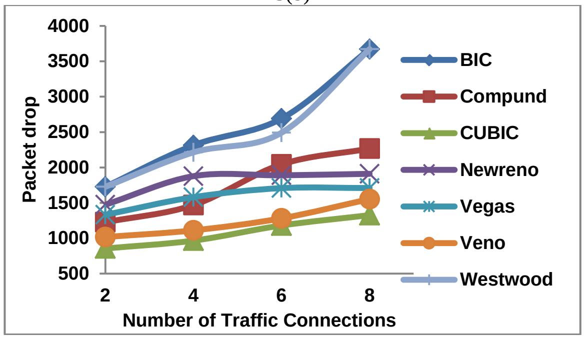
The final analysis model is to measure the traffic management capability of the wireless nodes. Number of active TCP connections varied from 2 to 8. From the graph (fig. 5), the average throughput decreases to 35.44%, average goodput decreases to 34.72%, packet drop increases to 38.11% and end to end delay 36.43%, when the active traffic connections varied from low to high.



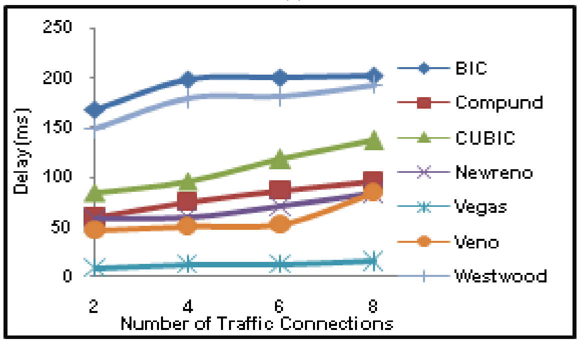
5(a)



5(b)



5(c)



5(d)

Fig. 5(a) No of traffic connections vs Throughput, (b) No of traffic connections vs Goodput, (c) No of traffic connections vs Packet dropped and (d) No of traffic connections vs Delay

From the simulation, it is observed that TCP performance is better in low traffic conditions than the high traffic conditions. At higher traffic the intermediary nodes exceed its processing capability and buffer size that leads to packet drop. There is no feedback mechanism available in standard TCP variants to notify the queue build up to sender.

4. CONCLUSION

In this paper, a micro level analysis of standard TCP implemented in Windows and Linux kernel were conducted to study the behavior of congestion control algorithm in multi-hop environment. The extensive simulation results of standard TCP variants demonstrate the performance degradation in wireless multi-hop network due to TCP's inability to recognize the difference between link failure and congestion. The standard TCP tested with four different models, the following suggestions were made from the analysis

i) Congestion algorithm should be modified with an ability to distinguish packet drop from congested link to wireless errors

ii) Conventional queue should be modified with feedback mechanism to alert sender

REFERENCES

[1] Allman, V. Paxson and E. Blanton, "TCP Congestion Control", RFC 5681, 2009.

[2] Mario Gerla and Leonard Kleinrock, "Vehicular networks and the future of the mobile internet," Elsevier Journal on Computer Networks, Vol. 55, no. 2, pp. 457–469, Feb. 2011.

[3] M.A. Shah, Sijing Zhang and C. Maple, "Cognitive radio networks for Internet of Things: Applications, challenges and future," IEEE Int. Conf. Automation and Computing (ICAC), London, pp. 1 – 6, Sept. 13-14, 2013.

[4] Marco Conti and Silvia Giordano, "Multihop Ad Hoc Networking: The Reality", IEEE Communications Magazine, 45 (4) , pp. 88-95, 2007.

[5] Larry Stotts, Scott Seidel, Tim Krout and Paul Kolodzy, "MANET Gateways: Radio Interoperability Via the Internet, Not the Radio", IEEE Communications Magazine, Vol 46 (6), pp. 51 – 59, 2008.

[6] Fekri M. Abduljalil and shrikant K. Bodhe, "A Survey of Integrating IP Mobility Protocols and Mobile Adhoc Networks", IEEE Communications Surveys & Tutorials, Vol 9 (1), pp. 14-30, 2007.

[7] Claudio Casetti, Mario Gerla, Saverio Mascolo, M.Y. Sanadidi and Ren Wang, "TCP Westwood: End-to-End Congestion Control for Wired/Wireless Networks", Wireless Networks 8, pp. 467–479, 2002.

[8] Cheng Peng Fu and Soung C. Liew, "TCP Veno: TCP Enhancement for Transmission Over Wireless Access Networks", IEEE Journal On Selected Areas In Communications, Vol. 21(2), 2003.

[9] Lawrence S. Brakmo, Larry L. Peterson, "TCP Vegas: End to End Congestion Avoidance on a Global Internet", IEEE Journal on Selected Areas in Communications, Vol 13, No 8, 1995.

[10] Injong Rhee, and Lisong Xu, "CUBIC: A New TCP-Friendly High-Speed TCP Variant", ACM SIGOPS Operating Systems Review - Research and developments in the Linux kernel, Vol 42(5), pp. 64-74, 2008.

[11] Kun Tan Jingmin Song, Murari Sridharan and Cheng-Yuan Ho, "CTCP: Improving TCP-Friendliness over Low Buffered Network Links", Microsoft Technical Report, 2008.

[12] Lisong Xu, Khaled Harfoush, and Injong Rhee, "Binary Increase Congestion Control (BIC) for Fast Long-Distance Networks", INFOCOM 2004. Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies, Vol 4, pp. 2514 – 2524, 2004.

[13] S. Floyd, T. Henderson and A. Gurtov, "The NewReno Modification to TCP's Fast Recovery Algorithm", RFC 3782, 2004.

[14] Joseph Auxilius Jude, Revathi Ganesan, "A Comprehensive Experimental Analysis of Standard TCP Variants in Vehicular Environment", In Proc. IEEE ICCCT'15, pp. 350-355, Feb. 2015.

[15] C. Callegari, S. Giordano, M. Pagano and T. Pepe, "Behavior analysis of TCP Linux Variants", Computer Networks 56, pp 462 – 476, 2012.

[16] Adnan Majeed, Nael B. Abu-Ghazaleh, Saquib Razak and Khaled A. Harras (2012), Analysis of TCP performance on multi-hop wireless networks: A cross layer approach, Ad Hoc Networks 10, 586–60.

[17] Gavin Holland and Nitin Vaidya, "Analysis of TCP Performance over Mobile Ad Hoc Networks", Wireless Networks 8, pp. 275–288, 2002.

[18] NS Simulator, http://www.isi.edu/nsnam /ns

[19] Bonnmotion, http://sys.cs.uos.de/bonnmotion/

[20] S. Floyd, Metrics for the Evaluation of Congestion Control Mechanisms, RFC 5166, 2008.

BIOGRAPHY

M.Shivaranjani received her Bachelor of Engineering in Electronics snd Communication Engineering in 2014 at Kongu Engineering College, Erode and received her Master of Engineering in Communication Systems in 2016 at Kongu Engineering College. Currently she is working as an Assistant Professor in the department of Electronics and Communication Engineering in M.P. Nachimuthu M.Jaganathan Engineering College, Erode. Her research area includes Transmission Control Protocol and Wireless Multihop Networks especially Wireless Sensor Network and Vehicular Ad Hoc Network.

A.Kameswaran received his Bachelor of Engineering in Electronics and Communication Engineering in 2008 at M.P. Nachimuthu M.Jaganathan Engineering College, Erode and Master of Engineering in VLSI Design in 2011 at Sasurie College of Engineering. Currently he is working as an Assistant Professor in the department of Electronics and Communication Engineering in M.P. Nachimuthu M.Jaganathan Engineering College, Erode. His area of interest includes Digital Transmission Systems and Spread Spectrum Communication