2014 International Conference on Advances in ICT for Emerging Regions (ICTer): 204 - 209

An Efficient Transport Layer Protocol with Reduced Packet Losses and Minimized Retransmission Time in Wireless Networks

Dhammika H De Silva [#*1], R S De Silva [#2], S B Mahakumbura [#3], I.Murray [*4]
[#]*Department of Information Technology, Sri Lanka Institute of Information Technology*
Malabe, Sri Lanka
[1]dhammika.d@sliit.lk, [2]roshanedesilva1@gmail.com, [3]shankaultimate@gmail.com
[*] *Department of Electrical &Computer Engineering, Curtin University*
Western Australia
[1]dhammika.koggalap@postgrad.curtin.edu.au
[4]I.Murray@curtin.edu.au

***Abstract* - This research will design and develop a protocol that supports less packet drops and consumes a low retransmission time which is able to transmit data in a speedy and efficient manner within a wireless environment. This is vital in real time systems; specifically in way finding applications in order to gather critical information of patients, evade hazardous situations for disable persons, and receive alternative paths in navigation systems in instances of impediments. This research examines the existing Transport Layer Protocols (TLP's); mainly TCP since it has a high rate of data packet loss and high retransmission time when operating in wireless networks. This paper proposes two modified algorithms that solves high packet loss rate by avoiding buffer overflows and high retransmission time by applying a spaced hop methodology. The simulations are performed through the Network Simulator 2 software operating on the Linux platform (Ubuntu) and will graphically and statistically display the improvement achieved by the modifications.**

***Keywords* - Transmission Control Protocol, Wireless networks, Way finding applications, buffer overflows, packet loss, spaced hop, retransmission time**

I. INTRODUCTION

In the present day and age the necessity for wireless applications has grown immensely due to its mobility, easy implementation and speed. Efficient wireless networking is beneficial when focusing on Way finding applications [1]. During an operation a surgeon should be able to instantly view the site where the surgery is performed through the way finder and even a millisecond delay in the application could result in a life-threatening situation of the patient. Similarly if a blind person using a navigation app on his smart phone doesn't receive the correct data on time it may even cause an injury to him. Transmission Control Protocol (TCP) was originally designed for wired networks and due to its high recognition and performance it is now used in wireless networks as well. Nevertheless it has been known to perform poorly in this situation. TCP assumes all packet losses are due to congestion and triggers rate reduction whenever a packet loss is detected. And TCP still relies on end to end retransmission to provide reliable data transport which ultimately consumes more energy and bandwidth than in hop by hop retransmission. [2]. This research presents two modified enhanced algorithms that evidently reduce the rate of packet loss and high retransmission time to benefit Way finding applications within the wireless environment.

Alterations have only been done to the Transport Layer since considering the rest of the layers will result in various other conflicts.

TABLE I

COMPARISON OF TLP'S

	TCP	UDP	DCCP	SCTP
Connection Oriented	Yes	No	No	No
Reliable	Yes	No	No	Yes
Congestion Control	Yes	No	Yes	Yes
Delivery	Ordered	Unordered	Unordered	Both
Re transmission	End to End	End to End	End to End	End to End

Table I compares the behaviour of the main Transport Layer Protocols (TLP's) and among them TCP is more reliable which is why it is used in wireless networks. Thus the base of this research focuses on eliminating the issues of TCP only.

II. RELATED WORK

In the research conducted by Hala Eidaw *et al* [3], computer and wireless communication require internet accessibility at anytime and anywhere, this includes high-speed mobile station such as in speedy trains, fast moving cars as vehicle to infrastructure communication. This increased the development of numerous schemes concerning the need of smooth handover of the mobile nodes.

Kulkarni.P *et al* [4] states that Responsive transport protocols e.g. TCP, back-off on packet loss. This is because packet loss is by default treated as a sign of congestion. However losses in a wireless network may not only arise from congestion but also because of the inherent characteristics of the wireless medium. If a TCP source assumes a wireless loss as a loss due to congestion and reduces its transmission rate, it fails to take advantage of the available capacity in the network thereby attaining less throughputs than it would have achieved otherwise.

Simon Heimlicher et al [5] states that in "End-to-end vs. Hop-by-hop Transport under Intermittent Connectivity" that end-to-end connectivity are often intermittent, limiting the performance of end-to-end transport protocols.

In the research done by Sankara Krishnaswamy on "Wireless Communication Methodologies & Wireless Application Protocol" [6] it is termed that during a study, 70% of the users rejected the idea of WAP enabled phones. Some of the disadvantages of WAP clearly made the users to decide not to like this. Because of the misguided use of design principles from traditional Web design, the usability of the current WAP services is reduced considerably.

According to Ana-Belen Garcia-Hernando *et al* [7] in wireless environments, both congestion and bit error can cause packet losses that degrade end-to-end reliability, while packet losses imply a decrease in energy efficiency.

Liw Jia Seng *et al* [8] proposed a new algorithm that differentiates the type of packet loss accurately since currently protocols such as TCP cannot do so. The algorithm categorizes the packet loss and invokes correct correction mechanisms.

III. SIGNIFICANCE OF THIS RESEARCH

Wireless applications operating with high reliability and transmitting timely data is vital for navigational purposes of vision impaired persons. In order to travel from one destination to another safely, avoiding obstacles the user should be able to rely on the network application. Another use of such a system would be for vehicle navigation systems to speedily update the road map when a road block lies ahead and provide alternative routes. In terms of security threats, such systems will have less probability of being attacked since the data being transmitted through the network will have less value compared to data transmitted in a network connecting a bank and its customer.

IV. METHODOLOGY

The overall goal of this research is to measure the ability of the TCP protocol to react to the created network topology change while continuing to successfully deliver data packets to their destinations and reduce retransmission delay. The designed protocol evaluations are based on the simulation of wireless nodes forming an ad hoc network, non-moving about over a rectangular flat space for 150 seconds of simulated time. A rectangular space was chosen in order to enforce the use of longer routes between nodes than in a square space with equal node density. For simulation purposes the default window size has been set to 20bytes in NS2. The default bandwidth set in NS2 is 1Mbps while packet transmission interval is generally set to 0.01 (100 packets per second) and background noise has been added to the algorithm to simulate a near to realistic condition. The algorithms have been developed and tested using the NS2 software operating in Ubuntu.

Table II displays the parameters used in the algorithms to generate realistic simulations while Figure 1 illustrates the methodology structure of this project.

TABLE II

SIMULATION PARAMETERS

Parameter	Value
Network	Wireless
Routing Protocol	AODV
Maximum packets in IFQ	50
No of nodes	5,10,20,50
Queue Type	Queue/Drop Tail/PriQueue
Antenna Type	Omni Antenna
MAC protocol	802.11
Simulation time(s)	150
Traffic Type	CBR

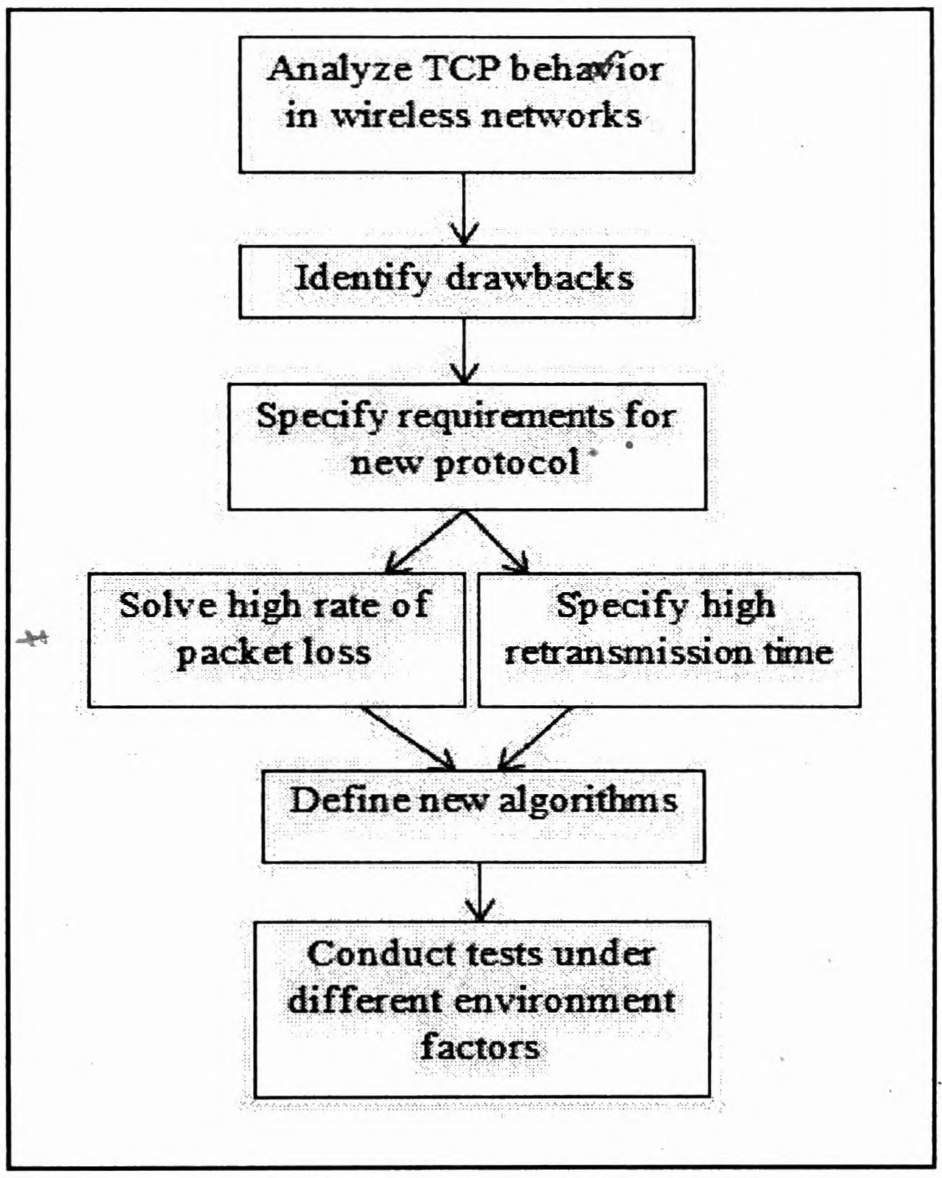

Fig. 1 Methodology of project

The algorithms begin by initialising the wireless topology variables and creating an instance for a new simulator. Next the GOD (General Operations Director) object is created to store information about the network. Further the necessary nodes are made, no of connections defined and the nodes are attached to TCP and Sink agents in order to setup traffic connections between the sender and the receiver. Lastly the time of simulation is specified and algorithm completed. In the packet loss algorithm the buffer size is set while in the retransmission algorithm, the no of intervals to store data is specified as well.

The results of number of packets lost, end to end delay times and throughput values are generated using .awk files given below in figures 2 and 3.

```
BEGIN {
seqno = -1;
droppedPackets = 0;
receivedPackets = 0;
count = 0;
}
{
#packet delivery ratio
if($4 == "AGT" && $1 == "s" && seqno < $6) {
seqno = $6;
} else if(($4 == "AGT") && ($1 == "r")) {
receivedPackets++;
} else if ($1 == "D" && $7 == "tcp" && $8 > 512){
droppedPackets++;
}
#end-to-end delay
if($4 == "AGT" && $1 == "s") {
start_time[$6] = $2;
} else if(($7 == "tcp") && ($1 == "r")) {
end_time[$6] = $2;
} else if($1 == "D" && $7 == "tcp") {
end_time[$6] = -1;
}
}
END {
for(i=0; i<=seqno; i++) {
if(end_time[i] > 0) {
delay[i] = end_time[i] - start_time[i];
count++;
}
else
{
delay[i] = -1;
}
}
for(i=0; i<count; i++) {
if(delay[i] > 0) {
n_to_n_delay = n_to_n_delay/count;
print "\n";
print "GeneratedPackets = " seqno+1;
print "ReceivedPackets = " receivedPackets;
print "Packet Delivery Ratio = " receivedPackets/(seqno+1)*100
"%";
print "Total Dropped Packets = " droppedPackets;
print "Average End-to-End Delay = " n_to_n_delay * 1000 " ms";
print "\n";
}
```

Fig. 2 awk file for packet loss and end to end delay

```
BEGIN {
      recvdSize = 0
      startTime = 400
      stopTime = 0
  }
    {
          event = $1
          time = $2
          node_id = $3
          pkt_size = $8
          level = $4

  # Store start time
  if (level == "AGT" && event == "s" && pkt_size >= 200) {
    if (time < startTime) {
          startTime = time
          }
      }
    # Update total received packets' size and store packets
arrival time
  if (level == "AGT" && event == "r" && pkt_size >= 200) {
      if (time > stopTime) {
          stopTime = time
          }
      # Rip off the header
      hdr_size = pkt_size % 200
      pkt_size -= hdr_size
      # Store received packet's size
      recvdSize += pkt_size
      }
```

Fig. 3 awk file for throughput

A. Methodology of solving high rate of packet loss

Currently, a TCP sender considers all losses as congestion signals and reacts to them by throttling its sending rate. With Internet becoming more heterogeneous with more and more wireless error-prone links, a TCP connection may unduly regulate its sending rate and experience poor performance over paths experiencing random losses unrelated to congestion. An algorithm known as the IAD scheme to differentiate packet loss types using a standard deviation for the packet inter-arrival time has previously been established [8].

Since buffer overflow is one main cause of wireless packet drops, firstly the default TCP algorithm was tested to analyse the general rate of packet drops. The algorithm was tested based on the number of nodes and number of connections. In here background data was added as well. Next the algorithm was modified to address the buffer overflow issue by adjusting the receiver window size accordingly. Table III explains the comparison of the background environment created for both testing sessions of the default and modified procedures.

TABLE III

COMPARISON OF BACKGROUND DATA

Background Data	Default TCP	Modified TCP
No of nodes	5,10,20,50	5,10,20,50
Window size(bytes)	20	200
Bandwidth (mbps)	1	2
Delay(mbps)	0.01	0.50
Noise	added	added

B. Methodology of solving high retransmission time

Traditional transport protocols perform poorly in wireless environments, especially in multi-hop scenarios. Many studies have shown that TCP, the universal transport protocol for reliable transport. The end-to-end retransmission delay is a major problem in network communication so the primary goal of this research is to come up with a solution to reduce end-to-end retransmission delay. As a solution, the hop-by-hop method for network communication was introduced. [9]

Firstly the default TCP algorithm was tested again in order to analyse the output of the end to end delay scenario in 5, 10 and 20 node topologies.

Next it was needed to test the algorithms using the hop by hop approach as well. Therefore the algorithm was modified and run in the NS2 software.

As the second method did not provide results to prove it better than end to end delay method, a new solution was proposed which is the spaced hop method. This technique does not store data in every node; rather it keeps an interval of 2 nodes and stores the data in the 3rd node. Once again this was tested in NS2 as well and results were analysed as to which was more efficient.

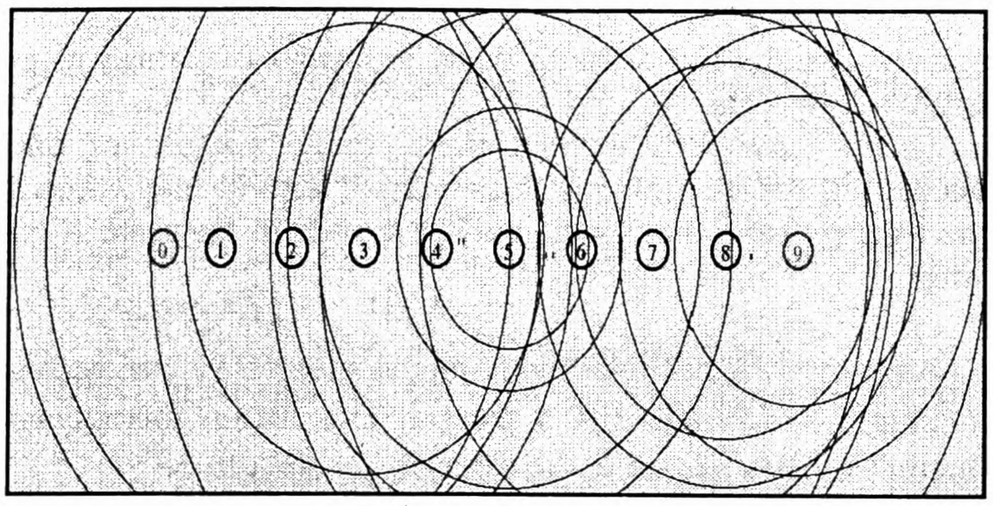

Fig.4 end to end delay scenario

Figure 4 illustrates the node pattern when the end to end simulation is run where the nodes are stored and forwarded from node 0 to node 9 within the 10 node topology.

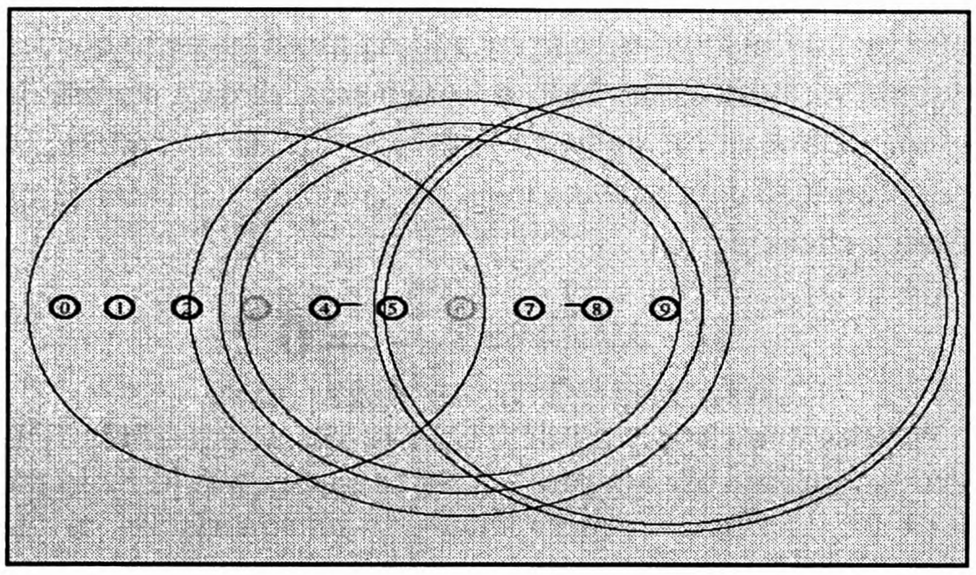

Fig 5 spaced hop approach

Figure 5 is the spaced hop approach that stores data in the 0^{th} node, 3^{rd} node, 6^{th} and 9^{th} nodes respectively unlike in the end to end situation.

V. RESULTS

A. Solution for high rate of packet loss

As the aim of this research is to decrease packet drops in wireless environment, the first testing was performed to observe the pattern of packet losses against an increasing no of connections in the default TCP environment. In here the no of connections were set to twice the amount of nodes created which is given in table IV.

TABLE IV

PARAMETERS FOR SIMULATION

No of nodes	No of connections
5	10
10	20
20	40
50	100

As a result the figure 6 illustrates the increasing rate of packet loss when the no of connections made increases simultaneously.

Such a situation is unhealthy since in the real world the number of wireless connections made at a time cannot be defined beforehand. Hence the network should be able to transfer the data to multiple connections concurrently without much losses taking place.

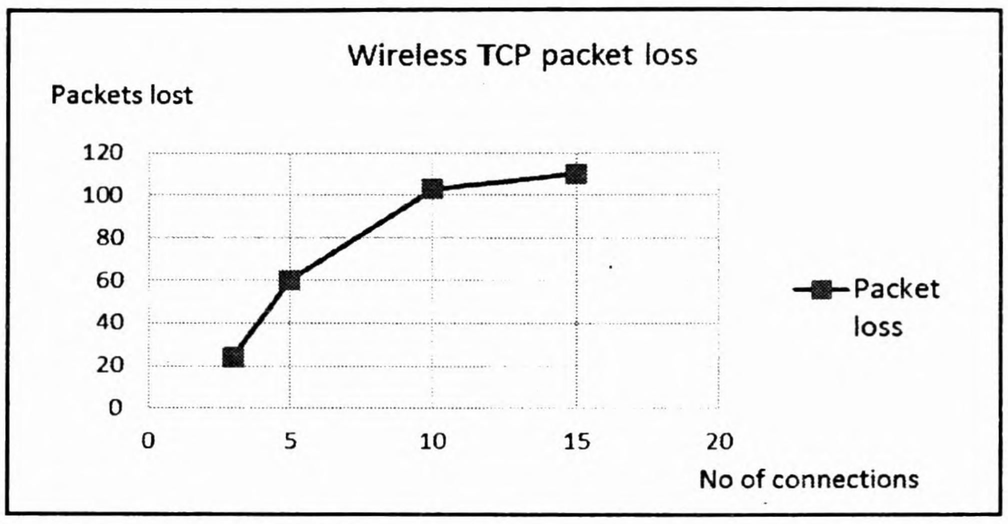

Fig 6 increasing packet loss in default TCP

The next step was to increase the receiver window size from default 20 bytes to 200 bytes and analyse the reduction in the packet loss for the same number of connections. The figure 7 is the comparison between the default TCP and modified algorithm packet loss rate. In here it can clearly be seen that the packet losses are comparatively lower than the normal TCP scenario.

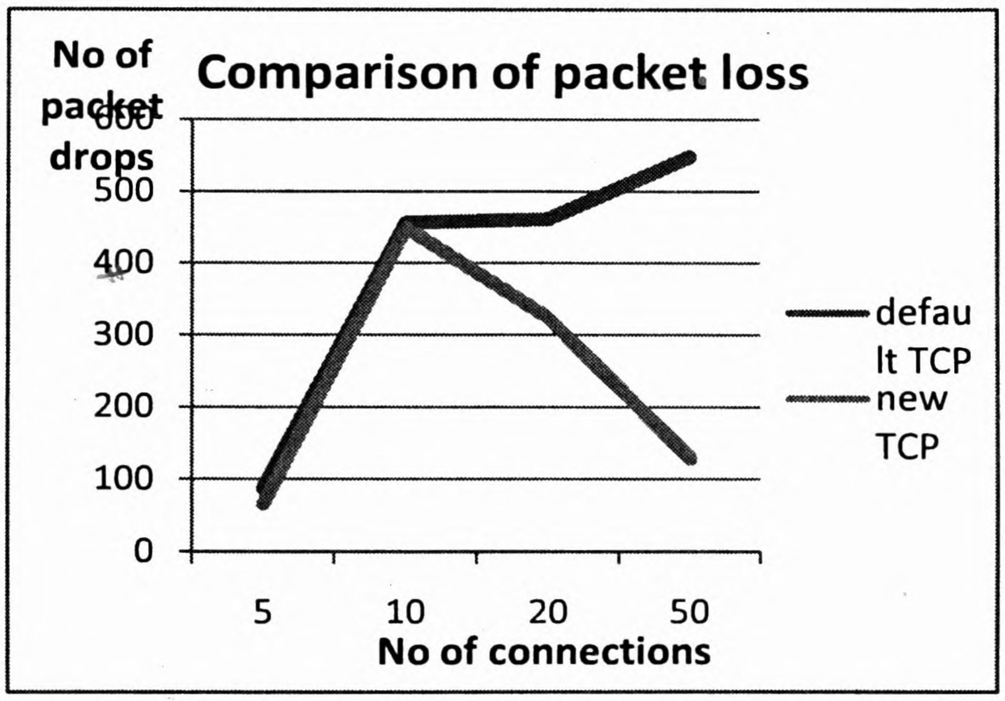

Fig 7 Comparison of rate of packet loss

In here the X axis represents the number of connections made while the Y axis represents the no of packet drops. The blue line in the graph shows the increasing linear curve of the default TCP algorithm and the red decreasing linear curve is the modified algorithm. Even though the level of packet losses rise at the beginning for less no of connections, eventually as the number of connections increase, the modified algorithm's curve decreases. This proves that the buffer overflow situation has been controlled in a near to realistic environment.

Apart from solving buffer overflow issue and addressing high rate of packet loss, this research further tested this scenario to examine if the result caused any adverse effects on the overall throughput of the transmission. The following table V shows the minor differences in the throughputs before and after modifications were done.

TABLE V

COMPARISON OF THROUGHPUT

No of connections	Default TCP	New Algorithm
5	802.07mbps	976.11mbps
10	1926.48mbps	1905.00mbps
20	2105.49mbps	2107.51mbps
50	655.75mbps	767.55mbps

B. Solution for high retransmission time

(i) Analyzing End-to-end parameters of TCP

1460kb (1.4mb) size data packets were sent across the 10 node topology 10 times to generate 10 trace files. Average value of end-to-end retransmission delay is 757.472ms. The results generated were as follows. To reduce this delay value the next tests were done.

Start Time	=	0.02000
Generated Packets	=	967
Received Packets	=	753
Packet Delivery Ratio	=	77.8697
Total Dropped Packets	=	21
Average End-to-End Delay [ms]	=	**757.472**
Average Throughput [kbps]	=	822.31
Stop Time	=	9.99000

Fig 8 End to end delay results

(ii) Analyzing Hop-by-hop parameters of TCP

To reduce the end-to-end retransmission delay in TCP, hop-by-hop implementation was done. It is assumed that all hops are independent and all nodes are behaving in the hop-by-hop manner therefore, we can model hop-by-hop transmission based on mean-value analysis.

Start Time	=	0.02
Generated Packets	=	1537
Received Packets	=	1167
Packet Delivery Ratio	=	75.9271
Total Dropped Packets	=	55
Average End-to-End Delay [ms]	=	**803.298**
Average Throughput [kbps]	=	762.36
Stop Time	=	10.00

Fig 9 Hop by hop results

At this step it was anticipated that end-to-end retransmission delay will reduce when this hop-by-hop approach was used. However, based on these outputs, in the actual situation the hop-by-hop implementation takes comparably more time than the end to end scenario.

Retransmission time (hop-by-hop): 803.298ms
Retransmission time (end-to-end): 757.472ms
Difference (803.338ms-757.472ms): **45.866ms**

Every node has to store, forward and check if the data packet is sent successfully to the next hopping node and if not successfully delivered, it resends each and every data packet. This is the queuing process, because of this, the queuing delay [9][10] occurs which is the reason Hop by hop retransmission is more time consuming than End to end retransmission. The queuing delay is further explained using Little's Law. [11]

Little's Law tells us that the average number of data packets in the queue L, is the effective arrival rate λ, times the average time that a data packet spends in the store W, or simply:

$$L = \lambda W$$

Assume data packets arrive at the rate of 10 per second and stays an average of 0.5 second. This means the average number of data packet in the store at any time to be 5.

$$L = 10 \cdot 0.5 = 5$$

Now suppose the arrival rate increases to 20 per second. The queue must either be prepared to host an average of 10 occupants or must reduce the time each data packets spend in the store to 0.25 second. The queue might achieve the latter by processing faster or by adding more space to queue. Assume we notice that there are on average 2 data packets in the queue and at the counter. We know the arrival rate is 10 per second, so data packets must be spending 0.2 seconds on average checking out.

$$W = \frac{L}{\lambda} = \frac{2}{10} = 0.2$$

We can even apply Little's Law to the queue itself. The average number of data packets at the queue would be in the range (0, 1) since no more than one person can be at the queue at a time.

In end-to-end situations queuing delay is neglected [12][13], but in the hop-by-hop situations queuing delay is the main reason behind high retransmission time.

Retransmission delay in End-to-end protocols =
N [TransDelay+PropDelay+ ProcDelay + ~~QueingDelay~~]
[12]

Retransmission delay in Hop-by-hop protocols =
N [TransDelay+PropDelay+ProcDelay+QueingDelay]
[12]

Due to these issues, finally a conceptual solution is presented in this research known as the spaced hop-by-hop concept.

All nodes do not act as hops. Instead the algorithm is set for every 3 nodes to act as hops and store data. The result should express the retransmission delay three times less than when in end to end retransmission. This is the most important outcome of this research. The result of the spaced hop approach is shown in figure 10.

Start Time	=	0.01
Generated Packets	=	1350
Received Packets	=	1237
Packet Delivery Ratio	=	91.6296
Total Dropped Packets	=	29
Average End-to-End Delay [ms]	=	**315.22**
Average Throughput [kbps]	=	770.26
Stop Time	=	9.9800

Fig 10 Spaced hop results

Therefore the conclusion could be shown as follows.

Retransmission time (hop-by-hop) :803.298ms
Retransmission time (end-to-end) :764.845ms
Retransmission (Spaced hop-by-hop): **315.228ms**

Ratio of hop-by-hop and spaced hop-by-hop of TCP can be calculated as

803.338ms / 315.228ms = **2.542ms**

This implementation was done for TCP-Reno and the output was

Retransmission time (hop-by-hop) :803.338ms
Retransmission time (end-to-end) :757.472ms
Retransmission (Spaced hop-by-hop) : **272.220ms**

Ratio of hop-by-hop and spaced hop-by-hop of TCP-Reno can be calculated as

803.338ms / 272.220ms = **2.9510616ms**

According to the proposition, both TCP and TCP Reno values are near to the assumed value of 3ms mentioned formerly since it is an improved version of TCP. The following table VI summarizes the results generated as well as the comparison of throughput.

TABLE VI

COMPARISON OF THROUGPUT

No of no-des	End-to-End		Hop-by-Hop		Spaced Hop-by-Hop	
	Delay (ms)	T-put	Delay (ms)	T-put	Delay (ms)	T-put
5	652.503	850.99	322.9 24	765.66	261.386	768.34
10	757.472	833.31	803.3 38	762.36	315.989	770.26
20	735.799	688.95	955.0 50	831.53	319.672	752.35

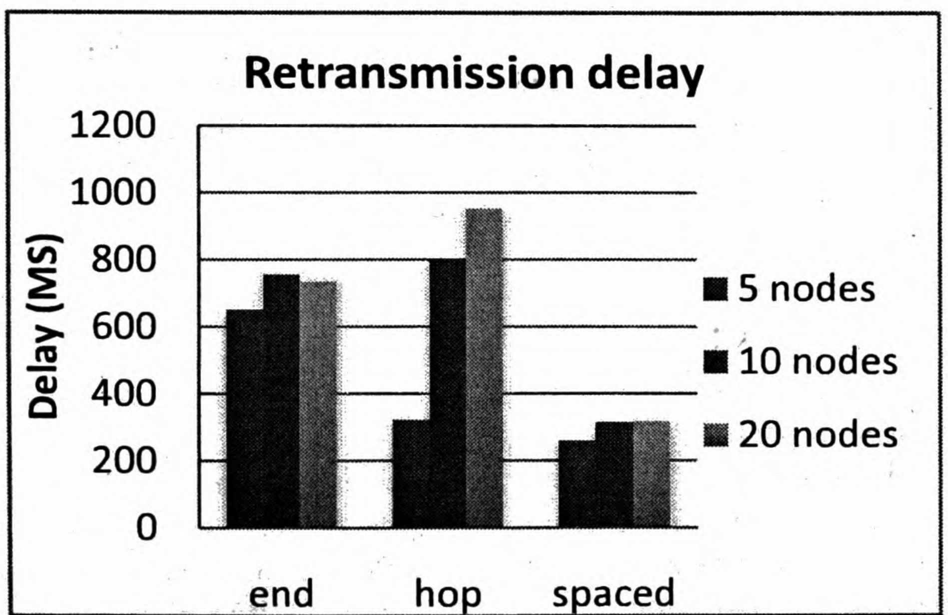

Fig 11 Comparison of all three techniques

Figure 10 illustrates the differences in retransmission time of all three methods tested and it can be clearly seen that the spaced hop method has far less time delay than the other two methods. Therefore the spaced hop method has proven to be an efficient technique to solve high retransmission time of TCP in wireless networks.

VI. CONCLUSIONS

This research presents an efficient enhanced algorithm that achieves low rate of packet losses and minimized retransmission time over wireless networks. However this research will continue on eliminating the other main drawbacks of TCP over wireless networks which are known as unnecessary congestion occurrences and high connection establishment time. Upon completion of this research, it will be of great importance in the field of Way finding applications to operate with high stability and speed.

ACKNOWLEDGMENT

This research is an on-going project focused on eliminating four key drawbacks of TCP and designing an efficient Transport Layer protocol suited for Wireless environment and the authors would like to thank staff at Sri Lanka Institute of Information Technology for their support.

REFERENCES

[1] Dhammika H De Silva, Dr.Ian Murray "An investigation in to Dynamic TLP's for Smartphone Communication", in International Conference on Advances in ICT for emerging regions ICTer, 2013, p.194 – 197
[2] C.Wang, K.Sohraby, B.LiM, Daneshmund Y.Hu, "A survey of transport protocols for wireless sensor networks", IEEE special issue on wireless sensor networks,Vol.20,pp 34-40,May,2006
[3] Hala Eldaw Idris Jubara, Sharifah Hafizah Syed Ariffin, Shiela N Fisal, Nurul Muazzah Abdul Latiff, Sharifah K Syed Yusof, Rozeha Rashid, (2012) Adaptive transport layer protocol for highly dynamic environment)[online]Available: http://jwcn.eurasipjournals.com/content/2012/1/229
[4] P.Kulkarni, M.Sooriyabandara, Li Lu "Improving TCP Performance in Wireless Networks by Classifying Causes of Packet Losses", in Wireless Communications and Networking Conference, 2009. WCNC 2009. IEEE, 2009, p.1
[5] Simon Heimlicher, Merkouris Karalipoulos, Hanoch Levy, Martin May, "End to end vs. hop by hop transport under Intercommitent Connectivity" in 1st International conference on Autonomic computing and communication systems, 2007, p.2-11
[6] Sankara Krishnaswamy (2001) "Wireless communication and Methodologies & Wireless Application Protocol", [online] Available: www.rivier.edu/faculty/vriabov/cs553a_Project_SKrishnaswamy.pdf
[7] Ana-Belen Garcia-Hernando, Juan-Manuel Lopez-Navarro, Jose-Fernan Martinez-Ortega, Aggeliki Prayati, Luis Redondo-Lopez, Problem Solving for Wireless Sensor Networks ,2008
[8] Liw Jia Seng, Mohd Noor Derahman and Azizol Abdullah, "Loss Discrimination Algorithm for Wired/Wireless Networks", Journal of Computer Science 7, Vol 7,pp. 1798-1804,2011
[9] V.Sundarapandian, Queueing Theory". *Probability, Statistics and Queueing Theory*. 2009
[10] Dimitri P. Bertsekas ,Robert G. Gallager, *Data Networks*, 2nd, United States of America, 1992
[11] Kandeepan Sithamparanathan, Mario Marchese, Marina Ruggieri, Igor Bisio, *Personal satellite services*, 2010
[12] Keith W. Ross; James F. Kurose, *Delay and Loss in Packet-Switched Networks*, 2000
[13] García-Hernando, A.-B., Martínez-Ortega, J.-F., López-Navarro, J.-M., Prayati, A., Redondo-Lopez, L, Computer Communications and Networks Series: Problem solving for wireless sensor networks, 2009