BONeS TCP PRIMITIVE MODULE VALIDATION

Georgios Y. Lazarou, Victor S. Frost

Technical Report TISL-10980-06

Telecommunication and Information Science Laboratory
The University of Kansas
Phone: (913) 864-4832
Fax : (913) 864-7789

Abstract

KU is responsible for developing techniques to predict the performance of the AAI. A Transmission Control Protocol (TCP) BONeS primitive module is created for modeling the AAI system. This report presents the validation results of this TCP BONeS primitive module.

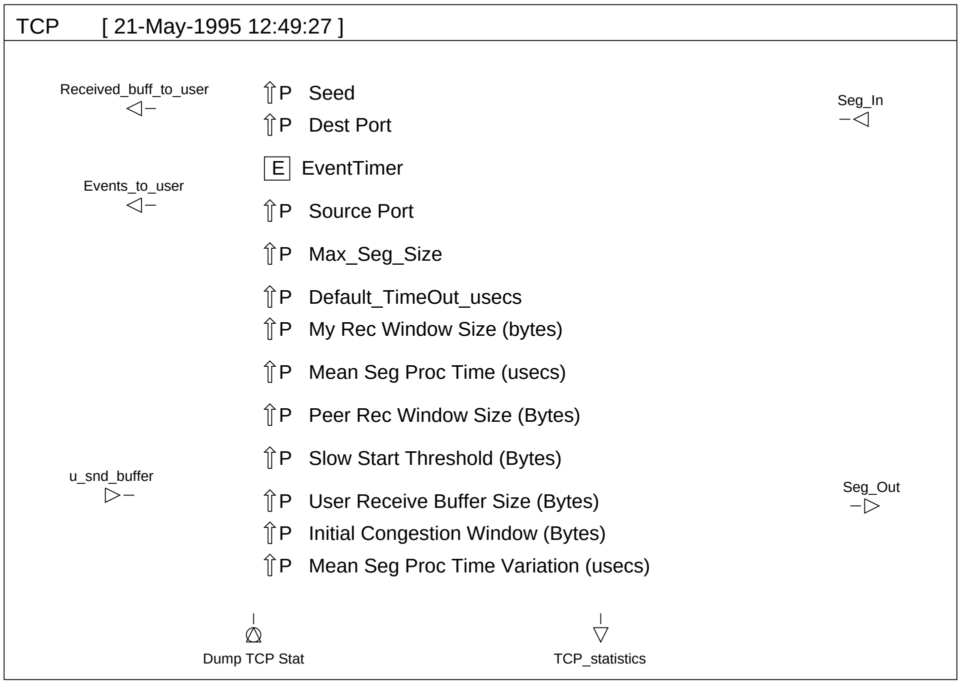

Figure 1: TCP BONeS primitive module interface

1 Introduction

This report presents the validation results of the Transmission Control Protocol (TCP) BONeS primitive module created for modeling the AAI system. The primitive TCP source code is based on the MIT Network Simulator (NetSim) [4] TCP model. Simulations of a simple network were performed in both BONeS and NetSim. Also, the simulation results from both simulations are compared with experimental results. Based on these validation results, the TCP BONeS primitive module functions correctly.

2 TCP BONeS Primitive Module

The BONeS primitive module interface of TCP is shown in Figure 1. It specifies all necessary input and output ports and user specified parameters. The module provides two ports for time depended statistics. Every time the

"Dump TCP Stat" port is triggered, the "TCP_statistics" port is activated and outputs the values of the most important TCP performance parameters. These performance parameters are:

1. Throughput (kbps)
2. Retransmission Percentage
3. Congestion Window Size (Bytes)
4. Slow Start Threshold value (Bytes)
5. Total Number of Bytes Transmitted
6. Maximum Sequence Number Send
7. Round Trip Time (RTT) (μsec)
8. Retransmission TimeOut (RTO) (μsec)
9. Current RTO with Exponential Backoff (μsec)

Figure 2 shows the main simulation events of TCP model. Since the TCP model is event driven model, it stays idle until an event is occurred. The action caused by the main simulation events are:

1. **Event Send Buffer (requested by user):**
    (a) Add new user buffer to input queue.
    (b) Send new data if possible.
2. **Event Receive Segment:**
    (a) Queue the received segment for processing
    (b) If processor not busy, set the Event Timer for another Demux Event.
3. **Event Retrans Timeout:**

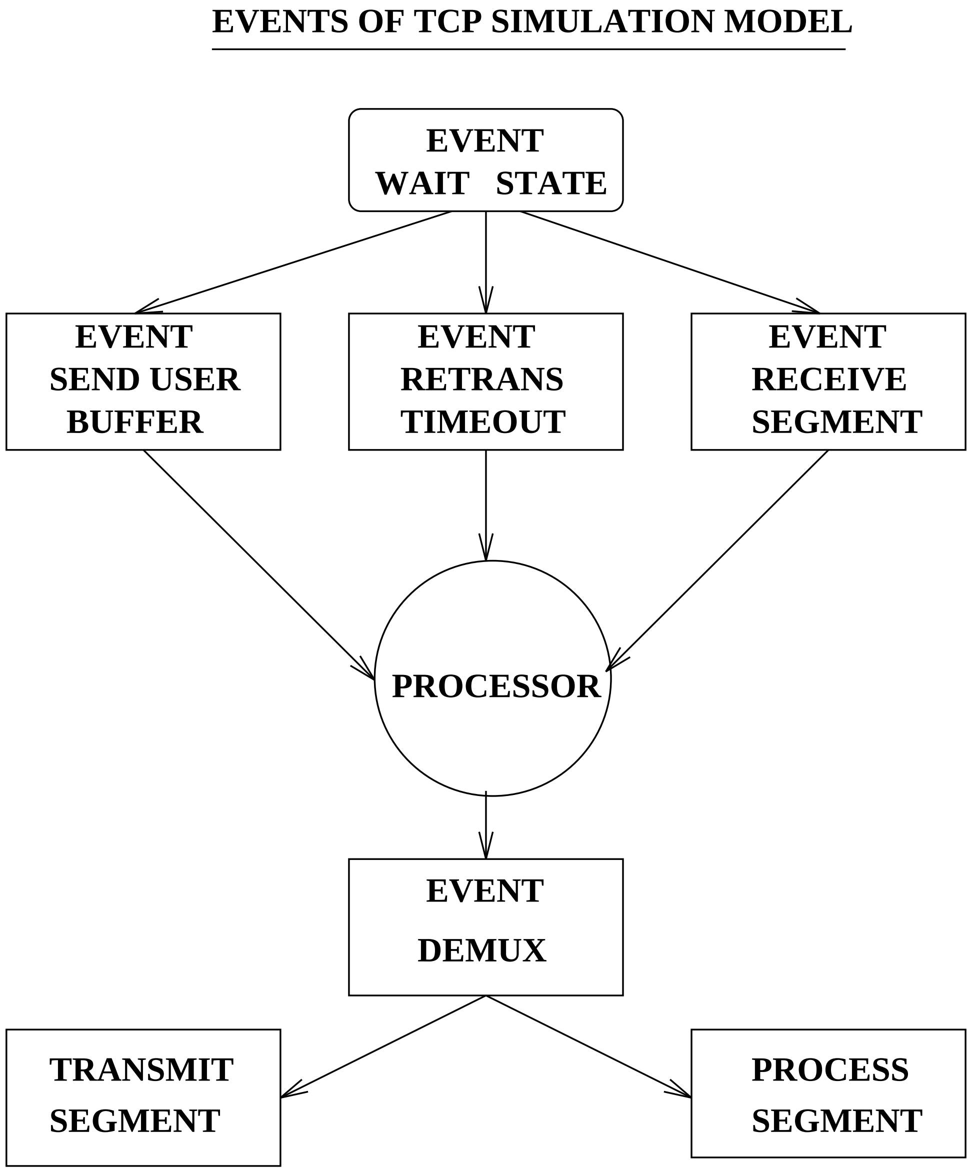

Figure 2: TCP main simulation events

(a) Reset the slow start threshold and congestion window size:

$$ssthresh = \frac{cwnd}{2} \tag{1}$$

$$cwnd = 1\ MTU \tag{2}$$

(b) Send lowest unAcked segment.

(c) Set another Retrans Timeout event using exponential backoff

$$RTO\ current \ = \ RTO\ current * 2 \tag{3}$$

4. **Event Demux:**

   (a) Is the segment output or Input?
   If output then Transmit Segment
   else
   Process Segment.

   (b) Set the event timer for the next Demux event.

5. **Transmit Segment:**

   (a) Transmit Segment.

   (b) If retransmit timer is off and segment just send is data segment, set the event timer for another Retrans Timeout event.

6. **Process Receiving Segment:**

   (a) If segment is corrupted, discard it.

   (b) If new window, update send window.

   (c) Recalculate RTO if clock information in segment using Van Jacobson's algorithm [3].

   (d) If segment in arrive in correct sequence process it,
   otherwise buffer it in resequent queue.

   (e) Process a new Ack.

   (f) Remove acked data from input queue.

   (g) Open congestion window using Van Jacobson's strategy.

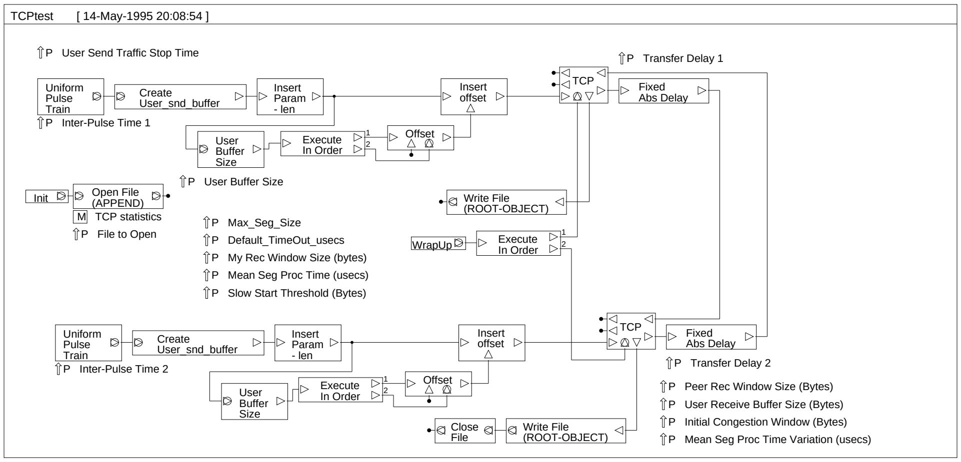

Figure 3: BONeS simple simulation network model

(h) Cancel event timer for Retrans Timeout event.

(i) If there are unacknowledged data in input queue, set the event timer for another Retrans Timeout event.

(j) Send new data if possible,
else
if received segment is not just ACK, send an ACK.

3 Simulation Network Models

Figure 3 shows the BONeS simple network model used for performance comparison with the NetSim simple network model as shown in Figure 4. The big difference between the two models is that in NetSim model an ATM host and ATM switch are used. However, in the BONeS model the ATM host and switch along with the link are model with a single "Fixed Abs Delay" module. It was impossible to make both model identical because:

1. A BONeS ATM host and switch were not used in this case, and

2. NetSim do not allow to connect the TCP block with Link block directly.

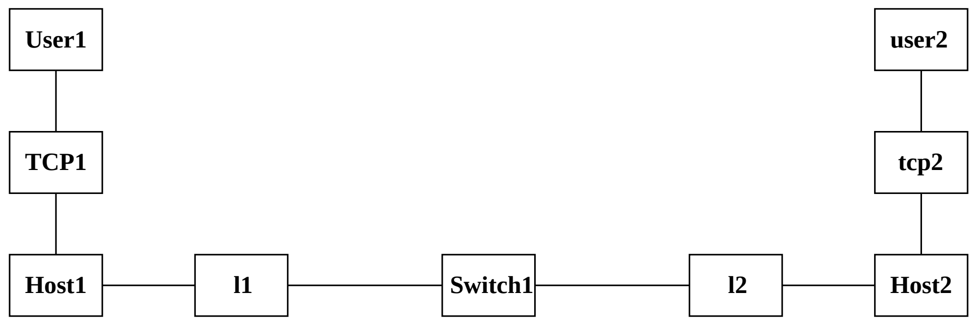

Figure 4: NetSim simple simulation network model

Later, we will include ATM SAR functionality in BONeS. 155 Mbps links were used in the simulation with 500 nsec of delay in each link. Figure 5 shows the BONeS model modified to cause segment errors.

4 Simulation Results

Figure 6 compares the simulation results, throughput Vs window size for MTU of 16384 bytes, from BONeS and NetSim model to experimental results from Ewy [2]. With the close match of the BONeS simulation results and the NetSim simulation and Ewy [2] experimental results, we have confidence in the TCP BONeS primitive module. Local area network experiments were performed using an MTU of 9180 bytes and the results were compared with BONeS simulations. Figure 7 compares BONeS and LAN experimental results for MTU of 9180 bytes. Figure 8 compares the WAN BONeS simulation results for MTU of 9180 bytes to emperical results and theoritical prediction from Ewy [2].

Figure 9 shows the congestion window size Vs time in case of link error event. The segment in error is discarded causing a retrans timeout event at time 0.27 sec. At this time the congestion window becomes equal to one MTU and increases by one MTU at every successful transmission. When the congestion window reaches the slow start threshold value which is equal to one half of the congestion window size prior of the retransmission timeout event, it increases very slowly. For example, looking at Figure 9, we see that the congestion window from 8192 bytes becomes 139264 bytes in 0.18 sec. But, from 139264 to 155648 bytes it takes 0.25 sec. This is Van Jacobson's

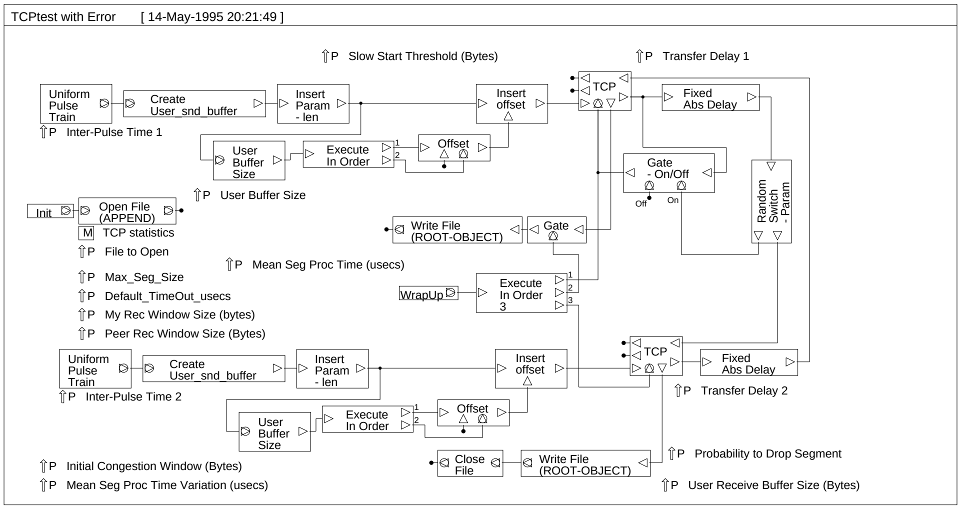

Figure 5: BONeS simple simulation network model with link error

slow start algorithm [3]. Looking again at the Figure 9, we see that another segment error causes the congestion window to drop to one MTU again and to increase according to Van Jacobson's strategy. Figure 10 shows the slow start threshold Vs time. It drops to one half of it's initial value at the first segment error, and then it decreases by one half of it's current value at the second segment error. Figure 11 shows the throughput Vs time. From this plot, we see that a single error causes the throughput to decrease 30% of it's value. Then it starts to increase very slowly until the occurrence of the second segment error. From Figure 12, we observe the exponential backoff technique for the current retransmission timeout.

At the occurrence of the first retrans timeout event, the current RTO doubles it's value until a successful retransmission. Then it becomes equal to the RTO value. RTO value is not affected by retransmissions and it's value is calculated by the RTT value according to Van Jacobson algorithm [3]. Figures 13, 14, and 15 show the RTO, RTT and Retransmission Percentage Vs time.

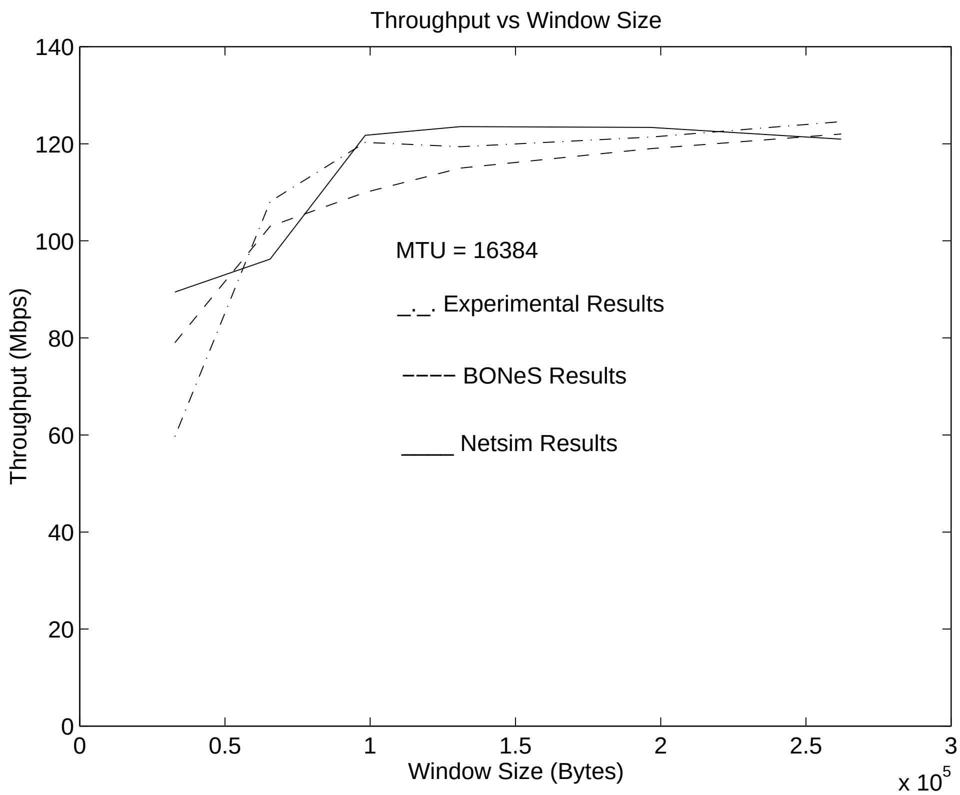

Figure 6: Throughput Vs Window comparison of experimental and simulation results

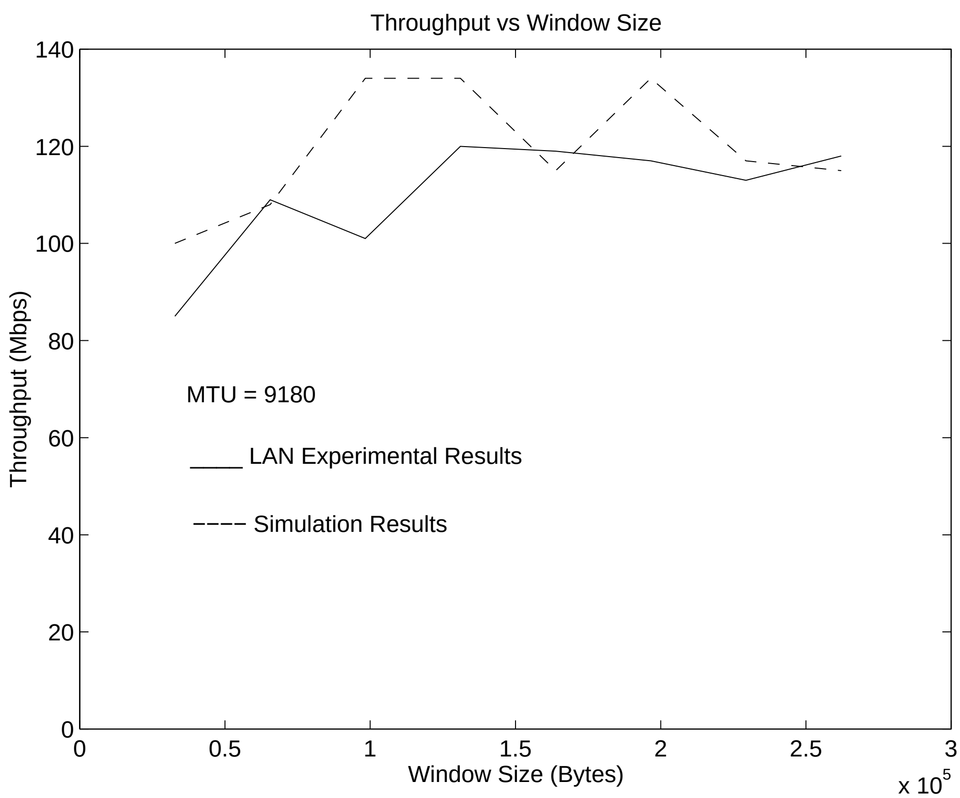

Figure 7: Comparison of LAN experimental and simulation results of MTU of 9180 bytes

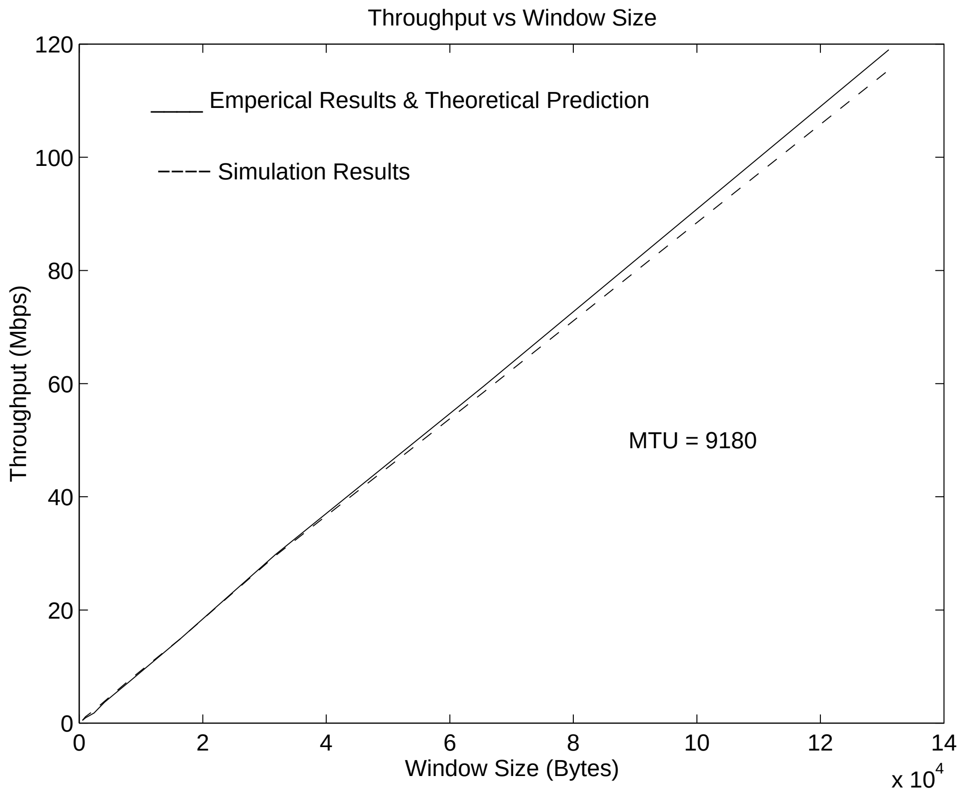

Figure 8: Wan simulation and emperical results and theoritical prediction comparison

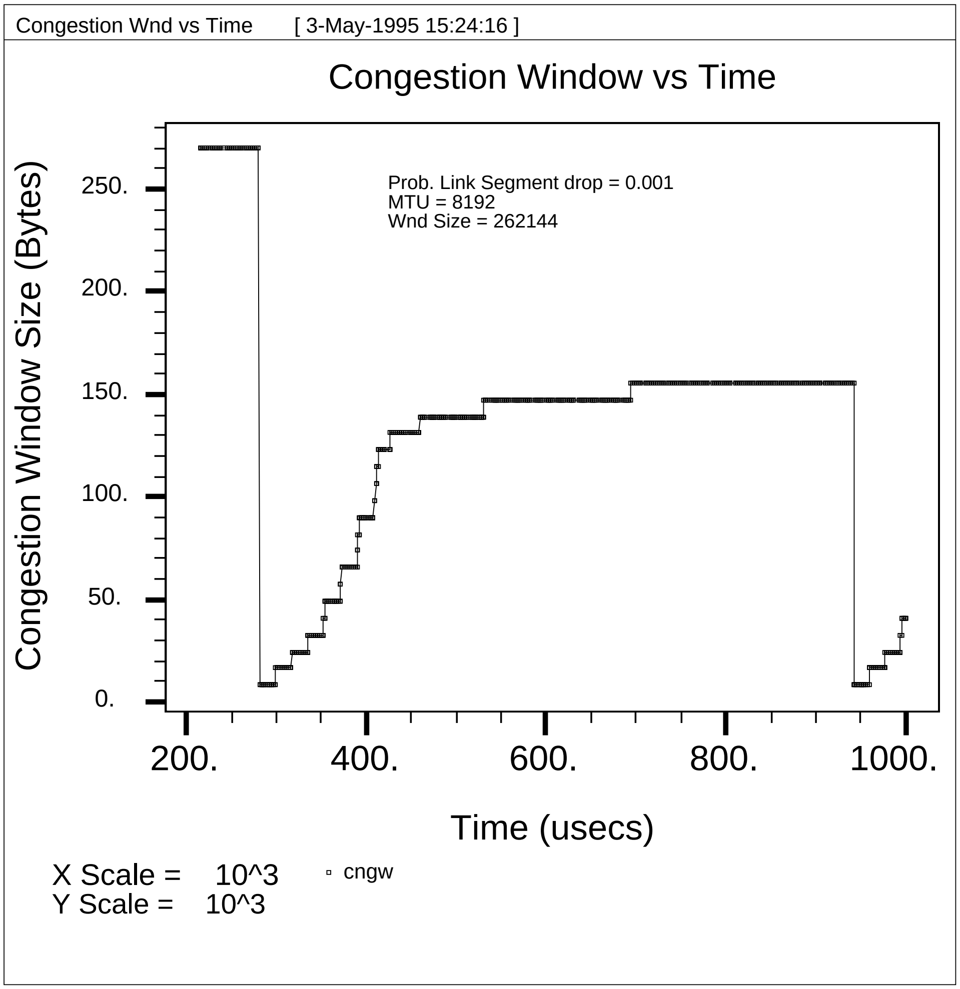

Figure 9: Congestion Vs Time

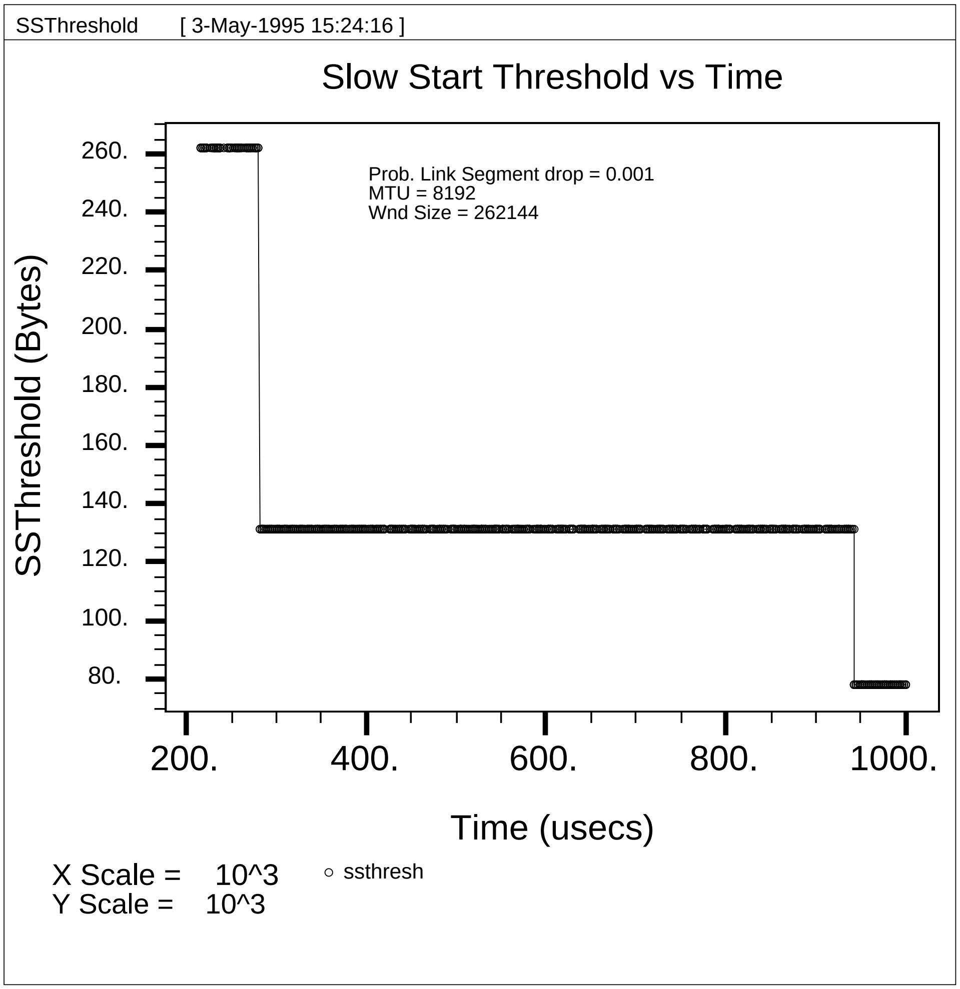

Figure 10: Slow Start Threshold Vs Time

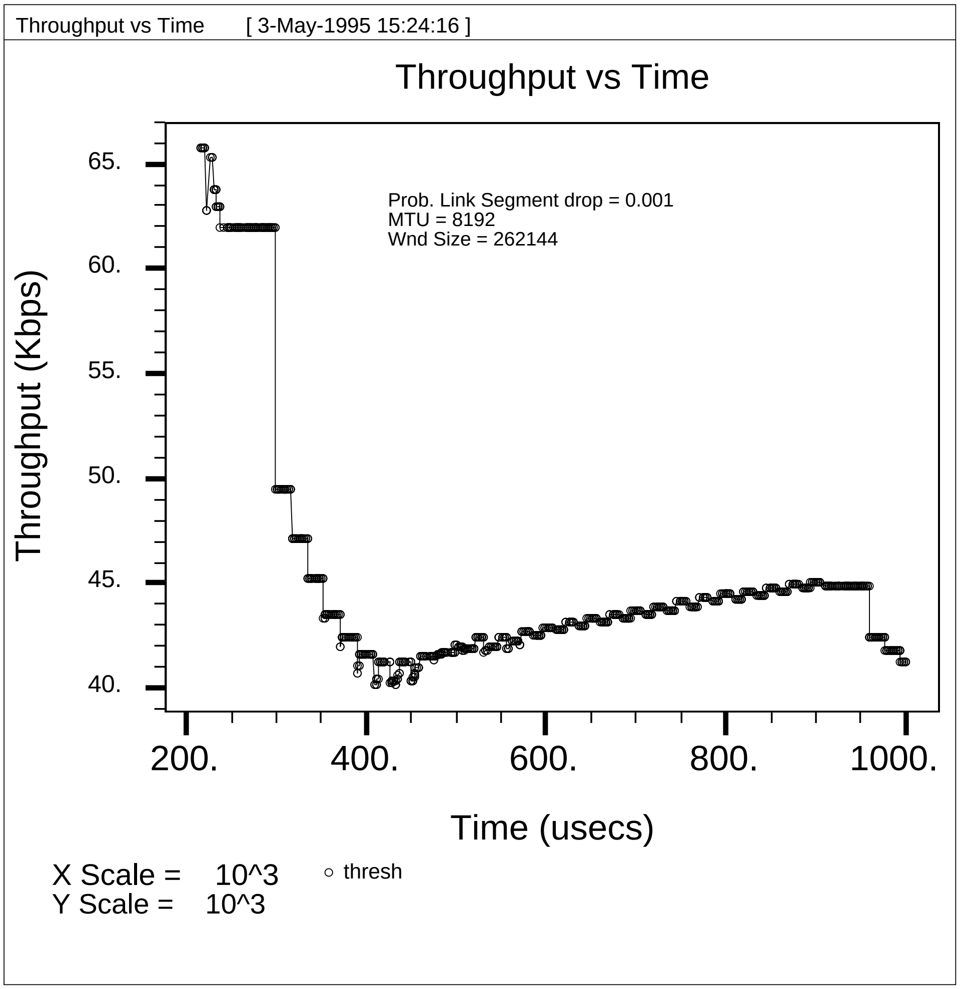

Figure 11: Throughput Vs Time

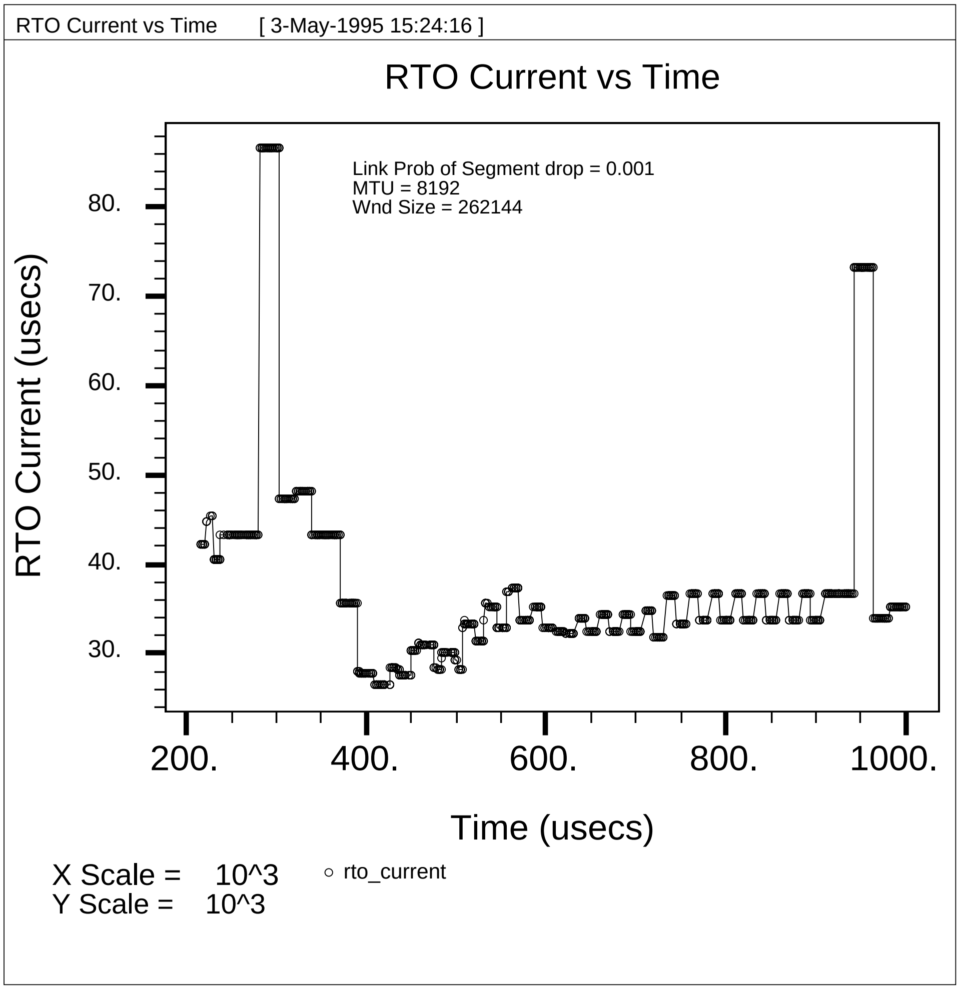

Figure 12: RTO Current Vs Time

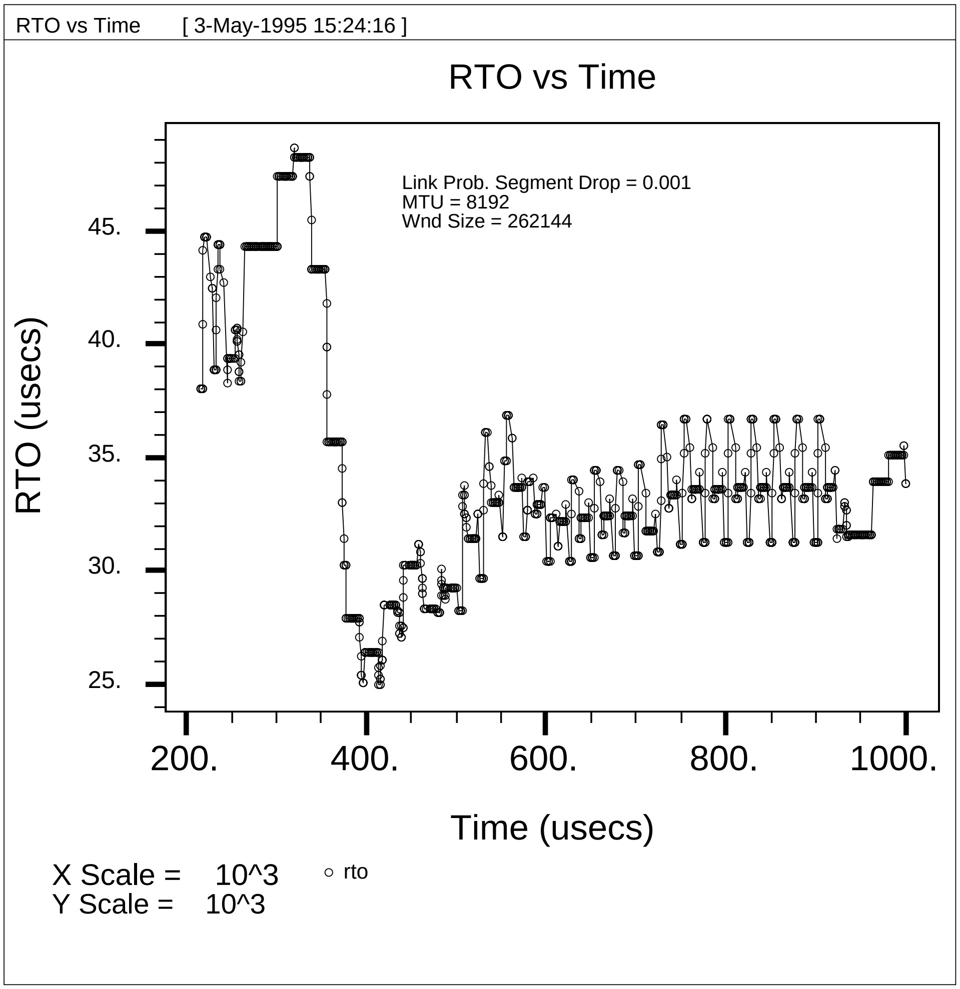

Figure 13: RTO Vs Time

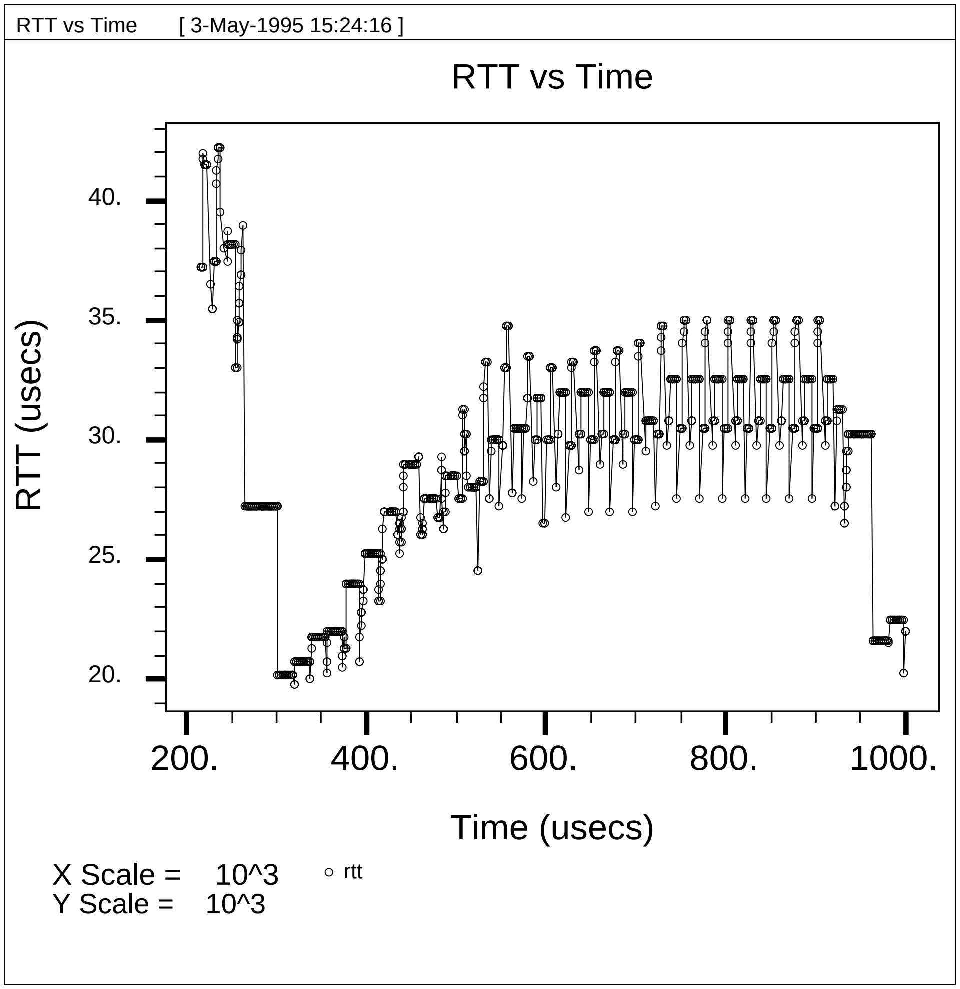

Figure 14: RTT Vs Time

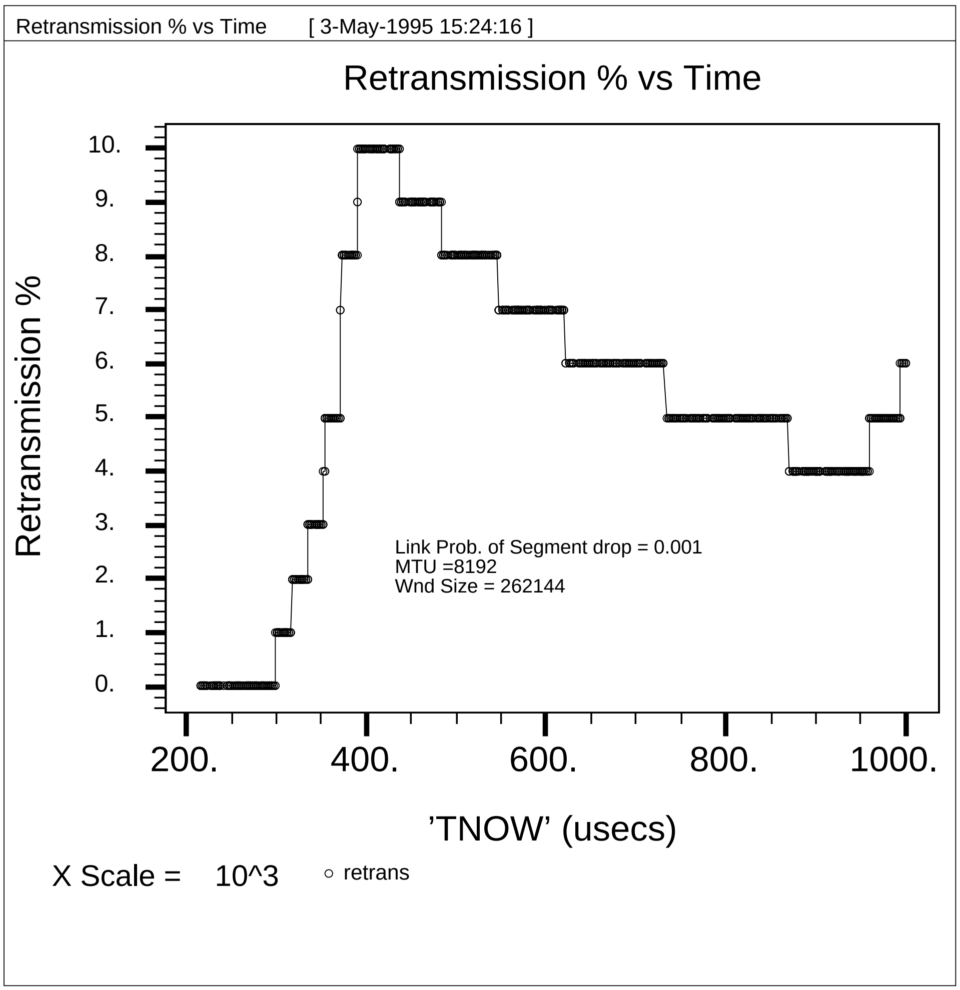

Figure 15: Retransmission Percentage Vs Time

5 Conclusion

Simulations were performed using BONeS and NetSim simulators in order to verify the correct functionality of the BONeS TCP primitive module created based on the NetSim TCP block. From the simulation results and the comparison to experimental measurements, we can say with confidence that the BONeS TCP primitive module created for modeling the AAI system functions correctly.

References

[1] D. E. Comer, *Internetworking with TCP/IP*, Volume I, II. Prentice–Hall, Englewood Cliffs, New Jersey, 1991.

[2] Benjamin J. Ewy, "The Effects of Traffic Shaping on Link Utilization and Congestion in ATM based WANs," Master Thesis, Electrical Engineering and Computer Science, TISL, University of Kansas, 1995.

[3] V. Jacobson, "Congestion Avoidance and Control," in Proc. SIGCOMM '88, AUG. 1988, pp. 314–329.

[4] D. Martin, *Network Simulator User's Manual*, MIT, 1988.