Trends in TCP/IP Retransmissions and Resets

Concordia Chen, Mrunal Mangrulkar, Naomi Ramos, and Mahasweta Sarkar
{cychen, mkulkarn, msarkar,naramos}@cs.ucsd.edu

Abstract

As the Internet grows larger, measuring and characterizing its dynamics becomes essential for purposes ranging from the optimization of future network equipment to modeling the effects of new protocols on the existing traffic mix. One important aspect of Internet connections is bandwidth utilization with data sent using the TCP protocol. In this paper, we discuss findings from a study of 65,535 TCP flows between 8083 Internet sites. We look into two major bandwidth utilization problems: HTTP connections reset unnecessarily by impatient users and bandwidth wasted on retransmissions. For HTTP connections, we provide an algorithm to distinguish a reset of an impatient user from a network reset. For bandwidth wasted, we calculated the amount of data retransmitted and also analyzed the goodput and throughput per flow. We discovered that in our one-hour trace, unnecessary resets occur 10.6% of the time and that retransmissions waste 3.47% of the total bandwidth.

1 Introduction

Analysis of Internet traffic behavior is a major area of research today. As the Internet grows larger, measuring and characterizing its dynamics becomes essential for purposes ranging from the optimization of future network equipment to modeling the effects of new protocols on the existing traffic mix.

Since the TCP/IP is the primary protocol being used in today's Internet, it is important to characterize its performance. While there are various aspects of TCP that can be analyzed, such as congestion control and end-to-end routing behaviors, bandwidth utilization is one of the most important statistics in TCP connections.

To understand bandwidth utilization, two main questions were put forth:

- What percentage of HTTP connections is reset unnecessarily?
- What percentage of bandwidth is wasted on retransmissions?

Using data collected by the NLANR/MOAT Network Analysis Infrastructure (NAI) project and analysis software from CAIDA's CoralReef project, trends were identified in TCP packet transmission over a one hour time period. The study was a passive analysis of 65,535 TCP flows between 8083 Internet sites. Statistics, such as throughput, goodput, and the number of resets that occurs due to "image- impatient" clients (e.g. individuals who reset their HTTP connections before the entire web page is transferred) were collected.

In §2, we discuss the methodology and the tools used for our analysis. In §3, we provide the results and the analysis used to answer our questions. We found that in our one-hour trace, unnecessary resets occur 10.6% of the time and that retransmissions waste only a small percentage - 3.47 % of the total bandwidth. In §4, we discuss other interesting observations made during the course of this project. In §5, we briefly summarize our findings and discuss future directions for evaluating retransmission and reset trends.

2 Analysis Methodology and Tools

The raw data used in the study was provided by the NLANR/MOAT Network Analysis Infrastructure (NAI) project. A monitor was placed on an OC-3c link at the New Zealand Internet Exchange and collected the data.[1]

[1] The monitor was connected to a Switched Port Analyzer (SPAN) using a 100BaseTX Fast Ethernet. Timestamps are consequently skewed compared to their arrival or departure times at the input/output ports of the switch as the total capacity of the switch is higher than the monitoring uplink and packets arriving

Due to limited resources, mainly disk space, we did our analysis on a one-hour trace extracted from a continuous five-day trace. Although the switching network used in this trace was ATM, the study concentrated on the TCP/IP layer and only the 40 bytes of header information extracted from the first ATM cell of each packet were analyzed. Since the traces were encoded in the DAG format, CAIDA's CoralReef software was modified to reduce the raw trace to a set of data that could be analyzed with other tools. Although the analysis for the study could have been done entirely within CoralReef, it was decided that an external program would be able to provide flexibility and spatial efficiency. This data set consisted of specific information found within the TCP/IP header, such as source/destination IP address, source/destination port address, window size, number of bytes of data sent, ACK number, sequence number, and the SYN, FIN, RST and ACK flags. In addition, the timestamp was recorded and the protocol number given by the monitor was used to verify that the packet was in fact TCP/IP.

Working with the one-hour trace of data extracted with the CoralReef utilities, flows characterized by the 4-tuple, *{IP source address, IP destination address, IP source port, IP destination port}* were individually analyzed to yield statistics on retransmissions and impatient resets. Table 1 summarizes the data gathering and analysis techniques.

Table 1: Data Gathering and Analysis Techniques

Tool	Input	Measurements			Output
		Active/passive	Functions	Time Scope	
Perl Script	Coral Reef extracted header information	Passive	• Count total bytes sent, retransmitted bytes, number of resets • Throughput calculation • Goodput calculation • Measure Bandwidth utility	1 hour from start of transmission	File and Graphs

Since the study was focused on retransmission and resets, the statistics calculated from observed measurements included bandwidth wasted due to retransmissions, throughput, goodput, and total number of "impatient" resets. The bandwidth wasted, *BW*, due to retransmit was calculated using the total number of bytes retransmitted, *r*, and the total transmission time, *t*, in seconds with the following equation.

$$BW = \frac{r}{t}$$

The total throughput, *T*, was calculated by looking at the total transmitted bytes, *b*, over the total transmission time, *t*, in seconds.

$$T = \frac{b}{t}$$

Total goodput, *G*, was defined with respects to the total transmitted bytes, *b*, total retransmitted bytes, *r*, over the transmission time, *t*, in seconds

$$G = \frac{b - r}{t}$$

---

from different ports at the same time need to be queued before being delivered to the SPAN port. The chance of traffic being lost before being monitored is fairly low as the total bandwidth at the switch currently peaks at around 10-12 MBits/sec.

In order to calculate resets due to "image-impatient" users, the following algorithm was used :

If either source or destination port = 80 [2]
  If RST = 1
    If bytes transmitted <10,000
      Increment number of resets

Since we were looking for retransmissions due to impatient users, it was necessary to observe the handshake at the beginning of the transmission. This algorithm was run only on the flows that began within the 1-hour time period. A distinction between connections was needed to decide which resets were due to network failures versus those due to "image-impatient" users. Looking at several images on the Internet, we discovered that a small image would be at least 20KB. We assumed that a user fitting this profile would terminate the connection halfway through. Although this number can vary greatly, this provides a reasonable estimate of connections reset unnecessarily.

3 Results and Analysis

All of the results have been obtained by analyzing the traces associated with a particular flow. Table 2, below, is a sample of some of the packets sent within a flow. We generated these traces for every 4-tuple that was found in the first hour of the trace.

Table 2: Sample of a Flow Data Structure

Time	Src IP	Dst IP	Length	Src Port	Dst Port	Seq Num	Window Size	Ack Number	ACK	FIN	RST	SYN
9.48764210E+08	10.0.19.232	10.0.0.26	44	1091	83	842060	8192	0	0	0	0	1
9.48764210E+08	10.0.0.26	10.0.19.232	44	83	1091	1749083406	8760	842061	1	0	0	1
9.48764210E+08	10.0.19.232	10.0.0.26	40	1091	83	842061	8760	1749083407	1	0	0	0
9.48764210E+08	10.0.19.232	10.0.0.26	40	1091	83	842061	32768	1749083407	1	0	0	0
9.48764210E+08	10.0.19.232	10.0.0.26	618	1091	83	842061	32768	1749083407	1	0	0	0

After running the algorithms on 65,535 TCP flows between 8083 Internet sites, a summary of the transmission characteristics for each flow was created. The flow summary contained the following information per connection: start time, end time, sent bytes, retransmitted bytes, duplicate ACKs, outstanding ACKs, and the number of RST bits set.

As described earlier, this data was used to calculate throughput per flow, goodput per flow, bandwidth wasted due to retransmissions, total bytes retransmitted during one hour transmission, and total number of resets.

3.1 Connections Reset

Whether or not a user resets a connection depends mostly on the individual using the link at the time the trace was taken. Hence, it becomes difficult to make a generalized statement about the mentality of users using passive data. Following the algorithm described in Section 2, the total number of connections reset by impatient users in the hour-long period totaled 6,947 out of 65,535 TCP connections, meaning that approximately 10.6% of the connections were reset. The number of resets is a significant portion of the total number of connections.

[2] We have assumed that in most cases packets sent between any port number higher than 1023 and a well-known port number below 1023 are generated by the same protocol (e.g., HTTP on port 80). This matches typical end host behavior, in which clients allocate ephemeral ports from the range 1024 to 32767.

It is highly likely that the major portion of these resets would have led to retransmission, wasted bandwidth, and increased congestion. It would be difficult to calculate the retransmissions that would have resulted from these resets because every time a new connection is established, a new port number is issued. By looking at the passive data, we were unable to decide whether successive connections to the same IP address were requesting the same data.

A more viable approach to understanding the percentage of HTTP connections that are reset unnecessarily would be to use an active methodology. By observing a large group of users and their browsing behavior, a more accurate generalization can be made.

3.2 Retransmissions

TCP retransmissions should only occur when it is certain that a packet to be retransmitted was actually lost. As seen in [2], redundant retransmissions can also occur because of lost acknowledgments, coarse feedback, and bad retransmission timeouts. This study looks at the bandwidth wasted due to all types of retransmissions. A summary of the statistics gathered for retransmissions can be found in Table 3 below.

Table 3: Summary Statistics for Retransmissions

Total bytes sent/received	878,172,211
Retransmitted bytes	3,0537,579
% of retransmitted bytes	3.47 %
Used bandwidth (bytes/sec)	243,936.7253
Wasted bandwidth (bytes/sec)	8483.5375
% of wasted bandwidth	3.47%
Total packets sent	1,910,081
Total packets retransmitted	65006
% of packets resent	3.403%

Figure 1 below, depicts the bandwidth wasted due to retransmissions as a percentage of the flows. This pie chart shows the amount of retransmissions occurred during the traces.

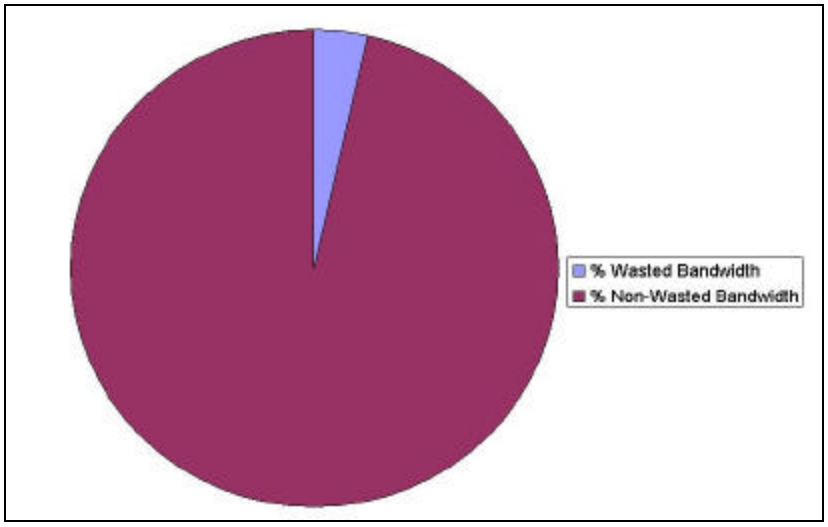

Figure 1 Bandwidth Wasted Due to Retransmissions

In order to compare our results, we looked the percentage of redundant retransmissions reported in [2]. This study indicated that approximately 1% to 6% of all packets sent was redundantly retransmitted. Our results fall

within this range. Although this observation could be misused to make a generalization of TCP connections, it is important to note that TCP can display a wide variety of behavior making generalization about any of its characteristics difficult.

For example, this can be observed using the *netstat* facilities available on Solaris and Windows operating systems to observe the segments retransmitted actively measured for each computer. The percentage of retransmissions for these systems were significantly smaller than our calculated percentage for retransmitted packets. On a Windows NT machine, approximately 0.010% segments were transmitted and on a Solaris machine, there was a reported 0.014% segments retransmitted. Because this facility looked solely at the amount of retransmissions sent for computers that were mainly clients, the smaller results were expected.

3.2.1 Throughput & Goodput

Throughput helps us estimate the utilization of the available bandwidth. With the virtual channel as a Classical-IP-over-ATM, the connection had a packet peak rate of 250,000 bytes/sec set in each direction (500,000/sec when you aggregate the two trace files). The throughput measured occupies 48.78% of the total available bandwidth on the OC3c link.

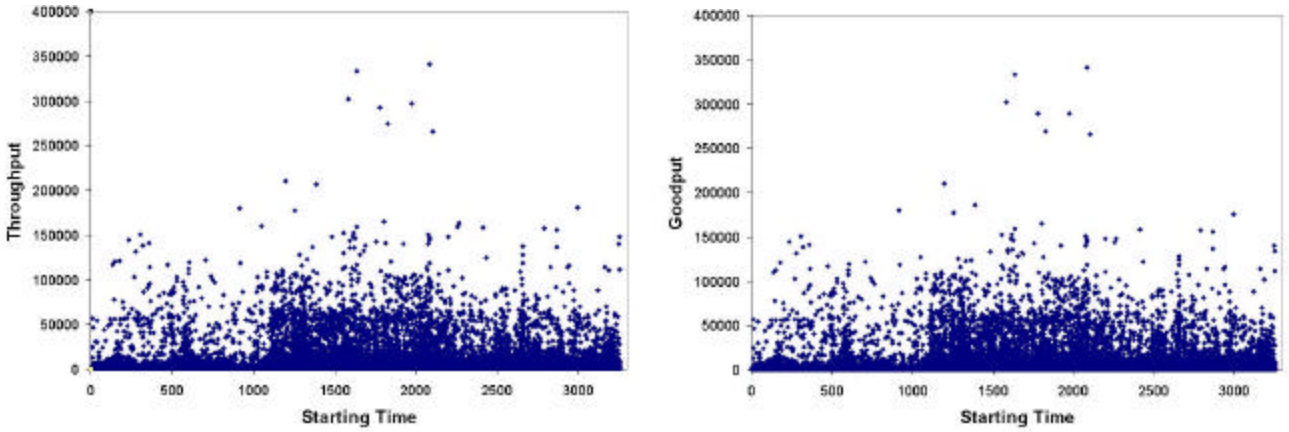

Figure 2a, b: Throughput and Goodput versus Starting Time

Figure 2a, above, shows the throughput per flow plotted against its reported start time. This graph shows the variation in throughput per flow throughout the one-hour period. Note that the high peaks indicate the throughput of the particular flow had high bandwidth utilization. The graphs are constructed using 30,000 flows randomly chosen from the data set. This is due to the limitation of the graphing program used to plot the data. Similarly, Figure 2b, below, illustrates the goodput obtained by these flows. Comparing the two graphs, it can be seen that these graphs resemble each other. With goodput defined as $G = \frac{b-r}{t}$ this result was predictable. With a minimal amount of bytes retransmitted, only accounting for 3.47% of the total bytes transmitted, the goodput results should have been similar to the throughput graph.

4. Other Observations

Piggybacking

An interesting observation in the traces was that some of the source/destination pairs were not piggybacking their acknowledgements. The same sequence number was used on two consecutive packets, one containing an acknowledgement for a packet received and the other containing a payload. While this pattern is expected following the initial handshake, it was observed several times within one flow. A possible explanation could be that the server/client did not want to delay acknowledgments to wait for the send buffer to fill.

Duplicate ACKs

By looking at the TCP flow traces, the congestion avoidance scheme being used for the TCP connection could be partially identified. Most of the connections waited for three duplicate ACKs before retransmitting a packet. This behavior identified the congestion avoidance scheme as being Reno or Tahoe. The total number of duplicate ACKs that was found in the one hour trace was 284,258. This is reasonable, considering that a packet is retransmitted for every three duplicate ACKs received on average.

Window Size

In our analysis, the window size stayed almost constant throughout the hour long period. This was contradictory to what was expected. TCP implements congestion avoidance by cutting down the window size to half of the value at which congestion started. TCP's policy of retransmitting after every three duplicate ACKs was also observed. This should have been accompanied by the decrease in window size. However no such observation was made.

5 Conclusion

After conducting an analysis of the TCP traffic workload, seen at a single measurement site, inside a major Internet traffic exchange point, the following conclusions can be made:

- We discovered that 10.6 % of the connections were reset due to image impatient users. We felt that this was a significant portion of the connection. We assume that a major portion of these resets would have led to retransmissions, leading to wasted bandwidth and increased congestion. It would be difficult to calculate the retransmissions that would have resulted from these resets because every time a new connection is established, a new port number is issued. By looking at the passive data, we were unable to decide whether successive connections to the same IP address were requesting the same data.

- The retransmission trends that we observed in our analyses showed that we had a 3.4% retransmission rate over all packets sent. This was in accordance to other retransmission results reported in previous studies [2]. However, due to the wide variation of TCP behavior, we could not generalize the results.

- Some common assumptions such as piggybacking of data with acknowledgements and decrease in window size at the onset of congestion were violated. However, we did observe that retransmissions occurred after three duplicate ACKs which is consistent with the TCP congestion avoidance mechanism of the most popular TCP implementations (e.g Reno and Tahoe)

Future directions for understanding trends regarding retransmission and resets would include more extensive analysis of traces over a longer period of time. It would help characterize TCP trends more accurately. An active measurement of resetting trends with image impatient users could lead to interesting conclusions.

5 Acknowledgements

This work would not have been possible without the CoralReef tools provided by Caida, data provided by the NLANR/MOAT Network Analysis Infrastructure (NAI) project, aid provided by David Moore, and the insight into TCP/IP given by Geoff Voelker.

References

[1] Peterson and Davie. *Computer Networks: A Systems Approach*. Morgan Kaufman, 2000 (Second Edition).

[2] Paxson, V. "End-to-End Internet Packet Dynamics", Proc. SIGCOMM'97, Sept 1997.

[3]Paxson, V. “End-to-End Internet Routing Behavior in the Internet”, Proc. SIGCOMM’96, August 1996.

[4] Stevens, W. *TCP/IP Illustrated, Volume1: The Protocols*. Addison Wesley, 1994.

[5] McCreary, S and Claffy, Kc “Trends in Wide Area IP Traffic Patterns”, 2000