Internet Engineering Task Force
INTERNET DRAFT
draft-floyd-tcp-highspeed-00.txt
June, 2002
Sally Floyd
ICSI

HighSpeed TCP for Large Congestion Windows

Status of this Memo

Abstract

This document proposes HighSpeed TCP, a modification to TCP's congestion control mechanism for use with TCP connections with large congestion windows. The congestion control mechanisms of the current, Standard TCP constrains the congestion windows that can be achieved by TCP in realistic environments. For example, for a Standard TCP connection with 1500-byte packets and a 100 ms round-trip time, achieving a steady-state throughput of 10 Gbps would require an average congestion window of 83,333 segments, and a packet drop rate of at most one congestion event every 5,000,000,000 packet (or equivalently, at most one congestion event every 1 2/3 hours). We do not consider this a realistic condition. To address this limitation of TCP, this document proposes HighSpeed TCP, and solicits experimentation and feedback from the wider community.

1. Introduction.

This document proposes HighSpeed TCP, a modification to TCP's congestion control mechanism for use with TCP connections with large congestion windows. In a steady-state environment, with a packet loss rate p, the current Standard TCP's average congestion window is roughly 1.2/sqrt(p) segments. This places a serious constraint on the congestion windows that can be achieved by TCP in realistic environments. For example, for a Standard TCP connection with 1500-byte packets and a 100 ms round-trip time, achieving a steady-state throughput of 10 Gbps would require an average congestion window of 83,333 segments, and a packet drop rate of at most one congestion event every 5,000,000,000 packet (or equivalently, at most one congestion event every 1 2/3 hours). To address this limitation of TCP, this document proposes HighSpeed TCP, and solicits experimentation and feedback from the wider community.

2. The Problem Description.

This section describes the number of round-trip times between congestion events required for a Standard TCP flow to achieve an average throughput of B bps, given packets of D bytes and a round-trip time of R seconds. A congestion event refers to a window of data with one or more dropped or ECN-marked packets.

From Appendix A, achieving an average TCP throughput of B bps requires a loss event at most every BR/(12D) round-trip times. This is illustrated in Table 1, for R = 0.1 seconds and D = 1500 bytes. The table also gives the average congestion window W of BR/(8D), and the steady-state packet drop rate P of 1.5/W^2.

TCP Throughput (Mbps)	RTTs Between Losses	W	P
1	5.5	8.3	0.02
10	55.5	83.3	0.0002
100	555.5	833.3	0.000002
1000	5555.5	8333.3	0.00000002
10000	55555.5	83333.3	0.0000000002

Table 1: RTTs Between Congestion Events for Standard TCP, for 1500-Byte Packets and a Round-Trip Time of 0.1 Seconds.

This document proposes HighSpeed TCP, a minimal modification to TCP's increase and decrease parameters, for TCP connections with larger congestion windows, to allow TCP to achieve high throughput with more realistic requirements for the steady-state packet drop rate. Equivalently, HighSpeed TCP has more realistic requirements for the number of round-trip times between loss events.

3. Design Guidelines.

Our proposal for HighSpeed TCP is motivated by the following requirements:

* Achieve high per-connection throughput without requiring unrealistically low packet loss rates.

* Reach high throughput reasonably quickly when in slow-start.

* Reach high throughput without overly long delays when recovering from multiple retransmit timeouts, or when ramping-up from a period with small congestion windows.

* No additional feedback or support required from routers:

For example, the goal is for acceptable performance in both ECN-capable and non-ECN-capable environments, and with Drop-Tail as well as with Active Queue Management such as RED in the routers.

* No additional feedback required from TCP receivers.

* TCP-compatible performance in environments with moderate or high congestion:

Equivalently, the requirement is that there be no additional load on the network (in terms of increased packet drop rates) in environments with moderate or high congestion.

* Performance at least as good as Standard TCP in environments with moderate or high congestion.

* Acceptable transient performance, in terms of increases in the congestion window in one round-trip time, responses to severe congestion, and convergence times to fairness.

Currently, users wishing to achieve throughputs of 1Gbps or more typically open up multiple TCP connections in parallel, or use MulTCP [GRK99], which behaves roughly like the aggregate of N virtual TCP connections. While this approach suffices for the occasional user on well-provisioned links, it leaves the parameter N to be determined by the user, and results in more aggressive performance and higher steady-state packet drop rates if used in environments with periods of moderate or high congestion. We believe that a new approach is needed that offers more flexibility, more effectively scales to a wide range of available bandwidths, and competes more fairly with Standard TCP in congested environments.

4. Non-Goals.

The following are explicitly *not* goals of our work:

* Non-goal: TCP-compatible performance in environments with very low packet drop rates.

We note that our proposal does not require, or deliver, TCP-compatible performance in environments with very low packet drop rates, e.g., with packet loss rates of 10^-5 or 10^-6. As we discuss later in this document, we assume that Standard TCP is unable to make effective use of the available bandwidth in environments with loss rates of 10^-6 in any case, so that it is acceptable and appropriate for HighSpeed TCP to perform more aggressively than Standard TCP is such an environment.

* Non-goal: Ramping-up more quickly than allowed by slow-start.

It is our belief that ramping-up more quickly than allowed by slow-start would necessitate more explicit feedback from routers along the path. The proposal for HighSpeed TCP is focused on changes to TCP that could be effectively deployed in the current Internet environment.

* Non-goal: Avoiding oscillations in environments with only one-way, long-lived flows all with the same round-trip times.

While we agree that attention to oscillatory behavior is useful, avoiding oscillations in aggregate throughput has not been our primary consideration, particularly for simplified environments limited to one-way, long-lived flows all with the same, large round-trip times. Our assessment is that some oscillatory behavior in these extreme environments is an acceptable price to pay for the other benefits of HighSpeed TCP.

5. Modifying the TCP Response Function.

The TCP response function, w = 1.2/sqrt(p), gives TCP's average congestion window w in MSS-sized segments, as a function of the steady-state packet drop rate p [FF98]. This TCP response function is a direct consequence of TCP's Additive Increase Multiplicative Decrease (AIMD) mechanisms of increasing the congestion window by roughly one segment per round-trip time in the absence of congestion, and halving the congestion window in response to a round-trip time with a congestion event. This response function for Standard TCP is reflected in the table below. In this proposal we restrict our attention to TCP performance in environments with packet loss rates of at most 10^-2, and so we can ignore the more complex response

functions that are required to model TCP performance in more congested environments with retransmit timeouts. From Appendix A, an average congestion window of W corresponds to an average of W/1.5 round-trip times between loss events for Standard TCP.

Packet Drop Rate P	Congestion Window W	RTTs Between Losses
10^-2	12	8
10^-3	38	25
10^-4	120	80
10^-5	379	252
10^-6	1200	800
10^-7	3795	2530
10^-8	12000	8000
10^-9	37948	25298
10^-10	120000	80000

Table 2: TCP Response Function for Standard TCP. The average congestion window W in MSS-sized segments is given as a function of the packet drop rate P.

To specify a modified response function for HighSpeed TCP, we use three parameters, Low_Window, High_Window, and High_P. To ensure TCP compatibility, the HighSpeed response function uses the same response function as Standard TCP when the current congestion window is at most Low_Window, and uses the HighSpeed response function when the current congestion window is greater than Low_Window. In this document we set Low_Window to 38 MSS-sized segments, corresponding to a packet drop rate of 10^-3 for TCP.

To specify the upper end of the HighSpeed response function, we specify the packet drop rate needed in the HighSpeed response function to achieve an average congestion window of 83000 segments. This is roughly the window needed to sustain 10Gbps throughput, for a TCP connection with the default packet size and round-trip time used earlier in this document. For High_Window set to 83000, we specify High_P of 10^-7; that is, with HighSpeed TCP a packet drop rate of 10^-7 allows the HighSpeed TCP connection to achieve an average congestion window of 83000 segments. We believe that this loss rate sets an achieveable target for high-speed environments, while still allowing acceptable fairness for the HighSpeed response function when competing with Standard TCP in environments with packet drop rates of 10^-4 or 10^5.

For simplicity, for the HighSpeed response function we maintain the property that the response function gives a straight line on a log-log scale (as does the response function for Standard TCP, for low to moderate congestion). This results in the following response

function, for values of the average congestion window W greater than Low_Window:

W = (p/Low_P)^S Low_Window,

for Low_P the packet drop rate corresponding to Low_Window, and for S as following constant [FRS02]:

S = (log High_Window - log Low_Window)/(log High_P - log Low_P).

For example, for Low_Window set to 38, we have Low_P of 10^-3 (for compatibility with Standard TCP). Thus, for High_Window set to 83000 and High_P set to 10^-7, we get the following response function:

W = 0.12/p^0.835. (1)

This HighSpeed response function is illustrated in Table 3 below. For HighSpeed TCP, the number of round-trip times between losses, 1/(pW), equals 12.7 W^0.2, for W > 38 segments.

Packet Drop Rate P	Congestion Window W	RTTs Between Losses
10^-2	12	8
10^-3	38	25
10^-4	263	38
10^-5	1795	57
10^-6	12279	83
10^-7	83981	123
10^-8	574356	180
10^-9	3928088	264
10^-10	26864653	388

Table 3: TCP Response Function for HighSpeed TCP. The average congestion window W in MSS-sized segments is given as a function of the packet drop rate P.

It has been suggested that a less "ad-hoc" guideline for a response function for HighSpeed TCP would be to specify a constant value for the number of round-trip times between congestion events. However, this seems inadvisable to us, as it gives less clearly-differentiated feedback in an environment with steady-state background losses at fixed intervals for all flows (as might occur with a wireless link with occasional short error bursts, giving losses for all flows every N seconds regardless of their sending rate). While it is not a goal to have perfect fairness in an environment with synchronized losses, it would be good to have moderately acceptable performance in this regime. This goal argues against a response function with a constant number of round-trip times between congestion events.

We believe that the problem of backward compatibility with Standard TCP requires a response function that is quite close to that of Standard TCP for loss rates of 10^-1, 10^-2, or 10^-3. We believe, however, that such stringent TCP-compatibility is not required for smaller loss rates, and that an appropriate response function is one that gives a plausible packet drop rate for a connection throughput of 10Gbps. This also gives a slowly increasing number of round-trip times between loss events as a function of a decreasing packet drop rate.

We do not in this document attempt to seriously evaluate the HighSpeed response function for per-connection bandwidths greater than 10Gbps. We believe that we will learn more about the requirements for sustaining the throughput of best-effort connections in that range as we gain more experience with HighSpeed TCP at 1 Gbps and 10 Gbps ranges. There also might be limitations to the per-connection throughput that can be realistically achieved for best-effort traffic in the absence of additional support or feedback from the routers along the path.

6. Fairness Implications of the HighSpeed Response Function.

The Standard and Highspeed Response Functions can be used directly to infer the relative fairness between flows using the two response functions. For example, given a packet drop rate P, assume that Standard TCP has an average congestion window of W_Standard, and HighSpeed TCP has a higher average congestion window of W_HighSpeed. In this case, a single HighSpeed TCP connection is receiving W_HighSpeed/W_Standard times the throughput of a single Standard TCP connection competing in the same environment.

This relative fairness is illustrated below in Table 4, for the parameters used for the Highspeed response function in the section above.

Packet Drop Rate P	Relative Fairness
10^-2	1.0
10^-3	1.0
10^-4	2.2
10^-5	4.7
10^-6	10.2
10^-7	22.1
10^-8	47.9
10^-9	103.5
10^-10	223.9

Table 4: Relative Fairness between the HighSpeed and Standard Response Functions.

Thus, for packet drop rates of 10^-4, a flow with the HighSpeed response function can expect to receive 2.2 times the throughput of a flow using the Standard response function, given the same round-trip times and packet sizes. With packet drop rates of 10^-6 (or 10^-7), the unfairness is more severe, and we have entered the regime where a Standard TCP connection requires a congestion event at most every 800 (or 2530) round-trip times in order to make use of the available bandwidth. Our judgement would be that there are not a lot of TCP connections effectively operating in this regime today, with congestion windows of thousands of packets, and that therefore the benefits of the HighSpeed response function would outweigh the unfairness that would be experienced by Standard TCP in this regime. However, one purpose of this document is to solicit feedback on this issue. The parameter Low_Window determines directly the point of divergence between the Standard and HighSpeed Response Functions.

7. Translating the HighSpeed Response Function into Congestion Control Parameters.

For equation-based congestion control such as TFRC, the HighSpeed Response Function above could be used directly by the TFRC congestion control mechanism. However, for TCP the HighSpeed response function would have to be translated into additive increase and multiplicative decrease parameters. The HighSpeed response function cannot be achieved by TCP with an additive increase of one segment per round-trip time and a multiplicative decrease of halving the current congestion window; HighSpeed TCP will have to modify either the increase or the decrease parameter, or both. We have concluded that HighSpeed TCP is most likely to achieve an acceptable compromise between moderate increases and timely decreases by modifying both the increase and the decrease parameter.

That is, for HighSpeed TCP let the congestion window increase by a(w)

segments per round-trip time in the absence of congestion, and let the congestion window decrease to w(1-b(w)) segments in response to a round-trip time with one or more loss events. Thus, in response to a single acknowledgement HighSpeed TCP increases its congestion window in segments as follows:

```
w <- w + a(w)/w.
```

In response to a congestion event, HighSpeed TCP decreases as follows:

```
w <- (1-b(w))w.
```

For Standard TCP, a(w) = 1 and b(w) = 1/2, regardless of the value of w. HighSpeed TCP uses the same values of a(w) and b(w) for w <= Low_Window. This section specifies a(w) and b(w) for HighSpeed TCP for larger values of w.

For w = High_Window, we have specified a loss rate of High_P. From [FRS02], or from elementary calculations, this requires the following relationship between a(w) and b(w) for w = High_Window:

```
a(w) = High_Window^2 * High_P * 2 * b(w)/(2-b(w).     (2)
```

We use the parameter High_Decrease to specify the decrease parameter b(w) for w = High_Window, and use Equation (2) to derive the increase parameter a(w) for w = High_Window. Along with High_P = 10^-7 and High_Window = 83000, for example, we specify High_Decrease = 0.1, specifying that b(83000) = 0.1, giving a decrease of 10% after a congestion event. Equation (2) then gives a(83000) = 72, for an increase of 72 segments, or just under 0.1%, within a round-trip time, for w = 83000.

This moderate decrease strikes us as acceptable, particularly when coupled with the role of TCP's ACK-clocking in limiting the sending rate in response to more severe congestion [BBFS01]. A more severe decrease would require a more aggressive increase in the congestion window for a round-trip time without congestion. In particular, a decrease factor High_Decrease of 0.5, as in Standard TCP, would require an increase of 459 segments per round-trip time when w = 83000.

Given decrease parameters of b(w) = 1/2 for w = Low_Window, and b(w) = High_Decrease for w = High_Window, we are left to specify the value of b(w) for other values of w > Low_Window. From [FRS02], we let b(w) vary linearly as the log of w, as follows:

```
b(w) = (High_Decrease - 0.5) (log(w)-log(W)) / (log(W_1)-log(W)) +
```

0.5.

The increase parameter a(w) can then be computed as follows:

a(w) = w^2 * p(w) * 2 * b(w)/(2-b(w)),

for p(w) the packet drop rate for congestion window w. From inverting Equation (1), we get p(w) as follows:

p(w) = 0.078/w^1.2.

We assume that experimental implementations of HighSpeed TCP for further investigation will use a pre-computed look-up table for finding a(w) and b(w). In the appendix we give such a table for our default values of Low_Window = 38, High_Window = 83,000, High_P = 10^-7, and High_Decrease = 0.1. These are also the default values in the NS simulator; example simulations in NS can be run with the command "./test-all-tcpHighspeed" in the directory tcl/test.

8. Slow-Start.

An companion internet-draft on "Limited Slow-Start for TCP with Large Congestion Windows" [F02b] proposes a modification to TCP's slow-start procedure that can significantly improve the performance of TCP connections slow-starting up to large congestion windows. For TCP connections that are able to use congestion windows of thousands (or tens of thousands) of MSS-sized segments (for MSS the sender's MAXIMUM SEGMENT SIZE), the current slow-start procedure can result in increasing the congestion window by thousands of segments in a single round-trip time. Such an increase can easily result in thousands of packets being dropped in one round-trip time. This is often counter-productive for the TCP flow itself, and is also hard on the rest of the traffic sharing the congested link.

[F02b] proposes Limited Slow-Start, limiting the number of segments by which the congestion window is increased for one window of data during slow-start, in order to improve performance for TCP connections with large congestion windows. We have separated out Limited Slow-Start to a separate draft because it can be used both with Standard or with HighSpeed TCP.

Limited Slow-Start is illustrated in the NS simulator, for snapshots after May 1, 2002, in the tests "./test-all-tcpHighspeed tcp1A" and "./test-all-tcpHighspeed tcpHighspeed1" in the subdirectory "tcl/lib".

In order for best-effort flows to safely start-up faster than slow-start, e.g., in future high-bandwidth networks, we believe that it

would be necessary for the flow to have explicit feedback from the routers along the path. There are a number of proposals for this, ranging from a minimal proposal for an IP option that allows TCP SYN packets to collect information from routers along the path about the allowed initial window [J02], to proposals with more power that require more fine-tuned and continuous feedback from routers. These proposals all are somewhat longer-term proposals that the HighSpeed TCP proposal in this document, requiring longer lead times and more coordination for deployment, and will be discussed in later documents.

9. Related Work in HighSpeed TCP.

HighSpeed TCP has been separately investigated in simulations by Sylvia Ratnasamy and by Evandro de Souza, and reports of some of these simulations should be available shortly. The simulations by Evandro verify the fairness properties of HighSpeed TCP when sharing a link with Standard TCP.

These simulations explore the relative fairness of HighSpeed TCP flows when competing with Standard TCP. The simulation environment include background forward and reverse-path TCP traffic limited by the TCP receive window, along with a small amount of forward and reverse-path traffic from the web traffic generator. Most of the simulations so far explore performance on a simple dumbbell topology with a 1Gbps link with a propagation delay of 50 ms. Simulations have been run both the Adaptive RED and with DropTail queue management.

Future work to explore in more detail includes convergence times after new flows start-up; recovery time after a transient outage; the response to sudden severe congestion, and investigations of the potential for oscillations. Additional future work includes evaluating more fully the choices of parameters for HighSpeed TCP. We invite contributions from others in this work.

Suggestions to other citations of related work would also be welcome.

10. Relationship to other Work.

Our assumption is that HighSpeed TCP will be used along with the TCP SACK option, and also with the increased Initial Window of three or four segments, as allowed by [AFP02]. For paths that have substantial reordering, TCP performance would be greatly improved by some of the mechanisms still in the research stages for robust performance in the presence of reordered packets.

11. Conclusions.

This is an initial proposal, and we are asking from feedback from the wider community. We have explored this proposal in simulations, though we have not yet finished our reports on these simulations. We would welcome additional analysis, simulations, and particularly, experimentation.

There are three parameters that determine the HighSpeed Response Function, and an additional parameter that determines HighSpeed TCP's tradeoffs between increases and decreases using that response function. We solicit feedback on our setting of these parameters as well as on other issues.

We also have not exhaustively explored the transient dynamics of HighSpeed TCP.

12. Acknowledgements

The HighSpeed TCP proposal is from joint work with Sylvia Ratnasamy and Scott Shenker. Additional investigations of HighSpeed TCP were joint work with Evandro de Souza and Deb Agarwal. We are grateful to the End-to-End Research Group and to members of the IPAM program in Large Scale Communication Networks for feedback, and to contributions and feedback from the following individuals: Tom Kelly, Jitendra Padhye, Brian Tierney.

13. Normative References

[RFC2581] M. Allman and V. Paxson, "TCP Congestion Control", RFC 2581, April 1999.

14. Informative References

[BBFS01] Deepak Bansal, Hari Balakrishnan, Sally Floyd, and Scott Shenker, "Dynamic Behavior of Slowly-Responsive Congestion Control Algorithms", SIGCOMM 2001, August 2001.

[FF98] Floyd, S., and Fall, K., "Promoting the Use of End-to-End Congestion Control in the Internet", IEEE/ACM Transactions on Networking, August 1999.

[FRS02] Sally Floyd, Sylvia Ratnasamy, and Scott Shenker, "Modifying TCP's Congestion Control for High Speeds", May 2002. URL "http://www.icir.org/floyd/notes.html".

[J02] Amit Jain, "Initial Congestion Window Discovery", rough draft, work in progress, 2002. Citation for acknowledgement purposes only.

15. Security Considerations

This proposal makes no changes to the underlying security of TCP.

16. IANA Considerations

There are no IANA considerations regarding this document.

A. TCP's Loss Event Rate in Steady-State

This section gives the number of round-trip times between congestion events for a TCP flow with D-byte packets, for D=1500, as a function of the connection's average throughput B in bps. To achieve this average throughput B, a TCP connection with round-trip time R in seconds requires an average congestion window w of BR/(8D) segments.

In steady-state, TCP's average congestion window w is roughly 1.2/sqrt(p) segments. This is equivalent to a lost event at most once every 1/p packets, or at most once every 1/(pw) = w/1.5 round-trip times. Substituting for w, this is a loss event at most every (BR)/12D)round-trip times.

An an example, for R = 0.1 seconds and D = 1500 bytes, this gives B/180000 round-trip times between loss events.

B. A table for a(w) and b(w).

This section gives a table for the increase and decrease parameters a(w) and b(w) for HighSpeed TCP, for the default values of Low_Window = 38, High_Window = 83000, High_P = 10^-7, and High_Decrease = 0.1.

w	a(w)	b(w)
38	1	0.50
118	2	0.44
221	3	0.41
347	4	0.38
495	5	0.37
663	6	0.35
851	7	0.34
1058	8	0.33
1284	9	0.32
1529	10	0.31
1793	11	0.30
2076	12	0.29
2378	13	0.28
2699	14	0.28
3039	15	0.27
3399	16	0.27
3778	17	0.26
4177	18	0.26
4596	19	0.25
5036	20	0.25
5497	21	0.24
5979	22	0.24
6483	23	0.23
7009	24	0.23
7558	25	0.22
8130	26	0.22
8726	27	0.22
9346	28	0.21
9991	29	0.21
10661	30	0.21
11358	31	0.20
12082	32	0.20
12834	33	0.20
13614	34	0.19
14424	35	0.19
15265	36	0.19
16137	37	0.19
17042	38	0.18
17981	39	0.18
18955	40	0.18
19965	41	0.17
21013	42	0.17
22101	43	0.17
23230	44	0.17
24402	45	0.16
25618	46	0.16

26881	47	0.16
28193	48	0.16
29557	49	0.15
30975	50	0.15
32450	51	0.15
33986	52	0.15
35586	53	0.14
37253	54	0.14
38992	55	0.14
40808	56	0.14
42707	57	0.13
44694	58	0.13
46776	59	0.13
48961	60	0.13
51258	61	0.13
53677	62	0.12
56230	63	0.12
58932	64	0.12
61799	65	0.12
64851	66	0.11
68113	67	0.11
71617	68	0.11
75401	69	0.10
79517	70	0.10
84035	71	0.10
89053	72	0.10
94717	73	0.09

Table 4: Parameters for HighSpeed TCP.

This table was computed with the following Perl program:

```
$top = 100000;
$num = 38;
if ($num == 38) {
  print "     w  a(w)  b(w)0;
  print "  ----  ----  ----0;
  print "    38     1  0.500;
  $oldb = 0.50;
  $olda = 1;
}
while ($num < $top) {
  $bw = (0.1 -0.5)*(log($num)-log(38))/(log(83000)-log(38))+0.5;
  $aw = ($num**2*2.0*$bw) / ((2.0-$bw)*$num**1.2*12.8);
  if ($aw > $olda + 1) {
     printf "%6d %5d  %3.2f0, $num, $aw, $bw;
     $olda = $aw;
  }
  $num ++;
}
```

Table 5: Perl Program for computing parameters for HighSpeed TCP.

AUTHORS' ADDRESSES

Sally Floyd
Phone: +1 (510) 666-2989
ICIR (ICSI Center for Internet Research)
Email: floyd@icir.org
URL: http://www.icir.org/floyd/

This draft was created in June 2002.