Internet Engineering Task Force
INTERNET DRAFT
draft-amit-quick-start-02.txt

Amit K. Jain
Array Networks
Sally Floyd
ICIR

October, 2002

Quick-Start for TCP and IP

Status of this Memo

Abstract

This draft outlines an optional Quick-Start mechanism for transport protocols to determine an optional allowed initial congestion window or initial sending rate at the start of a data transfer. This mechanism is designed to be used by a range of transport protocols; however, in this document we only describe its use with TCP and IPv4. By using Quick-Start, a TCP host, say, host A, would indicate its desired initial sending rate in packets per second in a Quick Start Request option in the IP header of the initial TCP SYN or SYN/ACK packet. Each router in turn could either approve the specified initial rate, reduce the specified initial rate, or indicate that nothing above the default initial rate for that protocol would be allowed. The Quick-Start mechanism also can determine if there are routers along the path that do not understand the Quick Start Request

option, or have not agreed to the initial rate described in the option. TCP host B communicates the final initial rate to TCP host A in a transport-level Quick-Start Response in the answering SYN/ACK or ACK packet. Quick-Start is designed to allow TCP connections to use high initial windows in circumstances when there is significant unused bandwidth along the path, and all of the routers along the path support the Quick-Start Request. This proposal is a request for feedback from the community.

Changes from draft-amit-quick-start-01.txt:

* Added a discussion in the related work section about the possibility of optimistically sending a large initial window, without explicit permission of routers.
* Added a discussion in the related work section about the tradeoffs of XCP vs. Quick-Start.
* Added a section on "The Quick-Start Request: Packets or Bytes?"

Changes from draft-amit-quick-start-00.txt:

* The addition of a citation to [KHR02].
* The addition of a Related Work section.
* Deleted the QS Nonce, in favor of a random initial value for the QS TTL.

1. Introduction

The life of a TCP connection begins with a question, that is, "With what rate I can transfer the data?" The question is not answered explicitly for TCP, but each TCP connection figures out the answer for itself. This is done by each connection starting with an initial congestion window (called cwnd) of from one to four MSS-sized segments, for an MSS the Maximum Segment Size in bytes. Currently, the TCP protocol allows an initial window of three or four segments [RFC3390]. The TCP connection then probes the network for available bandwidth using the slow-start procedure, essentially doubling its congestion window each round-trip time.

The probing mechanism of slow-start is time-consuming and causes an overhead in terms of queueing delay. It may take a number of round-trip times in slow-start before the TCP connection begins to fully use the available bandwidth of the network; it takes log N round-trip times to build up to a congestion window of N segments. This time in slow-start is not a problem for large file transfers, where the slow-start stage is only a fraction of the total transfer time. However, in the case of moderate-sized web transfers the connection might carry out its entire transfer in the slow-start phase. In an underutilized, high-bandwidth network, such a transfer can end up taking log N round-trip times to transfer the data, where one or two round-trip times might have sufficed.

A fair amount of work has already been done to address the issue of choosing the initial window for TCP, with RFC 3390 allowing an initial window of up to four segments [RFC3390]. Our underlying premise is that explicit feedback from all of the routers along the path would be required for best-effort connections to use initial windows higher than four segments.

The Congestion Manager proposes sharing congestion information among multiple TCP connections with the same endpoints [RFC2140]. With the Congestion Manager, a new TCP connection could start with a high initial window if it was sharing the path with a pre-existing TCP connection, with a high congestion window, to the same destination. It is also possible that a newly-starting TCP connection could make use of congestion information from a recently-terminated TCP connection to the same destination. However, neither of these approaches are of any use for a connection to a new destination when there is no existing or recent connection to that destination.

Active Queue Management and Explicit Congestion Notification (ECN) are both based on the router detecting congestion before buffer overflow [RFC3168]. In a similar but somewhat simpler fashion, Quick-Start is based on the ability of the router to determine whether or not the output link is significantly underutilized.

2. Assumptions, General Principles, and Open Questions

This section describes the assumptions and general principles behind the design of the Quick-Start mechanism.

Assumptions:

* A router can determine reasonably easily if the output link has been significantly underutilized over a period of time.

* The data transfer in the two directions of a connection traverses different queues, and possibly even different routers. Thus, any mechanism for determining the allowed initial window or initial sending rate would have to be used independently for each direction.

* Any new mechanism must be incrementally deployed, and may not be supported by all of the routers and/or end-hosts. Thus, any new mechanism must be able to accommodate non-supporting routers or end-hosts without disturbing the current Internet semantics.

* After the initial SYN exchange, a TCP data sender would be able to translate an initial sending rate in packets per second into an initial congestion window of MSS-sized segments.

General Principles:

* In order for best-effort connections to use initial windows higher than four segments, explicit feedback from all of the routers along the path would be required.

* A router should only allow an initial sending rate higher than the transport protocol's default initial rate if the router is significantly underutilized. Any other approach will result in either per-flow state at the router, or the possibility of a (possibly transient) queue at the router.

* No per-flow state is kept at the router for this mechanism. In Quick-Start, the only state kept at the router is the aggregate initial sending rate authorized over the most recent interval of time (e.g., quarter of a second).

There are also a number of open questions regarding the Quick-Start mechanism outlined in this draft.

Open Questions:

* Would the benefits of the Quick-Start mechanism be worth the added complexity?

One drawback of the Quick-Start mechanism is that the SYN and SYN/ACK packets containing the Quick-Start option would presumeably be processed in the slow path in routers. This would reduce the capacity of routers to handle Quick-Start requests, and delay the initial SYN exchange for the connection.

Another drawback is that Quick-Start would require functionality in the router to estimate the current link utilization, and to keep track of the aggregate Quick-Start rate authorized over the last interval of time. Adding new functionality to routers should not be done lightly, and any mechanisms that would require new functionality in routers would have to be carefully considered.

* Apart from the merits and shortcomings of the Quick-Start mechanism, is there likely to be a compelling need to add explicit congestion-related feedback from routers over and above the one-bit feedback from ECN?

* If the answer to the question above is yes, should we be considering mechanisms that, while more complex, are also sufficiently more powerful than Quick-Start?

There are a number of such mechanisms that have been proposed in the literature for more fine-grained congestion-related feedback from routers [KHR02]. Quick-Start is extremely coarse-grained, in that in its current form it only applies to the initial window of the connection. Quick-Start also focuses in on the specific issue of allowing very high initial sending rates for connections over underutilized, high-bandwidth paths. Quick-Start might first be deployed, for example, in an overprovisioned high-bandwidth Intranet, to allow much quicker transfers of best-effort traffic.

3. The Quick-Start Request in IPv4.

Quick-Start would require end-points and routers to work together, with end-points requesting a higher initial sending rate in the Quick-Start Request (QSR) option in IP, and routers along the path approving, modifying, or discarding or ignoring (and therefore disallowing) the Quick-Start Request. The receiver would use transport-level mechanisms to inform the sender of the status of the Quick-Start Request. We note that the Quick-Start Request implicitly assumes a unicast transport protocol; we have not considered the use of the Quick-Start Request for multicast traffic.

3.1. The Quick-Start Request Option for IPv4

The Quick-Start Request for IPv4 would be defined as follows:

```
      0          1          2          3
+----------+----------+----------+----------+
| Option   | Length=4 |  QS TTL  | Initial  |
|          |          |          | Rate     |
+----------+----------+----------+----------+
```

Figure 1. The Quick-Start Request Option for IPv4.

The first byte contains the option field, which includes the one-bit copy flag, the 2-bit class field, and the 5-bit option number.

The second byte contains the length field, which indicates an option length of four bytes.

The third byte contains the Quick-Start TTL (QS TTL) field. If the sender decides to use Quick-Start, then the sender sets the QS TTL field to a random value. Routers that approve the Quick-Start Request decrement the QS TTL (mod 256). The QS TTL is used by the sender to detect if all of the routers along the path understood and approved the Quick-Start option.

The TCP sender also calculates and remembers the TTL Diff, the difference between the TTL value and the QS TTL value in the transmitted SYN packet, as follows:

TTL Diff = (TTL - QS TTL) mod 256

The fourth byte is the requested Initial Rate (IR) field. The sender initializes the Initial Rate to the desired initial sending rate. The current proposal is for this field to be formatted in packets per 0.1 sec. Routers can approve the Quick-Start Request for a lower Initial Rate by decreasing the Initial Rate in the Quick-Start Request.

Note that the one-byte Initial Rate field, formatted in packets per 0.1 sec, limits the Initial Rate to at most 2560 packets/sec. For 1500-byte packets, this corresponds to an initial rate of 30 Mbps. A larger option field, or a different encoding for the one-byte requested Initial Rate option, would be needed to allow a higher range for the requested initial rate. One suggestion has been for an Initial Rate field encoded on a logarithmic scale, to allow a wider range of Initial Rates.

If the Quick-Start Request is not approved, then the sender uses the default initial rate or initial window for that transport protocol.

3.1.1. The Quick-Start Request: Packets or Bytes?

One of the design questions is whether the Initial Rate field should be in bytes per second or in packets per second. We will discuss this separately from the perspective of the transport, and from the perspective of the router.

For TCP, the results from the Quick-Start Request are translated into an initial window in bytes, using the measured round-trip time and the MSS. This initial window applies only to the bytes of data payload, and does not include the bytes in the TCP or IP packet headers. Other transport protocols would conceivably use the Quick-Start Request directly in packets per second, or could translate the Quick-Start Request to an initial window in packets.

The assumption of this draft is that the router only approves the Quick-Start Request when the output link is significantly underutilized. For this, the router could measure the available bandwidth in bytes per second, or could convert between packets and bytes using the MTU of the output link.

If the Quick-Start Request was in bytes per second, and applied only to the data payload, then the router would have to convert from bytes per second of data payload, to bytes per second of packets on the wire. If the Initial Rate field was in bytes per second and the sender ended up using very small packets, this could translate to a significantly larger number in terms of bytes per second on the wire.

It has been suggested that the router could possibly use information from the MSS option in the TCP packet header of the SYN packet to convert the Quick-Start Request from packets per second to bytes per second, or vice serva. One problem is that the MSS option is defined as the maximum MSS that the TCP sender expects to receive, not the maximum MSS that the TCP sender plans to send [RFC793].

We note that the sender does not necessarily know the Path MTU when the Quick-Start Request is sent, or when the initial window of data is sent. Thus, packets from the initial window could end up being fragmented in the network if the "Don't Fragment" (DF) bit is not set [RFC1191]. Interactions between larger initial windows and Path MTU Discovery are discussed in more detail in RFC 3390 [RFC3390].

We have chosen an Initial Rate field in packets per second rather than in bytes per second because it seems somewhat more robust (avoiding big surprises if the sender ends up using small packets). However, we note that more consideration of this issue is probably needed.

3.2. Processing the Quick-Start Request at Routers

Each participating router can either terminate or forward the Quick-Start Request. The router terminates the Quick-Start Request if the router is not underutilized, and therefore has decided not to grant the Quick-Start Request.

The preferable method for a router to terminate the Quick-Start Request is to delete the Quick-Start Request from the IP header. A less preferable but possibly more efficient method is to simply forward the packet with the Quick-Start Request unchanged, or with the Initial Rate set to zero.

If the participating router has decided to approve the Quick-Start Request, it does the following:

* It decrements the QS TTL by one.

* If the router is only willing to approve an Initial Rate less than that in the Quick-Start Request, then the router puts the new Initial Rate in that field of the Quick-Start Request.

A non-participating router forwards the Quick-Start Request unchanged, without decrementing the QS TTL. Of course, the non-participating router still decrements the TTL field in the IP header, as is required for all routers. As a result, the TCP sender will be able to detect that the Quick-Start Request is not valid.

A router that modifies or deletes the Quick-Start Request in the IPv4 header also has to update the IPv4 Header checksum.

3.3. Deciding the Permitted Initial Rate at a Router

In this section we briefly outline how a router might decide whether or not to approve a Quick-Start Request. As an example, the router could ask the following questions:

* Has the router's output link been underutilized for some time (e.g., several seconds).

* Would the output link remain underutilized if the arrival rate was to increase by the aggregate initial rate that the router has approved over the last fraction of a second?

Answering this question requires that the router have some knowledge of the available bandwidth on the output link for that output queue. It also requires that the router keep two counters, one indicating the total aggregate Initial Rates that have been approved over the recent interval of time, and one for the total aggregate Initial Rates approved over the previous interval of time. Thus, if an underutilized router experiences a SYN flood, then the router would begin to deny Initial Rate requests, even if the router remains underutilized.

* If the answer to both of the previous questions is Yes, then the router could approve the Quick-Start Request. The router could allow an Initial Rate that was a small fraction of the available unused bandwidth of the output link.

4. The Quick-Start Mechanisms in TCP.

This section describes how the Quick-Start mechanism would be used in TCP. If a TCP sender, say host A, would like to request Quick-Start, the TCP sender puts the requested initial sending rate in packets per second in the Quick-Start Request option in the IP header of the SYN or SYN/ACK packet, as described above. The TCP host B returns the Quick-Start Response option in the TCP header in the responding SYN/ACK packet or ACK packet, respectively, informing host A of the results of their request.

If the returning packet does not contain a Quick-Start Response, or contains a Quick-Start Response with the wrong value for the TTL Diff, then host A has to assume that its Quick-Start request failed.

In this case, host A uses TCP's default initial window.

If the returning packet does contain a valid Quick-Start Response, then host A uses the information in the response, along with its measurement of the round-trip time, to determine the initial congestion window. In order to use Quick-Start, the TCP host would also be required to use rate-based pacing to pace out the packets in the initial window at the rate indicated in the Quick-Start Response.

In a similar manner, if TCP host B requests Quick-Start in the IP header of the TCP SYN/ACK packet, then TCP host A returns the Quick-Start Response in the TCP header of the answering ACK packet. The two TCP end-hosts can independently decide whether to request Quick-Start.

4.1. The Quick-Start Response Option in the TCP header

The Quick-Start Response option is as follows:

```
     0          1          2          3
+----------+----------+----------+----------+
|   Kind   | Length=4 | Initial  |   TTL    |
|          |          |   Rate   |   Diff   |
+----------+----------+----------+----------+
```

Figure 2. The Quick-Start Response option in the TCP header.

The first byte of the Quick-Start Response option contains the option kind, identifying the TCP option.

The second byte of the Quick-Start Response option contains the length field, specifying the option length in bytes. The length field is set to four bytes.

The third byte of the Quick-Start Response option contains the allowed Initial Rate, formatted as in the Quick-Start Request option.

The fourth byte of the TCP option contains the TTL Diff. The TTL Diff contains the difference, in the received SYN or SYN/ACK packet, between the TTL field in the IP header and the QS TTL field in the Quick-Start Request Option.

4.2. Sending the Quick-Start Response

An end host, say host B, that receives a TCP SYN or SYN/ACK packet containing a Quick-Start Request passes the Quick-Start Request, along with the value in the TTL field in the IP header, to the receiving TCP layer.

If the TCP host is willing to permit the Quick-Start request, then a Quick-Start Response option is included in the TCP header of the answering acknowledgement packet. The Initial Rate in the Quick-Start Response option is set to the received value of the Initial Rate in the Quick-Start Request option, or to a lower value if the TCP receiver is only willing to allow a lower Initial Rate. The TTL Diff in the Quick-Start Response is set to the difference between the TTL value and the QS TTL value as follows:

TTL Diff = (TTL - QS TTL) mod 256

If the Quick-Start Response is being sent on the SYN/ACK, in response to a Quick-Start Request on the SYN, then the Quick-Start Response will be resent if the SYN/ACK has to be retransmitted. If the Quick-Start Response is being sent on the ACK, in response to the Quick-Start Request on the SYN/ACK, then the Quick-Start Response has to be resent on data packets until that TCP host receives an acknowledgement from the other end.

4.3. Receiving and Using the Quick-Start Response

A TCP host, say TCP host A, that sent a Quick-Start Request in a SYN or SYN/ACK, and receives an answering Quick-Start Response in the acknowledgement, first checks that the Quick-Start Response is valid. The Quick-Start Response is valid if it contains the correct value for the TTL Diff, and an equal or lesser value for the Initial Rate than that transmitted in the Quick-Start Request. If this check is not successful, then the Quick-Start request failed, and the TCP host uses the default TCP initial window.

If the checks of the TTL Diff and the Initial Rate are successful, then the TCP host sets its initial congestion window to R*T*MSS bytes, for R the Initial Rate in packets per second and T the measured round-trip time in seconds. The TCP host sets a flag that it is in Quick-Start mode, and while in Quick-Start mode the TCP sender uses rate-based pacing, pacing out packets at the specified Initial Rate.

Because the initial SYN packet with the Quick-Start Request presumeably was not able to use the fast path in routers, the initial round-trip time measurement might be unnecessarily large. The Quick-Start mode ends when the TCP host first receives an ACK for one of the data packets. If the initial congestion window has not been fully used, then the initial congestion window is decreased to the amount that has actually been used so far. This should address the problem of an overly-large congestion window from an overly-large initial measurement of the round-trip time.

After sending the initial window, the TCP sender remains in slow-start, and continues to increase its congestion window rather aggressively from one round-trip time to the next. To add robustness, the TCP sender would be required to use Limited Slow-Start along with Quick-Start. With Limited Slow-Start, the TCP sender limits the number of segments by which the congestion window is increased for one window of data during slow-start [F02a].

4.4. An Example Quick-Start Scenario with TCP

The following is an example scenario in the case when both hosts request Quick-Start:

* The TCP SYN packet from Host A contains a Quick-Start Request in the IP header.

* Routers along the forward path modify the Quick-Start Request as appropriate.

* Host B receives the Quick-Start Request in the SYN packet, and calculates the TTL Diff. If Host B approves the Quick-Start Request, then Host B sends a Quick-Start Response in the TCP header of the SYN/ACK packet. Host B also sends a Quick-Start Request in the IP header of the SYN/ACK packet.

* Routers along the reverse path modify the Quick-Start Request as appropriate.

* Host A receives the Quick-Start Response in the SYN/ACK packet, and checks the TTL Diff and Initial Rate for validity. If they are valid, then Host A sets its initial congestion window appropriately, and sets up rate-based pacing to be used with the initial window. If the Quick-Start Response is not valid, then Host A uses TCP's default initial window.

  Host A also calculates the TTL Diff for the Quick-Start Request in the incoming SYN/ACK packet, and sends a Quick-Start Response in the TCP header of the ACK packet.

* Host A repeats sending the Quick-Start Response in data packets at least once per round-trip time until it receives an acknowledgement from Host B for one of those data packets.

* Host B receives the Quick-Start Response in an ACK packet, and checks the TTL Diff and Initial Rate for validity. If the Quick-Start Response is valid, then Host B sets its initial congestion window appropriately, and sets up rate-based pacing to be used with the initial window. If the Quick-Start Response is not valid, then

Host B uses TCP's default initial window.

5. Evaluation of Quick-Start

The main benefit of Quick-Start is to the transport connection itself. For a small TCP transfer of one to five packets, Quick-Start is probably of very little benefit; at best, it might shorten the connection lifetime from three to two round-trip times (including the round-trip time for connection establishment). Similarly, for a very large transfer, where the slow-start phase would have been only a small fraction of the connection lifetime, Quick-Start would be of limited benefit. Quick-Start would not significantly shorten the connection lifetime, but it might eliminate or at least shorten the start-up phase. However, for moderate-sized connections of N packets in well-provisioned environments that allow Quick-Start requests of M packets per second, the use of Quick-Start could shorten the connection lifetime from log N round-trip times to at most N/M + 1 round-trip times. For large values of N and M, this would be a dramatic shortening of the connection lifetime.

The main cost of Quick-Start concerns the costs of added complexity at the routers. The added complexity at the end-points is moderate, and might easily be outweighed by the benefit of Quick-Start to the end hosts. The added complexity at the routers is also somewhat moderate, in that it involves estimating the unused bandwidth on the output link over the last several seconds, and keeping a counter of the aggregate Quick-Start rate approved over the last fraction of a second. However, this added complexity at the routers adds to the development cycle, and could prevent the addition of other competing functionality to routers. Thus, careful thought would have to be given to the addition of Quick-Start to IP.

Another drawback of Quick-Start is that packets containing the Quick-Start Request message presumeably would not take the fast path in routers. This would mean extra delay for the end hosts, and extra processing burden for the routers. This extra burden is mitigated somewhat by the following factors: only SYN and SYN-ACK packets would carry the Quick-Start Request option; very small flows of, say, one to five packets would receive little benefit from Quick-Start, and presumeably would not use the Quick-Start Request; flows from end hosts with low-bandwidth access links would receive little benefit from Quick-Start, and hopefully could be configured not to use the Quick-Start Request. In addition, in typical environments where most of the packets belong to large flows, the burden of the Quick-Start Option on routers would be considerably reduced. Nevertheless, it is still conceiveable, in the worst case, that up to 10% of the packets were SYN or SYN/ACK packets using a Quick-Start Request, and this could slow down the processing of SYN or SYN/ACK packets in routers considerably.

One limitation of Quick-Start is that it presumes that the data packets of a connection will follow the same path as the SYN or SYN/ACK packet. If this is not the case, then the connection could be sending the initial window, at the permitted initial rate, along a path that was already congested, or that became congested as a result of this connection. This is, however, similar to what would happen if the connection's path was changed in the middle of the connection, when the connection had already established the allowed initial rate.

A problem of any mechanism for feedback from routers at the IP level is that there can be queues and bottlenecks in the end-to-end path that are not in IP-level routers. As an example, these include queues in layer-two Ethernet or ATM networks. The hope would be that an IP-level router adjacent to such a non-IP queue or bottleneck would be configured to reject Quick-Start requests if that was appropriate.

The discussion in this paper has largely been of the Quick-Start mechanism with default, best-effort traffic. However, Quick-Start could also be used by traffic using some form of differentiated services, and routers could take the traffic class into account when deciding whether or not to grant the Quick-Start request. However, we would note that routers should be discouraged from granting Quick-Start requests for higher-priority traffic when this is likely to result in significant packet loss for lower-priority traffic.

The Quick-Start proposal, taken together with the recent proposal for HighSpeed TCP [F02], would go a significant way towards extending the range of performance for best-effort traffic in the Internet. However, there are many things that the Quick-Start proposal would not accomplish. For example, Quick-Start as it is currently specified would not allow flows to ramp-up quickly in the middle of the connection. Quick-Start would not help in making more precise use of the available bandwidth, that is, of achieving the goal of very high throughput with very low delay and very low packet loss rates. Quick-Start would not give routers any additional power in allocating bandwidth in the interests of greater fairness, or in having more control over slow decrease rates of active connections. One of the open questions is whether the limited capabilities of Quick-Start are sufficient to warrant standardization and deployment, or whether more work is needed to explore the space of potential mechanisms.

6. Other Mechanisms for Fast Start-ups.

Any evaluation of Quick-Start must include evaluating the relative benefits of approachs that use no explicit information from routers, and of approaches that use more fine-grained feedback from routers as part of a larger congestion control mechanism. We discuss three classes of proposals (no explicit feedback from routers; explicit feedback about the initial rate; and more fine-grained feedback from routers) in the sections below.

6.1 Faster Start-ups without Explicit Information from Routers

One possibility would be for senders to use information from the packet streams to learn about the available bandwidth, without explicit information from routers.

[JD02] explores the use of periodic packet streams to estimate the available bandwidth along a path. The idea is that the one-way delays of a periodic packet stream show an increasing trend when the stream's rate is higher than the available bandwidth. While [JD02] states that the proposed mechanism does not cause significant increases in network utilization, losses, or delays when done by one flow at a time, the approach could be problematic if conducted concurrently by a number of flows. [JD02] also gives an overview of some of the earlier work on inferring the available bandwidth from packet trains.

One possible path for future research would be to explore the limits of the ability of TCP flows to infer the available bandwidth using their own packet stream, without explicit feedback from the router. One advantage of explicit feedback from the router is that it allows the TCP sender to discover the available bandwidth immediately after the initial SYN exchange, possibly allowing a very large initial window. A second advantage of explicit feedback from the router is that the available bandwidth along the path does not necessarily map to the allowed sending rate for an individual flow; when multiple flows are trying to infer their allowed sending rate, the use of explicit feedback from the router adds considerable robustness. Nevertheless, it would also be useful to explore the limits of start-up behavior without explicit feedback from the router.

As an example, if the TCP sender sends four packets back-to-back in the initial window, and the TCP receiver reports the timing of the receipt of the data packets, and the data packets were received with roughly the same spacing as they were transmitted, does this mean that the flow can infer an underutilized path? What if the round-trip time is also considerably larger than the transmission time of the four packets? And if the sender can infer an underutilized path, can the sender increase faster than slow-start for the next window of data? While it seems clear that approaches *without* explicit feedback from the routers will be strictly less powerful that is possible *with* explicit feedback, it is also possible that some approaches that are more aggressive than slow-start are possible without explicit feedback from routers. Proposals for the TCP sender to infer about available bandwidth along the path without explicit feedback from routers include the Swift Start proposal in [PRAKS02]. Swift Start combines packet-pair and packet-pacing techniques, beginning with a four-segment burst of packets to estimate the available bandwidth along the path.

Another possibility that has been suggested [S02] is for the sender to start with a large initial window without explicit permission from the routers, and for the first packet of the initial window to contain information such as the size or sending rate of the initial window. The proposal would be that routers under congestion would use this information in the first data packet to drop or delay many or all of the packets from that initial window. In this way a flow's optimistically-large initial window would not force the router to drop packets from competing flows in the network.

Obviously there would be a number of questions to consider about an approach of optimistic sending. One question would be the potential complications of incremental deployment, where some of the routers along the path might not understand the packet information describing the initial window. There could also be concerns about congestion collapse if many flows used large initial windows, many complete sets of initial windows were dropped, and many congested links ended up carrying packets that are only going to be dropped downstream. A more thorough understanding of the dangers (or absence of dangers) of such optimistic larger initial windows would be useful.

6.2. Faster Start-ups with other Information from Routers

There have been several proposals similar to Quick-Start where the transport protocol collects explicit information from the routers along the path.

In related work, Joon-Sang Park and John Heidemann investigated the use of a slightly different IP option for TCP connections to discover the available bandwidth along the path. In that variant, the IP option would query the routers along the path about the smallest available free buffer size. Also, the IP option was sent after the initial SYN exchange, when the TCP sender already had an estimate of the round-trip time.

The Performance Transparency Protocol (PTP) includes a proposal for a single PTP packet that would collect information from routers along the path from the sender to the receiver [W00]. For example, a single PTP packet could be used to determine the bottleneck bandwidth along a path.

Additional proposals for end nodes to collect explicit information from routers include Explicit Transport Error Notification (ETEN), which includes a cumulative mechanism to notify endpoints of aggregate congestion statistics along the path [KAPS02].

6.3. Faster Start-ups with more Fine-Grained Feedback from Routers

Proposals for more fine-grained congestion-related feedback from routers include XCP [KHR02]. Proposals such as XCP are more powerful than Quick-Start, involving significant changes to the congestion control mechanisms of the Internet, but also are more complex to understand and more difficult to deploy.

We do not discuss proposals such as XCP in detail, but simply note that there are a number of open questions. One question concerns whether there is a pressing need for more sophisticated congestion control mechanisms such as XCP in the Internet. Quick-Start is inherently a rather crude tool that does not deliver assurances about maintaining high link utilization and low queueing delay, for example; Quick-Start is designed for use in environments that are significantly underutilized. More powerful mechanisms with more fine-grained feedback from routers can allow faster startups even in environments with rather high link utilization. Is this a pressing requirement?

A second question concerns whether mechanisms such as Quick-Start, in combination with HighSpeed TCP and other changes in progress, would make a significant contribution towards meeting some of these needs for new functionality. This could be viewed as a positive step of meeting some of the current needs with a simple and reasonably deployable mechanism, or alternately, as a negative step of unnecessarily delaying more fundamental changes. Without answering this question, we would note that our own approach tends to favor the incremental deployment of relatively simple mechanisms, as long as the simple mechanisms are not short-term hacks but mechanisms that lead the overall architecture in the fundamentally correct direction.

7. Conclusions

We are presenting the Quick-Start mechanism not as a mechanism that we believe is urgently needed in the current Internet, but rather as a proposal for a simple, understandable, and incrementally-deployable mechanism that would be sufficient to allow connections to start up with large initial rates, or large initial congestion windows, in overprovisioned, high-bandwidth environments. We are not making any predictions about what it likely to be a typical environment in the future Internet. However, we expect there will be an increasing number of overprovisioned, high-bandwidth environments where the Quick-Start mechanism, or another mechanism of similar power, could be of significant benefit to a wide range of traffic. We are presenting the Quick-Start mechanism as a request for feedback from the Internet community in considering these issues.

8. Acknowledgements

The authors wish to thank Mark Handley for discussions of these issues. The authors also thank the End-to-End Research Group, the Transport Services Working Group, and members of IPAM's program on Large Scale Communication Networks for both positive and negative feedback on this proposal. We also thank Mark Allman, Mohammed Ashraf, John Border, Tom Dunigan, John Heidemann, Dina Katabi, and Vern Paxson for feedback. This draft builds upon the concepts described in [RFC3390], [AHO98], [RFC2415], and [RFC3168].

This is a modification of a draft originally by Amit Jain for Initial Window Discovery.

9. Normative References

[RFC793] J. Postel, Transmission Control Protocol, RFC 793, September 1981.

[RFC1191] Mogul, J. and S. Deering, Path MTU Discovery, RFC 1191, November 1990.

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

[RFC3168] Ramakrishnan, K.K., Floyd, S., and Black, D. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168, Proposed Standard, September 2001.

[RFC3390] M. Allman, S. Floyd, and C. Partridge. Increasing TCP's Initial Window. RFC 3390, October 2002.

10. Informative References

[AHO98] M. Allman, C. Hayes and S. Ostermann. An evaluation of TCP with Larger Initial Windows. ACM Computer Communication Review, July 1998.

[FF99] Floyd, S., and Fall, K., Promoting the Use of End-to-End Congestion Control in the Internet, IEEE/ACM Transactions on Networking, August 1999.

[F02] Floyd, S., HighSpeed TCP for Large Congestion Windows, internet-draft draft-floyd-tcp-highspeed-01.txt, work in progress, August 2002.

[F02a] Floyd, S., Limited Slow-Start for TCP with Large Congestion Windows, internet-draft draft draft-floyd-tcp-slowstart-01.txt, work in progress, August 2002.

[JD02] Manish Jain, Constantinos Dovrolis, End-to-End Available Bandwidth: Measurement Methodology, Dynamics, and Relation with TCP Throughput, SIGCOMM 2002.

[KAPS02] Rajesh Krishnan, Mark Allman, Craig Partridge, James P.G. Sterbenz. Explicit Transport Error Notification (ETEN) for Error-Prone Wireless and Satellite Networks. Technical Report No. 8333, BBN Technologies, March 2002. URL "http://roland.lerc.nasa.gov/~mallman/papers/".

[KHR02] Dina Katabi, Mark Handley, and Charles Rohrs, "Internet Congestion Control for Future High Bandwidth-Delay Product Environments." ACM Sigcomm 2002, August 2002. URL "http://ana.lcs.mit.edu/dina/XCP/".

[PK98] Venkata N. Padmanabhan and Randy H. Katz, TCP Fast Start: A Technique For Speeding Up Web Transfers, IEEE GLOBECOM '98, November 1998.

[PRAKS02] Craig Partridge, Dennis Rockwell, Mark Allman, Rajesh Krishnan, James P.G. Sterbenz. A Swifter Start for TCP. Technical Report No. 8339, BBN Technologies, March 2002. URL "http://roland.lerc.nasa.gov/~mallman/papers/".

[RFC2140] J. Touch. TCP Control Block Interdependence. RFC 2140. April 1997

[RFC2309] B. Braden, D. Clark, J. Crowcroft, B. Davie, S. Deering, D. Estrin, S. Floyd, V. Jacobson, G. Minshall, C. Partridge, L. Peterson, K. Ramakrishnan, S. Shenker, J. Wroclawski, L. Zhang, Recommendations on Queue Management and Congestion Avoidance in the Internet, RFC 2309, April 1998.

[RFC2415] K. Poduri and K. Nichols. Simulation Studies of Increased Initial TCP Window Size. RFC 2415. September 1998.

[RFC2416] T. Shepard and C. Partridge. When TCP Starts Up With Four Packets Into Only Three Buffers. RFC 2416. September 1998.

[W00] Michael Welzl: PTP: Better Feedback for Adaptive Distributed Multimedia Applications on the Internet, IPCCC 2000 (19th IEEE International Performance, Computing, And Communications Conference), Phoenix, Arizona, USA, 20-22 February 2000. URL "http://informatik.uibk.ac.at/users/c70370/research/publications/".

[S02] Ion Stoica, private communication, 2002. Citation for acknowledgement purposes only.

11. Security Considerations

The only security consideration would be if the use of Quick-Start resulted in the sender using an Initial Rate that was inappropriately large, resulting in congestion along the path. Such congestion could result in an unacceptable level of packet drops along the path. Such congestion could also be part of a Denial of Service attack.

A misbehaving TCP sender could use a non-conformant initial congestion window even without the use of Quick-Start, so we restrict our attention to problems with Quick-Start with conformant TCP senders. (We also note that if the TCP sender is a busy web server, then the TCP sender has some incentive to be conformant in this regard.)

If a router that understands the Quick-Start Request deletes the Request, or zeroes the QS TTL in the request, then the chances of a downstream router or misbehaving receiver guessing the value of the QS TTL is at most 1/256. Thus, deleting the Quick-Start Request makes it unlikely that the receiver would be able to send a valid Quick-Start Response back to the sender.

If there are routers along the path that do not understand or approve of the Quick-Start Request, and that forward the Quick-Start Request unchanged, it would be not be easy for a downstream router or the receiver to cheat and modify the QS TTL field so that the request was considered valid, because the downstream routers do not know the initial value for the QS TTL.

It would be easy for routers or for the receiver to increase the Initial Rate, making the Initial Rate higher than that approved by upstream routers. Routers could only effectively cheat in this manner if there were no downstream routers that objected to that Initial Rate. Receivers, however, would easily increase the Initial Rate returned in the Quick-Start Response, causing unnecessary congestion for the next round-trip time along the path. However, this higher Initial Rate will only be considered valid by the TCP sender if it is no higher than the Initial Rate originally requested by the sender. Thus, this limits the ability of the TCP receiver to cheat in this regard.

12. IANA Considerations

The only IANA Considerations would be the addition of an IP option to the list of IP options, and the addition of a TCP option to the list of TCP options.

13. Discussion of a QuickStart Nonce

An earlier version of this document included a QuickStart Nonce that was initialized by the sender to a non-zero, 'random' eight-bit number, along with a QS TTL that was initialized to the same value at the TTL in the IP header. The QuickStart Nonce would have been returned by the TCP receiver to the TCP sender in the Quick-Start Response. A router could deny the Quick-Start request by failing to decrement the QS TTL field, by zeroing the QS Nonce field, or by deleting the Quick-Start Request from the packet header. The QS Nonce was included to provide some protection against broken downstream routers, or against misbehaving TCP receivers who might be inclined to lie about the Initial Rate. This protection is now provided by the use of a random initial value for the QS TTL field.

With the old QuickStart Nonce, along with the QS TTL field set to the same value as the TTL field in the IP header, the Quick-Start Request mechanism would have been self-terminating; the Quick-Start Request would terminate at the first participating router after a non-participating router has been encountered on the path. This would have minimized unnecessary overhead incurred by routers because of option processing for the Quick-Start Request. Thus, one disadvantage of the new approach with a random initial value for the QS TTL field is that intermediate routers can no longer determine when some upstream router has not understood the QuickStart option. However, a disadvantage of the old approach was that it offered no protection against downstream routers or the TCP receiver hiding evidence of upstream routers that do not understand the QuickStart option.

AUTHORS' ADDRESSES

Amit Jain
Array Networks
Email : ajain@arraynetworks.net

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