Tcpm Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 14, 2015

J. You
R. Huang
Huawei
R. Barik
University of Oslo
D. M. Divakaran
I2R, A*STAR

November 10, 2014

Configuring TCP's Initial Window
draft-you-tcpm-configuring-tcp-initial-window-03

Abstract

This document discusses that TCP's initial congestion window is not a constant in different use cases. It proposes a flexible method to configure the initial window in order to keep up with the current network state.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 14, 2015.

Copyright Notice

Table of Contents

1. Introduction

Google proposes to increase TCP's initial congestion window (InitCwnd) to at least ten segments (about 15KB) [RFC6928]. While for Taobao, the biggest online shopping mall in China, some public material from Taobao discloses that Taobao is using IW7 in their network instead of IW10. When the TCP's InitCwnd is set to 7, they get the best end-user experience in Taobao's experiments. In Google's experiments, the InitCwnd is configured using the InitCwnd option in the IP route command. Furthermore, all front-end servers within a data center are configured with the same InitCwnd. However, as the network properties at geographically diverse locations differ, one global InitCwnd for all servers cannot optimize the TCP performance.

This document discusses that TCP's initial congestion window is not a constant in different use cases. It proposes a flexible method to configure the initial window in order to keep up with the current network state.

2. Terminology

This section contains definitions of terms used in this document.

TCP: Transmission Control Protocol

RTT: Round-Trip Time

RTO: Retransmission Timeout

3. Current TCP's Initial Window Configuration Methods

The performance of initial TCP connection and congestion control is often affected by TCP parameters, such as initial congestion window, slow start threshold etc., which are usually default in systems. While with the global network access speeds growing, these default values set by systems may not be suitable for all the usages in current networks.

For example, Google believes a modest increase of InitCwnd to 10 is the best solution for the near-term deployment. Google's experiments consist of enabling a larger initial congestion window on front-end servers in several data centers at geographically diverse locations. In their experiments, the front-end servers run Linux with a reasonably standards compliant TCP implementation (the congestion control algorithm used is TCP CUBIC), and the initial congestion window is configured using the initcwnd option in the ip route command, i.e., InitCwnd=10.

Another example is Taobao, the biggest online shopping mall in China. Some public material from Taobao discloses that Taobao is using IW7 in their network instead of IW10. In Taobao's experiments, when the TCP's InitCwnd is set to 7, they get the best end-user experience.

As the application scenarios for Google and Taobao are definitely different, the InitCwnd values for optimal performance are different too. So using one fixed InitCwnd value (i.e. 10) for all cases is not appropriate.

Current TCP's InitCwnd is a global variable on a server or host, for example, a host is configured with the same initial congestion window for all the applications. Those parameters can be changed by modifying the regedit or kernel. However, they can't be modified dynamically, nor can be based on different granularities, such as per TCP flow. If they can be adjusted according to peak and off-peak times of Internet, server capability, network bandwidth, number of users, etc., maybe the performance of TCP connections could be effectively improved. For example, when there is no network

congestion or server overloading, TCP initial window size could be set bigger. While during the peak time of Internet, e.g. "Double-11" shopping festival of Taobao, the window size could be set smaller in order to avoid unnecessary congestion.

4. Factors affecting TCP's Initial Window

This section discusses some factors that need to be considered when determining an appropriate TCP initial congestion window. Regarding how to find the proper InitCwnd, it is TBD.

4.1. Web Object Size

[RFC3390] stated that the main motivation for increasing the initial window to 4 KB was to speed up connections that only transmit a small amount of data, e.g., email and web. The majority of transfers back then were less than 4 KB and could be completed in a single RTT (Round-Trip Time).

However, nowadays, the size of the average web page of the top 1000 websites passed 1600K for the first time in July 2014. At the same time the number of objects in the average web page increased to 112 objects. Google proposed to increase TCP's initial congestion window to at least ten segments (about 15KB) [RFC6928], while Taobao thinks 7 segments is optimal.

4.2. Bandwidth

For low bandwidth networks, such as GSM (Global System for Mobile Communication) and GPRS (General Packet Radio Service), injecting high-speed data into low-speed links leads to congestion or even collapse. In such case, small initial congestion window for TCP connections is relatively safe, e.g. 1 to 4, to prevent network congestion caused by a sudden influx of data into network. However, for networks with sufficient bandwidth capacity, the value of TCP initial congestion window could be set bigger, but we also need to consider other factors such as latency, packet loss, etc.

4.3. Latency

In high latency networks, the duration of slow start stage has a big impact on the whole TCP performance. A small initial congestion window usually leads to a long slow start stage, which may seriously decrease the TCP performance. Especially for short-lived services, e.g., HTTP Web transaction, most data would be transmitted at the low speed rate if the slow start stage is too long. For such kind of application, increasing InitCwnd enables transmission to be finished

in fewer RTTs. This would shorten the duration of slow start stage and avoid RTO (Retransmission Timeout).

Our experiments show the relations between the initial congestion window (horizontal axis) and transmission time when sending 50k data (vertical axis) in a lab simulation environment. We compare the results using different initial congestion windows (from 1 to 10) under different latency (50ms, 100ms, 200ms, 300ms) when sending 50k data. As we can see, when the initial congestion window is bigger, the duration is smaller.

4.4. Packet Loss Rate

Packet loss is usually caused by network congestion due to insufficient bandwidth discussed in Section 4.1.3, or network device problems, e.g. not enough storage in routers. Increasing InitCwnd would lead to data stream burst into networks then would aggravate the packet loss. So in high congestion environment, TCP initial congestion window should be set relatively small, e.g. 2 or 4.

Our experiments show the relations between the initial congestion window (horizontal axis) and transmission time when sending 50k data with different packet loss rate (vertical axis) in a lab simulation environment. We compare the results using different initial congestion windows (from 1 to 10) under packet loss rate (0.10%, 0.20%, 0.40%, 0.60%, 0.80%) when sending 50k data with fixed latency=50ms. As we can see, when the initial congestion window is bigger, the duration is smaller.

4.5. Concurrent TCP connections

For applications with too many concurrent TCP connections, it is not suggested to set too large initial congestion window since too many network resources would be occupied in a very short time. It's against the TCP fairness and also easily results in network congestion.

5. Configuring TCP's Initial Window

This document proposes that TCP's initial window could be automatically configured based on the flow size whenever this size is known, as shown in Figure 1.

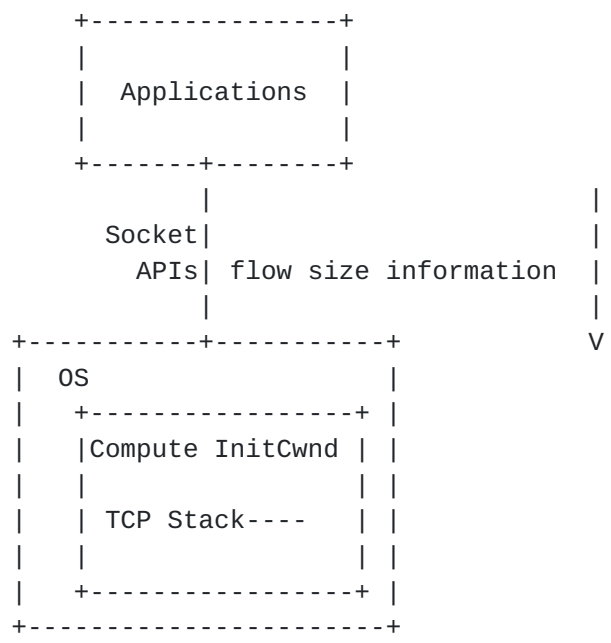

Figure 1: TCP's Initial Window Configuration

Intuitively, a function that determines InitCwnd of each flow based on its size can lead to better performance of small flows. In [WWIC][LCN][NoF] , this has been shown to be a benefit, irrespective of network conditions. The key idea is, the larger the flow-size, the smaller the InitCwnd. A lower bound for such an InitCwnd function can be the current standard for InitCwnd.

The results in [LCN] were derived using the following weighted function, where, for a flow of size s packets,

```
if s <= theta then
  InitCwnd = MaxIW
else
  InitCwnd = theta/s * MaxIW + (1-theta/s) * MinIW
end if
```

where theta is the flow size threshold used to distinguish between large flows and the rest, MinIW is the lower bound (four segments [RFC3390]), and MaxIW is the maximum InitCwnd that any connection can have (e.g. 10 [RFC6928]). The parameters, theta, MinIW and MaxIW are in number of TCP segments. Observe that, while small flows, defined by the threshold theta, will have InitCwnd as large as MaxIW, flows with size greater than theta will have InitCwnd closer to MinIW with increasing size.

This proposal assumes that flows will be able to know their sizes before the transfer begins. While this is true for many applications, for example, an HTTP query, a file transfer etc., there are also applications for which the flow-sizes can not be known in advance, for example, a streaming video. No changes are proposed for such flows. From our experiments, we can see that small flows benefit more from larger IW-size than large flows, while IW-size almost not affecting the performance of large flows.

If an application tries to cheat the system by splitting a large flow into small ones, then it would have to weigh the benefits of being able to use a larger IW against the cost of the handshakes for closing and re-opening a connection, at least. Depending on the IW calculation function above, there's an upper bound to protect the system even if such cheating might work.

6. IANA Considerations

This document does not introduce any new IANA considerations.

7. Security considerations

This document introduces a new method to configure the initial congestion window for TCP connections. This method facilitates application developers to tune TCP for their benefits. But it also has the possibility that packet loss may caused by inappropriate setting. However, as RFC6928 says, it is unlikely to lead to a persistent state of network congestion or collapse. So it does not introduce any new security issues.

8. Acknowledgement

The authors would like to thank Yoshifumi Nishida, Wesley Eddy and Michael Welzl for their detailed review and comments.

9. Normative References

[LCN] Barik, R. and Divakaran, D.M., "Evolution of TCP's initial window size", IEEE LCN 2013, Oct 2013.

[NoF] Barik, R. and Divakaran, D.M., "Development and Experimentation of TCP Initial Window Function", NoF 2014, Dec 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

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

[RFC6928] Chu, J., Dukkipati, N., Cheng, Y., and M. Mathis, "Increasing TCP's Initial Window", RFC 6928, April 2013.

[WWIC] Barik, R. and Divakaran, D.M., "TCP Initial Window: A Study", WWIC 2012, Jun 2012.

Authors' Addresses

Jianjie You
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, 210012
China

Email: youjianjie@huawei.com

Rachel Huang
Huawei
101 Software Avenue, Yuhuatai District
Nanjing, 210012
China

Email: rachel.huang@huawei.com

Runa Barik
University of Oslo
PO Box 1080 Blindern
Oslo N-0316
Norway

Email: runabk@ifi.uio.no

Dinil Mon Divakaran
I2R, A*STAR
1 Fusionopolis Way
#16-16 Connexix North Tower
Singapore 138632

Email: divakarand@i2r.a-star.edu.sg