Network Working Group
Internet-Draft
Intended status: Informational
Expires: September 5, 2012

R. Jesup
Mozilla
H. Alvestrand
Google

March 4, 2012

Congestion Control Requirements For Real Time Media
draft-jesup-rtp-congestion-reqs-00

Abstract

Congestion control is needed for all data transported across the Internet, in order to promote fair usage and prevent congestion collapse. The requirements for interactive, point-to-point real time multimedia, which needs by low-delay, semi-reliable data delivery, are different from the requirements for bulk transfer like FTP or bursty transfers like Web pages, and the TCP algorithms are not suitable for this traffic.

This document attempts to describe a set of requirements that can be used to evaluate other congestion control mechanisms in order to figure out their fitness for this purpose.

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 RFC 2119 [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 September 5, 2012.

Copyright Notice

Table of Contents

1. Introduction

The traditional TCP congestion control requirements were developed in order to promote efficient use of the Internet for reliable bulk transfer of non-time-critical data, such as transfer of large files. They have also been used successfully to govern the reliable transfer of smaller chunks of data in "as fast as possible" mode, such as when fetching Web pages.

These algorithms have also been used for transfer of media streams that are viewed in a non-interactive manner, such as "streaming" video, where having the data ready when the viewer wants it is important, but the exact timing of the delivery is not.

When doing real time interactive media, the requirements are different; one needs to provide the data continuously, within a very limited time window (no more than 100s of milliseconds end-to-end delay), the sources of data may be able to adapt the amount of data that needs sending within fairly wide margins, and may tolerate some amount of packet loss, but since the data is generated in real time, sending "future" data is impossible, and since it's consumed in real time, data delivered late is useless.

One particular protocol portofolio being developed for this use case is WebRTC [I-D.ietf-rtcweb-overview], where one envisions sending multiple RTP-based flows between two peers, in conjunction with data flows, all at the same time, without having special arrangements with the intervening service providers.

Given that this use case is the focus of this document, use cases involving noninteractive media such as YouTube-like video streaming, and use cases using multicast/broadcast-type technologies, are out of scope.

The terminology defined in [I-D.ietf-rtcweb-overview] is used in this memo.

2. Requirements

1. The congestion control algorithm must attempt to provide low-delay transit for real-time traffic, even when faced with intermediate bottlenecks and competing flows.

    A. It should also deal well with routing changes and interface changes (WiFi to 3G data, etc) which may radically change the bandwidth available.

2. The algorithm must be fair to other flows, both realtime flows (such as other instances of itself), and TCP flows, both long-lived and bursts such as the traffic generated by a typical web browsing session. Note that 'fair' is a rather hard-to-define term.

   A. The algorithm must not overreact to short-term bursts (such as web-browsing) which can quickly saturate a local-bottleneck router or link, but also clear quickly, and should recover quickly when the burst ends.

3. The algorithm should merge information across multiple RTP streams between the same endpoints, whether or not they're multiplexed on the same ports, in order to allow congestion control of the set of streams together instead of as multiple independent streams. This allows better overall bandwidth management, faster response to changing conditions, and fairer sharing of bandwidth with other network users.

   A. If possible, it should also share information and adaptation with other non-RTP flows between the same endpoints, such as a WebRTC data channel

4. The algorithm should not require any special support from network elements (ECN, etc). As much as possible, it should leverage existing information about the incoming flows to provide feedback to the sender. Examples of this information are the packet arrival times, packet timestamps, packet sizes, packet losses. Extra information could be added to the packets to provide more detailed information on actual send times (as opposed to sampling times), but should not be required.

   A. When signals such as ECN are available, it is good if they can be utilized.

5. Since the assumption here is a set of RTP streams, the backchannel typically should be done via RTCP; the alternative would be to include it in a reverse RTP channel using header extensions.

   A. In order to react sufficiently quickly, the AVPF/SAVPF RTP profile[RFC4585] must be used

   B. Note that in some cases, backchannel messages may be delayed until the RTCP channel can be allocated enough bandwidth, even under AVPF rules. This may also imply allowing a higher maximum percentage for RTCP data.

C. Note that RTCP is of course unreliable

D. Bandwidth for the feedback messages should be minimized (such as via RFC 5506 [RFC5506]to allow RTCP without SR/RR)

E. Header extensions would avoid the RTCP timing rules issues, and allow the application to allocate bandwidth as needed for the congestion algorithm.

F. Backchannel data should be minimized to avoid taking too much reverse-channel bandwidth (since this will often be used in a bidirectional set of flows). In areas of stability, backchannel data may be sent more infrequently so long as algorithm stability and fairness are maintained. When the channel is unstable or has not yet reached equilibrium after a change, backchannel feedback may be more frequent and use more reverse-channel bandwidth.

6. It should attempt to avoid bandwidth 'collapse' when facing a long-lived saturating TCP flow or flows. (I.e. a classic delay-sensitive algorithm will reduce bandwidth to keep delay down until the TCP flow has all the bandwidth). See the Cx-TCP algorithm discussed in a recent Transactions On Networking [cx-tcp] for an example of a delay-sensitive congestion-control algorithm that transitions to a loss-based mode when competing with TCP flows - at the cost of increased delay.

7. The algorithm should be stable and low-delay when faced with active queue management (AQM) in the channel.

8. The algorithm should quickly adapt to initial network conditions at the start of a flow; the adaptation may be faster than adaptation later in a flow. This should occur both if the initial bandwidth is above or below the bottleneck bandwidth.

A. it should allow for both slow-start operation (adapt up) and history-based startup (start at a point expected to be at or below channel bandwidth from historical information, which may need to adapt down quickly if the initial guess is wrong). Starting too low and/or adapting up too slowly can cause a critical point in a personal communication to be poor ("Hello!").

9. Where possible, the algorithm should leverage and piggyback on other RTCP communications, such as SR/RR, rctp-fb PLI, RPSI, SLI or application-specific NACK messages (such as for loss information).

10. It should be evaluated in how it works both with backbone-router bottlenecks, (asymmetric) local-loop bottlenecks, and local-lan (WiFi/etc) bottlenecks.

11. The algorithm should sense the unexpected lack of backchannel information as a possible indication of a channel overuse problem and react accordingly to avoid burst events causing a congestion collapse.

12. It should be stable if the RTP streams are halted or discontinuous (VAD/DTX); after a resumption of RTP data it may adapt more quickly (similar to the start of a flow).

3. IANA Considerations

This document makes no request of IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

4. Security Considerations

An attacker with the ability to delete, delay or insert messages in the flow can fake congestion signals, unless they are passed on a tamper-proof path. Since some possible algorithms depend on the timing of packet arrival, even a traditional protected channel does not fully mitigate such attacks.

An attack that reduces bandwidth is not necessarily significant, since an on-path attacker could break the connection by discarding all packets. Attacks that increase the percieved available bandwidth are concievable, and need to be evaluated.

Algorithm designers SHOULD consider the possibility of malicious on-path attackers.

5. Acknowledgements

This document is the result of discussions in various fora of the WebRTC effort, in particular on the rtp-congestion@alvestrand.no mailing list. Many people contributed their thoughts to this.

6. References

6.1. Normative References

[I-D.ietf-rtcweb-overview]
Alvestrand, H., "Overview: Real Time Protocols for Brower-based Applications", draft-ietf-rtcweb-overview-00 (work in progress), June 2011.

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

[RFC4585] Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey, "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585, July 2006.

6.2. Informative References

[RFC5506] Johansson, I. and M. Westerlund, "Support for Reduced-Size Real-Time Transport Control Protocol (RTCP): Opportunities and Consequences", RFC 5506, April 2009.

[cx-tcp] Budzisz, L., Stanojevic, R., Schlote, A., Baker, F., and R. Shorten, "On the Fair Coexistence of Loss- and Delay-Based TCP", December 2011.

Authors' Addresses

Randell Jesup
Mozilla
USA

Email: randell-ietf@jesup.org

Harald Alvestrand
Google
Kungsbron 2
Stockholm 11122
Sweden

Email: harald@alvestrand.no