Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 6, 2016

M. Zanaty
Cisco
V. Singh
Aalto University
S. Nandakumar
Cisco
Z. Sarkar
Ericsson AB
July 05, 2015

Congestion Control and Codec interactions in RTP Applications
draft-zanaty-rmcat-cc-codec-interactions-00

Abstract

Interactive real-time media applications that use the Real-time Transport Protocol (RTP) over the User Datagram Protocol (UDP) must use congestion control techniques above the UDP layer since it provides none. This memo describes the interactions and conceptual interfaces necessary between the application components that relate to congestion control, specifically the media codec control layer, and the components dedicated to congestion control functions.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

Interactive real-time media applications most commonly use RTP [RFC3550] over UDP [RFC0768]. Since UDP provides no form of congestion control, which is essential for any application deployed on the Internet, these RTP applications have historically implemented one of the following options at the application layer to address their congestion control requirements.

o For media with relatively low packet rates and bit rates, such as many speech codecs, some applications use a simple form of congestion control that stops transmission permanently or temporarily after observing significant packet loss over a significant period of time, similar to the RTP circuit breakers [I-D.ietf-avtcore-rtp-circuit-breakers].

- Some applications have no explicit congestion control, despite the clear requirements in RTP and its profiles AVP [RFC3551] and AVPF [RFC4585], under the expectation that users will terminate media flows that are significantly impaired by congestion (in essence, human circuit breakers).

- For media with substantially higher packet rates and bit rates, such as many video codecs, various non-standard congestion control techniques are often used to adapt transmission rate based on receiver feedback.

- Some experimental applications use standardized techniques such as TCP-Friendly Rate Control (TFRC) [RFC5348]. However, for various reasons, these have not been widely deployed.

The RTP Media Congestion Avoidance Techniques (RMCAT) working group was chartered to standardize appropriate and effective congestion control for RTP applications. It is expected such applications will migrate from the above historical solutions to the RMCAT solution(s).

The RMCAT requirements [I-D.ietf-rmcat-cc-requirements] include low delay, reasonably high throughput, fast reaction to capacity changes including routing or interface changes, stability without over-reaction or oscillation, fair bandwidth sharing with other instances of itself and TCP flows, sharing information across multiple flows when possible [I-D.welzl-rmcat-coupled-cc], and performing as well or better in networks which support Active Queue Management (AQM), Explicit Congestion Notification (ECN), or Differentiated Services Code Points (DSCP).

In order to meet these requirements, interactions are necessary between the application's congestion controller, the RTP layer, media codecs, other components, and the underlying UDP/IP network stack. This memo attempts to present a conceptual model of the various interfaces based on a simplified application decomposition. This memo discusses interactions between the congestion control and codec control layer in a typical RTP Application.

Note that RTP can also operate over other transports with integrated congestion control such as TCP [RFC5681] and DCCP [RFC4340], but that is beyond the scope of RMCAT and this memo.

2. Key Words for Requirements

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].

3. Conceptual Model

It is useful to decompose an RTP application into several components to facilitate understanding and discussion of where congestion control functions operate, and how they interface with the other components. The conceptual model in Figure 1 consists of the following components.

```
+-----------------------------+
|       +-----Config-----+    |
|       |       |        |    |
|       |     Codec      |    |
|       |     | | |      |    |
| APP   +---RTP | RTCP---+    |
|       |     | | |           |
|       |     | | |           |
|       +---Congestion-------|---Shared
|            Control         |   State
+-----------------------------+
|
+-----------------------------+
| Network      UDP           |
| Stack         |            |
|              IP            |
+-----------------------------+
```

Figure 1

o APP: Application containing one or more RTP streams and the corresponding media codecs and congestion controllers. For example, a WebRTC browser.

o Config: Configuration specified by the application that provides the media and transport parameters, RTP and RTCP parameters and extensions, and congestion control parameters. For example, a WebRTC Javascript application may use the 'constraints' API to affect the media configuration, and SDP applications may negotiate the media and transport parameters with the remote peer. This determines the initial static configuration negotiated in session establishment. The dynamic state may differ due to congestion or other factors, but still must conform to limits established in the config.

o Codec: Media encoder/decoder or other source/sink for the RTP payload. The codec may be, for example, a simple monaural audio format, a complex scalable video codec with several dependent

layers, or a source/sink with no live encoding/decoding such as a mixer which selectively switches and forwards streams rather than mixes media.

- RTP: Standard RTP stack functions, including media packetization / de-packetization and header processing, but excluding existing extensions and possible new extensions specific to congestion control (CC) such as absolute timestamps or relative transmission time offsets in RTP header extensions. RTCP: Standard RTCP functions, including sender reports, receiver reports, extended reports, circuit breakers [I-D.ietf-avtcore-rtp-circuit-breakers], feedback messages such as NACK [RFC4585] and codec control messages such as TMMBR [RFC5104], but excluding existing extensions and possible new extensions specific to congestion control (CC) such as REMB [I-D.alvestrand-rmcat-remb] (for receiver-side CC), ACK (for sender-side CC), absolute and/or relative timestamps (for sender-side or receiver-side CC), etc.

- Congestion Control: All functions directly responsible for congestion control, including possible new RTP/RTCP extensions, send rate computation (for sender-side CC), receive rate computation (for receiver-side CC), other statistics, and control of the UDP sockets including packet scheduling for traffic shaping/pacing.

- Shared State: Storage and exchange of congestion control state for multiple flows within the application and beyond it.

- Network Stack: The platform's underlying network functions, usually part of the Operating System (OS), containing the UDP socket interface and other network functions such as ECN, DSCP, physical interface events, interface-level traffic shaping and packet scheduling, etc. This is usually part of the Operating System, often within the kernel; however, user-space network stacks and components are also possible.

4. Implementation Model

There are advantages and drawbacks to implementing congestion control in the application layer. It avoids platform dependencies and allows for rapid experimentation, evolution and optimization for each application. However, it also puts the burden on all applications, which raises the risks of improper or divergent implementations. One motivation of this memo is to mitigate such risks by giving proper guidance on how the application components relating to congestion control should interact.

Another drawback of congestion control in the application layer is that any decomposition, including the one presented in Figure 1, is purely conceptual and illustrative, since implementations have differing designs and decompositions. Conversely, this can be viewed as an advantage to distribute congestion control functions wherever expedient without rigid interfaces. For example, they may be distributed within the RTP/RTCP stack itself, so the separate components in Figure 1 are combined into a single RTP+RTCP+CC component as shown in Figure 2.

```
+-----------------------------+
|       +-----Config          |
|       |       |             |
|       |     Codec           |
| APP   |       |             |
|       +---RTP+RTCP+CC------|---Shared
+-----------------------------+   State
                |
+-----------------------------+
| Network      UDP            |
| Stack         |             |
|              IP             |
+-----------------------------+
```

Figure 2

5. Codec - CC Interactions

The following subsections identify the necessary interactions between the Codec and congestion control layer interfaces that needs to be considered important

5.1. Allowed Rate

Allowed Rate (from CC to Codec): The max transmit rate allowed over the next time interval. The time interval may be specified or may use a default. The rate may be specified in bytes or packets or both. The rate must never exceed permanent limits established in session signaling such as the SDP bandwidth attribute [RFC4566] nor temporary limits in RTCP such as TMMBR [RFC5104] or REMB [I-D.alvestrand-rmcat-remb]. This is the most important interface among all components, and is always required in any RMCAT solution. In the simplest possible solution, it may be the only CC interface required.

5.2. Media Elasticity

Media Elasticity (from Codec to CC): Many live media encoders are highly elastic, often able to achieve any target bit rate within a wide range, by adapting the media quality. For example, a video encoder may support any bit rate within a range of a few tens or hundreds of kbps up to several Mbps, with rate changes registering as fast as the next video frame, although there may be limitations in the frequency of changes. Other encoders may be less elastic, supporting a narrower rate range, coarser granularity of rate steps, slower reaction to rate changes, etc. Other media, particularly some audio codecs, may be fully inelastic with a single fixed rate. CC can beneficially use codec elasticity, if provided, to plan Allowed Rate changes, especially when there are multiple flows sharing CC state and bandwidth.

5.3. Startup Ramp

Startup Ramp (from Codec to CC, and from CC to Codec): Startup is an important moment in a conversation. Rapid rate adaptation during startup is therefore important. The codec should minimize its startup media rate as much as possible without adversely impacting the user experience, and support a strategy for rapid rate ramp. The CC should allow the highest startup media rate as possible without adversely impacting network conditions, and also support rapid rate ramp until stabilizing on the available bandwidth. Startup can be viewed as a negotiation between the codec and the CC. The codec requests a startup rate and ramp, and the CC responds with the allowable parameters which may be lower/slower. The RMCAT requirements also include the possibility of bandwidth history to further accelerate or even eliminate startup ramp time. While this is highly desirable from an application viewpoint, it may be less acceptable to network operators, since it is in essence a gamble on current congestion state matching historical state, with the potential for significant congestion contribution if the gamble was wrong. Note that startup can often commence before user interaction or conversation to reduce the chance of clipped media.

5.4. Delay Tolerance

Delay Tolerance (from Codec to CC): An ideal CC will always minimize delay and target zero. However, real solutions often need a real non-zero delay tolerance. The codec should provide an absolute delay tolerance, perhaps expressed as an impairment factor to mix with other metrics.

5.5. Loss Tolerance

Loss Tolerance (from Codec to CC): An ideal CC will always minimize packet loss and target zero. However, real solutions often need a real non-zero loss tolerance. The codec should provide an absolute loss tolerance, perhaps expressed as an impairment factor to mix with other metrics. Note this is unrecoverable post-repair loss after retransmission or forward error correction.

5.6. Forward Error Correction

Forward Error Correction (FEC): Simple FEC schemes like XOR Parity codes [RFC5109] may not handle consecutive or burst loss well. More complex FEC schemes like Reed-Solomon [RFC6865] or Raptor [RFC6330] codes are more effective at handling bursty loss. The sensitivity to packet loss therefore depends on the media (source) encoding as well as the FEC (channel) encoding, and this sensitivity may differ for different loss patterns like random, periodic, or consecutive loss. Expressing this sensitivity to the congestion controller may help it choose the right balance between optimizing for throughput versus low loss.

5.7. Probing for Available Bandwidth

FEC can also be used to probe for additional available bandwidth, if the application desires a higher target rate than the current rate. FEC is preferable to synthetic probes since any contribution to congestion by the FEC probe will not impact the post-repair loss rate of the source media flow while synthetic probes may adversely affect the loss rate. Note that any use of FEC or retransmission must ensure that the total flow of all packets including FEC, retransmission and original media never exceeds the Allowed Rate.

5.8. Throughput Sensitivity

Throughput Sensitivity (from Codec to CC): An ideal CC will always maximize throughput. However, real solutions often need a trade-off between throughput and other metrics such as delay or loss. The codec should provide throughput sensitivity, perhaps expressed as an impairment factor (for low throughputs) to mix with other metrics.

5.9. Rate Stability

Rate Stability (from Codec to CC): The CC algorithm must strike a balance between rate stability and fast reaction to changes in available bandwidth. The codec should provide its preference for rate stability versus fast and frequent reaction to rate changes,

perhaps expressed as an impairment factor (for high rate variance over short timescales) to mix with other metrics.

6. Acknowledgements

The RMCAT design team discussions contributed to this memo.

7. IANA Considerations

This memo includes no request to IANA.

8. References

8.1. Normative References

[I-D.alvestrand-rmcat-remb]
Alvestrand, H., "RTCP message for Receiver Estimated Maximum Bitrate", draft-alvestrand-rmcat-remb-03 (work in progress), October 2013.

[I-D.ietf-avtcore-rtp-circuit-breakers]
Perkins, C. and V. Singh, "Multimedia Congestion Control: Circuit Breakers for Unicast RTP Sessions", draft-ietf-avtcore-rtp-circuit-breakers-10 (work in progress), March 2015.

[I-D.ietf-mmusic-sdp-bundle-negotiation]
Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-negotiation-22 (work in progress), June 2015.

[I-D.ietf-rmcat-cc-requirements]
Jesup, R. and Z. Sarker, "Congestion Control Requirements for Interactive Real-Time Media", draft-ietf-rmcat-cc-requirements-09 (work in progress), December 2014.

[I-D.welzl-rmcat-coupled-cc]
Welzl, M., Islam, S., and S. Gjessing, "Coupled congestion control for RTP media", draft-welzl-rmcat-coupled-cc-05 (work in progress), June 2015.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

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

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3551] Schulzrinne, H. and S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", STD 65, RFC 3551, July 2003.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, March 2006.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[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.

[RFC5104] Wenger, S., Chandra, U., Westerlund, M., and B. Burman, "Codec Control Messages in the RTP Audio-Visual Profile with Feedback (AVPF)", RFC 5104, February 2008.

[RFC5109] Li, A., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, December 2007.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, September 2008.

[RFC5450] Singer, D. and H. Desineni, "Transmission Time Offsets in RTP Streams", RFC 5450, March 2009.

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

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.

[RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, April 2010.

[RFC6330] Luby, M., Shokrollahi, A., Watson, M., Stockhammer, T., and L. Minder, "RaptorQ Forward Error Correction Scheme for Object Delivery", RFC 6330, August 2011.

[RFC6865] Roca, V., Cunche, M., Lacan, J., Bouabdallah, A., and K. Matsuzono, "Simple Reed-Solomon Forward Error Correction (FEC) Scheme for FECFRAME", RFC 6865, February 2013.

8.2. Informative References

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

[RFC3552] Rescorla, E. and B. Korver, "Guidelines for Writing RFC Text on Security Considerations", BCP 72, RFC 3552, July 2003.

Authors' Addresses

Mo Zanaty
Cisco

Email: mzanaty@cisco.com

Varun Singh
Aalto University

Email: varun@comnet.tkk.fi

Suhas Nandakumar
Cisco

Email: snandaku@cisco.com

Zaheduzzaman Sarker
Ericsson AB

Email: zaheduzzaman.sarker@ericsson.com