RTP AND THE DATAGRAM CONGESTION CONTROL PROTOCOL

Colin Perkins

Department of Computing Science
University of Glasgow

Ladan Gharai

Information Sciences Institute
University of Southern California

ABSTRACT

We describe how the new Datagram Congestion Control Protocol (DCCP) can be used as a bearer for the Real-time Transport Protocol (RTP) to provide a congestion controlled basis for networked multimedia applications. This is a step towards deployment of congestion control for such applications, necessary to ensure the future stability of the best-effort network if high-bandwidth streaming and IPTV services are to be deployed outside of closed QoS-managed networks.

1. INTRODUCTION

The Real-time Transport Protocol (RTP) [1] is widely used in video streaming, telephony, and other real-time networked applications. RTP can be run over a range of lower-layer transport protocols, and the performance of applications that use RTP is heavily influenced by the choice of lower-layer transport protocol [2]. The Datagram Congestion Control Protocol (DCCP) [3] is a newly specified transport protocol which provides desirable properties for real-time applications running on unmanaged best-effort IP networks. This paper describes how RTP can be framed over DCCP, and discusses issues inherent in running RTP over a congestion controlled transport. The contribution of this work is to describe some previously unknown issues with running RTP over congestion controlled transport protocols, and to outline some possible solutions to these problems.

Our paper is structured as follows: after reviewing the background to our work in Section 2, we present a proposal for mapping RTP onto DCCP in Section 3, and discuss issues in designing such a mapping in Section 4. We describe related work in Section 5, and conclude in Section 6.

2. BACKGROUND

RTP has been widely adopted as a standard basis for video conferencing, streaming, and other real time applications. It comprises a data transfer protocol that provides sequencing, timing, and payload and source identification, with a control protocol providing reception quality and presence data.

With the widespread adoption of RTP running over UDP have come concerns that many real time applications do not implement congestion control, leading to potential problems with the stability of the network. Accordingly, there is a push in the IETF to adopt congestion control for RTP applications. Such congestion control may be implemented as part of the RTP layer, or as part of a lower layer transport over which RTP is run. DCCP forms one such lower layer transport.

3. RUNNING RTP OVER DCCP

In the following we outline an approach to running RTP over DCCP which is presently under consideration in the Internet Engineering Task Force (see [4] for full details). We discuss how RTP data and control packets may be framed for transport over DCCP, multiplexing of data and control channels, and the differences between the RTP session model and the DCCP connection model.

3.1. Framing RTP Data Packets

DCCP provides a datagram service, similar to the datagram service provided by UDP, but with the addition of congestion control. Accordingly, each RTP packet can be encapsulated into a single DCCP datagram in exactly the same way is done for RTP over UDP: RTP header processing is not affected by DCCP framing, and fields in the RTP header are interpreted according to the RTP specification and any applicable RTP profile and payload format. There are, however, three major differences between DCCP and UDP which affect RTP: the connection oriented nature of the protocol, the congestion control algorithm, and partial checksums.

The connection oriented nature of DCCP is generally of benefit to RTP, but does cause some difficulties. A DCCP connection will be opened on joining an RTP session, and remains open for the duration of the session. The difficulty comes from knowing when the session ends: as we discuss further in Section 4.3, an RTP session does not necessarily correspond to a transport layer connection, and the receipt of an RTCP BYE packet does not necessarily indicate that the DCCP connection should be closed; it may be necessary to rely on non-RTP signalling (e.g. via SIP [5]) to indicate the end of the session.

The benefits of the connection oriented nature of DCCP come with NAT traversal: RTP systems typically use silence suppression resulting in periods where no data need be sent. To stop NAT bindings from timing out, these systems must send occasional comfort noise or RTP no-op packets [6] during the silent periods. The introduction of a no-op packet – a data packet that must be discarded without playout – into RTP is less than ideal since it requires a special case to be added to the media playout process. DCCP gives a cleaner solution: there is an explicit signal to the NAT when closing the connection, and keep alives can be implemented by sending periodic zero length DCCP-Data packets[1] invisible to the RTP layer.

An RTP implementation must obey the dictates of DCCP congestion control. In some cases, this may force a sender to transmit at a rate below that which the codec would otherwise use. Applications should use either rate adaptive codecs, or a range of codecs with seamless codec switching, to allow them to switch to a lower rate format when necessary. This rate adaptation is discussed further in Section 4.1.

Similar to UDP-lite [7], DCCP allows an application to choose the checksum coverage, with partial checksums allowing receipt of packets with corrupt payloads. Some RTP payload formats [8] make use of this feature in conjunction with payload-specific mechanisms to improve performance when operating in environments with frequent non-congestive packet corruption. If such a payload format is used, partial checksums can be enabled at the DCCP layer, in which case the checksum must cover at least the DCCP and RTP headers.

3.2. Framing RTP Control Packets

The RTP Control Protocol (RTCP) provides reception quality feedback and presence information. RTCP should be used with DCCP, grouping RTCP packets into compound packets in the usual manner, with each compound RTCP packet being transported in a single DCCP datagram.

RTCP augments the transport level packet loss metrics provided by DCCP, but as noted in [3] there is potential overlap between information conveyed in RTCP reception reports and that in DCCP acknowledgement options. In general this is not an issue: RTCP packets contain media-specific data not present in DCCP, while DCCP options contain network-level data. There is no overlap between standard RTCP reports and standard DCCP options, but the RTCP Extended Reports [9] define an optional run-length encoded loss report that overlaps with the DCCP Ack Vector. This wastes capacity but is otherwise harmless, and can be avoided with no loss of functionality by disabling one or the other feature.

The usual RTCP timing rules apply when running over DCCP, subject to the constraint that RTCP packets must be subject to DCCP congestion control. This constraint raises a number of issues that are discussed further in Section 4.2.

[1]An interesting feature of DCCP is that it permits an application to send datagrams with no payload. These can be used to maintain NAT bindings and congestion state, in the absence of any application data flow.

3.3. Multiplexing Data and Control Channels

A frequent criticism of RTP relates to the number of ports it uses: each RTP session uses two UDP ports, one for data and one for control. In recent years this has become a problem for large telephony gateways which could otherwise support more than 32768 flows between pairs of gateways, but are limited by the number of UDP ports available. Using multiple ports per RTP session also complicates NAT traversal, since it requires an extra NAT binding to be maintained.

The obvious mapping of RTP onto DCCP creates two DCCP connections for each RTP flow: one connection for data packets, one for control packets. One can multiplex data and control on a single DCCP connection, however, if care is taken in the choice of RTP payload type numbers to avoid shadowing control packet types. Such a change is possible since RTP payload type numbers are dynamically assigned and have no pre-defined semantics. With this change, problems with NAT traversal and port exhaustion are eased, so providing an incentive to use RTP over DCCP.

3.4. RTP Sessions and DCCP Connections

A system cannot assume only a single RTP synchronisation source (SSRC) will be present because it is using a unicast DCCP connection. An RTP session can span a number of transport connections, and can include mixers or translators bringing other participants into the session. The use of congestion control at the transport layer introduces a number of difficulties in the design of RTP mixers and translators, these are discussed further in Section 4.3.

3.5. Signalling RTP over DCCP

A signalling protocol is required to initiate and control an RTP session. A common example of such signalling is the Session Initiation Protocol, which is used in conjunction with the Session Description Protocol (SDP) offer/answer model [10, 5, 11]. Figure 1 is an example SDP file showing how the combination of RTP and DCCP can be signalled. We briefly explain the features of this example that are unique to DCCP.

SDP uses a media (“m=”) line to convey details of the media format and transport protocol used. The media line denotes the type of media, transport protocol, and transport port to which the media is sent (the destination address is conveyed in the connection (“c=”) line). New transport protocol identifiers can be registered following the conventions used with other connection oriented media [12, 13], and we have introduced the identifier “DCCP” to indicate DCCP with an unspecified upper-layer protocol, and “DCCP/RTP/AVP”, “DCCP/RTP/SAVP”, “DCCP/RTP/AVPF” and “DCCP/RTP/SAVPF” to indicate DCCP with the common RTP profiles [14, 15, 16, 17].

In addition to the port number a DCCP connection has an associated service code. SDP does not directly support sig-

```
v=0
o=alice 1129377363 1 IN IP4 10.0.0.47
s=-
c=IN IP4 10.0.0.47
t=0 0
m=video 51372 DCCP/RTP/AVP 99
a=rtpmap:99 h261/90000
a=dccp-service-code:52545020
a=setup:passive
a=connection:new
```

Fig. 1. Example SDP offer, soliciting RTP connections using H.261 video to DCCP port 51372 on IP address 10.0.0.47.

nalling DCCP service codes, but can be readily extended by means of new attribute ("a=") lines. We have introduced the attribute "a=dccp-service-code:" to convey the numeric value of the DCCP service code, and recommend default service codes for RTP-based applications.

Finally, it is necessary to specify which of the end points should initiate DCCP connection establishment (i.e. send the initial DCCP-Request). An existing SDP extension [12] for connection oriented media can be reused for this purpose.

4. DISCUSSION

As noted in Section 3, there are a number of issues that arise when running RTP on a congestion controlled transport such as DCCP. These can be grouped into three areas: adaptive media coding, interactions between congestion control and RTCP, and between congestion control and translators.

4.1. Congestion Control and Media Coding

A well known issue with congestion control for audio/visual media is rate adaptation for the media stream. Media codecs typically have somewhat limited adaptability, both in terms of how much they can vary their sending rate, and in terms of how quickly they can adapt, and often cannot adapt to match the dictates of the congestion control algorithm.

Furthermore, when the codec is capable of adapting to match the congestion control algorithm, it is not always clear that such adaptation is desirable from a human factors viewpoint. For example, much research has shown that human perception prefers uniform poor quality media over varying, but objectively higher, quality.

We do not believe that DCCP makes these issues worse than other congestion controlled transport protocols. It does not, however, make them easier to solve.

4.2. Congestion Control and RTCP

The rate at which an RTP implementation transmits RTCP packets is chosen to scale with the number of participants in the session. The reporting interval between packets, T_R is calculated as in Equation 1, where T_{min} is typically 5 seconds, N is the number of participants in the session, s is the average RTCP packet size, and B is the session bandwidth. The reporting interval is then randomly dithered by $\pm 50\%$ to avoid timing synchronisation across participants. If a participant sends no RTP or RTCP packets for some small number of reporting intervals it is assumed to have left the session.

$$T_R = \max\left(T_{min}, \frac{Ns}{0.375B}\right) \tag{1}$$

The reporting interval, T_R, is calculated independently by each participant in the session, and it is assumed that all can accurately estimate the necessary parameters. Unfortunately, the use of DCCP makes the session bandwidth ill defined: the available bandwidth on each half of a DCCP connection may vary due to congestion control.

It is possible for each participant to use their local value of T_R when calculating the RTCP interval. In this case each participant may calculate a significantly different reporting interval if bandwidth is asymmetric. As a result, participants may be timed out prematurely, leading to partition of the session (this is particularly significant if there are many participants in the session, for example when translating from RTP over DCCP to RTP in a multicast environment, since the reporting interval may be larger in those cases, and variations more noticeable). In addition, frequent changes of the session bandwidth due to the dictates of the congestion control algorithm will lead to frequent reconsideration of the reporting interval, increasing the load on the system.

Alternatively, one may assume a constant T_R throughout the session, based on some nominal value for the media in use. This approach is simpler, but has the disadvantage that the RTCP traffic may grow to an excessive rate for a low bandwidth link, since the actual session bandwidth is neglected.

For an RTP session with two participants, an appropriate solution would be to adjust the reporting interval based on the actual session bandwidth, relying on the presence of the DCCP connection, augmented by zero length packets as a liveness indicator, to indicate that the other participant should not be timed out. The solution for multiparty sessions, including an RTP translator or mixer is subject to further research.

4.3. Congestion Control, Mixers and Translators

In addition to end systems, RTP supports active intermediate systems called mixers and translators. These systems allow a single RTP session to bridge multiple networks and transport connections, and provide media transcoding, mixing and adaptation functions.

The simplest translation scenario occurs when a single RTP session spans multiple DCCP connections (Figure 2). The RTP translator in this scenario must be aware of the congestion state of all three DCCP connections, and must adapt the media to the available capacity of each according to their

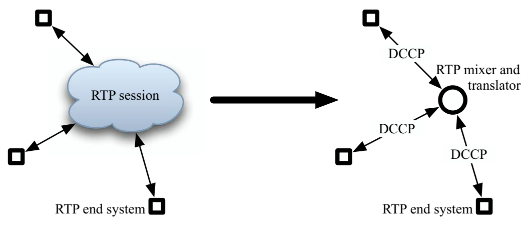

Fig. 2. RTP Mixers and Translators

congestion state. The introduction of DCCP complicates this scenario since each connection has an independent response to congestion, and each may use a different CCID and so adapt on different time-scales and with varying dynamics. As a result, it is difficult to determine a single sending rate which will suit all connections, forcing the translator to generate a new RTP stream for each connection. This is difficult since rate adaptation is generally done by the RTP translator transcoding the media, a computationally expensive process.

An RTP session may also span a DCCP connection and some other transport connection. Examples might include a translator between DCCP/IP and multicast UDP/IP, or a DCCP/IP connection that links two PSTN gateways. Issues that arise for such an RTP translator or mixer are similar to those when linking multiple DCCP connections, except that congestion control algorithms on either side of the translator may not be compatible, and indeed one side of the translator may not require or implement congestion control. The design of effective translators for such an environment is nontrivial.

We see that, compared to traditional RTP translators that produce output matching the smoothed lowest sending rate, a DCCP based system will be fairer to the network, but may not be feasible or economic to implement.

5. RELATED WORK

In addition to running RTP over a congestion controlled transport, it is possible to implement congestion control within the RTP layer. We have documented an approach to this [18], running TCP Friendly Rate Control within RTP, using RTCP to convey congestion feedback information. Such an approach differs significantly in detail from the work we describe here, but many of the issues discussed in Section 4 remain. In particular, the issues with congestion control and media coding, and with the RTCP interval. Mixers and translators also cause problem, but these are somewhat less severe, since there is more flexibility in the implementation of translators when the congestion control is done at the application (RTP) level.

6. CONCLUSIONS

We have described how RTP can be framed for transport over DCCP, and how such sessions can be signalled. Several issues arise as a result of the congestion controlled nature of DCCP: in particular, we discuss media coding, RTCP, and translators and mixers, and propose solutions to improve the performance of RTP running over DCCP. It is essential to the stability of the network that future RTP systems implement congestion control. We believe that, once the issues that we discuss in Section 4 are resolved, DCCP will provide an effective base for congestion controlled transport of RTP sessions.

7. ACKNOWLEDGEMENTS

This work was supported by a UK EPSRC GridNet2 award and by NSF grant 0334182. Thanks to Philippe Gentric, Magnus Westerlund and members of the IETF DCCP working group for comments on this work.

8. REFERENCES

[1] H. Schulzrinne et al, "RTP: A transport protocol for real-time applications," IETF, July 2003, RFC 3550.

[2] D. D. Clark and D. L. Tennenhouse, "Arch. consid. for new generation protocols," in *Proc. ACM SIGCOMM*, Sep. 1990.

[3] E. Kohler, M. Handley, and S. Floyd, "Datagram congestion control protocol," IETF, December 2005, Work in progess.

[4] C. S. Perkins, "RTP and the Datagram Congestion Control Protocol (DCCP)," IETF, November 2005, Work in progress.

[5] J. Rosenberg et al, "SIP: Session initiation protocol," IETF, June 2002, RFC 3261.

[6] F. Andreasen, D. Oran, and D. Wing, "A No-Op payload format for RTP," IETF, May 2005, Work in progress.

[7] L.-A. Larzon, M. Degermark, and S. Pink, "UDP-Lite for real-time m.media appl.," in *Proc. IEEE Intl. Conf. Comm.*, 1999.

[8] J. Sjoberg et al, "RTP payload format and file storage format for the AMR and AMR-WB audio codecs," IETF, June 2002, RFC 3267.

[9] T. Friedman, R. Caceres, and A. Clark, "RTP control protocol extended reports," IETF, November 2003, RFC 3611.

[10] M. Handley, V. Jacobson, and C. S. Perkins, "SDP: Session Description Protocol," IETF, July 2005, Work in progress.

[11] J. Rosenberg and H. Schulzrinne, "An offer/answer model with the session description protocol," IETF, June 2002, RFC 3264.

[12] D. Yon and G. Camarillo, "TCP-based media transport in sdp," IETF, September 2005, RFC 4145.

[13] J. Lazzaro, "Framing RTP and RTCP packets over connection oriented transport," IETF, September 2005, Work in progress.

[14] H. Schulzrinne and S. Casner, "RTP profile for audio and video conf. with minimal control," IETF, June 2003, RFC 3551.

[15] M. Baugher et al, "Secure rtp," IETF, March 2004, RFC 3711.

[16] J. Ott et al, "Extended RTP profile for RTCP-based feedback," IETF, August 2004, Work in progress.

[17] J. Ott and E. Carrara, "Extended secure RTP profile for RTCP-based feedback," IETF, November 2005, Work in progress.

[18] L. Gharai, "RTP profile for TCP friendly rate control," IETF, October 2005, Work in progress.