Content-Aware Selective Reliability for DCCP Video Streaming

Michael Schier
University of Innsbruck

Michael Welzl
University of Oslo

Abstract

The growing use of real-time video in wireless networks, where loss rates are high yet bandwidth and CPU power of receivers can be scarce, calls for new methods to maintain good quality when packets are dropped. These methods should ideally only involve the sender and avoid increasing the amount of data sent across the network. We introduce a scheme that satisfies these requirements by selectively retransmitting only the most important packets in case of loss based on the video content at macroblock granularity. In a Linux implementation using the DCCP protocol, we show that our mechanism outperforms content-unaware retransmission strategies such as earliest-first.

1. Introduction

The consumption of multimedia content, especially of videos, still experiences a considerable growth in popularity. Besides commercial IPTV services, social networking platforms like YouTube and DailyMotion, enabling their users to freely publish clips, are the main drivers of this hype. Video content is increasingly consumed on mobile devices, which brings along new challenges for the design of video streaming frameworks: In contrast to their wired counterparts, wireless environments suffer from significantly higher packet loss rates and delay fluctuations and are characterized by lower bandwidths. Unfortunately, the commonly used protocols TCP and UDP can not handle this problem in an optimal way. TCP increases transfer delay and is fully reliable, which is inappropriate for real time applications. UDP, which is unreliable and unaware of congestion, confronts application programmers with the burden of implementing both congestion control and partial reliability by themselves. The novel transport protocol DCCP fills this gap by offering built-in congestion control and facilitating the implementation of application-specific selective reliability.

Timely delivery of multimedia content is a critical factor for live video streaming systems as well as video on demand applications, whose users prefer uninterrupted over error-free playback suffering from occasional interruptions due to receiver buffer underflows. To provide good video quality, both timeliness and relevance of content should be considered during retransmissions. Our approach combines both aspects and is easy to deploy because no client support is needed.

The remainder of this paper is structured as follows: In Section 2, a survey of previous work is provided. Section 3 introduces our distortion estimation scheme used for prioritizing video content and Section 4 presents our approach for content-aware selective reliable video streaming over DCCP. The results of an extensive performance evaluation are briefly summarized in Section 5. Finally, Section 6 concludes.

2. Related Work

Some prior publications exist which examine error recovery and concealment mechanisms for real-time videos over DCCP. Most of them focus on networking aspects and do not take content analysis into consideration. As example, Nosheen et al. evaluate the video streaming performance of DCCP and SCTP over wireless networks and underline that they both achieve higher throughput than UDP [6]. When bandwidth gets scarce, DCCP performs better than SCTP and causes less packet loss. A semi-reliable multimedia streaming approach is proposed by Huszák et al., which aims at minimizing the playout buffer delay [3]. All retransmit decisions solely depend on the playback buffer, the round-trip time (RTT) and the allowed sending rate. Yuan-Cheng et al. designed a partial reliability extension to optimize the decodable frame ratio [10], but prioritize content only at the level of frames. Finally, McDonald et al. try to improve DCCP video conferencing by using a "Best Packet Next" strategy [5]. They also consider expiry time, but only categorize content into high priority audio and low priority video packets.

3. Macroblock Based Distortion Estimation

By efficiently detecting and reducing temporal and spatial redundancy, the vast majority of existing video

codecs achieve high compression ratios. They consider macroblocks as the central element of compression. In order to determine the importance of a single packet carrying video content, we estimate the perceptual quality distortion of the overall video stream caused by its loss. We use our macroblock based distortion estimation scheme discussed in [8], which considers the macroblock composition of single frames—their types as well as their temporal and spatial dependencies. Inter-frame dependencies are of special interest as they allow errors to propagate over several frames. Consequently, the loss of i-macroblocks and f-macroblocks has a high impact on quality as other macroblocks may depend on them. On the other hand, macroblocks of type b, bi, d and s are considered as less important because they can only act as dependency sinks.

In the remainder of this work, we refer to our scheme as $\Psi_{\text{MB}} : \Pi \rightarrow [0; 1]$ with Π being the set of transmitted video packets. $\Psi_{\text{MB}}(p_j)$ is calculated by doing a type weighting over all macroblocks of packet p. It is further refined by taking the frame's position within the current group of pictures into account and by implementing a scene cut detection mechanism. The latter is responsible for protecting frames having very low temporal similarities amongst themselves which is usually the case at positions close to scene cuts. For a detailed specification of $\Psi_{\text{MB}}(p)$, we refer to [8].

4. Sender-Driven Selective Reliability

DCCP, the Datagram Congestion Control Protocol, is a transport protocol which implements bidirectional, congestion controlled, unicast delivery of datagrams. As specified in [4], it is an unreliable transport protocol which does not support retransmissions. For transmitting time-critical video content, we use profile CCID3 (TCP-Friendly Rate Control) as congestion control mechanism because it provides a much smoother sending behavior and less delay fluctuations than profile CCID2 (TCP-Like Congestion Control).

We propose a selective retransmission scheme which only requires support from the sender. SACK-like feedback can be obtained from DCCP by using its ACK-vector option. ACK-vectors are run-length encoded, placed in packets' headers and can contain positive or negative feedback of up to 16192 packets (per option) preceding the ACKed packet. To reduce the number of wrong decisions caused by out-of-order packet arrivals, similarly to dupACKs in TCP, we consider NACKed packets as lost if they have a sequence number of less than the last feedback-packet's ACK number minus three. Another way of detecting packet loss is to use timeouts. According to [1], the receiver sends DCCP-ACK packets at least each RTT; on this basis, we chose $^{5RTT}/_{2}$ as the timeout interval which in experiments turned out to be a good compromise.

The only administrative messages which are exchanged in our scheme are *start-display*, *stop-display* and *position-change*. Usually, the receiver which wants to consume multimedia content establishes the connection. At some point, he decides that there is enough data in his playback buffer, starts the playback and notifies the sender. Analogously, when the consumer stops the video, the sender will get to know this. Finally, *position-change* informs the sender about a change of the current playback position. Each of those messages is used by the sender to deduce the receiver buffer's fill level, which is essential for calculating deadlines of packets. These are used to decide whether a packet stays in the queue, gets sent or is skipped.

The selective reliability mechanism establishes a mapping between the sender application's video packet numbers and the DCCP sequence numbers: When a packet leaves the sender system's network interface, the sequence number used can be obtained and mapped accordingly. Consequently, the ACK numbers from received ACK-vectors can be back-translated to the application's packet numbers. Non-ACKed packets stay in the set of pending packets until they get ACKed or, as previously explained, until they get dropped when the playback deadline can not be met. Such a mapping is necessary especially in conjunction with retransmissions because DCCP sequence numbers can not be reused and therefore some may get mapped onto the same packet number.

In the following, we describe the two phases of our scheme: the initial Buffer-Fill phase which is responsible for establishing the receiver buffer, and the Selective-Retransmit phase which aims at keeping the receiver buffer filled up to a certain threshold.

4.1. Buffer-Fill Phase

After connection establishment, the sender starts transmitting video content at the highest allowed sending rate and selects packets using the earliest-first strategy, further referred to as D_{EF}. During the whole Buffer-Fill phase, the transmission is fully reliable because there are no playback time constraints. The duration of this phase depends on several factors: the RTT, the average packet size and the packet loss ratio, which together determine the allowed sending rate, and the receiver application's playback buffer threshold, which specifies when to start the playback. We use a receiver which leaves the Buffer-Fill phase after having exceeded this threshold and when the receiving rate gets higher than

the average video bitrate of the received stream. Upon receipt of a *start-display* message, the sender leaves the Buffer-Fill phase and stores $t_S = t_{now} - {}^{RTT}/_2$ and $t_B = T(p_l)$, where $T(p)$ is the video timestamp of packet p and p_l is the most recently transmitted packet.

4.2. Selective-Retransmit Phase

We distinguish between four packet states, as depicted in Figure 1: *Pending* packets which have not yet been sent or for which negative feedback was received, *unACKed* packets which were sent but for which no feedback has been received, *ACKed* packets for which positive feedback was received and *dropped* packets which were not ACKed and which are assumed not to have arrived on time. During the Selective-Retransmit

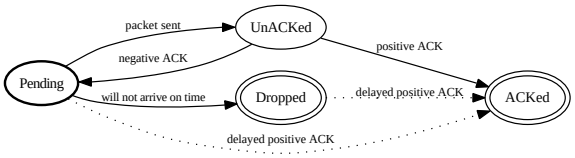

Figure 1: Possible packet states

phase, the decision mechanism $D_{\text{MB}}(\Pi_{pe}, t_{now})$ determines which packet from the set of pending packets Π_{pe} to transmit each time the sender is allowed to send. We define D_{MB} as

$$D_{\text{MB}}(\Pi, t) = \{p \in \Pi | \Phi(p,t) = \max(\{\Phi(q,t) | q \in \Pi\})\}$$

where Φ provides an estimate of the relevance of sending packet p at time t and is defined as follows:

$$\Phi(p,t) = \omega_C \cdot \Psi_{\text{MB}}(p) + \omega_T \cdot \Psi_{\text{T}}(p,t) \in [0;1]$$
$$\omega_C + \omega_T = 1$$

In the previous equation, ω_C and ω_T are weights which determine the importance of content and timeliness respectively. Ψ_{T}, which reflects the temporal importance of packets, is defined as

$$\Psi_{\text{T}}(p,t) = \begin{cases} e^{4 \cdot \frac{t_R - t}{t_B}} & \text{if } t_R - t_B \leq t \leq t_R \\ undef & \text{otherwise} \end{cases}$$
$$t_R = t_S + T(p) - {}^{RTT}/_2$$

Ψ_{T} is designed to let the importance of packets exponentially increase as they approach t_R, which is the latest possible moment to successfully transmit p. After this point in time, p would arrive after its playback deadline and would therefore only waste bandwidth. On the other hand, sending a packet too early ($t < t_R - t_B$) would exceed the initial receiver buffer, can cause an unnecessarily high transmission rate and is hardly possible in real-time scenarios.

5. Performance Evaluation

In order to benchmark the efficiency of our approach, we deployed its implementation in our testbed, depicted in Figure 2. For emulating packet loss and delay fluc-

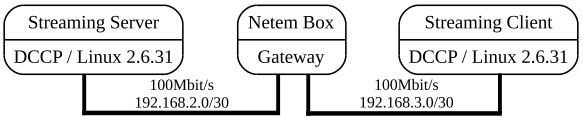

Figure 2: Test environment

tuations, NetEm[2] is used at the gateway as queueing discipline for the network 192.168.3.0/30. To restrict the bandwidth, we use a token bucket filter with a sufficiently large buffer. On both, sender an receiver, we installed Linux kernel 2.6.31 which is compatible with the newest DCCP kernel module from the official development tree. We set $tx\_qlen = 0$ using the `sysctl` interface to force blocking behavior and adjust CCID and kernel send buffer size appropriately via the `sockopt` interface. An exceedingly large send buffer would introduce unnecessary delay and would diminish the benefits of selective retransmissions. On the other hand, using a send buffer which is too small limits the throughput, as shown in Figure 3.

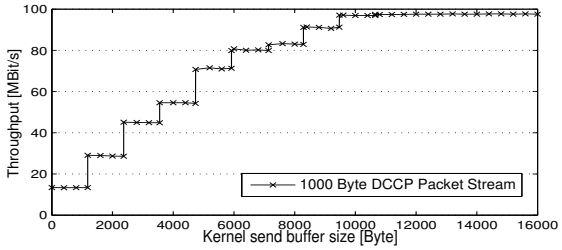

Figure 3: Influence of send buffer on throughput

In order to establish the sequence number mapping, we extracted information about assigned sequence numbers and received ACK vectors directly from the sender's DCCP module. This was done by extending the `sockopt` interface with two additional functions. Furthermore, we extended the lookup table of the TFRC equation to deliver finer-grained transmission rates.

To keep the measurement results as general as possible, we used 24 test sequences out of four different categories, which are characterized by low/high spatial/temporal complexity and have a length of 2000 to 30000 frames. All sequences were encoded as variable bit rate MPEG4/ASP video using the common group of pictures structure $I(BBP)^3BB$. We used packet loss ratios of 0.1% to 5% with a burst probability of 25 % and a normally distributed RTT (μ=60 ms, σ=6 ms) as suggested in [7]. In the experiments, we applied our

content-aware selective reliability scheme D_{MB}. It was tested against a non-reliable video transmission scheme which we denote as D_{NR}, and the earliest-first selective reliable strategy D_{EF}, which can be defined as

$$D_{\text{EF}}(\Pi, t) = \{p \in \Pi | \Psi_{\text{T}}(p,t) = \max(\{\Psi_{\text{T}}(q,t) | q \in \Pi\})\}$$

D_{EF} always selects the packet with the earliest timestamp which still can meet its playback deadline.

The mean of all average video bitrates of the test sequences is 273 KB/s with a standard deviation of 63 KB/s. During all tests, the bandwidth of the link connecting gateway and receiver (which can be interpreted as the "last mile") was restricted to 8 MBit/s. Besides emulated packet loss, we set up two competing TCP flows—one trying to fully saturate the link (e.g. a download) and the other being characterized by a bursty rate (e.g. web browsing), using the traffic generator *Harpoon* [9].

The results of the performance evaluation are summarized in Figure 4. As expected, the earliest first selective retransmit scheme D_{EF} performs significantly better than D_{NR}. By using our approach D_{MB}, the achieved perceptual video quality improvement can be further increased (up to 4 dB at loss rates of 5%).

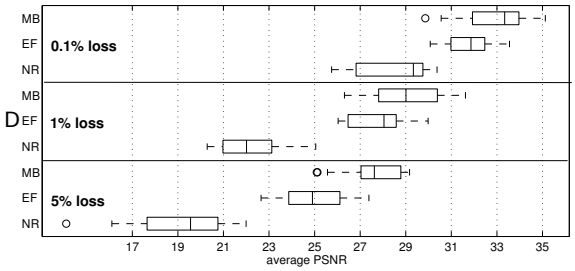

Figure 4: Performance of schemes at different PLRs

To further demonstrate the benefits of D_{MB}, one specific sequence, WARGAMES, was picked from the set of test videos. The evaluation results, as depicted in Figure 5, indicate that most of the time, both selective retransmit mechanisms provide the same video quality. This is the case when the DCCP send rate is above the video's bit rate and the receiver buffer level is high enough to let all retransmission attempts succeed. Occasionally, the congestion situation worsens (especially during the high-loss phase) and decisions have to be made which packets to skip and which ones to transmit. In nearly all such situations, D_{MB} masters this task better than D_{EF}.

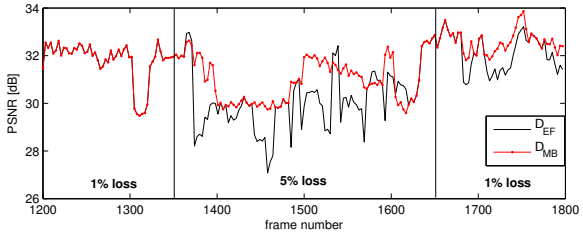

Figure 5: PSNR of test sequence WARGAMES

6. Conclusion

In this paper, we proposed a selective retransmission scheme for DCCP, which uses a sophisticated, macroblock-composition based method to decide which packet to send next. This decision not only depends on playback timestamps of packets, but also incorporates estimates regarding their impact on the perceptual quality in case of loss. Our method is especially designed for wireless environments, where bandwidth and CPU power can be scarce, making it attractive to use a pure sender-side scheme which does not increase the amount of data sent across the network.

Acknowledgements: This work was funded by the EU IST project EC-GIN under the contract STREP FP6-2006-IST-045256, TransIDEE and ASFiNAG.

References

[1] S. Floyd, E. Kohler, and J. Padhye. Profile for DCCP Congestion Control ID 3: TCP-Friendly Rate Control (TFRC). RFC 4342 (Proposed Standard), March 2006.

[2] S. Hemminger. Network Emulation with NetEm. In *Linux Conf Au*, April 2005.

[3] A. Huszák and S. Imre. Source Controlled Semi-reliable Multimedia Streaming using Selective Retransmission in DCCP/IP Networks. *Computer Communications*, 31(11):2676–2684, July 2008.

[4] E. Kohler, M. Handley, and S. Floyd. DCCP. RFC 4340 (Proposed Standard), March 2006.

[5] I. McDonald and R. Nelson. Application-Level QoS: Improving Video Conferencing Quality through Sending the Best Packet Next. In *International Conference on Next Generation Mobile Applications, Services, and Technologies*, pages 507–513, September 2008.

[6] S. Nosheen, S. Malik, Y. B. Zikria, and M. K. Afzal. Performance Evaluation of DCCP and SCTP for MPEG4 Video over Wireless Networks. In *Internat. Multitopic Conference*, pages 1–6, December 2007.

[7] Y. Sato, S. Ata, I. Oka, and C. Fujiwara. Using Mixed Distribution for Modeling End-to-End Delay Characteristics. In *Asia-Pacific Network and Management Symposium*, volume 105, pages 59–64, September 2005.

[8] M. Schier and M. Welzl. Selective Packet Discard in Mobile Video Delivery based on Macroblock-Level Distortion Estimation. In *IEEE Infocom MoViD Workshop*, pages 1–6, April 2009.

[9] J. Sommers and P. Barford. Self-configuring network traffic generation. In *4th ACM SIGCOMM Conference on Internet Measurement*, pages 68–81, 2004.

[10] L. Yuan-Cheng and L. Ching-Neng. DCCP partial reliability extension with sequence number compensation. *Computer Networks*, 52(16):3085–3100, Nov. 2008.