# Analysis and Evaluation of a Scalable QoS Device for Broadband Access to Multimedia Services

Wenyu Zhu, Thomas Dreibholz, Erwin P. Rathgeb
University of Duisburg-Essen, Institute for Experimental Mathematics
Ellernstrasse 29, 45326 Essen, Germany

***Abstract*—This paper presents the initial evaluation of a novel network device being located in edge nodes. It provides relaxed QoS guarantees to certain flows on a congested link by focussing packet discard on selected flows. In contrast to classical IntServ solutions, our approach requires minimal signalling and therefore provides both efficiency and scalability. In this paper, we first describe the ideas of our QoS device and then provide first results of our ongoing simulative performance evaluation and optimization.**

**Keywords: Quality of Service, Broadband Internet, Flow Routing, Multimedia, Intelligent Packet Discard**

## I. Introduction

With DSL technology becoming widespread, a rising amount of customers gets connected to high-speed Internet backbones. Such links make new multimedia services like video and audio on demand possible. However, unlike for best-effort applications, such services have tighter QoS requirements. In particular, they need an assured bandwidth. We assume that core bandwidth is usually over-provided and only the link to the customer becomes the bottleneck. When multiple equal-priority flows exceed a DSL link's bandwidth, the quality of *all* flows suffers due to packet loss. Using RSVP to establish per-flow reservations would solve the problem – but introduce complex signalling procedures and limited scalability [1]. In our paper, we present a novel, simple and scalable approach to ensure relaxed QoS guarantees while only requiring a minimum effort on signalling. Our approach [2], [3] is based on a QoS device located inside an edge node before the bottleneck link. It is currently under consideration by ITU-T and ETSI [4], [5].

## II. Our QoS Device

Key idea of our QoS device is – in case of congestion – to focus packet discard on selected flows instead of dropping arbitrary packets. Then, only the selected flows would suffer from quality loss instead of all flows. This "last straw"[1] principle, applied at the ATM cell level, was first suggested in [6] and its value has been shown in other publications since [7]. We apply it to our device by making the latest flow(s) the subject of discard (as the default policy; arbitrary other schemes may be applied as well). Therefore, our device has to know when a new flow starts, and has to maintain a record of it.

For the device to recognise the start of a flow, its sender is only required to send a *start packet*; no further signalling is necessary. There is no need to wait for any acknowledgement – the sender may just start transmitting. The start packet contains flow identity information to identify the data flow and an estimation of the flow's bandwidth. This information is recorded by our device. Flow identification and record keeping can be realized easily as part of a flow router [8].

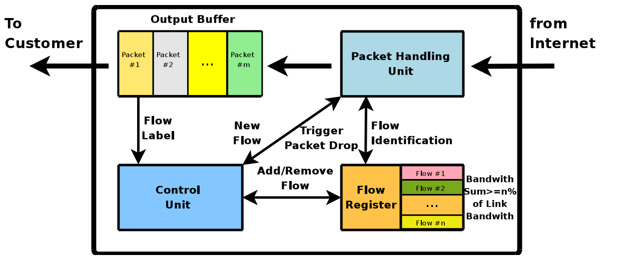

Figure 1. QoS Device Overview

The device (see figure 1) maintains a window of flows that are vulnerable to packet discard in case of congestion: the *drop window*. The identities of the drop window flows are stored in the flow register. As new flows start up, a flow moves through the drop window, until it is eventually removed from the window (by being overwritten by new entries) – according to a configured flow register policy. In the simplest case, it selects oldest flow entry: new flows should not disturb already running flows.

On removal of a flow identity from the drop window, it becomes a *guaranteed area* flow – except under extreme traffic conditions (i.e. when dropping all drop window flows' packets is still not sufficient). There is *no* storage of guaranteed area flow identities. All packets not belonging to flows of the drop window are implicitly assumed to be guaranteed flows. This ensures the scalability of our device. Protection against denial of service attacks (e.g. by filling the drop window with non-existent flow identities) can be provided by a proxy, as described in [3].

Based on OMNeT++ and the SimProcTC tool-chain [9], we have designed a simulation model of the QoS device and validated it against a former, fast-track model [2]. Furthermore, source and sink models have been created. The sources can either use CBR/VBR traffic or use media traces for realistic traffic patterns.

Upon congestion (i.e. full output queue), a *dropper strategy* is applied to remove certain packets using one of the following strategies: *DropAll* traverses the output buffer and drops *all* packets belonging to drop window flows. *DropEnough* only drops enough packets to fit in the

[1]"It's the last straw which breaks the camel's back" – Proverb.

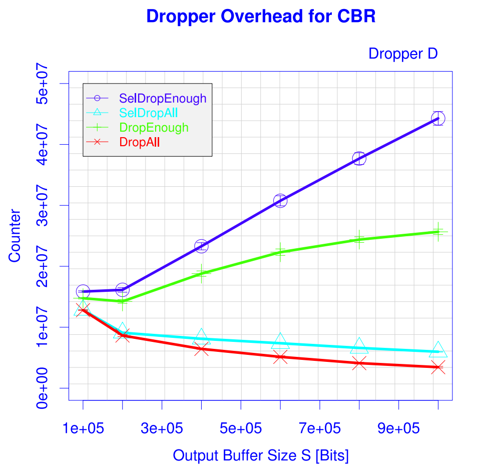

Figure 2. Dropper Overhead Results

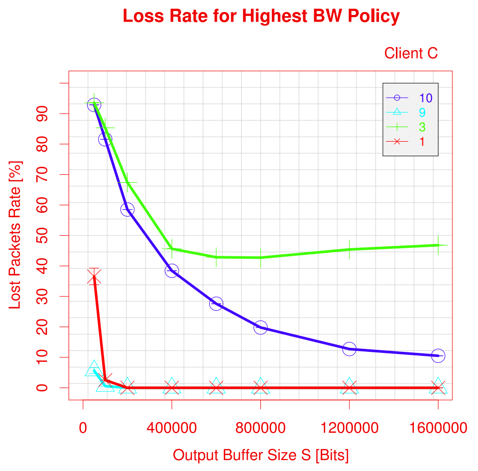

Figure 3. Using the "Highest Bandwidth" Policy

new guaranteed area packet. *SelectiveDropAll* first tries to drop all packets of the first drop window flow. If there is still not enough space, it continues with the second flow and so on (i.e. the position in the drop window becomes similar to a priority). *SelectiveDropEnough* stops dropping as soon as there is enough space.

Initial performance metrics are delay and packet loss rate, due to independence of specific media codecs.

## III. First Performance Results

As generic proof of concept, we have varied the output buffer size in a setup of 5 flows using 50 frames/s and a frame size 5,000 bytes (i.e. 10 Mbit/s total bandwidth) over an 8 Mbit/s bottleneck link. The register contains 2 flows, i.e. 2 Mbit/s. Comparing the loss rates for the 4 dropper strategies, the drop window flows are in focus of packet discard. When the output buffer is sufficiently large (here: $\geq 5 * 10^5$ bits), there is no significant loss for guaranteed area flows. The selective strategies reduce the loss rate of the second flow register flow – at cost of the first one. Furthermore, using DropEnough or SelectiveDropEnough, the packet loss for drop window flows is minimized – at cost of a slightly higher delay.

Comparing the dropping effort (i.e. buffer maintenance overhead) – as depicted in figure 2 – the effort for SelectiveDrop is only slightly higher than for DropAll. Furthermore, SelectiveDropEnough and DropEnough have significantly higher overhead, since DropAll/SelectiveDropAll remove a set of packets – which gains space for the next burst of guaranteed area messages. But DropEnough/SelectiveDropEnough only obtain space for the next packet. So, at a high probability, the dropper will be invoked soon again. In summary, SelectiveDropAll is the most useful strategy, achieving good performance at reasonable overhead.

For a more realistic evaluation, we have used 10 flows – using MP3, H.263 and MPEG traces from [1] – of about 12.5 Mbit/s total bandwidth over a 10 Mbit/s link. The drop window contains flow #10 (MPEG stream) and flow #9 (MP3 stream). In this setup, the overload exceeds the data rate in the drop window – resulting in losses for guaranteed area flows. Therefore, additional drop window flows have to be chosen by deriving flow identities from queued packets. Two selection strategies have been considered: (1) random selection and (2) using the flow having the largest number of bytes queued.

The first strategy may select one of the MP3 flows – which still would not solve the congestion. The second strategy is better, as shown in figure 3: flow #3 – a high-bandwidth MPEG stream – is added to the drop window. Given a reasonable buffer size, there is no loss for guaranteed area flows any more. Also, the MP3 flow #9 (as last entry in the drop window) has no significant loss: applying SelectiveDropAll, removing packets of the two MPEG flows is already sufficient.

## IV. Ongoing Work

Currently, we are evaluating the system parameters in detail. Particularly, we intend to realize dynamic adaptation to changing scenarios and customer needs. We will also evaluate implications on the user's perceptual quality (e.g. using metrics like PEAQ/PEVQ). Also, we go to realize the QoS device as Linux queuing discipline – to perform lab experiments with real-world applications.

## References

[1] T. Dreibholz, "Management of Layered Variable Bitrate Multimedia Streams over DiffServ with Apriori Knowledge," Masters Thesis, University of Bonn, Institute for Computer Science, Feb. 2001.

[2] T. Dreibholz, A. IJsselmuiden, and J. L. Adams, "An Advanced QoS Protocol for Mass Content," in *Proceedings of the IEEE Conference on Local Computer Networks (LCN) 30th Anniversary*, Sydney/Australia, Nov. 2005, pp. 517–518.

[3] T. Dreibholz, A. J. Smith, and J. L. Adams, "Realizing a scalable edge device to meet QoS requirements for real-time content delivered to IP broadband customers," in *Proceedings of the 10th IEEE International Conference on Telecommunications (ICT)*, vol. 2, Papeete/French Polynesia, Feb. 2003, pp. 1133–1139.

[4] A. IJsselmuiden and J. L. Adams, "Delivery of assured QoS content in NGNs," ITU-T, British Telecom Contribution D-,Q4,6,10,11,16,SG13, Feb. 2004.

[5] ——, "Delivering QoS from remote content providers," ETSI, British Telecom Contribution TISPAN 01(03)TD132, Sept. 2003.

[6] A. J. Smith, J. L. Adams, C. J. Adams, and A. G. Tagg, "Use of the Cell Los Priority Tagging Mechanism in ATM switches," in *Proceedings of the ICIE 1991*, Singapore, Dec. 1991.

[7] K. Kawahara, K. Kitajima, T. Takine, and Y. Oie, "Performance evaluation of selective cell discarding schemes in ATM networks," in *Proceedings of the IEEE Infocom '96*, vol. 3, San Francisco, California/U.S.A., Mar. 1996, pp. 1054–1061.

[8] L. G. Roberts, "The Next Generation of IP – Flow Routing," in *Proceedings of the International Conference on Advances in Infrastructure*, L'Aquila/Italy, July 2003, pp. 25–.

[9] T. Dreibholz and E. P. Rathgeb, "A Powerful Tool-Chain for Setup, Distributed Processing, Analysis and Debugging of OMNeT++ Simulations," in *Proceedings of the 1st ACM/ICST OMNeT++ Workshop*, Marseille/France, Mar. 2008.