A Scalable QoS Device for Broadband Access to Multimedia Services

Wenyu Zhu, Thomas Dreibholz, and Erwin P. Rathgeb
University of Duisburg-Essen, Institute for Experimental Mathematics
Ellernstrasse 29, 45326 Essen, Germany
{thomas.dreibholz,erwin.rathgeb}@uni-due.de

Xing Zhou
Hainan University, College of Information Science and Technology
Renmin Avenue 58, 570228 Haikou, Hainan, China
zhouxing@hainu.edu.cn

Abstract

Nowadays, an increasing number of users get high-speed broadband access to the Internet. These broadband connections make new multimedia applications possible. However, such applications also introduce new Quality of Service (QoS) requirements to the network. Particularly, they require an assured bandwidth even in the case of network overload. The network itself has to provide such bandwidth assurances.

This article introduces our concept of a novel network QoS device being located in network edge nodes. It provides a solution for relaxed QoS guarantees to certain flows on a congested link by focussing packet discard on selected flows. However, unlike for IntServ solutions like RSVP, our approach only requires minimal signalling and therefore provides both efficiency and scalability. We furthermore provide a quantitative performance evaluation of our QoS device by using simulations.

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

1. Introduction

With DSL technology becoming widespread, a rising number of customers is getting connected to high-speed Internet backbones. Such links do not only speed up existing applications but also make services like video and audio on demand possible. However, unlike for best-effort applications, such services demand stricter 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 (see figure 1). When multiple equal-priority flows exceed a DSL link's bandwidth, the quality of *all* flows suffers due to packet loss.

As long as the user does not request more media flows than available link bandwidth, there should be no problem. However, let us consider the following scenario shown in figure 2: one family member requests a sports video at 5 Mbit/s, another one requests a soap opera at 3 Mbit/s and yet another one requests an online game at 5 Mbit/s via a single 10 Mbit/s link. This is a situation where packet loss is likely to occur. Since all flows have equal priority, the packet loss is likely to affect all flows. It is possible that the quality reduction is so severe that all flows are of unacceptable quality. Clearly, an edge node that was able to apply an intelligent discard policy, focusing loss on a single (or as few as possible) flow, would minimise disruption to the total number of flows.

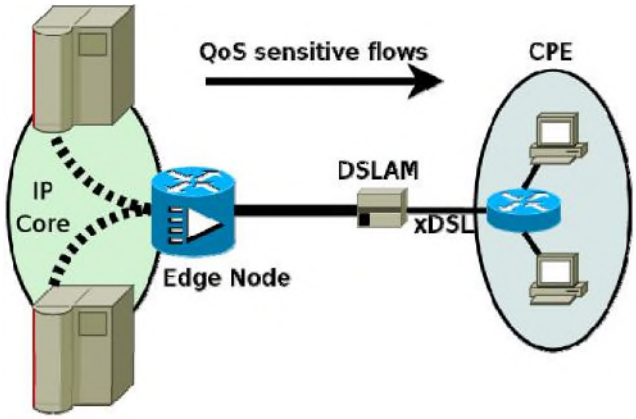

Figure 1. A Typical xDSL Broadband Scenario

Figure 2. A Bottleneck Link Overload Example

Using RSVP [1], [2] to establish per-flow reservations would solve the problem – but also introduce complex signalling procedures and a lack of scalability [3]. In our article, we present a novel, simple and scalable approach to ensure relaxed QoS guarantees while only requiring a minimum effort on signalling. Our approach [4]–[12] is based on a QoS device located inside an edge node before the bottleneck link. It is currently under consideration by the Standards Bodies ETSI [13] and ITU-T [14]–[16].

2. Our QoS Device Concept

The key idea of our QoS device is that – in case of congestion – it is better to focus packet discard on selected flows than to discard 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 [17] and its value has been shown in other publications since [18], [19]. 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. In particular, the sender is not required to wait for any acknowledgement – it may just start sending data. The Start Packet contains flow identity information (i.e. packet header fields; see also [20]) 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 [21], [22].

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

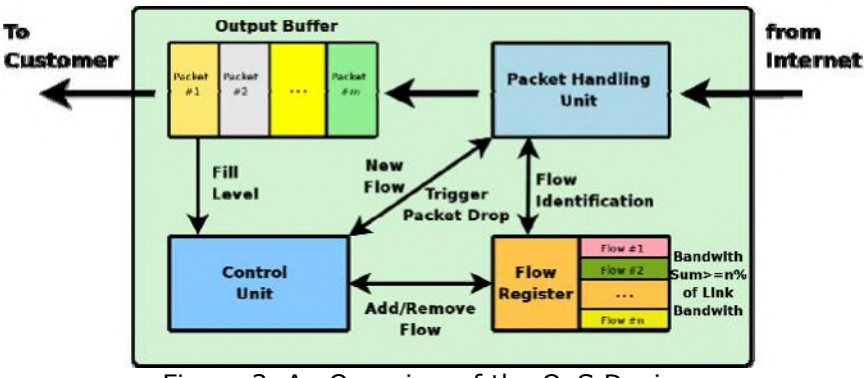

Figure 3. An Overview of the QoS Device

The device – which is illustrated in figure 3 – 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), when one of the following conditions is satisfied:

1) The sum of the rates of the flows in the Drop Window, minus rate of the oldest flow, is greater than R, where R is a percentage of the link bandwidth.
2) The flow's entry is older than time t_{min}.
3) It has received at least $_{pmin}$ packets since startup.

When a flow identity is removed 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). Note, that the list of Guaranteed Area flow identities is not stored. All packets not belonging to flows of the Drop Window are implicitly assumed to be guaranteed flows. This ensures the scalability of our device.

By default, we assume that new flows go into the Drop Window first, in order to avoid disturbing already running flows. Consider a customer viewing a video flow for quite some time: clearly, he would be annoyed if his running flow would be considered for focussed packet loss whenever another user causes overload by new flows. Instead, it is recommended to try to guarantee older flows while discriminating new ones in case of congestion. Nevertheless, our QoS device approach allows the implementation of arbitrary policies, depending on the customers' requirements.

3. Implementation Notes

3.1. Flow Identification and Marking

To recognise different flows, it is necessary to map packets to flows. Using IP-based protocols, flows can be separated by examining source and destination IP address and source and destination port number of the Transport Layer protocol (e.g. UDP); e.g. the RTP payload type [23] can be used to separate sub-components used for layered encoding [3]. But to examine all of these fields, a significant fraction of CPU power is required. IPv6 simplifies flow identification by using flow labels (see [24], [25]). Flow labels combined with the source IP address are network-unique pseudo-random values. The usage of flow labels can simplify the flow identification for IPv6. For IPv4, there is an Internet Draft [20] to add a flow label within an IPv4 option header.

3.2. Using Mechanisms for Explicit Congestion Notification

Using IPv4 or IPv6, congestion notification is standardized as ECN (Explicit Congestion Notification, see [26]). Instead of using specific messages, ECN uses the Type of Service field (IPv4) or the Traffic Class field (IPv6) of the data packets to inform the receiver of experienced congestion. Then, the receiver can e.g. request a lower bitrate from the server or an ECN-aware Transport Layer protocol (e.g. TCP [26], SCTP [27] or DCCP [28]) can inform the sender to reduce its congestion window.

3.3. Coping with the Loss of Start Packets

Start packets may be lost during transport since the IP protocol does not provide acknowledgements and retransmissions. We propose that the server transmits two Start Packets for each flow, before commencing the flow itself. The probability of both packets being lost is very low. However, in the case where both Start Packets are lost, the unknown flow would simply go straight into the Guaranteed Area and be unlikely to cause a problem. If this happens during periods of high congestion and packet discard, the mechanism for extreme congestion which picks on flows in the Guaranteed Area would start to operate and provide a recovery position for the network. Note that in this situation, the mechanism is only designed to allow the network to recover from congestion, and not to detect the flow which caused the problem. To minimize the loss probability, it is recommended to use a special DiffServ [3] class for the transmission of control messages.

For an additional reduction of the Start Packet loss probability, it is possible to send n Start Packets within the first m seconds of the flow and store the flow's identity for at least $\alpha * m$ seconds (e.g. $n = 3$, $m = 5$ and $\alpha = 2$ to cope with network delay), even if the flow has already been moved to the Guaranteed Area. Note, that it is not possible to simply retransmit the Start Packet regularly. Since the identities of the Guaranteed Area flows are not stored, it is impossible for the device to decide whether a received Start Packet is for a new flow or the flow has already been moved to the Guaranteed Area. In the first case it would have to handle the packet, while in the second case it would have to know that it should be ignored.

3.4. Handling of Best Effort Flows

Best effort flows using protocols that are able to handle congestion itself do not need the congestion control behaviour of our device. Such protocols (e.g. TCP [29] and SCTP [27]) are able to adapt their bandwidth usage to the current congestion state of the network by monitoring their packet loss rate. Therefore, using the flow admission mechanism of the device does not make sense for such transmissions. Furthermore, if such flows went into the Guaranteed Area, these flows would assume a congestion-free network and, if there is sufficient data to transmit, increase their bandwidth until packet loss is detected. But packet loss in this case means that there is such extreme congestion that packets of flows within the Guaranteed Area must be dropped.

The simple solution for Best Effort flows is therefore to route them to a separate queue which is not controlled by our QoS mechanism. Both this queue and the multimedia queue operate as weighted fair queues and are assigned a specific partition of the available bandwidth. This is recommended to prevent the device from e.g. blocking all TCP and SCTP flows when the link bandwidth is utilized with multimedia flows. For identification of Best Effort flows, the Transport Layer protocol IDs can be examined (e.g. ICMP, TCP and SCTP are Best Effort, all other ones use the Drop Window/Guaranteed Area mechanism).

3.5. Security Considerations

Since the edge node is the central connection point of many customers to the Internet, there is a high risk of Denial of Service (DoS) attacks. Since downtimes are critical in commercial networks, a very high level of security is mandatory here. Since our QoS device does not save the identities of flows within the Guaranteed Area, and packets that do not belong to a flow within the Drop Window are implicitly assumed to belong to flows of the Guaranteed Area. The advantage of this approach is that it is not necessary to maintain active/ceased states. On the other hand, this makes the QoS device vulnerable to DoS attacks: an attacker only has to send junk to a customer without supplying a Start Packet. The edge node assumes this junk to be within the Guaranteed Area and may drop packets of real customer flows currently within the Drop Window.

The security concept, proposed by us in [10], is shown in figure 4: all content servers must be located within a protected subnet, preferably directly connected to the edge node. Access to content providers within the Internet is realized using a proxy or gateway within this protected subnet. Proxies/gateways provide connection establishment to the outside world, generate Start Packets and handle congestion notifications. For security reasons, there should be no direct connection from the subnet to the Internet. Instead, the proxies/gateways should be dual-homed and their connection to the Internet should be protected by an external firewall. Further, the edge node must handle all traffic not having its source within the subnet as Best Effort traffic (see subsection III-D), shown in figure 4 as Best Effort Access. Specifically, an access control list must filter out IP-spoofed packets from the public Internet claiming to have its source in the subnet. This makes sure that it cannot be harmful to the Drop Window/Guaranteed Area mechanism. The device should also be protected by firewalls on the customer links and the Internet link for Best Effort data. Intrusion Detection Systems (IDS) can monitor network segments for unusual behaviour. In case of attacks, an IDS can inform the administrator or trigger defence actions like adapting firewall rules.

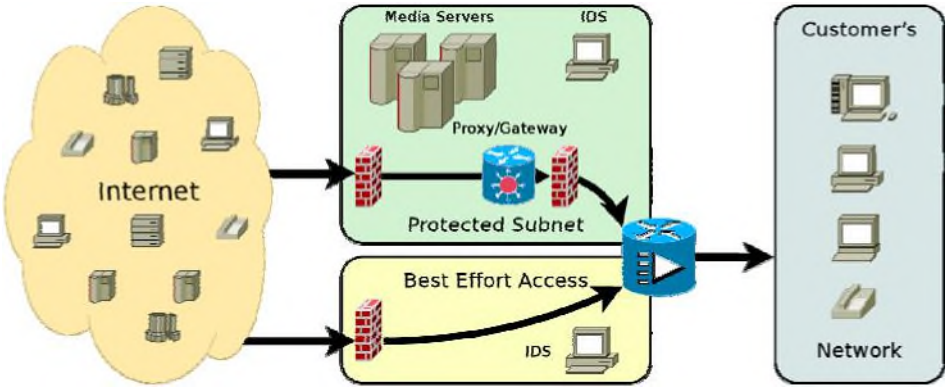

Figure 4. Our Security Concept for the QoS Device Deployment

As described above, all traffic from the protected subnet may be handled as Drop Window/Guaranteed Area traffic while all other transmissions are handled as Best Effort. All flows using the device's flow control mechanism have their source in the subnet, either directly at a server or at a proxy/gateway for flows from providers within the Internet. Therefore, only the traffic from the subnet is critical. Since the edge node relies on the integrity of the servers and proxies, it is mandatory that customers requesting services from

these systems are always authenticated. This ensures that no other customer can spoof another user's identity and start unwanted data transmissions to a spoofed customer. The approach of delegating authentication to the servers or proxies instead of implementing it into the edge node simplifies the device. Furthermore, it is scalable since the authentication effort is shared among all servers and proxies.

4. The Simulation Model and its Performance Metrics

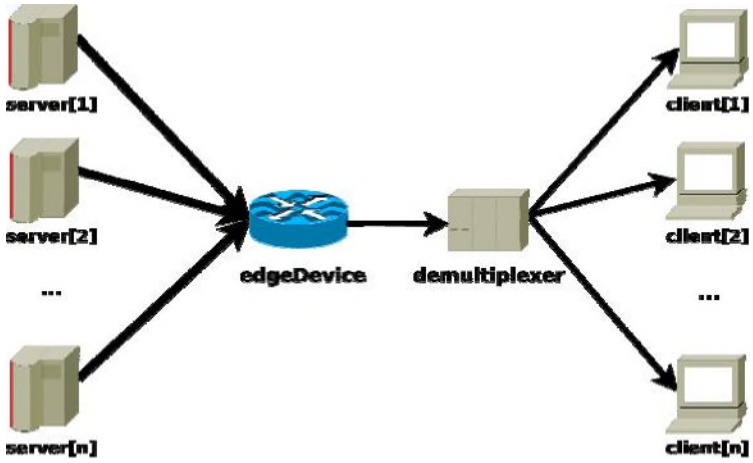

Figure 5. The Simulation Setup

For our performance analysis, we have modelled the QoS device using the OMNET++ [30] simulation environment and the SIMPROCTC [31], [32] tool-chain. The basic simulation setup is illustrated in figure 5 and consists of the QoS edge device module as well as media sources and receivers. Each source can either generate CBR traffic using a configured frame size or by reading the frame sizes from a media trace file. For our analysis, we used MPEG traces from [33] as well as H.263 and MP3 traces from [3]. According to the frame size, frames are generated with a fixed frame rate (38fps for MP3, 30fps for video). The inter-frame time randomly varies by ±25% to avoid synchronization among the sources. Frames are segmented to packets, the packet MTU is 1,000 bytes (header overhead is neglected).

As performance metrics for our evaluation, we have chosen delay and packet loss rate. These metrics of the service user's perspective are independent of media codecs and easy to capture. As part of future work, however, more sophisticated methods of evaluating perceptual quality [34] – like PEAQ for audio – are being considered. Delay and packet loss rate are recorded for each receiver.

Our QoS device model has been validated against a former, LISP-based fast-track simulation approach being described by us in [7], [9]. But unlike the simple model, our new model contains support for a number of dropper strategies. The packet dropper is invoked each time the output buffer is too full for storing the incoming packet of a Guaranteed Area flow:

- *DropAll* traverses the output buffer and drops all packets belonging to Drop Window flows.
- *DropEnough* only drops enough packets to fit in the 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 Drop Window position becomes similar to a priority).
- *SelectiveDropEnough* stops dropping as soon as there is enough space.

The number of total packet/flow identity comparison steps within the dropper function is denoted as the *dropper overhead*. It represents the additional effort of the QoS device in comparison to a regular router (i.e. the service provider's performance metric).

GNU R [35] has been used for the statistical post-processing of the results. Each resulting plot shows the average of multiple simulation runs and the corresponding 95% confidence intervals.

5. Performance Analysis

In our performance analysis, we first illustrate the general behaviour of the device as proof of concept in a simple CBR flow setup in subsection V-A, before we go to MP3 and heterogeneous multimedia scenarios in the following subsections.

5.1. A Proof of Concept

Our basic setup (see figure 5) consists of 5 senders and their receivers (i.e. 5 flows). Each flow uses a frame rate of 50 fps and a fixed frame size of 5,000 bytes. Due to the MTU setting, a frame consists of a burst of 5 packets. The resulting total bandwidth is 10 Mbit/s, while we restrict the link bandwidth to only 8 Mbit/s. The size of the output buffer is crucial for the delay, so we vary it between 10^5 bits and 10^6 bits. The configured Drop Window bandwidth has been 2 Mbit/s, i.e. the Flow Register memorizes two flows for focussed packet discard; in our case: flow #4 and flow #5.

The simulation results are shown in figure 6: packet loss rate (left-hand side) and delay (right-hand side) for the strategies DropAll (solid lines) and DropEnough (dotted lines) at the clients #4 and #5 (Drop Window) and client #1 (Guaranteed Area; other Guaranteed Area flow curves have been omitted, since they are similar). As expected, the Drop Window flows suffer of focussed packet loss while the Guaranteed Area flows' loss rate reduces to 0% when the output buffer size is large enough ($\geq 5 * 10^5$ bits). Furthermore, the loss rate for DropAll is higher, in particular when buffer space is scarce.

Clearly, the reduced loss rate of DropEnough leads to an increased delay. Comparing the delay results of both strategies, an interesting observation can be made: for DropAll, the delay of the Drop Window flows is lower than for the Guaranteed Area flow. Drop window packets either get sent early, or the buffer fills up and they are dropped. However, for DropEnough, an inverse observation can be made: here, Drop Window packets may "survive" for some time, but get lost when a burst of Guaranteed Area packets comes in.

Figure 7 presents the results for the SelectiveDropAll and SelectiveDropEnough strategies: while there is no difference for the Guaranteed Area flow in comparison to DropAll/DropEnough, the selective strategies discriminate flow #5 for the benefit of flow #4. SelectiveDropEnough intensifies this effect by further reducing the loss rate of flow #4 at cost of flow #5. The delay result reflects the expectation from DropAll: SelectiveDropAll leads to the lowest delay for flow #5 (packets are either forwarded quickly or dropped) and the highest delay for the Guaranteed Area flow. Again, the order is reversed for SelectiveDropEnough.

Clearly, SelectiveDropAll and SelectiveDropEnough are the better choices from the media user's perspective. But from the QoS device provider's perspective, it is clearly useful to keep the additional effort for the dropper small. Figure 8 therefore presents the overhead (as

defined in section IV) for the simulations above. For DropAll and SelectiveDropAll, the effort decreases with the output buffer size – a large buffer leads to fewer dropper invocations. On the other hand, the effort increases for DropEnough and SelectiveDropEnough: these strategies just free enough space for storing the next incoming Guaranteed Area packet. A burst of packets – caused by segmentation of frames larger than the MTU – leads to a series of dropper invocations. Furthermore, an increased buffer size leads to an increased effort to find packets to drop. Clearly, this effect is amplified by SelectiveDropEnough – which requires multiple iterations for different Drop Window flow identities.

In summary, our proof of concept has shown that the device using SelectiveDropAll works well from the user's perspective and is also efficient from the provider's perspective.

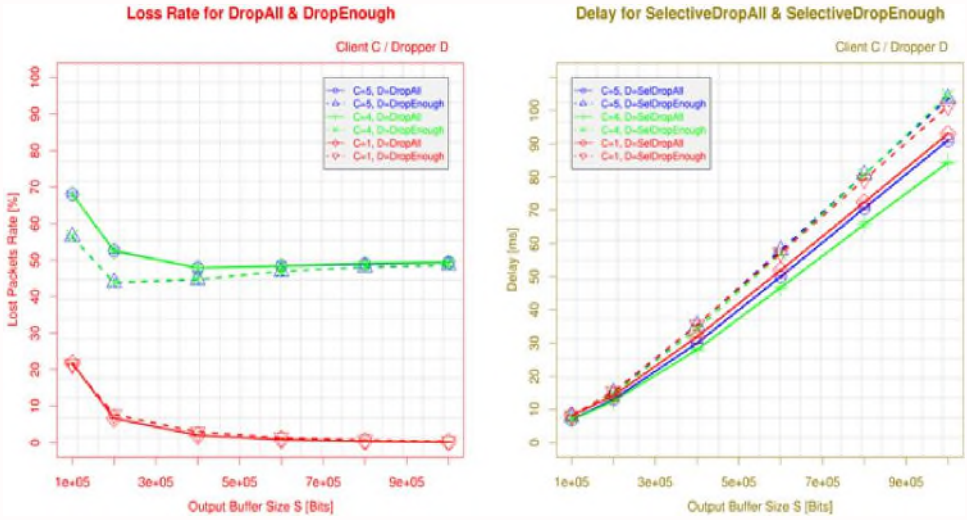

Figure 6. Proof of Concept: DropAll/DropEnough

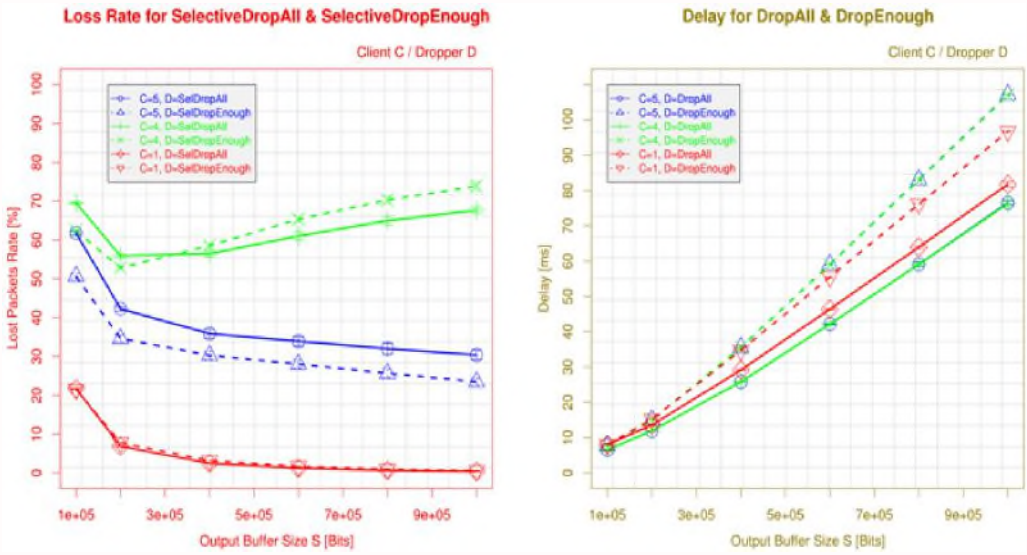

Figure 7. Proof of Concept: SelectiveDropAll/SelectiveDropEnough

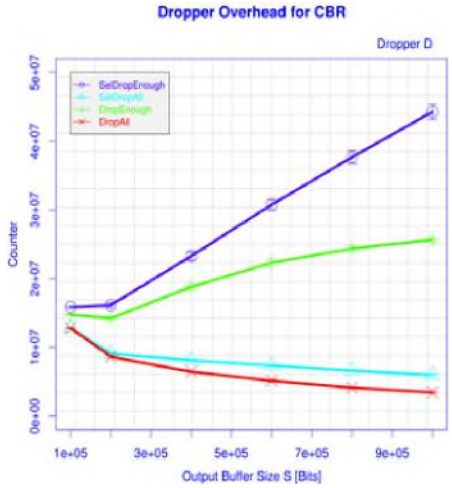

Figure 8. Proof of Concept: Dropper Overhead

5.2. MP3 Flows Scenario

For the following simulations, we have set up realistic environments. Nowadays, the most popular multimedia application on the Internet is online radio. Therefore, our scenario contains 12 flows using the VBR MP3 traces from [3]. The total bandwidth estimation is about 2.5 Mbit/s, while the link bandwidth is only 2 Mbit/s. The DropWindow bandwidth is 500 kbit/s (i.e. 25% of the link bandwidth), leading to 2 flows (here: flow #11 and flow #12) in the Flow Register.

While the results for the delay are as expected from the proof-of-concept simulations in subsection V-A (and therefore a plot has been omitted), the packet loss rate is depicted on the left-hand side of figure 9 for the SelectiveDropAll strategy. Again, we have chosen only one "representative" Guaranteed Area flow (flow #1, the other flows behave in the same way) and the two Drop Window flows (flow #11 and flow #12). Flow #11 is the first flow in the Drop Window and therefore suffers – as expected – from the highest loss rate. Interesting is the loss rate hollow at an output buffer size of S=25,000 bits: for smaller settings of S, packets get dropped very often – regardless of their flow's status (Guaranteed Area or Drop Window). On the other hand, for higher values of S, the QoS device's functionality comes into play: the buffer size is large enough to store sufficient Drop Window packets which can be dropped when the space gets scarce. That is, space is gained for Guaranteed Area packets, leading to reduced Guaranteed Area drops but increased loss for Drop Window flows.

While significant dropper overhead differences have been found in the proof-of-concept scenario in subsection V-A, the results for the MP3 flow scenario (shown on the right-hand side of figure 9) appear unexpected: the differences between the strategies are quite small – and SelectiveDropAll even has a slightly higher overhead than SelectiveDropEnough. The reason for this behaviour is that the MP3 flows have significantly smaller frame sizes. The proof-of-concept simulation used a frame size of 5000 bytes, i.e. each frame has been segmented into 5 packets. With regard to the frame rate, the network bandwidth is much higher, so the interval between two packets has been much smaller than the interval between two frames. Therefore, a large number of packets has arrived in a very short time – leading to a high congestion probability during this period. However, for the MP3 flows, the maximum

frame size is 1044 bytes and most frames consist of much less than 1000 bytes. That is, the packet rate is almost equal to the frame rate, so the interval between two packets is approximately equal to the interval between two frames. Therefore, multi-packet congestion (see subsection V-A) during this time is not likely. Therefore, SelectiveDropAll and SelectiveDropEnough strategies have similar behaviour from the viewpoint of the dropper overhead. But since SelectiveDropEnough stops immediately after sufficient buffer space has been gained, SelectiveDropAll checks for further packets to be dropped (which is not necessary here to prevent congestion, but nevertheless increases the overhead).

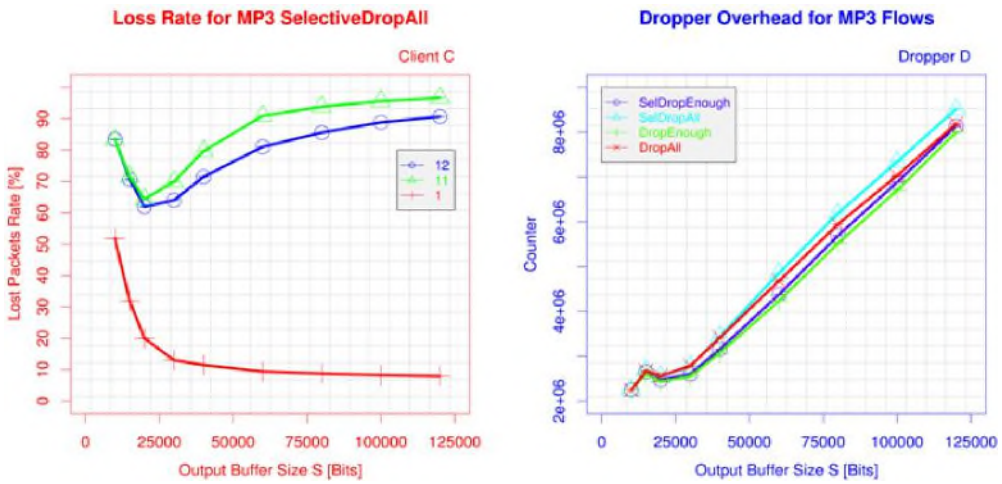

Figure 7. MP3 Flow Scenario

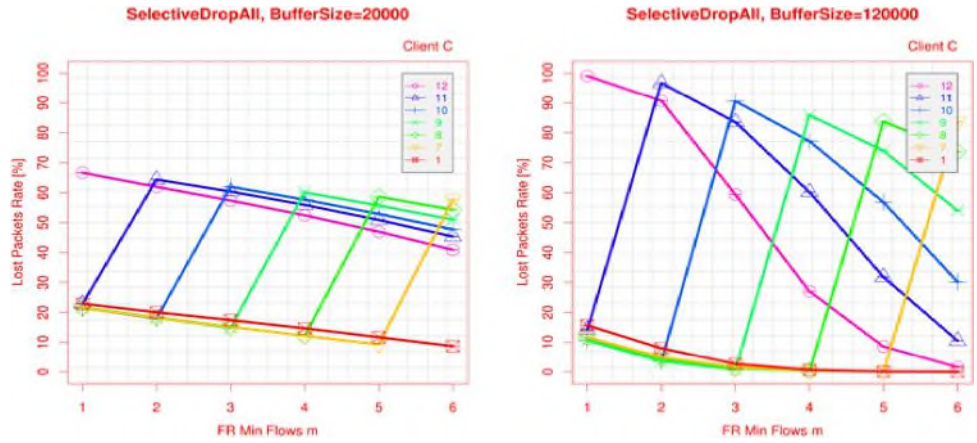

Figure 9. Variation of the Drop Window Size

5.3. Adapting the Drop Window Size

Another important parameter to be analysed is the number of flows in the Drop Window. We reuse the MP3 flow setup of subsection V-B and adjust the number of Flow Register

entries by an appropriate setting of the Flow Register bandwidth (each MP3 flow has an average bandwidth of about 200 Kbit/s).

In the results analysis of the simulation, we stress on the packet loss rate – the delay results are as expected. Figure 10 shows the packet loss rates of the flows for the SelectiveDropAll strategy and output buffer sizes 20,000 bits (left-hand plot) and 120,000 bits (right-hand plot). Flow #1 is again the "representative" Guaranteed Area flow, the other flows – beginning from flow #12 – are successively added to the Drop Window with increasing number of Drop Window entries *m*. That is, for a single entry, the Drop Window contains only flow #12, while it includes flow #12 to flow #7 for 6 entries.

For a too small output buffer (20,000 bits, left-hand plot), it becomes difficult for the dropper to find a sufficient number of Drop Window packets to gain space. Therefore, even for a Flow Register bandwidth of 60% of the link bandwidth (i.e. 6 flows here), the Guaranteed Area packet losses do not reach 0%. Using a more appropriate buffer size (120,000 bits, right-hand plot), a Flow Register size of 40% (i.e. 4 flows) already achieves a lossless transport of the Guaranteed Area packets.

In summary, it is crucial for the QoS device's functionality to have a certain output buffer space to find Drop Window packets to discard in case of overload. Clearly, the number of Drop Window flows also has to be sufficiently large. Therefore, the administrator of the device has to carefully provision these parameters to achieve the highest benefit for the user. However, unexpected things can always happen – and a configuration may be non-optimal, resulting in Guaranteed Area losses. The counter-measure of the QoS device in such cases is the subject of our following analyses.

5.4. Heterogeneous Multimedia Flows

In order to test our model in a more complex and heterogeneous environment, ten different media trace files (MP3 and H.263 from [3], MPEG from [33]) are used for the following simulation. The flows' total estimated bandwidth is about 12.5 Mbit/s, so the output rate is set to 10 Mbit/s to trigger congestion. The Drop Window is 2.5 Mbit/s, which consists of flow #10 (MPEG stream) and flow #9 (MP3 stream). In our analysis, we also present two selected Guaranteed Area flows: flow #1 (a low-bandwidth MPEG stream) and flow #3 (a high-bandwidth MPEG stream).

The left-hand side of figure 11 presents the packet loss rate using the SelectiveDropAll strategy. Although two flows (flows #10 and #9) are in the Drop Window, this is clearly insufficient: even for an output buffer size of as large as $1.6*10^6$ bits, flow #3 in the Guaranteed Area still suffers from a loss rate of more than 10%. The reason for this problem is the Flow Register policy: originally, the flows in the Drop Window follow the FIFO policy. That is, when a new flow identity is pushed into the tail of the Flow Register, the first flow identity in the Flow Register will be popped out. Finally, the latest flows are kept in the register.

To cope with this problem, we change the Flow Register policy: when a Start Packet comes in, the
flows which have the highest bandwidth estimation are put into the Drop Window. In this case, the Drop Window will finally consist of high-bandwidth flows only. The right-hand side of figure 11 shows the results for using the new policy. Now, the Drop Window consists of the high-bandwidth flows #3 and #10, which are in the focus of packet discard. All the other flows situated in the Guaranteed Area benefit from lossless transmission.

However, while the new "Highest Bandwidth" policy achieves a significant benefit over "FIFO" in

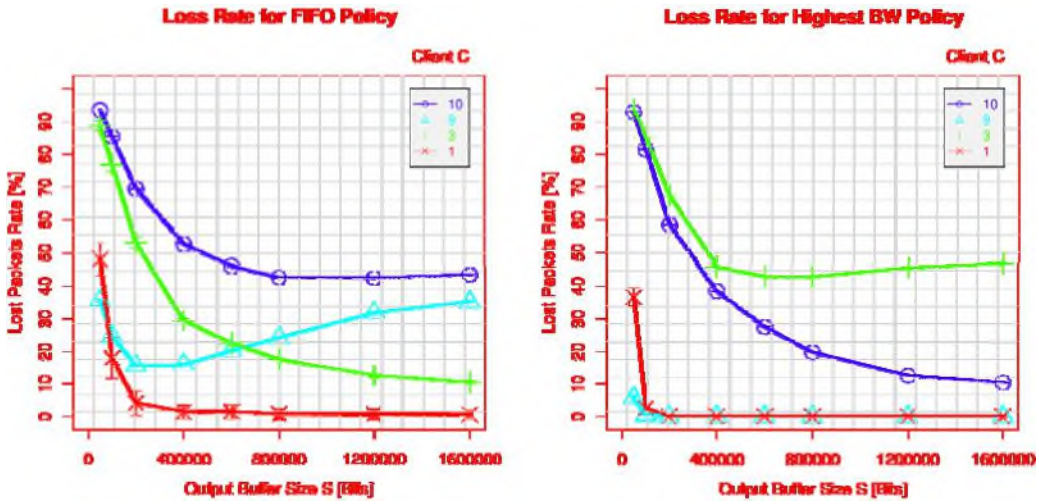

Figure 10. Multimedia Flow Scenario

case of congestion, it has an obvious disadvantage: high-bandwidth flows are target of packet drops – without regard to how long they are already transmitting. Taking the example of the video user from section II, a viewer already watching a movie for some time would clearly be disappointed when its stream gets in the focus of packet discard in favour of another user's newly started soap opera stream.

A better approach for such a scenario will be presented in the following.

5.5. Avoiding Guaranteed Area Losses

The solution for avoiding Guaranteed Area losses in case of extreme overload is clearly to add another flow to the Drop Window, which – hopefully – leads to sufficient discard possibilities for the dropper to resolve the Guaranteed Area congestion. Appending the new flow to the tail of the Flow Register, SelectiveDropAll will touch it only as last resort – after all other Drop Windows packets have been discarded from the output buffer. This should minimize the user's quality loss. Selecting a Guaranteed Area flow for move into the Drop Window is challenging: as described in section II, the identities of Guaranteed Area flows are not stored. Our solution is simple: we derive identities from packet headers (addresses, ports, etc., see [20]) in the output buffer. That is, selecting a Guaranteed Area packet results in its flow identity.

Two policies for selecting a new Drop Window flow have been considered: "Add Random" simply selects a random Guaranteed Area packet and adds its flow identity. "Add Max Packets" adds the flow currently having the largest number of packets in the output buffer (i.e. the largest contributor of the congestion). For the evaluation of our flow selection policies, we reuse the multimedia flow setup from subsection V-D but add two more MP3 streams (flow #11 and flow #12). Initially, only the flows #12, #11 and #10 are in the Drop Window. Flow #3 and flow #10 are the highest bandwidth flows (MPEG streams). Obviously, this setup creates high overload, which cannot be solved by the two small-bandwidth MP3 entries. Therefore, the QoS device has to select a further Drop Window entry.

The left-hand side of figure 12 presents the packet loss rate for using the "Add Random" policy. As shown, this policy still does not entirely solve the problem: even for an output

buffer size of $1.6*10^6$ bits, there are still Guaranteed Area losses of up to 10%. The problem of this policy is that selecting a random flow may only add another small-bandwidth flow to the Drop Window. In this case, the benefit remains small.

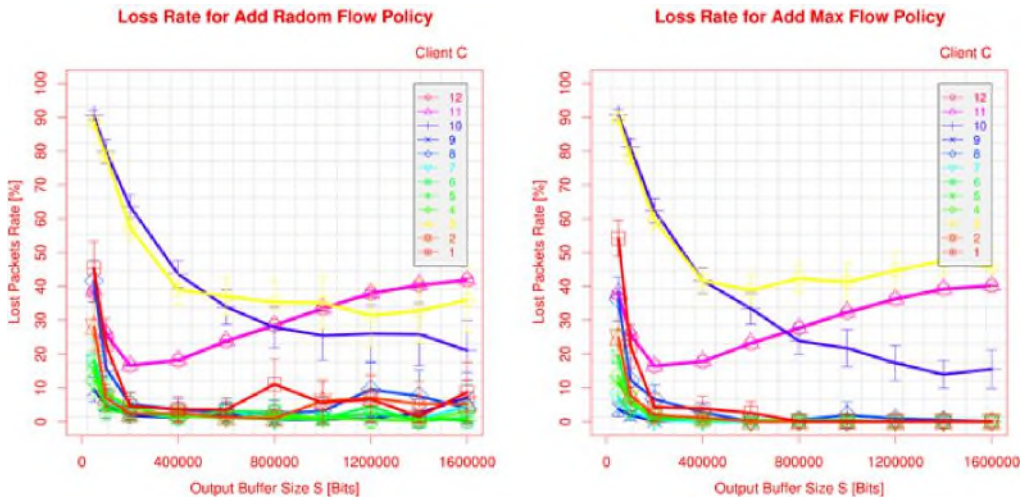

Figure 11. Flow Selection Policies

On the other hand, the "Add Max Packets" policy achieves the desired result (see the right-hand side of figure 12): it selects the high-bandwidth flow #3 for focused discard. After that, the Drop Window size is large enough to avoid Guaranteed Area losses.

6. Conclusions

In this article, we have described our QoS device approach to handle multimedia flow congestion on bottleneck links to broadband customers: instead of dropping packets of all streams indiscriminately (which leads to reduced QoS for all users), our device focuses packet discard on certain target flows. Since the device only has to store these selected flow identities and only requires minimal signalling overhead (i.e. the transmission of a Start Packet), it is – unlike classical IntServ approaches – very scalable. In our simulative performance evaluation, we have presented the general behaviour of the device for multimedia scenarios: using reasonable settings for output buffer size, Flow Register and policies, a significant performance benefit can be achieved for the users.

As part of future work, more analyses of the device parameters are necessary. In particular, it is useful to develop an algorithm to automatically adapt the parameters to the current flow scenario and changing needs of the customers. Furthermore, we are going to perform more codec-specific evaluations of the QoS: instead of only observing packet loss and delay, it is interesting to analyse the implications to the user's perceptual quality (e.g. metrics like PEAQ and PEVQ). Next to the simulative evaluation, we also intend to realize the QoS device functionality as a queuing discipline (QDisc) for Linux, in order to perform lab experiments using real-world applications.

Acknowledgements

The authors would like to thank Avril IJsselmuiden and John L. Adams for their support of the project. Furthermore, the authors would like to thank Jens Radeck and Stefan Monhof for reformatting the paper.

References

[1] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification," IETF, Standards Track RFC 2205, Sept. 1997.
[2] J. Wroclawski, "The Use of RSVP with IETF Integrated Services," IETF, Standards Track RFC 2210, Sept. 1997.
[3] 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.
[4] W. Zhu, T. Dreibholz, E. P. Rathgeb, and X. Zhou, "A Scalable QoS Device for Broadband Access to Multimedia Services," in Proceedings of the IEEE International Conference on Future Generation Communication and Networking (FGCN), Sanya, Hainan/People's Republic of China, Dec. 2008.
[5] W. Zhu, T. Dreibholz, and E. P. Rathgeb, "Analysis and Evaluation of a Scalable QoS Device for Broadband Access to Multimedia Services," in Proceedings of the 33rd IEEE Conference on Local Computer Networks (LCN), Montreal/Canada, Oct. 2008, pp. 504–505, ISBN 978-1-4244-2413-9.
[6] W. Zhu, "Analysis and Evaluation of a Scalable QoS Device for Broadband Access to Multimedia Services," Master's thesis, University of Duisburg-Essen, Institute for Experimental Mathematics, Mar. 2008.
[7] 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, ISBN 0-7695-2421-4.
[8] J. L. Adams, A. IJsselmuiden, and L. Roberts, "An advanced QoS protocol for real-time content over the internet," in Proceedings of the 13th International Workshop on Quality of Service (IWQoS), Passau/Germany, June 2005, pp. 164–177.
[9] T. Dreibholz, A. IJsselmuiden, and J. L. Adams, "Simulation of an advanced QoS protocol for mass content," in Second International Conference on Performance Modelling and Evaluation of Heterogeneous Networks (HET-NET), Ikley, West Yorkshire/United Kingdom, July 2004.
[10] 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, ISBN 0-7803-7661-7.
[11] J. L. Adams and A. J. Smith, "A New QoS Mechanism for Mass-Market Broadband," IETF, Individual Submission, Internet-Draft Version 00, Dec. 2001, draft-adams-qos-broadband-00.txt.
[12] A. J. Smith and J. L. Adams, "Packet discard control for broadband services," British Telecom, Tech. Rep. EP 01 30 5209, June 2001.
[13] A. IJsselmuiden and J. L. Adams, "Delivering QoS from remote content providers," ETSI, British Telecom Contribution TISPAN 01(03)TD132, Sept. 2003.
[14] A. IJsselmuiden and J. L. Adams, "Description of the QoS device," ITU-T, Contribution , Dec. 2004.
[15] 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.
[16] A. IJsselmuiden and J. L. Adams, "Proposal for a new IP transfer capability and future QoS studies," ITU-T, Tech. Rep. D-,Q4,6,10,11,16,SG13, Feb. 2004.
[17] 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.
[18] S. Floyd and V. Jacobsen, "Random early detection gateways for congestion avoidance," IEEE/ACM Transactions on Networking, vol. 1, no. 4, pp. 397–413, 1993.
[19] 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, ISBN 0-8186-7292-7.

[20] T. Dreibholz, "An IPv4 Flowlabel Option," IETF, Individual Submission, Internet-Draft Version 09, Jan. 2009, draft-dreibholz-ipv4-flowlabel-09.txt, work in progress.
[21] L. G. Roberts, "The Next Generation of IP – Flow Routing," in Proceedings of the International Conference on Advances in Infrastructure for e-Business, e-Education, e-Science, e-Medicine on the Internet 2003S International Conference, L'Aquila/Italy, July 2003, pp. 25–.
[22] T. Dreibholz and E. P. Rathgeb, "Towards the Future Internet – An Overview of Challenges and Solutions in Research and Standardization," in Proceedings of the 2nd GI/ITG KuVS Workshop on the Future Internet, Karlsruhe/Germany, Nov. 2008.
[23] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications," IETF, Standards Track RFC 3550, July 2003.
[24] J. Rajahalme, A. Conta, B. Carpenter, and S. Deering, "IPv6 Flow Label Specification," IETF, Standards Track RFC 3697, Mar. 2004.
[25] C. Partridge, "Using the Flow Label Field in IPv6," IETF, Informational RFC 1809, June 1995.
[26] K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," IETF, Standards Track RFC 3168, Sept. 2001.
[27] R. Stewart, "Stream Control Transmission Protocol," IETF, Standards Track RFC 4960, Sept. 2007.
[28] S. Floyd and E. Kohler, "Profile for Datagram Congestion Control Protocol (DCCP) Congestion Control ID 2: TCP-like Congestion Control," IETF, Standards Track RFC 4341, Mar. 2006.
[29] J. Postel, "Transmission Control Protocol," IETF, Standards Track RFC 793, Sept. 1981.
[30] A. Varga, OMNeT++ Discrete Event Simulation System User Manual - Version 3.2, Technical University of Budapest/Hungary, Mar. 2005.
[31] T. Dreibholz, X. Zhou, and E. P. Rathgeb, "SimProcTC – The Design and Realization of a Powerful Tool-Chain for OMNeT++ Simulations," in Proceedings of the 2nd ACM/ICST OMNeT++ Workshop, Rome/Italy, Mar. 2009, ISBN 978-963-9799-45-5.
[32] 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, ISBN 978-963-9799-20-2.
[33] University of W¨urzburg, "MPEG-1 Traces Archive," 1995.
[34] A. Richards, G. Rogers, M. Antoniades, and V. Witana, "Mapping User Level QoS from a Single Parameter," in Proceedings of the 2nd International Conference on Multimedia Networks and Services (MMNS), Versailles/France, 1998.
[35] R Development Core Team, R: A language and environment for statistical computing, R Foundation for Statistical Computing, Vienna/Austria, 2005, ISBN 3-900051-07-0.

Authors

Wenyu Zhu was born in Chengdu, China in 1981. He graduated from Tianjin University in 2003 with a Bachelor Degree in Computer Science and Technology. From 2003 to 2005, he was the assistant engineer in charge of network construction and administration at Chengdu Ouqi Technology Co.Ltd.. In 2008, he finished his study in Computer Science and Communication Engineering at the University of Duisburg-Essen in Duisburg, Germany and received his Masters degree for his thesis "Analysis and Evaluation of a Scalable QoS Device for Broadband Access to Multimedia Services".

Currently, he is responsible for the development of the project "MetalOne" at ATS AG in Oberhausen, Germany.

Thomas Dreibholz was born in Bergneustadt, Germany in 1976. He studied Computer Science at the University of Bonn, Germany and received his Diploma (Dipl.-Inform.) degree in 2001 for his thesis "Management of Layered Variable Bitrate Multimedia Streams over DiffServ with Apriori Knowledge". In 2007, he received his Ph.D. degree from the University of Duisburg-Essen for his thesis "Reliable Server Pooling – Evaluation, Optimization and Extension of a Novel IETF Architecture". Since 2001, he is a member of the scientific staff in the Computer Networking Technology group at the Institute for Experimental Mathematics, University of Duisburg-Essen, Germany.

Currently, his main research topic is Reliable Server Pooling (RSerPool). He is not only the author of various research papers – at international conferences and in journals – on this subject, but he also realized the first implementation of the RSerPool standard as part of a research cooperation project with Siemens, Munich, Germany. Furthermore, he contributed multiple Working Group and Individual Submission Drafts to the IETF RSerPool Working Group's standardization process. He is also coauthor of multiple RFC documents published by the IETF.

His research interests also include the Stream Control Transmission Protocol (SCTP), Quality of Service (QoS) and network security as well as concepts and protocols for the Future Internet.

Erwin P. Rathgeb was born in Ulm, Germany in 1958. He received his diploma and Ph.D. degrees in Electrical Engineering from the University of Stuttgart, Germany in 1985 and 1991, respectively. From 1985 to 1990, he was member of the scientific staff at the Institute of Communication Networks and Computer Engineering (Prof. Paul J. Kühn) at the University of Stuttgart where he was head of a research group on design and analysis of distributed systems.

From 1990 to 1991, he was a member of technical staff at Bellcore, Morristown, New Jersey, U.S.A., before joining Bosch Telekom in Backnang, Germany. In 1993, he joined Siemens in Munich, Germany. He contributed to concepts for commercial ATM nodes and ATM-based multiservice networks in a variety of positions in systems engineering and product planning. Since January 1999, he has held the Alfried Krupp von Bohlen und Halbach-Chair for "Computer Networking Technology" at the Institute for Experimental Mathematics, University of Duisburg-Essen, Germany. He is the author of a book on ATM and has published more than 50 papers in journals and at international conferences. Professor Rathgeb is a senior member of IEEE and a member of GI, IFIP and ITG where he is chairman of the expert group on network security.

His current research interests include network security as well as concepts and protocols for next generation internets, in particular the Stream Control Transmission Protocol (SCTP) and Reliable Server Pooling (RSerPool).

Xing Zhou was born in Chongqing, China in 1958 and is a full professor for the Computer Science Department at Hainan University. She received her Bachelor and Master degrees from the Electrical Engineering Department of Chongqing University in 1982 and 1988 respectively. From 1982 to 1992, she was a lecturer of the scientific staff at Yuzhou University of Chongqing. Since February 1992, she has been the scientific staff of Hainan University and primarily engaged in teaching and scientific research. From September 2001 to January 2003, she pursued research on computer network and information security technology as an advanced visiting scholar at Computer Science Department of Shanghai Jiaotong University. From September 2006 to September 2007, she undertook research on Reliable Server Pooling as a visiting scholar at the Institute for Experimental Mathematics, University of Duisburg-Essen, Germany. She is an author of 6 books and has published more than 40 articles in Journals and at international conferences as well as received the Award of Science and Technology Progress in Hainan province.

Her current research interests include network security, the protocols for next generation internet and computer application, in particular, Reliable Server Pooling and security concepts for the Reliable Server Pooling framework.