INTEGRATION OF A TRAFFIC CONDITIONER FOR DIFFERENTIATED SERVICES IN END-SYSTEMS VIA FEEDBACK-LOOPS

M. Bechler, H. Ritter, J. Schiller
University of Karlsruhe (TH), Institute of Telematics
Zirkel 2, 76128 Karlsruhe, Germany
[mbechler, ritter, schiller]@telematik.informatik.uni-karlsruhe.de

Abstract More and more applications on the Internet would benefit from Quality of Service provisioning. Unfortunately, the IP-based Internet works according to a best-effort model and gives no QoS guarantees. Thus, the (compared to Integrated Services) simple Differentiated Services architecture has been developed to provide a service better than best-effort. This paper presents components of the Differentiated Services architecture and shows the benefits of local pre-marking and traffic conditioning. This traffic conditioning is combined with a feedback architecture and a simple user interface where a user can express his or her request for more QoS via a simple button. The traffic is formed in a way that QoS is supported and no packet is discarded or is downgraded at the next domain. First results demonstrate the benefits of the approach in a real testbed.

Keywords: Differentiated Services, Operating Systems, QoS, Feedback Mechanisms

1. INTRODUCTION

As more and more people do business on the Internet, many multimedia applications have been introduced, and people pay for network access, it was soon discovered that the traditional best effort packet switching is not sufficient any longer to provide satisfying Quality of Service (QoS). While in the ATM-world an elaborated scheme for QoS provisioning exists (at the expense of higher complexity), the IP-based Internet in its original design lacks QoS support. Thus, the Integrated Services architecture [1] (with the resource reservation protocol RSVP) has been developed to provide "soft" guarantees for traffic streams (so-called flows) across the Internet. However, it was soon discovered that this model has only a limited scalability as all intermediate systems (the IP routers) have to store and manage state per traffic flow. As the Internet community did not want to adopt ideas of signaling, connections, and resource reservation as done in ATM (which would provide QoS), a very simple architecture, Differen-

tiated Services (DS), has been developed. DS does neither use signaling nor does it store per flow state in intermediate systems. The basic idea of DS is the aggregation of traffic streams with a common identifier and the per hop treatment of data packets. Intermediate systems receive a packet, check a single identifier (or few fields), and forward the packet. All packets carrying the same identifiers are treated identically, no matter what source or destination addresses they contain. This local view on packet forwarding makes the approach scalable compared to per flow treatment.

The following section gives a short introduction of the DS approach and focuses on its architectural elements [2, 3]. The third section proposes two different types of DS enabled end-systems, a simpler system that is able to mark packets according to a traffic contract between the sender and a service provider, and a more complex system that is able to shape traffic in a way that outgoing traffic never violates the traffic contract. Section 4 discusses our overall architecture of a DS capable end-system with easy-to-use user interaction via a Q(uality) button in the user interface, while the 5th section presents the elements of our system which condition traffic according to profiles. Finally, section 6 explains the implementation and test scenarios, and presents first results demonstrating the behavior of our implementation.

2. DIFFERENTIATED SERVICES

Primary goal of the DS approach is to provide a simple, efficient, and thus scalable mechanism that allows for better than best effort services in the Internet [2, 3]. DS does not perform any reservation of resources as needed in, e.g., Integrated Services or ATM, nor does it differentiate traffic based on single applications or flows in each networking element (typically a router in IP networks, a switch in ATM networks). In DS, a user has a service agreement with, e.g., an ISP (Internet Service Provider), the ISP gives service guarantees based on aggregates of traffic. For example, the ISP could offer 2 Mbit/s Premium Service [4, 5] and up to 5 Mbit/s total bandwidth to the customer. As long as the user stays within 2 Mbit/s with Premium Service traffic, the ISP guarantees special treatment. Premium Service traffic exceeding 2 Mbit/s will be dropped. The agreements between a user and an ISP or between ISPs are rather of long-term nature and reflect the typical traffic characteristics and not the current requirements of a single short-living connection. The special treatment of different types of traffic is based on the PHB (Per Hop Behavior) of all nodes within a DS domain. Each node only checks the DS field (which carries the DS code point) and applies simple rules (e.g., remarking the packet, inserting into special queues) before forwarding. Several PHBs have been proposed, e.g., “Expedited Forwarding” [4] and “Assured Forwarding” [6]. For the DS field the TOS field in IPv4 or the Traffic Class in IPv6 is used respectively. Using a single DS field allows for a scalable service discrimination without per-flow state and signaling at every hop. Nodes only perform a simple mapping of the DS code point to a local behavior (PHB), there are no rules how to achieve a certain end-to-end service for users by setting the DS field to a certain value.

The DS architecture defines several architectural elements [3]: A DS domain consists of DS interior nodes and DS boundary nodes. DS boundary nodes build the border of a DS domain and can be ingress nodes (handling traffic entering the DS domain) or egress nodes (handling traffic leaving the DS domain). DS boundary nodes connect a DS domain to a node in another DS domain or to a non-DS-compliant node. DS boundary nodes classify and possibly condition ingress traffic in addition to the application of the PHB. Traffic conditioning of ingress traffic is based on the TCA (Traffic Conditioning Agreement) and may comprise metering, shaping, policing, and remarking of the DS field. The DS boundary node may be an ingress router of an ISP or may be co-located with a host generating traffic. Only DS boundary nodes perform classification and conditioning, DS interior nodes only apply the appropriate PHB based on the DS field. Thus, scalability is achieved by shifting all complexity to some boundary nodes.

The classification of traffic in a DS boundary node can be based on two different classifiers: the BA (Behavior Aggregate) classifies traffic based on the DS field only. The MF (Multi Field) classifier may use several fields of the packet such as source address, destination address, DS field, protocol type, ports etc. Now assume the following scenario: a user has three applications running and decides to favor one of them (i.e., the user wants to have a better quality for this application, e.g., higher bandwidth, lower delay). How can this application on a host get a better forwarding behavior? After aggregation the DS domain cannot distinguish different applications or different hosts. But also classification at a boundary node (which is not the sending host) based only on packet fields is problematic. Users do not know in advance which application may be more important as this often depends on the content. Consider, for example, a file upload using ftp – one file could be time-critical business data, the other file just some updates of manuals. Although using the same program from the same host, both packet streams have a different importance for the user and should be treated differently. Thus, nodes besides the host cannot classify packets according to their importance and the use of MF for classification is useless in this case. Signaling the “importance” of traffic originating from single applications as done in RSVP or ATM (via resource reservation) is no option due to the well-known scalability and complexity problems.

RFC2475 states that traffic sources may perform traffic classification and conditioning. Traffic originating from the source domain across a boundary may be marked by the traffic sources directly or by intermediate nodes before leaving the domain which is called initial marking or pre-marking. The advantage of this initial marking is that the traffic source can more easily take an application’s preferences into account when deciding which packets should receive better forwarding treatment. This classification of packets is much simpler compared to classification of aggregated traffic (e.g., aggregated traffic from all hosts of a company) as less classification rules are needed. The host can now set the bits in the DS field according to the local importance of an application and according to the TCA with an ISP. The host can ensure the local conformance of its traffic to the TCA to avoid packet loss or remarking due to a TCA viola-

tion. So-called bandwidth brokers may support mid-term changes in QoS requirements or the distribution of resources within administrative domains.

3. END-SYSTEMS FOR DIFFERENTIATED SERVICES

A host performing initial marking is inside an own DS domain. Thus the boundary node of the DS domain is co-located with a host generating traffic. Two approaches of such hosts, realizing different functions of traffic conditioning, are presented in this paper.

3.1 The marking approach

A simple approach to integrate service differentiation into the end-system is to implement the function of packet marking without the knowledge of the TCA. This simple approach will be further called the marking approach. The end-system does not perform any traffic shaping or policing. Therefore, the DS node closest to the end-system acts as a DS boundary node [3]. This DS boundary node is responsible for ensuring that the traffic generated by this end-system conforms to the appropriate TCA. The complete set of traffic conditioning elements as described above (metering, shaping, policing and/or remarking) may be performed on the aggregated traffic coming from the end system.

The data packets of an application can be marked according to the service level currently needed. If an application is started, it will use a default service, in most cases the best effort service. If this service is not sufficient for the application to perform well, the packets can be marked as assured service or even premium service packets. In general, best effort packets are more likely to be discarded than packets marked as assured packets, respectively suffer more delay than premium service packets. Therefore, marking packets as assured service packets will in most cases result in a better throughput of the respective application.

Due to traffic conditioning in the ingress boundary node there is no guarantee for packets marked by the end-system to keep their DS field. If the end-system marks packets for example as premium service packets, the ingress boundary node only passes the packets not exceeding the data rates specified in the TCA. Out-of-TCA packets are discarded. This can be seen as a disadvantage, because traffic generated in the end-system may be discarded at the next hop. Therefore, the end-system should only mark packets with the premium service DS code point if an application really needs this service; an example is a telephony application requiring a minimum amount of guaranteed rate.

Considering assured service packets, the situation differs. If assured service packets exceed the TCA, they are not discarded, only their DS field is cleared and they will be handled further as best effort packets. Due to this behavior, the application generating assured service packets uses the maximum amount of assured service without suffering from packet loss at the ingress boundary node. The main advantage of this approach is its simplicity. There is no need for any-

thing else than a simple packet marker, resulting in a small, CPU-saving code especially suited for small mobile devices.

3.2 The shaping approach

Nevertheless, if the end-system provides additional functions like shaping based on the TCA, this has major advantages: In the case of premium service, the end-system can inform the applications if the requested rate of premium service traffic is supported. Furthermore, the traffic generated by the end-system will most likely not be discarded at the next ingress boundary node, as the traffic already conforms to the TCA. The end-system can also adapt to the available, probably very small bandwidth on the link up to the next boundary node. This more sophisticated approach realizing TCA-aware shaping, is further on called shaping approach. The end-system acts as DS boundary node for traffic from applications running on that host and monitors the conformance to the TCA, i.e., it performs both marking and shaping.

Realizing the shaping approach, the end-system must be aware of the TCA and the rules of classification and conditioning defined by the TCA. This is the main difference to the marking approach described before. The traffic generated by the end-system is therefore more likely not to be discarded at the next hop, and thus the goodput of the end-system can be increased. However, the realization of the end-system as a DS boundary node requires additional functionality and thus complexity compared to the simple marking approach.

4. USERS, FEEDBACK, AND HIERARCHICAL SCHEDULERS

As mentioned above, the classification of traffic flows of different applications in the end-system by the boundary node is quite problematic if it is not the sending host, because the boundary node does not know the relative importance of certain applications. The proposed marking and shaping approaches avoid this problem by performing the traffic conditioning in the end system, where the application's preference can be more easily taken into account. Nevertheless, the user preferences regarding an application can change rapidly depending on the content or a change of the user's focus.

Therefore, the user needs an interface for expressing his or her current preferences. A simple button related to each application is sufficient to express the demand for a better performance of the application. On top of Figure 2 an example of this simple button is shown. A button labeled with the letter Q (denoting Quality of Service) is integrated into the window title bar of all applications running on the system. The user expresses the current preference for an application by clicking on the Q-Button. This approach might be implemented in any windowing system (e.g., X Windows as well as Windows CE) independent of the application. The operating system is provided via a system call with the basic information which comprises the user request of a somehow "better" performance of the application running in the respective window.

This information influences the sharing of resources in favor of the related application. In a first step, the resources “CPU time” and “network bandwidth” are taken into account. The resulting feedback architecture is shown in Figure 1. The upper feedback loop is constituted by the process scheduler. Clicking on the Q-Button influences the share of computation time in favor of the related application. In the simple case of a multimedia application this might be enough to improve its performance, e.g., given a MPEG player requiring more computation time for encoding. Feedback to the user is therefore provided via the visible improvement of the application’s performance. This is described in more detail in [7].

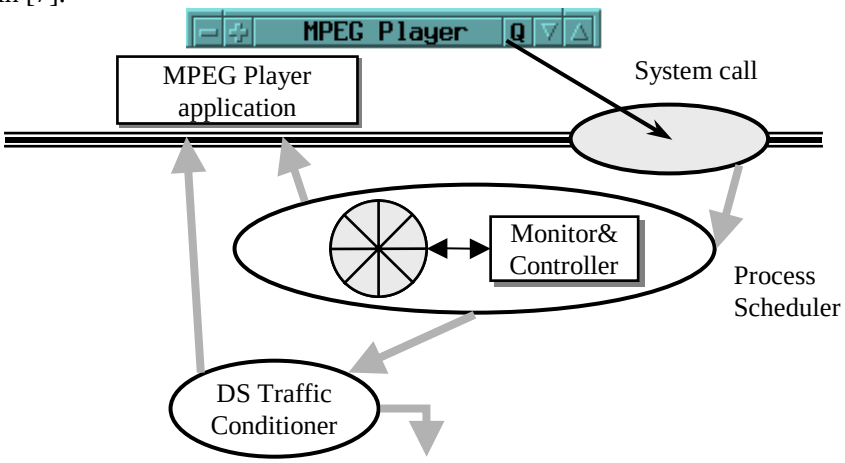

Figure 1 Integration of the DS Traffic Conditioner in the end system.

The focus of the research described in this paper is on the lower feedback loop constituted by the DS traffic conditioner. The DS traffic conditioner realizes the functions of packet marking and/or shaping. The information about the user’s preference (the click on the Q-Button) is passed down from the monitor&control-entity of the process scheduler to the traffic conditioner. The elements of the DS traffic conditioner are described in the next section. Feedback to the user is provided by a better performance of applications requiring more bandwidth or a better service level.

5. TRAFFIC CONDITIONER IN END SYSTEMS

In order to integrate the functionality of service differentiation in an end system, the operating system has to be extended by a DS Traffic Conditioner. The main component is the (generic) traffic conditioner (adapted from [8]), that realizes an adaptive resource manager for scheduling the data packets. The consideration of user preferences is realized by a profile that contains the DS parameters that are needed if the application sends data. For every (running) application such a profile is implicitly specified. If a user requests a better transmission

quality for an application, a profile manager is responsible for modifying the DS parameters in the profile of this application.

These three components are sufficient for realizing the marking approach described above. As an example, the profile manager modifies the profile of an application to send packets using premium service. Then, the traffic conditioner marks the DS field of the packets generated by this application with the corresponding values. Note that this approach does not require shaping functionality as it is the job of the boundary node to perform the complete set of traffic conditioning functions according to the TCA. The traffic conditioner in the end-system just sets the DS field in this marking approach, independent of the TCA.

The realization of the more interesting shaping approach requires a conditioner that takes the TCA into account. The traffic conditioner is additionally responsible for shaping the packet stream that leaves the network interface, conforming to the TCA, i.e., the functionality of a boundary node is realized in the end system. The traffic conditioner has to be adaptive, as the sharing of a service has to consider the dynamically varying user's preferences on the one hand, and the restrictions determined by the TCA on the other hand. Therefore, the traffic conditioner works in two steps:

1. First, the DS field of each packet is marked according to the requirements stored in the profile of each application. In contrast to the marking approach, the profile for a connection has to be more extensive as more information about the transmission characteristics is needed. Note that the content of the profile does not depend on the TCA.
2. In the second step, traffic conditioning is performed in order to fulfill the TCA. Therefore, the requirements of the applications are adapted to the available resources of the service.

The strategy for traffic conditioning depends on the algorithm that is used. Mainly, two algorithms are conceivable: Priority Based Sharing and Proportional Fair Sharing. Using the Priority Based Sharing, an order of the applications will be established that expresses the priority of each application. The requirements of the applications for a service are then satisfied in turn, depending on the priority.

Using Proportional Fair Sharing leads to an even degradation of the requirements of the applications competing for a service. Therefore, the resource of the service is proportionally shared between the competing applications. A short example shows this in more detail. Assume, there are two applications A1 and A2 that like to send data using premium service. The data rate specified in the profile for A1 is 8 Mbit/s, for A2 it is 4 MBit/s. The specified premium service (in the TCA) is only 9 MBit/s, and thus the requirements of A1 and A2 (12 MBit/s together) cannot be met. Therefore, a degradation of A1 and A2 will be performed by the traffic conditioner to conform to the TCA. The Proportional Fair Sharing reduces the rate of A1 to 6 MBit/s, the rate of A2 to 3 MBit/s, respectively, in order to distribute the resources proportionally.

However, it is quite difficult to decide what a request for a better transmission quality really means. An improvement can be reached, e.g., by extending the share of the current service, but also by using a better service with the original requirements. Thus, the decision for the best conditioning strategy mainly

depends on the requirements of the applications. We implemented the Proportional Fair Sharing algorithm as it is more flexible compared to the Priority Based Sharing. It is intuitive for a user, that the sending rate of an application using a service will be degraded, if another application uses this service for transmission. In this case, the user has the possibility to click several times on the Q-Button until the wanted characteristics of the service is reached.

6. IMPLEMENTATION

The integration of DS functionality in an end-system requires changes in the kernel of the operating system. Especially for the transmission of data, modifications on the IP stack are necessary to set the DS field in the header of the outgoing IP packets. Therefore, we use Linux as the platform for implementation of the marking and the shaping approach.

The notification of the DS Traffic Conditioner to achieve a better characteristic for data sent by an application was discussed in chapter 4. This characteristic is stored in the profile that exists for every application running on the system. A profile comprises the following components:

- *service:* Specifies the service that should be used for sending data packets. This could be (currently) the mentioned premium service, assured service, or best effort service.
- *send_priority:* This parameter specifies the current priority of an application for sending data. Using this value, the share of the service for this application is calculated to define the sequence of the outgoing packets.
- *last_send_time:* The time- stamp of the last packet t7he application transmitted is stored in this parameter. This value is also needed to calculate the sequence of the outgoing packets.

The marking approach only uses the service entry of the profile as it just sets the DS field of all data packets generated by an application depending on this value. Therefore, the traffic conditioner identifies the application that sent the data, fetches the service of the profile, sets the DS field, and computes the checksum of the IP header to generate a valid IP packet.

The more sophisticated shaping approach also sets the DS field in the IP header as mentioned above. In a further step the time stamp for transmitting the packet is computed, based on the *send_priority* and the *last_send_time* parameters with the following formula:

$$next\_send\_time = last\_send\_time + length_{packet} \cdot \frac{rate_{CPU} \cdot \sum send\_priority}{rate_{service} \cdot send\_priority}$$

where $length_{packet}$ is the length of the current packet, $rate_{CPU}$ the speed of the CPU in the end system, and $rate_{service}$ the specified rate of the service. The sum of the *send_priorities* covers the *send_priority* of every application that currently sends data. The traffic conditioner is then responsible for sending the packets at the correct time.

In order to show the feasibility of the approach and to gain experience with the handling of a DS-capable end-system we set up measurements using the

marking and the shaping approach. The tests and measurements were performed using the DS implementation of the UNIQuE environment [9].

Marking approach: Clicking the Q-Button results in both approaches in a better service for the related application. In the marking approach applications are thus shifted from best effort to assured service and finally up to premium service. Pressing the Q-Button in favor of an application already using premium service nevertheless does not change the service class, because there is currently no "better" service class defined. Using the marking approach the user can not further influence the bandwidth share of two applications using premium service. The fraction of totally available bandwidth an application gets is equal to its fraction of the total bandwidth demand. Given a link of 1 Mbit/s reserved for premium service and applications demanding 0.7 Mbit/s (application A) and 0.466 Mbit/s (application B) respectively, this results in an *average* share of 60% of the totally available bandwidth for application A and an *average* share of 40% for application B. This behavior is a result of the premium service queuing discipline realized in the boundary node closest to the end system.

Shaping approach: Using the shaping approach, the user can influence the bandwidth share of two applications using the same service. The fraction of totally available bandwidth an application gets is equal to the relation of the current user priorities. The user priorities are stored in the profile of each application as described above. Given again a link of 1 Mbit/s reserved for premium service and applications demanding 0.7 Mbit/s (application A) and 0.466 Mbit/s (application B) respectively, the share depends now on the relation of the user priorities for application A and B. Figure 2 shows a scenario with application A beginning to send 0.7 Mbit/s at 0 s and application B starting at 10 s. Given equal user priorities for both applications, the shaper limits the throughput of application A to 0.6 Mbit/s and that of application B to 0.4 Mbit/s as in the marking approach. After 20 s the user prioritizes application A by clicking the Q button and thus rises the throughput to 0.65 Mbit/s. The throughput of application B drops to 0.35 Mbit/s. Given limited bandwidth, this partitioning perfectly reflects the current user preference, expressed by clicking on a simple button.

7. CONCLUSION

The future Internet will need some kind of QoS support. As neither protocols nor applications currently integrate mechanisms for resource reservation and the Internet should remain very simple in its architecture, the Differentiated Services approach claims to introduce at least some better QoS compared to the current best-effort model into the Internet. While typically routers at the edge of administrative domains act as boundary nodes in the DS architecture to protect the domain, we believe in the benefits of hosts acting as additional boundary nodes which already pre-mark packets according to application and user demands, and perform traffic conditioning to avoid packet downgrading or discarding.

Future work comprises more detailed evaluation of the approach in our UNIQuE [9] testbed as well as the integration of the approach in mobile and

wireless scenarios. Particularly these environments could benefit from pre-conditioning as it makes no sense at all to transmit packets via the bandwidth limited air interface only to drop them at a boundary router due to the violation of a traffic contract.

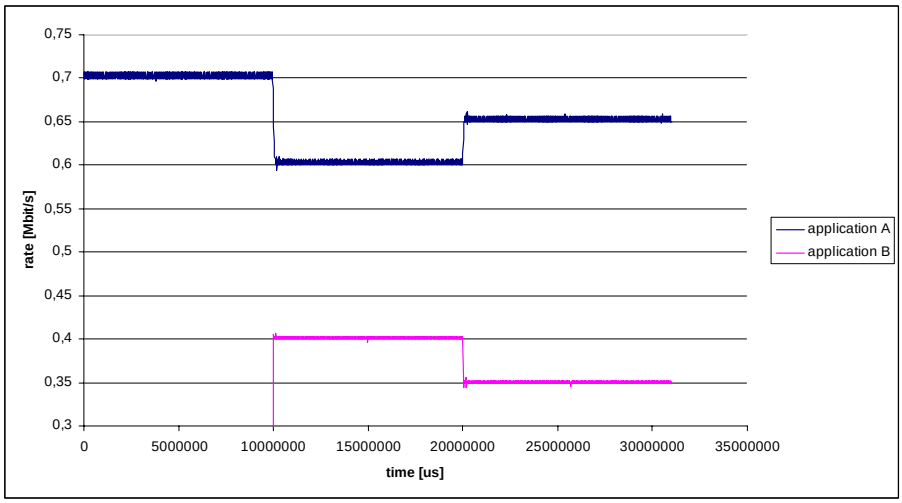

Figure 2 Example result of the shaping approach.

References

[1] R. Braden, D. Clark, S. Shenker: Integrated Services in the Internet Architecture: An Overview, RFC 1633, 1984.

[2] K. Nichols, S. Blake, F. Baker, D. Black: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, RFC 2474, 1998.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss: An Architecture for Differentiated Services, RFC 2475, December 1998.

[4] V. Jacobson, K. Nichols, K. Poduri: An Expedited Forwarding PHB, Internet Draft, RFC2598, 1999.

[5] K. Nichols, V. Jacobson, L. Zhang: A Two-bit Differentiated Services Architecture for the Internet, Internet Draft, <draft-nichols-diff-svc-arch-00.txt>, November 1997.

[6] J. Heinanen, F. Baker, W. Weiss, J. Wroclawski: Assured Forwarding PHB Group, Internet Draft, RFC2597, 1999.

[7] J. Schiller: Feedback controlled scheduling for QoS in communication systems, IFIP Conference on High Performance Networking (HPN'98), Vienna, Austria, 1998.

[8] J. Schiller, P. Gunningberg: Feasibility of a Software-Based ATM Cell-Level Scheduler with Advanced Shaping, Broadband Communications '98, Stuttgart, 1998.

[9] UNIQuE, Universal Networking Infrastructure with QoS Enhancements, http://www.telematik.informatik.uni-karlsruhe.de/forschung/UNIQuE/.