

Simulations of a TCP rate controller in inter-domain scenarios*

Peter Dorfinger, Christof Brandauer
Salzburg Research,
Jakob Haringer Str. 5/III,
A - 5020 Salzburg, Austria
{peter.dorfinger,christof.brandauer}@salzburgresearch.at

Ulrich Hofmann
Fachhochschule Salzburg,
Schillerstrasse 30,
A - 5020 Salzburg, Austria
ulrich.hofmann@fh-sbg.ac.at

Abstract

In this paper a new mechanism for providing an assured rate to a long-lived TCP flow is evaluated. The mechanism is called TCP rate controller (TRC) and operates as a traffic conditioner at the ingress of a network. The TRC seeks to achieve the requested rate by imposing well directed drops and (artificial) delays on the flow's packets. The choice of drop probability and delay is based on an analytical model of TCP sending behavior. The requirements on downstream domains are highlighted. It is shown in a simulation study that the TRC performs well in inter-domain scenarios.

Keywords: *Quality of Service, TCP rate control, TCP rate assurance, inter-domain*

1 Introduction

In the recent years there has been a growing interest in IP Quality of Service (QoS). New applications that have high requirements on network performance are being developed. Some of those applications, for example video conferencing or IP telephony, require some minimum network quality to be useful at all. From the operator's point of view it is hoped that services that deliver high QoS can be profitable.

Currently, a lot of research work is based on the paradigm of Differentiated Services (DiffServ) [8, 1]. DiffServ seeks to provide QoS in IP networks in a simple and scalable fashion. It is tried to remove complex tasks from the core and shift them to the edges of the network instead. As an example, traffic controllers that operate on single flows are only acceptable at the ingress/egress of a DiffServ network.

The focus of this paper is on the topic of rate assurance for TCP flows in inter-domain networks. Given that TCP is the number one transport protocol [4] in today's Internet,

*This work was partly funded by the IST project AQUILA by contract IST-1999-10077

we believe that such a service could be of interest. The topic of assuring TCP rates has been investigated in several other publications, e.g. [11, 3, 6].

The basis of this work is a traffic conditioning mechanism that can be used to assure a certain level of goodput to long-lived TCP flows. The proposed conditioner could be used in a DiffServ network to enable such a service class. We proposed a TCP rate controller (TRC) that regulates the goodput of a TCP flow by controlling packet drops and the round trip time (RTT) of the flow's packets. The TRC is based on a model of TCP sending behavior. The TRC is proposed in [5] where the intra-domain behavior is evaluated. The goal of this work is to specify the behavior of the TRC in inter-domain scenarios and to evaluate the behavior of the TRC in such an environment.

The remainder of this paper is organized as follows: Section 2 discusses related work. Section 3 summarizes the basic concept of the TRC and describes the essential network environment for the TRC. Inter-domain issues of the TRC are discussed in Section 4. Exemplary simulation results for inter-domain scenarios are presented in Section 5. Section 6 concludes the paper.

2 Related Work

The Capped Leaky Bucket (CLB) as proposed in [6] is an improved Leaky Bucket traffic conditioner. To take the behavior of TCP into account it is tried to estimate the RTT by measuring the time between two bursts. If the input rate is higher than the target rate one packet each two RTTs is marked as out-profile. Simulations in [6] show performance that is not appropriate to give assurances and a bias against big reservations.

In [11] equations how to set the parameters of a token bucket marker for achieving a requested rate are proposed. The parameter setting depends on the requested rate (R_{req}), drop probability of out-profile packets (p_2) and the RTT . With the equations in [11] it is possible to make correct goodput assumptions for a known value of RTT and p_2 .

But the crucial aspect in the application of this model is that the RTT and p_2 are not constant for different connections and also strongly vary over time due to changes in the level of congestion. This especially arises in wide-area networks where the traffic traverses several domains. In each domain queuing may arise, and thus the queuing delay is the determining factor of the RTT . In a recent work [2] we analyzed the applicability of the TBM algorithm, showing weaknesses of this approach.

The changing conditions in network (RTT, p_2) can not be captured by *static* parameter setting for TBM. An adaptive marking algorithm has been proposed in [3].

3 TRC in intra-domain Environment

The TRC controls the achieved rate of a TCP connection by well directed drops and artificial delay. When a user requests a rate an admission control algorithm is executed. If the request is for a class for long-lived TCP flows and accepted by the admission control the TRC is initialized. Based on a TCP model the TRC calculates drop probability (p) and RTT for the request. The delay at the TRC (d_{TRC}) is set to the RTT minus the network RTT (RTT_{net}). The TRC is placed at the network ingress of the flow. The TRC drops each $1/p$ 'th packet and delays all other packets by d_{TRC} . The TRC can be configured to be used in ECN or in drop mode. For the Slow Start phase a special algorithm is implemented to take the Slow Start behavior into account.

The TCP rate controller essentially requires some conditions from the network. The first aspect is that at the network ingress the TRC must exclusively control single long-lived TCP flows that are not multiplexed with a different kind of traffic (e.g. short-lived TCP flows or UDP flows) [12]. Second, an admission control framework must ensure that the sum of requested rates over all accepted reservations is in fact available. This condition provides for an over-provisioned service class. Further, the used code-point can be different in different domains. In the core network the traffic can be multiplexed with any kind of traffic, but it has to be ensured that the drop probability (p) is zero and the network RTT (RTT_{net}) is predictable and nearly constant.

The TRC does not need any special queue management mechanism, because no packet should be dropped in the network. It has to be ensured that a few packets can be buffered in the queue.

The receiver window has to be larger than the maximum congestion window (W_{max}) otherwise the achieved rate will be controlled by the receiver and not by the TRC. Further also TCP's Slow Start threshold ($ssthresh$) should be larger than W_{max} otherwise the performance during Slow Start will be worse.

The TRC has to be placed at the ingress point of the network. Only one TRC can be applied to one flow, because if two TRCs are working on the same flow two times the packets needed to control the flow are dropped.

It has to be ensured that from the receiver to the sender there is no congestion, because RTT_{net} is assumed to be constant. Consequently also ACKs have to be marked with the same code-point as used for packets from the sender to the receiver.

More details about the TRC can be found in [5]. It is shown by simulations, that for a known RTT the TRC is able to control the achieved rate of a TCP connection. Simulations are performed for a mixture of different requested rates and RTTs. All flows achieve the requested rates and confidence intervals are small. All simulation scenarios are intra-domain scenarios.

4 TRC in inter-domain Environment

The goal of this work is it to extend the approach of the TRC to an inter-domain environment. It is analyzed what is needed for the TRC to work in inter-domain scenarios.

In the TRC controlled domain at each ingress a TRC controls the rate of a single TCP flow. The TRC exclusively controls packet drops. By delaying packets for d_{TRC} which is based on RTT_{net} the achieved rate is controlled. The input traffic of the downstream domain is an aggregate of TRC controlled flows. Consequently the TRC could not be applied, because there are no single TCP flows. A TRC at the ingress of the downstream domain is not even needed because as soon as the packets pass the TRC TCP flows are already rate controlled. The TRC must exclusively control drops and RTT , thus it has to be ensured by the downstream domains that no packets are dropped and that the delay jitter is small. Especially in service classes where loss should be zero, the incoming traffic is strictly policed. For such service classes peak rate policers are often used. Traffic outside of the profile is dropped because for each forwarded packet the guarantees have to be given.

It is the task of the TRC controlled domain to bring the outgoing traffic to a form, such that no packets are dropped at the policer. Therefore we are using a traffic shaper at the egress of the TRC controlled domain, which shapes traffic to the bandwidth requested for this specific service class.

In the downstream domains TRC controlled traffic can be mixed with other traffic. A service class that ensures no loss and low jitter could be based on an EF PHB [7]. The TRC controlled traffic can be mixed with such without degrading its performance.

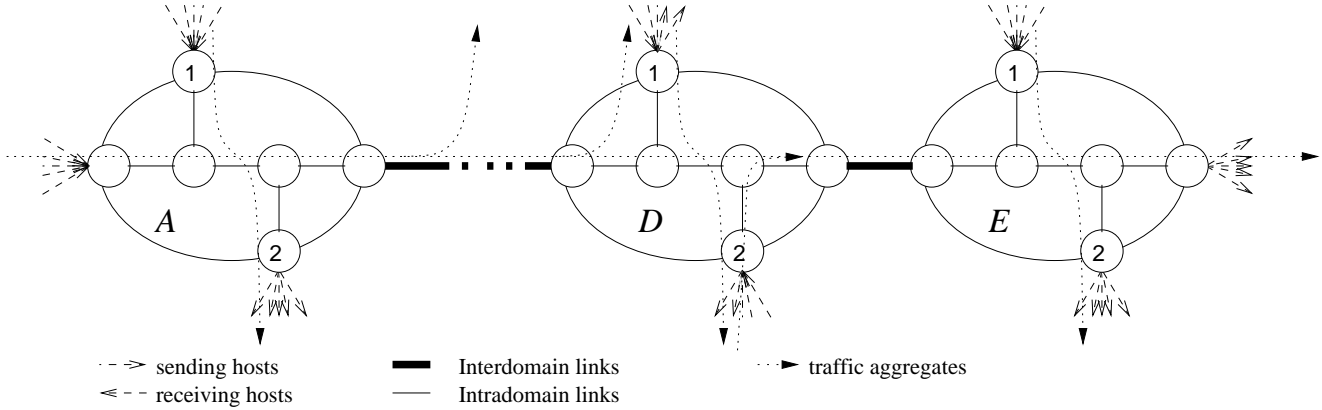


Figure 1. Simulation topology

5 Simulation study

5.1 Simulation Topology

The behavior of the TCP rate controller is studied by means of network simulations using `ns-2` [9]. Each domain is simulated as a domain with 6 routers as shown in Figure 1. The dotted lines indicate the path of a set of flows through the network. Each set consists of 50 FTP flows, where 20 flows request 50 kbps, and the other flows request 10x100 kbps, 10x500 kbps, and 10x1000 kbps respectively. FTP senders start sending data at a random point of time within the first 10 seconds and last the full simulation time. All routers are configured with a DropTail queue. Traffic conditioning is done by the TRC on a per-flow basis at each sending host (instead of at the ingress, because this is simpler for implementation and makes no difference in simulation results). Simulations last for 500 seconds and are repeated 100 times. All simulations are run in ECN and in drop mode and show equal results. Within one domain all links have the same bandwidth.

5.2 Simulation results for scenario with five domains

The performance of the TCP rate controller is evaluated in a scenario with five (A–E) domains. The intra-domain links of A–C, E have a bandwidth of 100 Mbps and the intra-domain links of D have a bandwidth of 50 Mbps. The link between A and B has a bandwidth of 60 Mbps, link from B to C 40 Mbps, link from C to D 20 Mbps and link from D to E 40 Mbps. In each domain a set of 50 flows as described above sends traffic from host 1 to host 2. One traffic set sends from A to B, one from A to C one from A to E and one from D to E. At each domain egress, traffic is shaped to the bandwidth of the inter-domain link. Each domain that

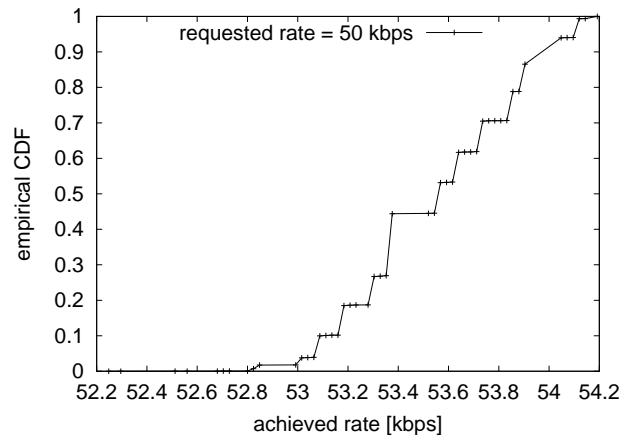


Figure 2. Requested rate 50 kbps

accepts traffic from an upstream domain employs a peak rate policer at the ingress.

Figures 2 - 5 are showing the empirical CDFs of the flows requesting 50 kbps - 1 Mbps. All flows achieve the requested rate. The difference between achieved rate and requested rate is small. Therefore a good service differentiation is possible.

5.3 Simulations with UDP traffic

In this section we demonstrate that TRC controlled TCP traffic can be mixed with UDP traffic without service degradation. Simulations are run in one domain of the topology shown in Figure 1. A set of TRC controlled traffic is mixed with CBR UDP traffic.

Figure 6 shows the empirical CDF of the achieved rate of flows requesting 1Mbps. All flows achieve the requested rate. The deviation between requested rate and achieved rate is small. The achieved rate of a flow mixed with UDP

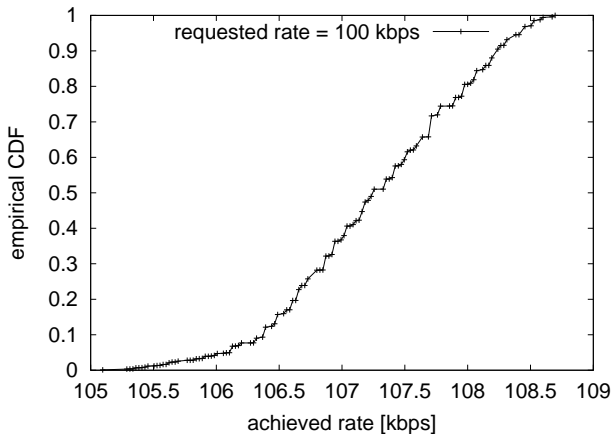


Figure 3. Requested rate 100 kbps

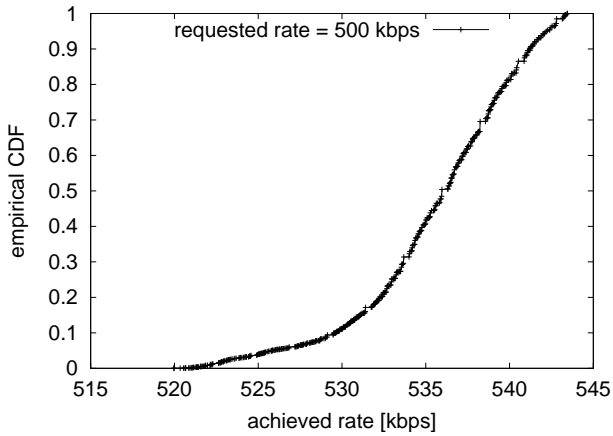


Figure 4. Requested rate 500 kbps

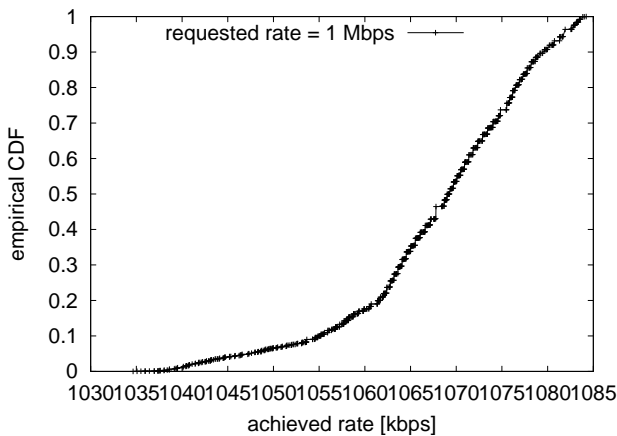


Figure 5. Requested rate 1 Mbps

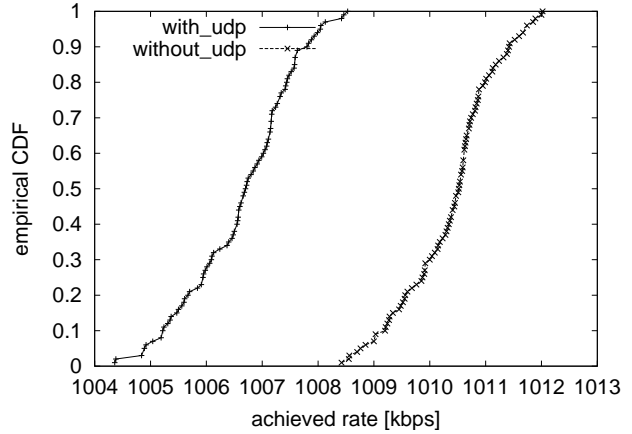


Figure 6. TCP traffic mixed with UDP traffic

traffic is about 0,4% lower than the achieved rate not mixed with UDP traffic. Results for flows requesting 50 kbps to 500 kbps show equivalent results.

6 Conclusion

In this paper the TCP rate controller (TRC) is evaluated in an multi-domain environment. The TRC seeks to achieve goodput assurances for long-lived TCP flows.

The task of the TRC is it to control the achieved goodput of a TCP connection by controlling a connections RTT and p . The TRC is based on a TCP model which predicts the sending behavior for known values of RTT and p . The idea of the TRC is it to fix RTT and p of a connection. Therefore the TRC drops packets to control the window size of the TCP connection and delays packets to increase the RTT . Consequently the achieved rate of the connection is controlled. The TRC is implemented as traffic conditioner which has to be placed at the network ingress. A TCP model for ns-2 TCP SACK implementation was derived. Based on this model the TRC is constructed.

The TRC is evaluated by simulations using ns-2. It is shown that the TRC performs well in a multi-domain environment. The requested rate is achieved with a very high probability. The deviation between requested rate and achieved rate is small.

Overall concluding from the simulation results the TRC seems promising to be used for TCP rate control in DiffServ networks. Especially the feasibility to mix UDP and TCP traffic is a major advantage compared to other approaches.

The whole concept of the TRC is based on simulations in ns-2. So the TRC has to be evaluated by real measurements in TCP/IP networks, because the accuracy of the TRC depends on the TCP model. In real TCP/IP networks there exist a lot of slightly different TCP implemen-

tations [10]. Consequently the applicability of the TRC in such an environment has to be evaluated, therefore for each TCP implementation a precise TCP model has to be derived.

The TRC does not need more suppositions as needed in QoS networks. The diversity of TCP implementations is a general problem of all attempts that try to control or estimate TCP rates on the basis of a model for TCP sending behavior.

References

- [1] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. RFC 2475: An architecture for differentiated services, Dec. 1998.
- [2] C. Brandauer and P. Dorfinger. An implementation of a service class providing assured TCP rates within the AQUILA framework. In *Workshop on Architectures for Quality of Service in the Internet*, Mar. 2003.
- [3] Y. Chait, C. Hollot, V. Misra, D. Towsley, H. Zhang, and J. Lui. Providing Throughput Differentiation for TCP Flows Using Adaptive Two-Color Marking and Two-Level AQM. to be presented at INFOCOMM 2001.
- [4] K. Claffy. Workload Characterization, 2002. <http://www.caida.org/analysis/workload> (May 15, 2002).
- [5] P. Dorfinger, C. Brandauer, and U. Hofmann. A Rate Controller for Long-Lived TCP Flows. In *Protocols and Systems for Interactive Distributed Multimedia*, pages 154–165, Nov. 2002. <http://www.salzburgresearch.at/cbrand/pub/trc.proms.02.pdf>.
- [6] A. Elizondo-Armengol. TCP-Friendly Policy Functions: Capped Leaky Buckets. In *Seventeenth International Teletraffic Congress (ITC17)*, December 2001.
- [7] V. Jacobson, K. Nichols, and K. Poduri. RFC 2598: An Expedited Forwarding PHB, June 1999.
- [8] K. Nichols, S. Blake, F. Baker, and D. Black. RFC 2474: Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers, December 1998.
- [9] Network Simulator ns-2, see <http://www.isi.edu/nsnam/ns/>.
- [10] J. Padhye and S. Floyd. TBIT The TCP Behavior Inference Tool, see <http://www.icir.org/tbit/>.
- [11] S. Sahu, P. Nain, D. Towsley, C. Diot, and V. Firoiu. On Achievable Service Differentiation with Token Bucket Marking for TCP. In *Proc. of the ACM SIGMETRICS'2000 Int. Conf. on Measurement Modeling of Computer Systems, Santa Clara, CA, USA*, June 2000.
- [12] S. Yilmaz and I. Matta. On Class-based Isolation of UDP, Short-lived and Long-lived TCP Flows. Technical Report BU-CS-2001-011, Boston University, 2001.