FoU

Forskningsrapport/Scientific Report

R&D R 18/2002

Ragnar Andreassen, Mechthild Opperud

Applications of explicit congestion notification in providing service quality differentiation and price/quality guarantees

telenor

R&D Report R 18/2002

Title **Applications of explicit congestion notification in providing service quality differentiation and price/quality guarantees**

ISBN	**82-423-0535-8**
ISSN	**1500-2616**
Project No	**CX1504 / Fifth Framework Project 11429 (M3i)**
Program	**F-COM**
Security Gr.	**OPEN**
No. of pages	**38**
Date	**2002.03.18**

Author(s)

Ragnar Andreassen, Mechthild Opperud

Subject headings

Active queue management, differentiated services, explicit congestion notification, guaranteed service provider

Abstract

We investigate by traffic laboratory experiments whether it is possible to differentiate service quality by explicit congestion notification, i.e., by marking packets under congestion. We compare the quality differences achieved by ECN marking with those achieved by a comparable diffserv queue management. The qualities measured were delay for VoIP and download times for web traffic.

Assuming that an internet service provider charges customers by the number of received ECN marks (dynamic pricing), we also experiment with a service providing quality and price guarantees, by averaging out the risks of varying prices and varying qualities. The provider of this “guaranteed service” (GSP) may pursue a business separate from the ISP’s (refining the ISP’s dynamically priced best-effort service) or be integrated with the ISP, who would offer a best-effort and a guaranteed service. We identify critical factors for the GSP’s business.

Contents

1 Introduction

1.1 Motivation and overview

This report describes results of experimental investigations on how to provide service and price differentiation for Internet transport services using end-to-end control policies. The alternative to end-to-end control policy is network-centric control policy, where policies for resource usage are coded into network nodes. The debate on which control-regime to employ is as old as the invention of data networks. Most network concepts for public data transport rely on network-centric control policies, with X.25 and ATM as notable examples. The TCP/IP based Internet is as such an exception in using end-to-end control. This exception has however by the process of technological and commercial "natural selection" and "survival of the fittest" achieved dominance in public data transport.

One reason for this is that the resource control applied in TCP/IP has turned out to fit well to needs of application services using the network as transport. Most of these have been elastic, i.e., they can derive utility of any transfer capacity. Also, TCP/IP was developed in a non-commercial environment, where there was no need to perform market differentiation at the transport network level. The success of the Internet has led to the new requirement that it should cater both for inelastic services, such as telephony and video that need a given minimal bandwidth to work, and for quality differentiation for elastic services.

The answers to these requirements in the Internet industry and standardisation fora have been the developments of the Integrated Services and Differentiated Services concepts. Both represent steps away from the original successful end-to-end policy model of the Internet towards network centric models. A more recent development is however the employment of so called "active queue management" in order to implement service differentiation. In active queue management the network makes network state information available to end nodes, which decide on network service deployment based on locally implemented policies. One way of transporting the network state information to the end nodes is by explicit congestion notification (ECN), which begins to mark packets (by setting the so-called ECN bit) at the onset of congestion.

The notion of service quality differentiation is closely related to price discrimination, as service differentiation implies price discrimination (but not conversely so). It is possible to combine active queue management with charging by giving network state information signals to users a simultaneous semantic as price signals, for example, by making users pay a small price per marked packet. In effect the network will then become a marketplace for network resource capacity. The upside of such pricing, commonly referred to as "dynamic pricing", is that contested resources are priced simultaneously proportional to the degree of contestation and to the consumption of contested resources. The downside is the degree of risk (in the form of price variation) to which users will be exposed. They are not used to such risks in connection with pricing of telecommunication services, and may not find the exposure to them as acceptable. An answer to this is to introduce the concept of risk brokering in conjunction with dynamically priced network services. Such risk brokers would hide the risks from the users at the cost of averaging between them, much like the insurance companies do in other business areas. Especially, they would give price and quality guarantees.

The experimental investigations to be reported here are aimed at answering the following questions related to the above:

- Can one enforce a service quality control for inelastic services by active queue management similar to what is achievable in a DiffServ environment? If so, what are the consequences in terms of network performance and impact on quality for remaining elastic traffic in the network?
- Which are the critical factors in performing risk brokering, and can operational methods be found facilitating such risk brokering?

The questions are generally related to the notion of a “guaranteed stream provider” (GSP). A GSP is an actor buying transport services with dynamically varying prices and/or service qualities from an Internet connectivity provider and selling transport services with given tariffs and qualities to end customers. The concept is developed and discussed in [1], and its architectural implications are further discussed in [3]. The current experiments relate to the GSP/ECN variant defined in [1], where the vehicle for dynamic pricing are the explicit congestion notifications [13] generated in the network by means of the random early drop (RED) mechanism [7]. The services offered by the considered GSP to its customers are access to two service classes, one designed to carry inelastic voice traffic and another designed for carrying elastic data traffic.

The investigations are made in an experimental setting. This facilitates freedom to choose the quality metrics most relevant to the services in question and comparisons between alternative QoS-provisioning solutions executing on the same platform. This latter may be important, as the link capacity in itself might not be the only critical resource. Bottlenecks may also exist in the execution of the mechanisms required to perform differentiated packet forwarding.

The traffic in the experiments is generated synthetically, but in a manner that is proposed to be realistic representations of web browsing and Internet telephony traffic. Also, the assumptions are made that the real time traffic is price insensitive and that the network accepts all calls that are initiated. These latter assumptions will not apply in the general case, but are relevant for the specific investigations to be made here.

The presentation proceeds by briefly relating the current investigation to some recent research in the field of active queue management. Chapter 2 then describes the common technical setting – network configuration and traffic generation – of the investigations. There are two main experiments relating to each of the bullet points listed above. The first experiment relates to the possibilities of service control by RED/ECN based active queue management, and is presented in Chapter 3. The last experiment relates to critical operational aspects of a GSP, in particular whether a GSP may infer something on the qualities of services by observing price levels of limited test streams flowing in the network. This experiment is reported in Chapter 4. The conclusions for both experiments are summarized in Chapter 5.

1.2 Related work

Packet marking can serve various purposes: load control, call acceptance based on the present marking rate, and charging (for each marked packet).

There is a growing volume of work on the usage of marking algorithms for the purpose of controlling certain network performance measures, such as the utilization, queue lengths. The purpose of marking is to warn sources of congestion already before packet loss occurs. This work goes under the title “active queue management”. We present some papers. The most commonly used marking algorithm is Random Early Detection – RED (presented here in Section 3.1.2). The algorithm is described in [7], and its use with ECN marking is standardized in RFC 3168 [13].

Diot et al. [5] investigate, by experiments and simulations, the influence of RED parameters on tcp goodput, tcp and udp loss rate, probability of consecutive packet losses, and queueing delay.

Hollot et al. [12] approach the problem of parameter setting in RED from the side of control theory, backed up by some simulations. The main goal is to make the system of (long-lived) tcp streams and RED control stable to variations in system loads, and thereafter to tune propagation delays and load levels.

Kunniyur and Srikant [16] similarly employ control theory to study another marking algorithm: adaptive virtual queue, which marks packets overflowing from a virtual queue whose capacity is updated according to the load level.

Athuraliya et al. [2] present yet another marking algorithm: random exponentialmarking, which is specially designed to stabilize utilization and queue size around given values. They test the algorithm's design goals by simulated tcp flows.

In the papers named up to now, marking was employed to control tcp flows. In a next step, one assumes that tcp reacts properly to marking, and that one now can use measured marking rates to estimate the congestion or load level, and depending on this, to accept or reject non-tcp connections.

Tom Kelly [15] experiments with an ECN-probe based call acceptance scheme applied to voice traffic. Prior to call setup, a voice connection probes the congestion level by sending packets at increasing rate, and abandoning call setup when the percentage of marked probes is larger than a fixed threshold, or when a packet was lost. A problem found here was that at high offered loads, ftp traffic grabbed an increasing percentage of the carried load, while voice connections were blocked (due to high marking rates).

Karsten [14] designs and tests a whole system with connection acceptance control at ingress and packet marking in the internal nodes. The system attempts to protect udp traffic from greedy tcp traffic and vice versa, by using only single FIFO queues in the internal nodes, though equipped with differentiated packet marking algorithms depending on the type of traffic. Marking rates are solely used to estimate load levels, and provide feedback to tcp sources. Special attention is given to so-called load-based marking (see [18]), which starts to mark packets when a load estimate exceeds a given value.

In the work presented up to now it is assumed that tcp reacts to marked packets in a standardized way (as if they were dropped packets) and that udp does not react at all. This may not be true in the future, so the idea appeared to control congestion by assigning an economic value to each marked packet. This idea was presented first by Gibbens and Frank Kelly [8]. This so-called congestion pricing or dynamic pricing opens a natural way to assign prices to traffic flows depending on their usage of contested resources and on the congestion they cause to others. It also opens the way to more refined protocols attempting to optimize the user's presumed utility function. A first invention (also from [8]) was the rate-controlled tcp, which tries to stabilize the fraction of marked packets over time around a user-given value, the willingness-to-pay.

Siris et al. [18] experiment with three different marking algorithms, and with two different tcp implementations of the rate control algorithm with willingness-to-pay. They compare willingness-to-pay, marking probability and throughput for different combinations of marking algorithms with various parameter settings.

Wischik [21] investigates, for certain marking algorithms, the dependence of marking rates (i.e., the charge) on the "effective bandwidth" of the traffic.

In our work, we first view packet marking solely as a technical mechanism to protect udp flows (non-reacting) from greedy tcp flows (reacting to marks as if they were dropped), and compare certain quality measures with the more standard diffserv situation using several queues, one for each traffic class. Secondly, we take the more economical viewpoint, and experiment with a “guaranteed service”, which not only gives quality guarantees, but also price guarantees in a dynamic pricing environment.

2 Network configuration and traffic generation

2.1 Network configuration

A test network was used for QoS testing with given dynamic pricing under various QoS strategies, traffic mixtures and load levels. The figure below illustrates the test network.

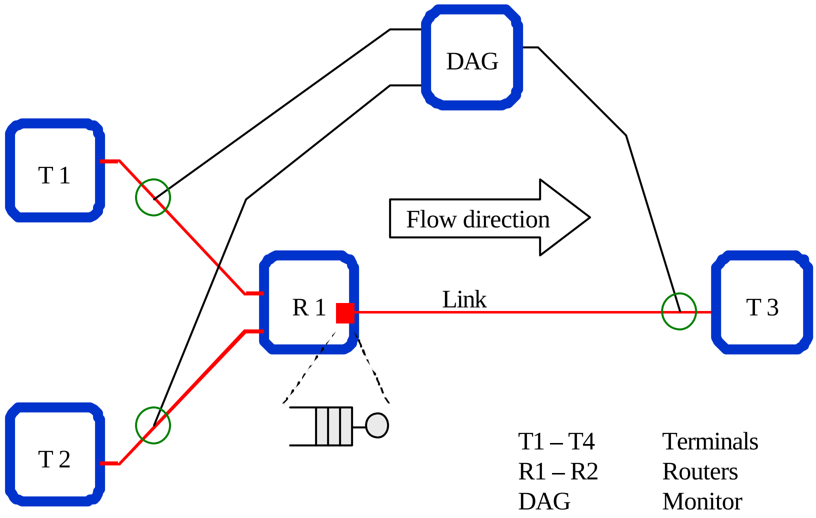

Figure 1: Test network configuration

The network is composed of three terminal nodes connected together by a router as indicated. Links between T1/T2 and R1 are 100 Mbit/s full-duplex Ethernet while the link between R1 and T3 is 100 Mbps half duplex. The capacities and resource allocation policies between router R1 and terminal T3 can be controlled using the ALTQ for FreeBSD software at the outbound interface of R1 [4]. Traffic flowing across the three access links to the router R1 is monitored by copying the flows into a DAG monitor [9]. At the T1/T2-R1 interfaces, this monitoring is facilitated by copying traffic flows by means of a Cisco 2924 Ethernet switch. As the R1-T3 interface is configured in broadcast mode, the DAG system can get its input data there through passive listening. Actual transmission capacity in the direction from R1 to T3 on the 100 Mbit/s half duplex bottleneck link was usually configured at 40 Mbit/s. As this is well below the link rate, and traffic was mainly unidirectional (only TCP acknowledgements in the opposite direction), the effect of the underlying Ethernet link arbitration mechanism is assumed to be insignificant. The DAG monitor performs hardware-supported real-time remote-synchronised (by GPS or direct link) time stamping of each packet entering the monitor system, enabling accurate one-way delay and loss analyses. These analyses are performed off-line by comparing monitor output at the sending and receiving ends of the communication.

The router is a 1.7 GHz PC running ALTQ on FreeBSD, acting as ECN-capable network node. Terminals T1 and T2 (also PCs, running either Linux Redhat 7.1 or FreeBSD 4.3) generate traffic towards terminals T3 and T4 using the GenSyn traffic generation system as described below. This system is capable of producing both elastic and inelastic traffic on the basis of a configurable statistical source model. In the test setup, one terminal generated elastic traffic while the other terminal generated inelastic traffic. In the experiments involving ECN, both elastic and inelastic traffic types were configured to produce ECN-enabled traffic streams.

2.2 Traffic generation

As mentioned above, the experiments were carried out using the GenSyn traffic generator. This generator, and its underlying modelling approach, is documented thoroughly in the publications [10] and [11]. A short description is included here in order to make this document reasonably self contained.

GenSyn is a Java process that generates IP traffic using a flexible, scalable, stochastic, modelling framework for describing the user behaviour of sources. This stochastic behaviour model is linked to the underlying protocol stack and generates real packets into the network. This is an approach that combines flexibility and scalability of composite state models with accuracy in the protocol behaviour. Several parameter-controlled model examples are described in the GenSyn framework, including web, ftp, VoIP, MPEG video, and constant packet rate. In the current experiments, the web/ftp and VoIP models were used.

2.2.1 Voice sources

A voice stream is considered to be completely price-insensitive at the packet level after it has started. It thus represents a specific type of inelastic source. Two voice models are possible, with and without silence suppression. In the current experiments, only the latter type was used. Neither models include control traffic, i.e., call setup and disconnection signalling.

The user behaviour model of voice streaming

The voice over IP model uses a simplified model of a telephony user. The media stream is uni-directional, i.e., the A and B parties are included in the same model but no synchronisation (two-way communication) between the two parties exists.

The user behaviour model consists of two states as illustrated in Figure 2.

Idle: a source generates on the average three calls per hour and each call is generated according to a Poisson process,

Connect: the call duration time follows a negative exponential distribution with average of 3 minutes

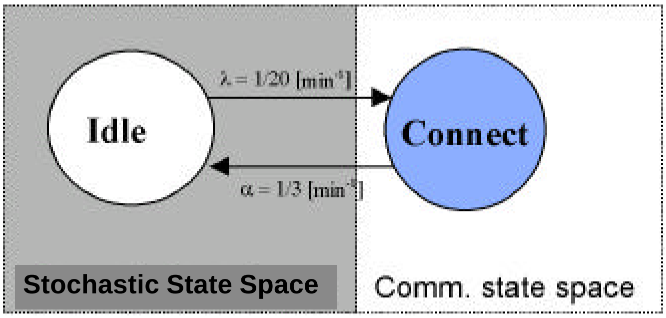

Figure 2: The model of a voice source without call signaling

Interface module – sending UDP packets

Two approaches, with and without silence suppression, are used to model the details of the connect state of the user behaviour model. Both approaches assume that the voice channels have a capacity of 8 kbit/s that is modeled as a stream of udp packets of fixed size of 100 bytes payload generated every 100 ms. Both the packet size and inter-packet times are given as parameters to the interface module and hence are easily changed.

In the current experiments only the interface module without silence suppression was used, which means that the udp packet stream was sent continuously in the connect state.

The VoIP connections are not aware of congestion signals in the form of ECN marks. The udp transport protocol is however modified in order to allow the setting of the ECT flag in the IP header, such that the network queuing mechanism may mark udp packets rather than drop them in case of congestion. We may justify this by assuming that the network operator uses ECN marking solely to control web traffic in order to give VoIP connections diffserv-like delay guarantees. In the GSP experiments we establish a correlation between ECN marking rate and VoIP quality, which may later be used to control admission of VoIP connections.

2.2.2 Web/ftp sources

In addition to voice sources we also used web/ftp sources as background traffic.

The user behaviour model of web and ftp clients

The overall model of the web and ftp clients describes the user behaviour within and between sessions by the 3-state model in Figure 3 below. A session is essentially the same as the web sessions defined in [20] as a sequence of packets with less than 30 minutes between two consecutive web page requests. The following states are defined:

Idle: the user is in between sessions.

Read: the user reads the downloaded web page or a file and considers what to do next, download another one, or close the session?

Download: the user opens a connection to a url-address, randomly selected from a list of addresses, and downloads this page or file. In the web interface module the page is parsed and all corresponding image files and applets are downloaded.

The Idle and Read states are stochastic states. The state sojourn times of these two states are sampled from a probability density distribution. Each user that starts to download a web page enters the Download communication state. When the web page, and all its content (text, images, applets), is downloaded, the user returns to the Read state. The soujourn time in the download state is also stochastic in a sense, depending on the stochastically varying available network capacity. The modelling framework enables the (random) setting of an upper limit of the download time, the impatience factor. This factor allows the connection to be closed before the entire web page is downloaded.

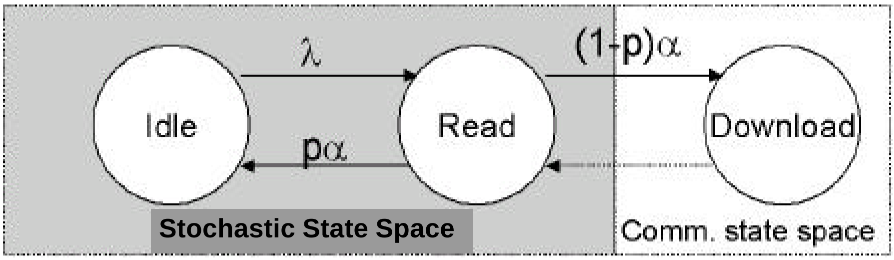

Figure 3: The overall model of a web or ftp client

The parameters of the state model are extracted from the work described in [20] where an aggregated stream of several HTTP connections was separated and broken into web sessions.

Time between web-sessions - The Idle state uses the web session separator criterion as the expected sojourn time $T_{idle} = l e^{-1}$, where 1/λ = 1800 seconds.

Time between requests - Within a web session the mean time between requests (*X*) is *E(X)* = 42.8 seconds, with a coefficient of variation equal to *S/E(X)* = 2.9. This means that the Read state in the overall model is not negative exponential distributed. This state is therefore substituted by a hyper-exponential distribution with 4 branches, each with different time constants. The parameters are determined to fit the truncated-Pareto model used in [20].

Ftp interface module – downloading a file

The ftp interface module uses http to download a file. The file can be downloaded from any machine that is set up as a web server and read directly into the memory on the receiving machine, i.e. the machine that is hosting the GenSyn process running the ftp client. In contrast to the web interface module in [10], which downloads web pages, parses them for embedded objects, and subsequently downloads these objects, the ftp module we used simply downloads a file and discards it. This adds very little processing time to the download time, and hence most of the download time is given by the server response and transfer delays.

The pointer to the files that can be downloaded by the ftp interface module are given as url-addresses in a separate parameter list as input. This list is in a text file that can easily be changed and hence the user of GenSyn can create any empirical file distribution.

In the current experiments, the “heavy tailed” empirical file-size distribution of Figure 4 was used.

All web users employ a tcp version that reacts to ECN marked packets as if they were dropped packets. (We were not able to implement more advanced rate control schemes in our experiment setting.)

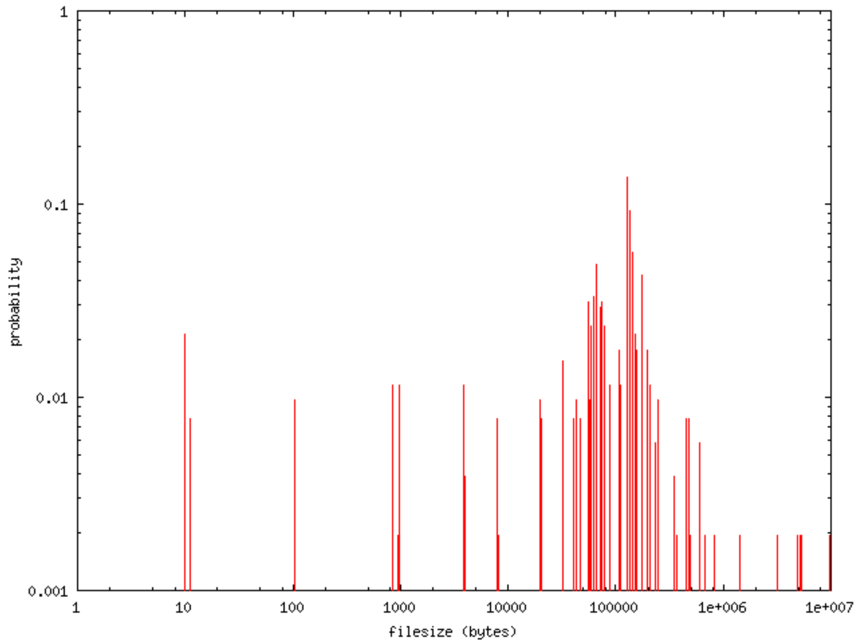

Figure 4: Empirical distribution of download file sizes

2.3 Conceptual model and compared measures

The conceptual model of the experiment network and the employed quality metrics are illustrated in Figure 5. As shown, the elastic traffic constitutes a feedback system, where firstly the behaviour of TCP depends on the network service. The behaviour of the web source that submits jobs to TCP is again dependent on the TCP job service rate.

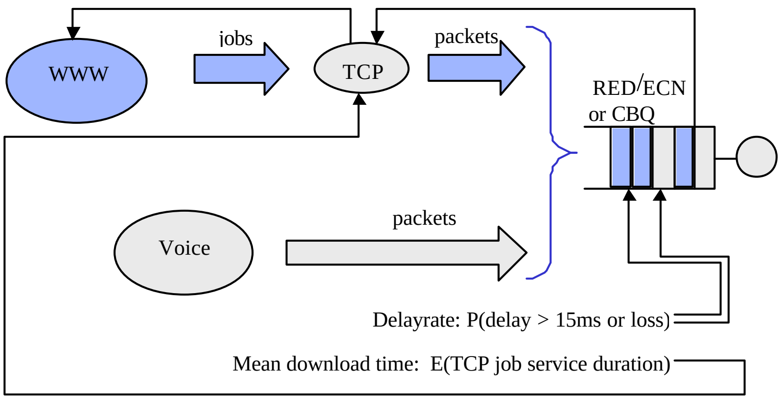

Figure 5: Network conceptual model

Emphasis has been put on using quality metrics relevant for the services in question. For voice services the delay rate = prob(packet delayed more than 15 ms or packet lost) was used. This metric is directly relevant for the dimensioning of the resynchronising buffer in the voice decoder at the receiving end, thus for the overall network delay. Delays of individual packets are irrelevant for a web-service, where the relevant units are web-pages or files. Typically, files or web-pages are transferred in tcp-sessions consisting of transmission and possibly retransmissions of several packets. The quality metric for the web service was therefore chosen to be the average duration of tcp-sessions as observed in the network. Overall throughput and average delays were also measured for the two services.

3 Service differentiation by explicit congestion notification

A main trend for providing service differentiation in IP-based networks is to employ diffserv technology, where nodes in the network are configured to serve IP-packets belonging to different service classes differently. In the realisation that diffserv is a main development, and dynamic pricing is not, the method of research for the current experimental investigations has been chosen to be a comparison between diffserv scenarios and GSP/ECN scenarios. More specifically, a given traffic mix of elastic/inelastic traffic is served by a node employing class-based queuing with two service classes, configured such that certain quality objectives are obtained for the real-time traffic. The same traffic mixes are then served in an ECN-enabled environment using the same hardware/software systems. By varying the way congestion notifications are created in the network, hence the behaviour of the elastic sources, it is possible to observe under which conditions the real-time service gets comparable quality to the diffserv case, and then to assess the overall efficiency of the two service differentiation schemes by comparing resulting qualities for elastic traffic.

3.1 Experiment scenarios

The diffserv scenario has two classes, one designed for real-time traffic and another for best-effort traffic. We use a link of 40 Mbit/s. The diffserv scenario is implemented by a class-based queuing (CBQ) scheme supported in ALTQ. We set up CBQ such that it guarantees udp traffic (VoIP) 7% of the bandwidth (2.8 Mbit/s) and tcp traffic (web) 90% of the bandwidth (36 Mbit/s). These guarantees apply in case of congestion. When there is no congestion, one class may prey on the bandwidth of another class. ALTQ reserves the rest of 3% of the bandwidth to control traffic. The bandwidth percentages were chosen in this way because our traffic generator was not able to generate more than 3 Mbit/s of VoIP traffic. A buffer of 50 ms is allocated to both the udp and tcp class in relation to their weights. The tcp class employs Random Early Drop (RED) [7] to avoid congestion. The RED parameters are given in Table 1. The maximum threshold is equivalent to the buffer size allocated to the tcp class. The minimum threshold is one half of the maximum threshold, the maximum marking probability is 1/10, and the weighting factor in exponential queue averaging is 2^{-9}, which is the default setting suggested many places in the literature and used in other experiments at Telenor.

Class	bandwidth perc. [%]	packet size [Byte]	min_{th} [packets]	max_{th} [packets]	max_p	w_q
VoIP	7	142				
Best-effort	90	1457	77	154	1/10	2^{-9}

Table 1: Parameters of diffserv scenario

We experiment with two types of sources, namely web users and VoIP. Both models are described in Section 2.2.

In the diffserv scenario the web users use the best-effort class, and the VoIP users use the real-time class.

In the ECN scenario all classes use the same queue towards a bottleneck of 40 Mbit/s. The queue marks incoming packets using ECN marks, according to a RED algorithm. The parameters of the RED algorithm are described in Section 3.1.2.

Note that in the GENSYN traffic model the number of users is not the number of *active* users, but the size of the total user population out of which some users may be idle. This gives burstier traffic than if all sources were in communication state all the time.

3.1.1 Choice of traffic mixes

We experiment with traffic mixes satisfying the following criteria

- In the diffserv scenario no VoIP user may in the long run decrease his delay rate by permanently switching from the udp class to the tcp class. We assume implicitly that VoIP users are lazy and do not switch between priority classes depending on immediate delay measurements. They may, however, want to switch when the long run delay rate of the udp is larger than the long run delay rate of the tcp class.
- In the diffserv scenario the VoIP delay rate prob(delay > 16 ms or loss) should not exceed 10^{-2}.

After some preliminary experiments we arrived at the traffic mixes described in Table 2.

3.1.2 Calibration of ECN marking rate

For given number of web users, *m*, and given number of VoIP users, *n*, we adjust the ECN marking rate in the ECN scenario such that the delay rate becomes approximately the same as in the corresponding diffserv scenario. By increasing the marking rate one may decrease the traffic offered by web sources and thereby decrease VoIP delay rates. Note that ECN marking does not drop VoIP packets, it only marks them, but VoIP does not react to these markings, so VoIP is not directly affected by ECN marking. However, the tcp protocol used for web traffic *does* react to marked packets by entering its slow start/congestion avoidance procedures.

ECN marking is achieved by the RED algorithm [7]. The RED algorithm marks (or drops) packets with a certain probability when the average queue size contains more than min_{th} packets. The marking probability increases linearly from 0 to max_p between min_{th} and max_{th}. When the average queue size contains more than max_{th} packets, all incoming packets are marked. In our case max_{th} was always 6 times min_{th}. The average queue size is according to [7] computed by exponential smoothing with a parameter w_q. Figure 6 shows the dependence of the marking rate on the queue size.

In our experimental setting we only vary max_{th} and keep all other parameters fixed as follows. We set max_p to 1, which gives a smoothly increasing marking rate. We set $min_{th} = max_{th}/6$, and vary max_{th}. This approximates the gentle RED version with its convex marking function used in literature (e.g. [6]). We set w_q to 1, that is, we use the instantaneous queue size. Diot et al. [5] found that using instantaneous queue length in RED leads to less variation in the packet delays in the presence of web traffic than using an average queue size, and that smoothly increasing marking rates leads to less consecutive packet losses.

For finding the right max_{th}, we chose six values of max_{th} between 1 and 6 times the equivalent of 16 ms buffer expressed in number of packets. This was computed as link size x 16 ms / average packet length (in bytes). The average packet length is dependent on the traffic mix. If max_{th} is set to the lower limit, RED will mark all tcp packets whenever the buffer is approximately 16 ms long. This should guarantee zero delay rate to udp traffic. On the other hand if min_{th} is set to the equivalent of 16 ms, and max_{th} to 6 x min_{th} then RED will start marking packets with a low probability whenever the buffer contains 16 ms worth of tcp packets. This explains the range from which max_{th} was chosen.

3.2 Comparisons and results

For given number of web users, *m*, and given number of VoIP users, *n*, we adjust the ECN marking rate in the ECN scenario such that the delay rate becomes approximately the same as in the corresponding diffserv scenario with the same number of web and VoIP users. We are now able to compare the web traffic's quality of the ECN scenario with the diffserv scenario.

3.2.1 Comparison of diffserv with ECN

Table 2 shows the results for the diffserv scenario.

#voice	delay rate	avg. delay (ms)	Mbps	#web	delay rate	avg. delay (ms)	Mbps	avg. download time (ms)
700	1.26E–3	1.02	1.13	30	1.81E–1	7.50	22.01	117.09
800	5.83E–3	1.30	1.29	25	1.72E–1	7.27	20.09	178.69
900	9.70E–3	1.30	1.52	20	1.44E–1	6.94	18.88	151.31
1000	8.28E–3	1.18	1.59	15	7.51E–2	5.53	15.94	118.01
1100	1.08E–2	1.21	1.76	10	3.86E–2	5.00	14.20	102.53

Table 2: Results for diffserv scenario

Table 3 shows the results for the ECN scenarios, whose max_{th} was set such that the voice delay rate is approximately the same as in the diffserv scenario. We have included the diffserv results into the table (first line in every group). The second and third lines in every group correspond to the ECN scenarios whose voice delay rates (third column) were, respectively, lower and higher than the voice delay rate of the diffserv scenario. For the last group (of 1100 VoIP users and 10 web users) we were not able to generate an ECN scenario with a voice delay rate of more than $1.08E^{-2}$.

max_{th}	#voice	delay rate	avg. delay (ms)	Mbps	#web	delay rate	avg. delay (ms)	Mbps	avg. download time (ms)
ds	700	1.26E–3	1.02	1.13	30	1.81E–1	7.50	22.01	117.09
288	700	3.65E–4	2.30	0.88	30	3.82E–4	3.26	21.54	251
360	700	2.29E–3	2.57	0.84	30	2.22E–3	3.89	20.59	233.64
ds	800	5.83E–3	1.30	1.29	25	1.72E–1	7.27	20.09	178.69
408	800	5.19E–3	2.83	0.95	25	6.00E–3	4.27	21.06	217.20
438	800	9.70E–3	2.81	1.01	25	1.31E–2	4.40	20.44	188.03
ds	900	9.70E–3	1.30	1.52	20	1.44E–1	6.94	18.88	151.31
444	900	1.21E–3	1.83	1.10	20	2.33E–3	3.56	15.51	116.09
474	900	8.94E–3	2.67	1.18	20	1.06E–2	4.24	19.71	181.39
ds	1000	8.28E–3	1.18	1.59	15	7.51E–2	5.53	15.94	118.01
480	1000	2.54E–3	1.87	1.24	15	4.00E–3	3.87	14.81	115.12
540	1000	1.11E–2	2.13	1.23	15	2.13E–2	4.17	15.94	126.24

ds	1100	1.08E–2	1.21	1.76	10	3.86E–2	5.00	14.20	102.53
600	1100	4.73E–3	1.28	1.39	10	1.52E–2	3.96	9.70	91.42

Table 3: ECN scenarios compared to diffserv scenarios

It can be seen that, compared to the diffserv scenario, the average delay of voice packets is doubled, from appr. 1 ms to 2 ms, which is not a large delay. We may say that voice quality is approximately equal in the two queuing regimes.

The throughput for web traffic does not change when comparing diffserv with ECN. Throughput for voice traffic seems however to be systematically somewhat lower in the ECN scenarios. As voice constitutes non-responsive udp-traffic, and losses are moderate, these differences cannot easily be ascribed to differences in network configuration. Also, the generation of voice traffic is independent (being created at a separate computer) of web traffic generation. As voice traffic is only in the range of 5 – 10 percent of the overall traffic, this rather surprising observation does not however significantly affect the overall throughput efficiencies in the ECN-scenarios. Concerning quality for web users, the web delay rates and average delays are significantly lower than in the diffserv scenario, but download times are somewhat higher (double) or equal. The difference is more marked for the traffic mixes with higher number of web users. Apparently the tcp version used in the ECN scenarios reacted as it should: In times of high congestion, tcp bit rates were suppressed (resulting in higher download times), and at times of low congestion, tcp bit rates were increased (resulting in lower average delays per packet). Tcp has become more conservative, so to speak.

We may conclude that it is possible to devise a marking algorithm operating on a single queue, such that voice quality is approximately the same as in diffserv, and ftp quality is somewhat reduced as compared to the two-queue diffserv model. In the worst case (large number of web users), download times are doubled, and in the best case (small number of web users), they were approximately the same.

3.2.2 Influence of marking thresholds on quality measures and marking rate

Table 4 shows the influence of changed marking thresholds in the ECN scenarios on diverse measures. The scenarios are grouped after number of voice and web users. The columns are grouped by voice and web data. The new columns titled marking rate give the ratio of marked packets over total sent packets of voice or web traffic, respectively.

Max_{th}	#voice	delay rate	avg. delay	Mbps	marking rate	#web	delay rate	avg. delay	Mbps	avg.dl time	marking rate
72	700	2.89E–5	0.92	0.87	4.02E–2	30	9.39E–5	1.42	17.69	288.90	1.08E–2
144	700	2.92E–5	1.60	0.88	2.14E–2	30	7.05E–5	2.17	20.60	450.96	6.09E–3
216	700	9.97E–6	2.11	0.86	1.01E–2	30	5.24E–5	2.83	22.21	331.86	3.47E–3
288	700	3.65E–4	2.30	0.88	5.52E–3	30	3.82E–4	3.26	21.54	251.00	2.16E–3
360	700	2.29E–3	2.57	0.84	2.51E–3	30	2.22E–3	3.89	20.59	233.64	1.39E–3
384	700	6.04E–3	3.11	0.87	2.54E–3	30	6.47E–3	4.42	22.67	293.71	1.56E–3
414	700	9.38E–3	3.04	0.88	2.04E–3	30	1.05E–2	4.45	21.90	234.79	1.24E–3
432	700	1.38E–2	3.45	0.89	2.42E–3	30	1.52E–2	4.80	23.37	307.64	1.37E–3
78	800	7.81E–6	1.10	1.02	5.05E–2	25	1.47E–4	1.51	19.80	460.97	1.25E–2
156	800	1.74E–5	1.42	0.98	1.27E–2	25	1.90E–4	2.13	19.31	271.21	4.49E–3
234	800	8.14E–5	1.98	1.06	6.91E–3	25	6.44E–5	2.86	20.84	291.24	3.07E–3
312	800	3.63E–4	2.33	0.98	3.83E–3	25	3.18E–4	3.30	21.43	248.63	1.85E–3
390	800	2.83E–3	2.62	1.04	2.43E–3	25	3.01E–3	3.88	20.99	207.45	1.36E–3

408	800	5.19E–3	2.83	0.95	2.30E–3	25	6.00E–3	4.27	21.06	217.20	1.27E–3
438	800	9.70E–3	2.81	1.01	1.36E–3	25	1.31E–2	4.40	20.44	188.03	1.02E–3
468	800	1.42E–2	3.08	1.05	1.29E–3	25	1.80E–2	4.88	20.51	203.49	1.01E–3
84	900	7.59E–6	0.97	1.12	3.77E–2	20	8.96E–5	1.46	17.57	235.65	1.09E–2
168	900	0	1.24	1.12	9.63E–3	20	2.76E–5	1.99	17.24	186.74	3.67E–3
252	900	2.84E–5	1.86	1.06	5.45E–3	20	3.41E–5	2.86	19.19	196.76	2.51E–3
336	900	4.97E–5	1.71	1.11	1.85E–3	20	8.33E–5	3.18	16.23	132.15	1.15E–3
420	900	1.40E–3	1.90	1.13	9.89E–4	20	2.77E–3	3.52	16.42	126.11	6.99E–4
444	900	1.21E–3	1.83	1.10	7.03E–4	20	2.33E–3	3.56	15.51	116.09	5.40E–4
474	900	8.94E–3	2.67	1.18	1.34E–3	20	1.06E–2	4.24	19.71	181.39	9.59E–4
504	900	1.00E–2	2.41	1.11	7.60E–4	20	2.26E–1	4.45	13.70	196.04	5.65E–4
90	1000	0	0.80	1.21	2.52E–2	15	4.42E–5	1.41	14.66	142.74	9.10E–3
180	1000	4.59E–7	1.04	1.24	7.66E–3	15	1.89E–5	1.89	14.47	112.75	3.12E–3
270	1000	3.15E–5	1.62	1.28	3.86E–3	15	9.23E–5	2.65	17.85	155.93	2.00E–3
360	1000	3.00E–5	1.52	1.25	9.40E–4	15	6.10E–5	3.09	14.76	108.49	7.05E–4
450	1000	2.25E–3	2.25	1.25	1.16E–3	15	3.12E–3	3.98	17.49	139.35	8.26E–4
480	1000	2.54E–3	1.87	1.24	6.45E–4	15	4.00E–3	3.87	14.81	115.12	5.07E–4
516	1000	1.99E–3	1.59	1.30	3.70E–4	15	3.69E–3	3.59	13.33	98.05	3.00E–4
540	1000	1.11E–2	2.13	1.23	4.58E–4	15	2.13E–2	4.17	15.94	126.24	4.11E–4
102	1100	4.12E–7	0.52	1.38	1.15E–2	10	2.52E–5	1.37	9.03	105.24	7.09E–3
204	1100	4.10E–7	0.47	1.39	1.30E–3	10	1.05E–6	1.87	6.03	87.74	1.64E–3
306	1100	0	0.97	1.38	1.05E–3	10	5.93E–7	2.48	10.72	90.44	7.80E–4
408	1100	1.76E–3	1.50	1.40	7.98E–4	10	1.25E–3	3.17	13.88	109.07	5.77E–4
528	1100	2.57E–3	1.55	1.36	1.96E–4	10	5.79E–3	3.90	11.99	99.04	2.62E–4
564	1100	2.39E–3	1.55	1.40	1.28E–4	10	5.08E–3	3.83	12.03	95.15	1.51E–4
600	1100	4.73E–3	1.28	1.39	6.09E–5	10	1.52E–2	3.96	9.70	91.42	1.06E–4

Table 4: Influence of marking threshold on ECN scenarios

The marking rate changes by increasing max_{th}. Note that min_{th} was set to $1/6^{th}$ of max_{th}. This means that for lower max_{th}-values, marking starts earlier, depending on the number of packets in the queue. Therefore, the lower max_{th}, the more packets should be marked.

Apparently with increasing max_{th}, the delay rates and average delays of voice and ftp packets increase, and the marking rates decrease, as expected. The average download times are expected to decrease, but this effect, if at all, can only be seen for a larger number of web users. Maybe the measurement period was too short (30 min), so statistical variations in download times had a too large influence. The changes in marking rate do not seem to have an effect on the throughput for either voice or web users, except maybe for the lowest max_{th} -values (indicating the most restrictive marking algorithm). The throughput values are much more influenced by the traffic mix. Figure 9 illustrates the dependence of the VoIP delay rate and the web download times on a byte-scaled version of the min_{th} threshold, which is described in Section 3.2.4. Here, a line denoted by *n*v_*m*f refers to the traffic mix with *n* voice sources and *m* web/ftp sources.

3.2.3 Comparison of marking rates for voice and for web traffic

One may observe from Table 4 that the percentage of marked voice packets is higher than the percentage of marked tcp (web) packets, especially when max_{th} is small (i.e., marking starts early). If marks would correspond to prices, this means that voice traffic has to pay more per sent packet than web traffic. If prices were given as marked packets / total sent bytes, the relationship of voice prices versus web prices would grow by a factor of 10, because web packets are generally ten times as large as voice packets.

It is only fair that voice traffic has to pay more per sent packet than web traffic. This is the price voice has to pay for being protected from web traffic, especially when voice delay rates are kept very small with the help of a small marking threshold max_{th}. One should also note that the udp version used for voice traffic is declared ECN-capable, but does not react to ECN marks, in contrast to the tcp version which does react to ECN marks. So udp is much more aggressive than tcp, and is thus rightly punished by higher ECN prices.

Figure 10 shows how the marking rates change with a byte-scaled version of the min_{th} threshold, which is described in Section 3.2.4. Here, a line denoted by *n*v_*m*f refers to the traffic mix with *n* voice sources and *m* web/ftp sources.

3.2.4 Dependence of delay rates on byte-value of RED thresholds

We claim that in order to achieve a certain delay rate P(delay > 16ms), one should set min_{th} as a fixed fraction of the queue size (in bytes) corresponding to 16 ms. The value of *16ms(B)* is the linkrate (40Mbps) times 16ms, scaled to become bytes. Figure 6 shows the marking probability in dependence on the queue length in bytes.

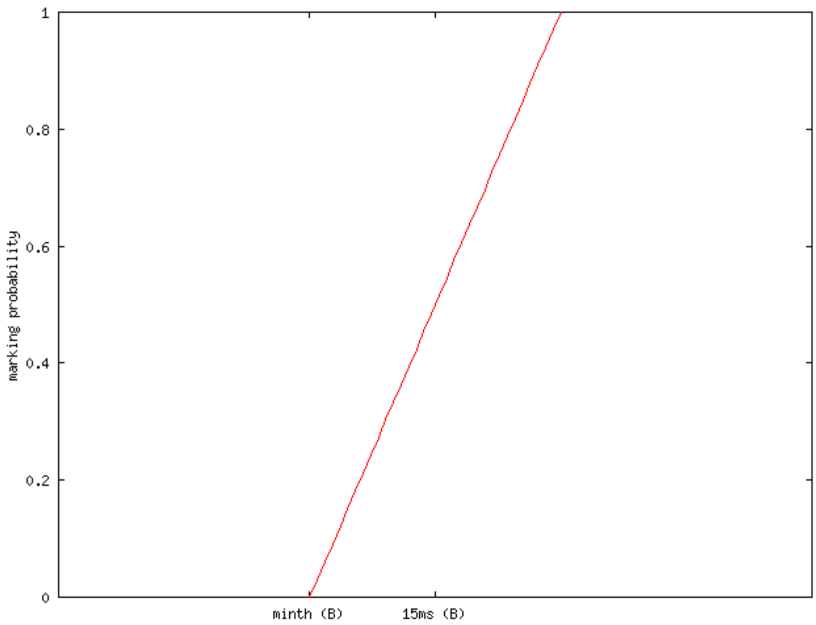

Figure 6: Marking probability in dependence on queue length in bytes

Table 5 shows the necessary data to compute $min_{th}(B)$ as min_{th} times the average packet length, and the ratio of min_{th} *(B)/16ms(B).*

max_{th}	min_{th} (pack)	#voice	#web	delay rate (voice)	Avg.pack. (Bytes)	min_{th} (B)	$min_{th}(B)$/ 16ms (B)
72	12	700	30	2.89E–5	967.54	11610.45	0.148
144	24	700	30	2.92E–5	1011.95	24286.84	0.309
216	36	700	30	9.97E–6	1037.86	37362.79	0.475
288	48	700	30	3.65E–4	1022.22	49066.70	0.624
360	60	700	30	2.29E–3	1022.48	61348.51	0.780
384	64	700	30	6.04E–3	1038.31	66451.96	0.845
414	69	700	30	9.38E–3	1027.16	70874.16	0.901
432	72	700	30	1.38E–2	1040.90	74944.81	0.953
78	13	800	25	7.81E–6	954.33	12406.27	0.158
156	26	800	25	1.74E–5	958.42	24918.94	0.317

234	39	800	25	8.14E–5	957.16	37329.06	0.475
312	52	800	25	3.63E–4	987.57	51353.55	0.653
390	65	800	25	2.83E–3	966.11	62797.26	0.799
408	68	800	25	5.19E–3	994.18	67604.01	0.860
438	73	800	25	9.70E–3	965.96	70514.97	0.897
468	78	800	25	1.42E–2	956.47	74604.68	0.949
84	14	900	20	7.59E–6	886.05	12404.69	0.158
168	28	900	20	0	881.93	24694.04	0.314
252	42	900	20	2.84E–5	932.80	39177.70	0.498
336	56	900	20	4.97E–5	865.11	48446.43	0.616
420	70	900	20	1.40E–3	863.70	60459.01	0.769
444	74	900	20	1.21E–3	854.22	63212.31	0.804
474	79	900	20	8.94E–3	907.34	71680.23	0.911
504	84	900	20	1.00E–2	887.52	74551.99	0.948
90	15	1000	15	0	805.07	12076.09	0.154
180	30	1000	15	4.59E–7	792.09	23762.67	0.302
270	45	1000	15	3.15E–5	849.30	38218.65	0.486
360	60	1000	15	3.00E–5	795.21	47712.55	0.607
450	75	1000	15	2.25E–3	850.40	63780.32	0.811
480	80	1000	15	2.54E–3	798.98	63918.48	0.813
516	86	1000	15	1.99E–3	750.27	64522.80	0.820
540	90	1000	15	1.11E–2	826.96	74426.80	0.946
102	17	1100	10	4.12E–7	607.19	10322.27	0.131
204	34	1100	10	4.10E–7	491.30	16704.07	0.212
306	51	1100	10	0	660.57	33689.15	0.428
408	68	1100	10	1.76E–3	737.74	50166.50	0.638
528	88	1100	10	2.57E–3	701.81	61759.54	0.785
564	94	1100	10	2.39E–3	692.96	65138.29	0.828
600	100	1100	10	4.73E–3	628.88	62887.96	0.800

Table 5: Dependence of delay rates on byte-values of thresholds

Figure 7 shows that the delay rate seems to be a function of $min_{th}(B)$ for delay rates larger than 10^{-4}, independent of the traffic mix! According to the figure, if one wishes to achieve a delay rate of, say 0.005, one may set $min_{th}(B)$ to approximately 0.8 times *16ms(B)*.

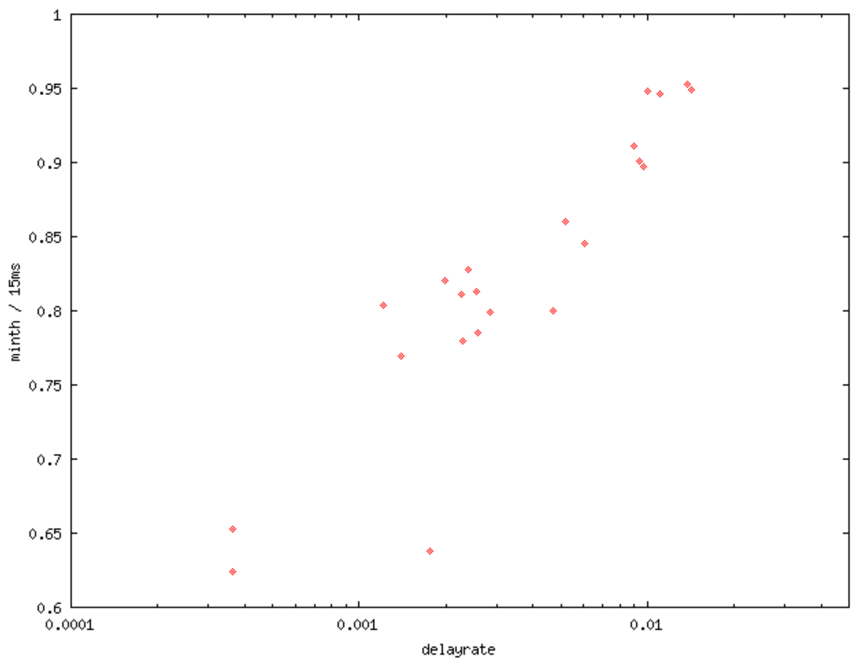

Figure 7: Delayrate versus min_{th} (B)/16ms(B)

To show the contrast, we plotted the delay rate versus the original min_{th} given in packets. Here, there does not seem to be any dependence, or if there is, then it is strongly influenced by the traffic mix. This means one would have to tune the min_{th} parameter anew for every traffic mix.

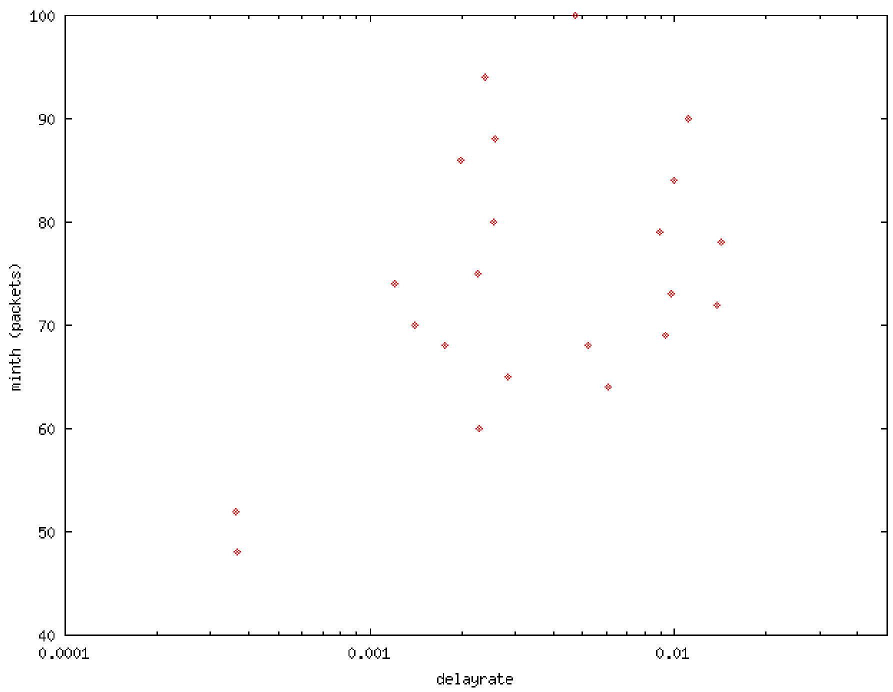

Figure 8: Delayrate versus min_{th} (packets)

We conjecture that in order to keep a delay rate under a certain threshold *dr*, one should use a RED-ECN algorithm that marks in dependence on the queue length in bytes, and where min_{th} is set to a value of *f(dr) x d* ms *x* link rate, and where *f(dr)* can be found by sampling from the network. The function *f(dr)* would be independent of the traffic mix, but dependent on other RED parameters (max_{th}, queue averaging factor, max_p).

The following figures illustrate how delay rates and marking rates for the two traffic types change as a function of the byte-scale RED min_{th} -value.

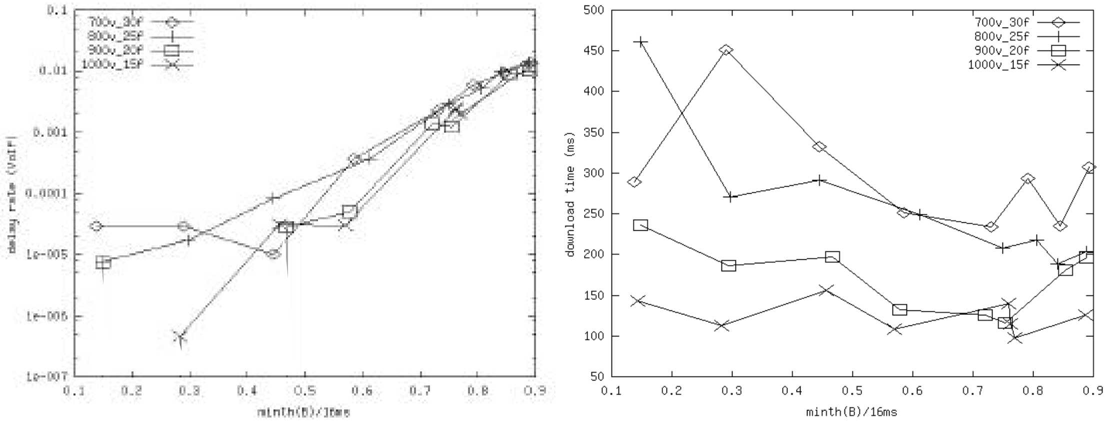

Figure 9: Quality in ECN scenarios as functions of RED min_{th} in bytes relative to 16 ms quality threshold.
Left panel: delay rates, right panel: average download time

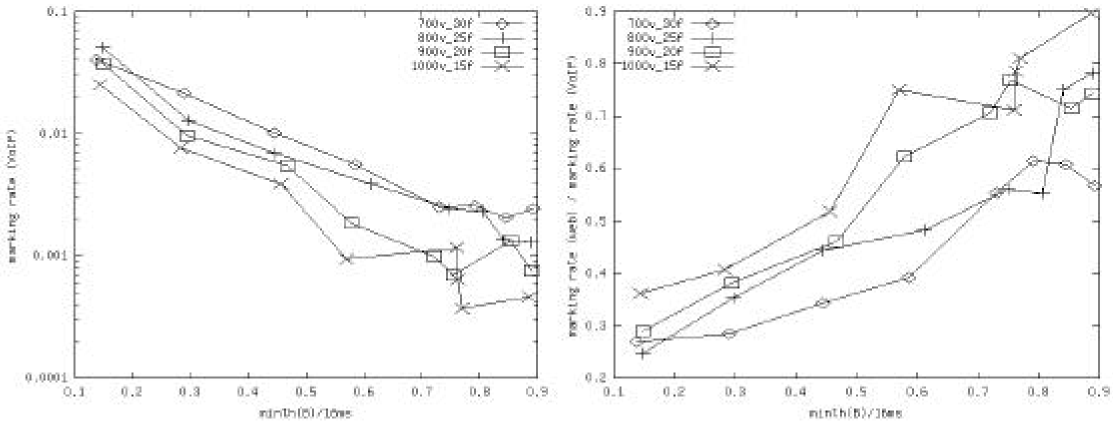

Figure 10: Marking rates.
Left panel: VoIP marking rates as function of RED configuration.
Right panel: Web marking rate relative to VoIP marking rate as function of RED configuration

3.2.5 Critical assumptions

Our findings are dependent on the behaviour of the VoIP and web sources. Especially, we assume that all VoIP users in the population never change their diffserv priority class (during the course of the experiment...). This may be justified by inherent laziness and lack of tools to measure instantaneous delay rates.

We also assume that VoIP users in the ECN scenario do not refrain from sending when the ECN marking rate is high (and delay rates are high). That is, there is no admission control for VoIP connections based on ECN marking rates. This may be excused, because in our ECN experiments the "network operator" is assumed to set the marking rate such that the delay rate always is acceptable to the VoIP user. We do not say anything about the network operator's charges towards the VoIP user ? let us assume the VoIP user is prepared to pay almost anything for an acceptable quality.

We assume that VoIP users in the ECN-experiments set their ECT bit, i.e., they signalled that their protocol reads ECN marks (but in reality, they do not care about ECN marks). The set ECT bit has the effect that the router does not drop packets but rather marks them in case of congestion. Our VoIP protocol's usage of the ECT bit is not quite in line with the philosophy behind ECN marking. We may excuse this by the fact that ultimately the (nasty) VoIP users will be punished by having to pay, e.g., the equivalent of received ECN marks.

We assume that web users all use the same ECN-enabled TCP protocol. This may be justified, if the willingness-to-pay value is hard-coded into the transmission protocol ? the average web user may not want to reprogram his transmission protocol. The situation may change when the user interface allows the user to make his transfer protocol more aggressive by just turning a knob, or when the ftp agent discussed in D11 [19] becomes available, which transfers data at either maximum speed or not at all depending on the present ECN marking rate. It would be interesting to see whether smarter ftp agents lead to increased quality for aggregated web traffic, always assuming that the network operator sets the marking rate such that quality experienced by real-time users is not compromised.

3.3 Conclusion

The development towards service differentiation in the Internet has for several years been a large effort, but the impact on the commercial scene has up to now been rather limited. A main trend is the development of the Differentiated Services concept, in which differentiation is performed in network nodes at an aggregate level. The fact that uptake of DiffServ is rather slow may be taken as a hint that even this paradigm, designed with simplicity and scalability in mind, might be complicated to implement and operate. The experimental investigation reported here considers a differentiation scheme that is perhaps even simpler than DiffServ: The network is left basically best effort as it always has been, but the end nodes themselves decide how the network should be used. In order to give end nodes relevant information, the network supplies information on network state in terms of binary congestion notifications.

The investigation demonstrates that such a differentiation scheme can be applied to achieve the same performance for real-time services in situations where non-responsive real-time traffic is 5 – 10 percent of the overall traffic. In order for the ECN based control regime to be incentive compatible, it should be combined with a price per received congestion notification. Thus used, the experiments showed that the greedy voice sources get duly punished for excess use of contested resources. The next chapter on the GSP experiments indicates how to translate dynamic prices related to ECN marks into more predictable charges.

The experiments indicate that a byte oriented marking algorithm should have given better control over delay performances, and an empirical functional relation between RED parameters and delay rates, considering byte oriented measures, was found.

4 Providing services with quality and price guarantees (GSP)

The introduction of dynamic pricing of internet traffic brings about a large disadvantage for customers – they can no longer predict the price of an internet session. One may envisage a new service alleviating the unpredictability of price and quality. The new service offers connections of limited length and bandwidth and with a certain quality guarantee against a fixed charge. The provider of this guaranteed service (GSP = Guaranteed Service Provider) is himself charged by the underlying ISP (Internet Service Provider) with a price varying dynamically according to congestion. The GSP may nevertheless make a profit, because some customers are willing to pay more than the average dynamic price for a predictable service.

In the next sections we describe a GSP business model, and describe how a GSP may construct tariffs and guarantees from traffic observations. The GSP model and its workings are described in more detail in [17]. We tested whether our GSP could make a profit in the (experimental) surroundings as described in Chapter 2, involving VoIP and web traffic, and dynamic prices corresponding to received ECN marks. The rather disappointing results are reported in Section 4.3. Further research or an adjustment of the business model is needed, as pointed out in Section 4.4.

4.1 GSP business model

The ISP offers a best-effort service where the receiver has to pay a charge per received marked packet. It is up to the ISP how to mark packets, but presumably he marks packets when the network is congested.

The GSP uses the ISP's service to offer the subscribers a more predictable service: The subscriber may choose a connection type from a table containing

- Maximum connection time
- Peak rate (eventually more complicated traffic descriptors may be used)
- Maximum blocking rate (applied to multiple connection requests)
- Maximum delay rate (i.e., ratio of lost packet plus packets delayed by more than x ms over sent packets).
- Origin/destination, period of day, or other parameters differentiating the tariff

Associated with each connection type the GSP offers a fixed charge (*before* the connection is made!), and a reimbursement in case the maximum delay rate is not met.

Once the subscriber has made a choice of connection type, the GSP may choose to either accept the connection or not. If he accepts the connection, he sends it as best-effort through the ISP's network and charges the subscriber the promised fixed amount. If the subscriber was not satisfied with the connection's quality, the GSP reimburses the subscriber. The ISP does not distinguish between the GSP's connections.

Figure 11 gives a schematic overview of the traffic flow and the GSP's and ISP's position with respect to the subscriber.

The GSP's profit is the difference between the fixed charge paid by the subscriber, and the dynamically varying charge he needs to pay the ISP. In case the quality guarantee is not met, he incurs the reimbursement fee in addition.

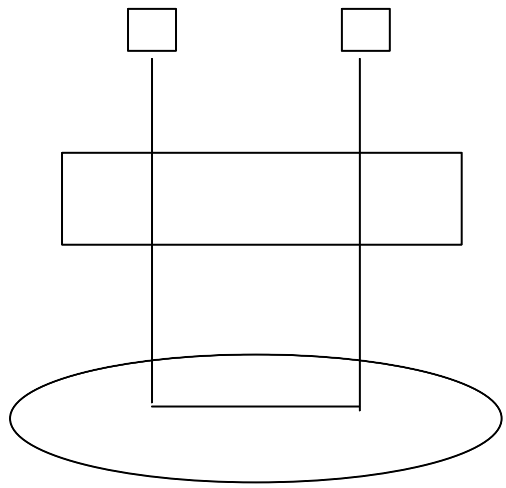

Figure 11: Schematic view of scenario

Business assumptions

In principle the GSP may be identical with the ISP. But here we assume that the GSP is independent of the ISP. This has the following consequences:

- The GSP does not know the exact nature of traffic inside the ISP's network. If the GSP produces a sizeable amount of the ISP's traffic, he may know more about the traffic, but we assume rather that the GSP is small, to begin with. The GSP may, however, differentiate between different traffic mixes by designing tariffs for different times of day.
- The GSP has no access to the ISP's routers.
- The GSP does not necessarily know the inner workings of the ISP's marking algorithm (e.g. RED + RED parameters).
- The GSP cannot monitor the quality of the subscriber's connection, but he may send test traffic.

4.2 Ideal GSP in operation

R. Lorentzen [17] developed a model on how the GSP may set the connection charges so as to have a certain expected profit. Here, we present a simplified model.

The GSP infers quality and price levels from test traffic sent through the ISP's network. The test traffic simulates the varying connection types. In particular test traffic is characterized by

- Maximum connection time = sample period = T
- peak rate (or other traffic descriptors)

With each traffic sample are collected

- The relative frequency M of ECN marks = marked packets / sent packets ("marking rate")
- The relative frequency L of packets lost or delayed by more than 15ms ("delay rate")

M and L can be regarded as random variables.

From the collected statistical material the GSP determines

- A guaranteed maximum call blocking rate: B^G
- A guaranteed maximum delay rate: L^G

- Price per connection: V, which the GSP's customer has to pay.

Other economical data, which the GSP has little or no influence on, are the reimbursement R to the customer if the delay rate exceeds L^G, and the cost per marked packet, to be paid to the ISP by the GSP. The cost per marked packet is assumed to be 1 for simplicity.

When a customer requests a connection of given type, the GSP blocks it, if the present marking rate M is above the $(1-B^G)$-median $ß^G$ of the distribution of marking rates M. In this way, he can guarantee that at most a fraction of B^G connections are blocked. The reason for blocking connections is that the non-blocked connections have a hopefully smaller delay rate than the blocked connections.

The profit from an accepted connection is

- $-$ R – M, if L > L^G
  V – M, otherwise

Denote by p the probability of an accepted connection having delay rate larger than guaranteed, and by μ the average cost of an accepted connection. On the average, the profit from an accepted connection is then

$$P = V(1-p) - Rp - \mu$$

Given a profit margin P, reimbursement R, estimators for the average cost μ and for the reimbursement probability p, the GSP may now compute the price V of the connection.

The full paper [17] derives estimators for μ, p, and $ß^G$, assuming that M is Beta-distributed.

A central assumption in the model is that the marking rate M is positively correlated with the delay rate L. Otherwise it would make no sense to block connections, because one would not be able to guarantee a better delay rate by blocking connections of high marking rate.

4.3 Experimental GSP

4.3.1 Experiment goals

We wanted to test experimentally whether there was a correlation between marking rate M and delay rate L in test traffic samples sent over a link with background traffic. Maybe one could even construct a function L ˜ l(M) relating marking rate with delay rate (depending, of course, on the connection type).

We also wanted to test how the marking rate (which is the GSP's cost) depended on parameters of the test traffic. If, for example, the marking rate depends in a predictable way on the peak rate (or other traffic descriptors), it is sufficient for the GSP only to send test traffic of one fixed peak rate instead of several. Moreover, the GSP may more easily construct a tariff depending on the traffic descriptor of the requested connection.

Lastly, we wanted to test the dependence of marking and delay rate distribution on parameters outside the GSP's influence, for example, the background traffic mix, or the marking algorithm.

4.3.2 Experiment set-up and parameters

The experimental set-up was the same as described in Section 2. Background traffic of types VoIP and web traffic was sent through a bottleneck link of 40 Mb/s. In addition we also sent GSP test traffic. The test traffic consisted of three parallel udp streams of constant packet rate and packet size. This was to simulate several types of constant bit rate connections.

The router in front of the bottleneck link ran a RED/ECN marking algorithm similar to the one used in the previous ECN experiments (Section 0). The background traffic was made to react to ECN marking as before: The web traffic (tcp) reacted to marked packets as if they were lost, and the udp streams (representing VoIP background traffic and GSP test traffic) did not react to packet marks. The udp packets' ECT bit was set, which had the effect that the marking algorithm *marked* these packets instead of *dropping* them. The queue size was large enough to make the loss rates negligible.

In this set-up it was possible to vary the following parameters:

- Background traffic mix specified by
- number of VoIP users in background traffic (note that according to the VoIP user model in Section 2.2.1 some of these VoIP users may be idle)
- number of web users in background traffic (some of which may be idle)
- Three parallel test traffic streams, each of which had a constant
- packet rate and
- packet size
- RED/ECN parameters. Here we kept $max_p = 1$, $w_q = 1$, and $min_{th} = max_{th} / 6$, as in Section 3.1.2. The only free parameter is
  max_{th}

The standard setting of the parameters was:

- 700 VoIP users, 30 web users
- 3 test streams of a udp payload size of 100 bytes (=128 bytes with udp and ip headers) sent with a rate of 10/20/30 packets per second, respectively. This corresponds to bit rates (including headers) of 10/20/30 Kbps. The smallest test stream is of the same packet size and packet rate as a single VoIP stream.
- $max_{th} = 360$

This setting appears also in the previous ECN experiments (Table 4). Most of our experiments changed only one or two of the parameters in the standard setting.

In a post-processing step, each of the three test traffic streams was cut into samples of length *T*. The standard length was 3 minutes, which is the same as the average connection length of a VoIP connection in the VoIP model (Section 2.2.1). For each sample we stored the

- Marking rate = marked packets / sent packets
- Delay rate = (lost packets + packets delayed by more than 15 ms) / sent packets.
- Marking rate of the previous sample

Each experimental run lasted for one and a half hours, excluding a warming-up phase of half an hour, inside which the web and VoIP traffic generation should approach steady state. Inside each run of 1.5 hours about 30 samples were generated, each of three minutes length.

The tested parameter settings were:

VoIP users	Web users	max_{th}
700	30	360
		564
900	20	360
		420
		444
		564
1100	10	360
		564

Table 6: Tested parameter settings

Note that these parameters already appear in the diffserv/ECN comparisons reported in the previous chapter (see Table 4). In addition we made a few more runs with the standard setting (700VoIP/30web/360 max_{th}) with different types of test traffic. More on this below.

Due to limitations in time, we were not able to systematically test our findings, especially those that did not correspond to our expectations. Therefore we were not able to revise some of our parameter choices. But even with our limited test material, we are able to make some statements, as reported in the next few sections.

4.3.3 Correlation cost/quality

When plotting marking rate (cost) against delay rate, we expected to see a picture like the one in Figure 12. By discarding the samples (= connections) of high cost to the right of the circled point, it should be possible to limit the delay rate, i.e. guarantee a certain quality.

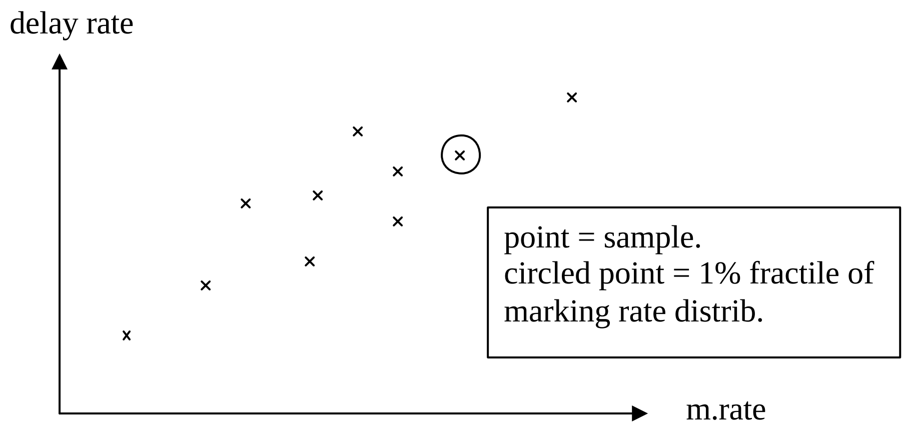

Figure 12: Ideal picture: Marking rate correlated with delay rate

However, a typical picture looks like the one in Figure 13. Here, it is not easy to find a point on the marking-rate axis, beyond which one may discard connections, in order to guarantee a quality for the remaining connections.

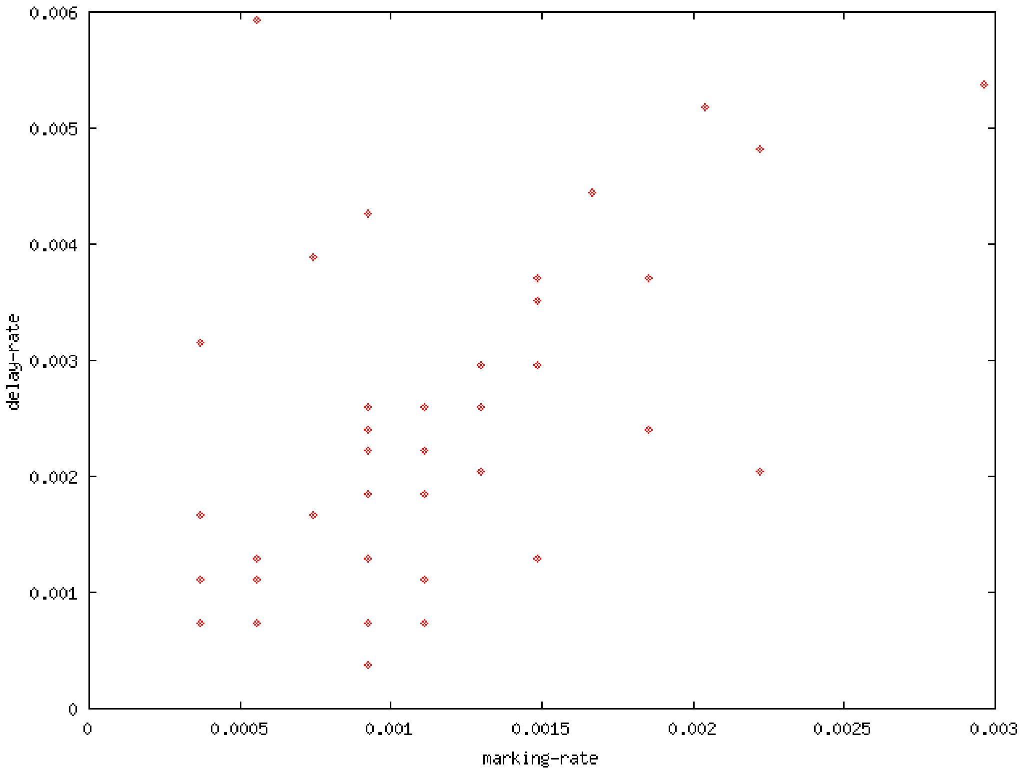

Figure 13: Marking rate versus delay rate for 41 three-min samples of 30 Kbps with 700VoIP/30web background traffic, max_{th}=360, two-hours run

Indeed, the correlation of *marking rate versus delay rate* was never higher than 0.75 in all our runs. In all except two runs it was below 0.5. In the example shown in Figure 13, the correlation was 0.52.

In Figure 14 we plotted the development of delay rate and marking rate over time, for the same example as in Figure 13. The marking and delay rates seem to increase at the same times, but not with varying factors.

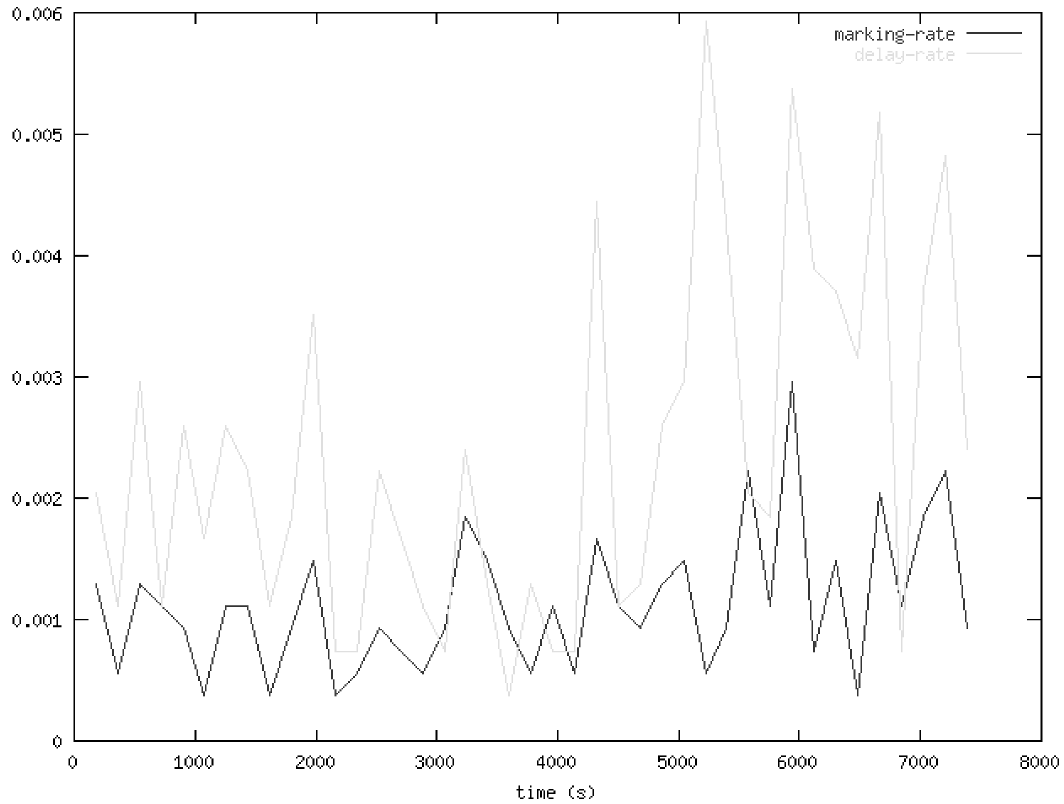

Figure 14: Development of delay rate and marking rate over time

The correlation seemed to increase with decreasing sample size (thereby also increasing the number of samples). This is however irrelevant for the GSP, because customers will probably not request connections of 20 seconds.

The correlation seemed to increase, too, with increasing test traffic size and increasing measurement length. However, as time ran out, we were not able to properly test these claims.

The marking rate in one sample and the marking rate in the previous sample were even less correlated: between –0.17 and 0.34. Even worse: the delay rate of one sample was weakly negatively correlated with the marking rate of the previous sample in more than half of our runs. The correlation did not improve by decreasing the averaging interval from three minutes to 20 seconds, as one should have expected. This indicates that in our setting it makes no sense for the GSP to discard a connection when the previous connection has a “large” cost.

4.3.4 Dependence of average cost on test traffic rate

Inside one measurement, we compared the marking rates of each of the three parallel test streams, averaged now over the whole measurement period. For the measurements listed in Table 6, the three test streams were a 10/20/30 Kbps streams, generated as 128B packets with a packet rate of 10/20/30 packets per second. The marking rate, defined as the number of marked packets over the number of sent packets inside the measurement period, appeared to be the same for all three streams, independent of the experiment. This means that the 30 Kbps stream that sends three times as many packets as the 10 Kbps stream, costs three times as much in terms of collected marks.

Now we compared three test streams sending at the same packet rate of 10 packets per second, but at three different packet sizes, namely 128/228/328 bytes. This corresponds to bit rates of, respectively, 10/17.8/25.6 Kbps. We repeated this measurement three times for the standard traffic mix of 700 VoIP/30 web users, and $max_{th} = 360$, and varying measurement lengths. In none of the measurements were there a noticeable difference between the average marking rates of the three streams. This means that the stream of higher bit rate costs as much as the stream of lower bit rate! Clearly, it pays off to increase the packet size in the chosen setting.

Probably, if the RED/ECN algorithm, which marks the packets, had taken into account queue size in terms of bytes, not only in terms of number of packets, streams of higher bit rate would have been punished by a higher marking rate independent of their packet size.

Last, we compared the average marking rate of the three test traffic streams with the average marking rate of aggregate udp traffic (consisting of test traffic and of the background VoIP traffic). Recall that a single VoIP connection lasts for an average of three minutes, and sends packets of size 128 bytes at a constant packet rate of 10 packets per second, similar to the smallest test traffic stream. In contrast to the test traffic streams, the number of simultaneous VoIP connections was highly varying (with a maximum of 700). We performed measurements for the standard test parameter setting of 700 VoIP/30 web users, max_{th}=360. Three measurements were done, in which only the set of test traffic streams varied. The duration of each measurement was 30 minutes only (due to processing restrictions in our analysis software), excluding a warming up phase of 30 minutes.

As said before, when comparing only the test traffic streams, there was no difference in average marking rate, nor in the average delay rate inside one measurement. But the average marking rate of the aggregate udp traffic was at least twice as high as the average marking rate of any test traffic stream. In addition, the average delay rate of the aggregate udp traffic was at least 1.35 as high as the average delay rate of any test traffic stream. This observation indicates that the “cost” of a connection (in terms of marked packets) is dependent not only on the peak rate but also on the relationship of peak to mean rate (burstiness). Note that the “connection” we are talking about here is very long: 30 minutes. The GSP may be advised to construct tariffs based on peak and mean rate also for shorter connections. Wischik [21] investigates, for certain marking algorithms, the dependence of marking rates on the “effective bandwidth” of the traffic (a measure lying between the mean and the peak rate, depending on the traffic stream’s burstiness and the traffic mix).

Table 7 gives the details for the measurements, which compared marking rates for test traffic and aggregate udp traffic.

4.3.5 Dependence of average cost on background traffic mix and marking algorithm

As we already observed in the diffserv/ECN comparisons (cf. Table 4), variations in background traffic and delay rate have a significant effect on the delay rate and the marking rate. This is also true with respect to the delay rate, when observing only test traffic of 10/20/30 pps and 128 B packets. Table 8 gives the details.

scenario	pps	packet size	kbps	delay rate	marking rate
small test streams	10	128	10	1.92E–3	9.32E–4
of increasing pps,	20	128	20	1.89E–3	9.32E–4
const. packet size	30	128	30	2.02E–3	1.15E–3
	aggreg.	udp traffic		3.07E–3	2.45E–3
small test streams	10	128	10	4.60E–3	1.22E–3
of increasing pack.	10	228	17.8	4.66E–3	1.75E–3
size, const. pps	10	328	25.6	4.84E–3	1.52E–3
	aggreg.	udp traffic		7.38E–3	4.41E–3
large test streams	80	128	80	2.80E–3	1.09E–3
of increasing pps,	159.7	128	159.7	2.74E–3	1.20E–3
const. packet size	238.9	128	238.9	2.75E–3	1.19E–3
	aggreg.	udp traffic		3.71E–3	2.01E–3

Table 7: Comparison of marking rates between traffic streams. Experiment with 700 VoIP/30 web users, max_{th} = 360, duration = 30 min

VoIP	web	max_{th}	test str.	delay rate	marking rate
700	30	360	1	4.08E–3	1.30E–3
			2	3.61E–3	1.19E–3
			3	3.74E–3	1.19E–3
700	30	564	1	5.70E–2	5.27E–4
			2	5.51E–2	3.86E–4
			3	5.50E–2	3.70E–4
900	20	360	1	7.34E–4	8.47E–4
			2	7.34E–4	1.01E–3
			3	6.52E–4	9.78E–4
900	20	420	1	2.28E–3	4.90E–4
			2	2.31E–3	4.90E–4
			3	2.53E–3	3.82E–4
900	20	444	1	5.95E–3	6.59E–4
			2	5.99E–3	8.37E–4
			3	5.97E–3	7.90E–4
900	20	564	1	3.37E–2	3.57E–4
			2	3.37E–2	3.76E–4
			3	3.29E–2	3.95E–4
1100	10	360	1	0	1.32E–4
			2	0	4.52E–4
			3	7.53E–5	5.65E–4
1100	10	564	1	5.66E–3	1.32E–4
			2	5.99E–3	1.32E–4
			3	5.74E–3	1.38E–4

Table 8: Delay and marking rates for varying background and marking threshold. Test streams of 10/20/30 Kbps and 10/20/30 pps. Duration 1.5 h

The effect of the marking threshold on the marking rate is not very significant. It needs to be remarked that when we repeated the first experiment of 700 VoIP/30 web users and max_{th}=360, the average marking rates varied between 9.32E–4 and 1.30E–3, and the average delay rates varied between 1.89E–3 and 5.42E–3. It can also be seen from the table that the marking rate was approximately the same for all test streams inside one measurement.

It is unfortunately not possible to compare delay rates from Table 4 directly with delay rates from Table 8, because the former delay rate was defined as the frequency of packets lost or delayed more than 16 ms, whereas the latter was defined with 15 ms. The results in Table 7 are not affected by this unintended discrepancy – there we used 15 ms for both test and aggregate traffic.

Concerning the GSP, the results show that variations in background traffic and in marking parameters must have an effect on his tariff and quality guarantee calculations. Time-varying background traffic variations can be compensated by time-varying tariffs, but some research is needed to find out how stable the marking algorithms are. If routes change unpredictably from one domain (with one marking algorithm) to another domain (with another marking algorithm), the GSP's costs and quality guarantees become unpredictable as well.

4.4 Repairing the experimental and the ideal GSP

Although our experiments were not very extensive, we can draw some conclusions, give some advices for "upcoming" GSPs, and propose further research directions.

The low correlations between marking rates and delay rates in our experiments do not provide a good basis for computing quality guarantees from observations of marking rates. Maybe a simpler tariff computation model, which assumes that quality is independent of marking rate, would give a safer profit. The correlations seem, however, to be quite dependent on the marking algorithm, the number of observations, the size of test traffic, and the sample length.

The even lower correlations (almost non-existent) between the quality and marking rates of two connections following each other in time do not promise well for connection acceptance control based on observation of marking rates. It is quite possible that the fault lies within the chosen marking algorithm. Maybe also real-world web and VoIP traffic is less bursty than the simulated traffic used in our experiments.

As expected, average delay and marking rates depend on the traffic mix and the marking algorithm.

An upcoming GSP would thus have to put much more effort than us into investigating

- The effects of the ISP's marking algorithm and the ISP's background traffic mix on quality, cost, and correlations
- The sensibility of quality guarantees and connection acceptance control
- The stability over time of the ISP's marking algorithm (which the our presumed GSP has no control over)
- The stability over time of the ISP's background traffic, and the derivation of time-dependent tariffs

Concerning the GSP's *tariff structure* our findings were:

- If the underlying marking algorithm counts packets only (not their sizes) then the GSP's costs and thus his tariffs may be based on packet rates, not byte rates.
- For constant packet rate traffic of constant sizes the cost and thus the tariff may depend linearly on the packet rate.
- For non-constant (that is, bursty) traffic, the cost and thus the tariff are dependent not only on the mean rate, but also on other traffic descriptors (peak rate?)

We may also need to reconsider the assumptions imposed on our *ideal GSP* (see Section 4.1). Since most of our findings are dependent on the marking algorithm and the background traffic mix, it appears that a GSP who possesses knowledge or even control over marking and traffic may have a much better business chance. This leads to a change in the GSP's *business model*, namely the integration of the GSP with the ISP. The next paragraph describes what kind of business this GSP/ISP might be.

Business model for GSP integrated with ISP

The new GSP (integrated GSP/ISP) applies an automated queue management scheme in his routers, by marking packets with ECN marks. All traffic is best-effort traffic, i.e., there is only one queue in each router, and no classification and prioritization of traffic. The new GSP does not quite trust all his customers to use protocols reacting in a nice and predictable way to ECN marks, so he enforces the proper reaction by charging customers proportional to received ECN marks. He knows, however, that some (or all) of his customers prefer stable prices to dynamically changing prices, and stable qualities to changing qualities, so he offers some or all customers a more predictable service: The customer may thus choose a connection type from a table containing

- Maximum connection time
- Peak rate (eventually more complicated traffic descriptors may be used)
- Maximum blocking rate (applied to multiple connection requests)
- Maximum delay rate (i.e., ratio of lost packet plus packets delayed by more than x ms over sent packets).
- Origin/destination, period of day, or other parameters differentiating the tariff

Associated with each connection type the GSP offers a fixed charge, and a reimbursement in case the maximum delay rate is not met.

Once the customer has chosen the connection type, the GSP may choose to either accept the connection or not. If he accepts the connection, he sends it through his network and charges the customer the promised fixed amount. If the customer was not satisfied with the connection's quality, the GSP reimburses the customer.

The new GSP does not really incur costs in the form of ECN marks. We may nevertheless assume that he attempts to maximise a "profit" per connection type, whose ingredients are, as before, the fixed charge paid to the customer, the reimbursement fee, and a "cost" equal to the average number of generated ECN marks for this connection type. These "costs" are no longer paid to an ISP, but one may consider them as "lost revenue" anyway, because the GSP/ISP could have charged them to the customer, especially if he offers a dynamically priced "best effort" traffic class alongside the "guaranteed service".

In this way, the GSP/ISP keeps his network simple (best-effort with marking), the subscribers of the "guaranteed service" get a service of predictable prices and quality, and the economic incentives carried by ECN marks are not lost.

The new GSP has the advantage that he controls the marking algorithm and possesses some knowledge of the traffic mixes. This may lead to an improved algorithm for tariff computations. Anyway, before an ISP decides to upgrade his business with a guaranteed service, he will have to consider the same advices given to the GSP above, namely he needs to investigate in more detail:

- The effects of the marking algorithm and the background traffic mix on quality, cost, and correlations of cost/quality
- The sensibility of quality guarantees and connection acceptance control.

5 Concluding remarks

Our experiments comparing diffserv with an ECN-marking regime show that (under certain assumptions) it is possible to replace diffserv classifying and scheduling in the routers by simple best-effort queueing together with marking of packets under congestion. It will still be possible to control delay for delay-sensitive traffic like VoIP without compromising too much the quality of more flexible traffic (in our case download times of web traffic).

Combining charges with ECN marks offers even better control with the traffic, but may be unacceptable to customers expecting predictable charges and predictable quality. We present a theoretical guaranteed service provider (GSP) that offers volume-based tariffs in an environment of ECN-based charges. The business model is interesting, especially when the GSP's role is combined with the internet service provider's (ISP) role. A GSP reselling a dynamically priced service from a separate ISP may encounter high economic risks due to the GSP not possessing critical knowledge of the ISP's network. Experiments with the GSP mainly point out which factors (marking algorithm, traffic mixes) are critical for the GSP and need more investigation.

Acknowledgments

We would like to thank Brynjar Viken for his help in setting up the technical platform, and for the development, together with Poul Heegaard, of the traffic generation system. Moreover, we give credit to Ralph Lorentzen, whose mathematical specification of the GSP's workings laid the ground for our GSP experiments. The work underlying this report was in part financed by the Fifth Framework Project 11429 (M3i), and on this occasion we express our thanks for many valuable discussions with our partners in M3i.

References

1 Andreassen, R. (ed). *Requirements specification: Reference model*. Fifth Framework Project 11429 (M3i), 2000 (Deliverable 1). – URL: http://www.m3i.org/results/main.html

2 Athuraliya, S., Li, Victor H., Low, S. H., Yin, Qinghe. REM: Active queue management. *IEEE Network*. May/June 2001

3 Briscoe, B. (ed). *Architecture*. M3I-project, 2000 (Deliverable 2). – URL: http://www.m3i.org/results/main.html

4 Cho, K. *The design and implementation of the ALTQ traffic management system*. dissertation, Keio University. January 2001 – URL: ftp://ftp.csl.sony.co.jp/pub/kjc/papers/dissertation.ps.gz/

5 Diot, C., Iannaccone, G., May, M. Aggregate traffic performance with active queue management. *ACM SIGCOMM Computer Communication Review*, 31 (3), 4–13, 2001.

6 Floyd, S. *RED: Discussions for setting parameters*. [online] – URL: http://www.icir.org/floyd/REDparameters.txt

7 Floyd, S., Jacobson, V. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, 1 (4), 397–413, 1993. – URL: http://www.icir.org/floyd/papers/red/red.html

8 Gibbens, R. J., Kelly, F. P. Resource pricing and the evolution of congestion control. *Automatica*, 35, 1969–1985, 1999. – URL: http://www.statslab.cam.ac.uk/~frank/PAPERS/evol.html

9 Graham, D. Donnelly, S. F., Martin, S. et al. Nonintrusive and accurate measurements of unidirectional delay and delay variation on the internet. In *Proceedings of INET 1998*. – URL: http://www.comms.uab.es/inet99/inet98/6g/6g_2.htm. Dag project homepage http://dag.cs.waikato.ac.nz/

10 Heegaard, P. E. GenSyn – a Java based generator of synthetic Internet traffic linking user behaviour models to real network protocols. In *Proceedings from ITC Specialist Seminar on IP Traffic Measurement, Modeling and Management*, Sept. 18-20, 2000

11 Heegaard, P. E., Lu, M. *GenSyn – Java based generator of synthetic Internet traffic*. SINTEF (Technical report STF40 A99078)

12 Hollot, C. V., Misra, V., Towsley, D. et al. A control theoretic analysis of RED. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies INFOCOM 2001*, 1510–1519

13 IETF. Ramakrishnan, K., Floyd, S., Black, D. *The addition of Explicit Congestion Notification (ECN) to IP*. 2001 (RFC 3168)

14 Karsten, M. *GSP/ECN technology & experiments*. M3i-project, 2002 (Deliverable 15.3, Part 3, Appendix 3). – URL: http://www.m3i.org/results/main.html

15 Kelly, T. An ECN-probe based connection acceptance control. *ACM SIGCOMM Computer Communication Review*, 31 (3), 14–25, 2001.

16 Kunniyur, S., Srikant, R. Analysis and design of an Adaptive Virtual Queue (AVQ) algorithm for active queue management. In *Proceedings SIGCOMM'2001* (ACM Computer Communication Review 31 (4), 123–134, 2001.)

17 Lorentzen, R. *GSP pricing policies*. M3i-project, 2002. (Deliverable 11, Part 2, Appendix B)

18 Siris, V., Courcoubetis, C., Margetis, G. Service differentiation in ECN networks using weighted window-based congestion control for various packet marking algorithms. In *Proceedings of 2nd International Workshop on Quality of Future Internet Services, Coimbra, Portugal, 2001* (Lecture Notes on Computer Science 2156, Springer-Verlag, 190–206, 2001)

19 Songhurst, D. (ed). *Global interaction models*. M3I-project, 2002. (Deliverable 11, Part 1). – URL: http://www.m3i.org/results/main.html

20 Vicari, N. *Models of WWW-Traffic: a comparison of Pareto and logarithmic histogram models*. University of Würzburg, Institute of Computer Science, Germany, 1998 (Research Report Series, Report no. 198)

21 Wischik, D. How to mark fairly. In *Proceedings of Workshop on Internet Service Quality Economics, MIT*, 1999. – URL: http://www.statslab.cam.ac.uk/~djw1005