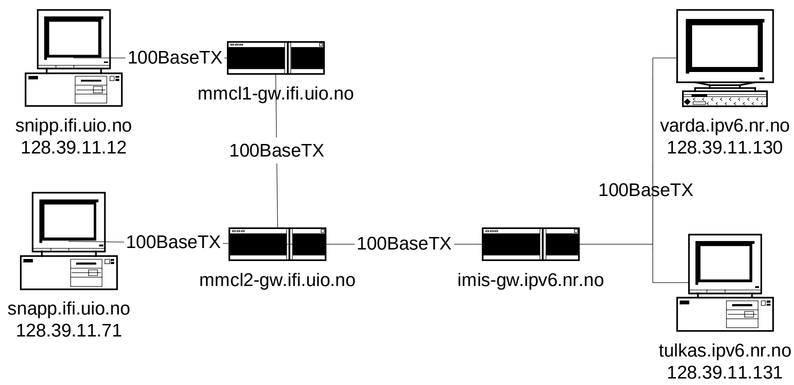
ENNCE -
Final QoS-controllable Network Infrastructure



IMEDIA/02/01

Lars Aarhus
Espen Fjellheim
Jan-Olav Eide

February 2001

NR Norsk Regnesentral
ANVENDT DATAFORSKNING

Notat/Note

Tittel/Title:
ENNCE - Final QoS-controllable Network Infrastructure

Dato/Date: February
År/Year: 2001

Notat nr.:
Note no.: IMEDIA/02/01

Forfatter/Author:

Lars Aarhus, Espen Fjellheim, Jan-Olav Eide

Sammendrag/Abstract:
This document describes the final status of the QoS-controllable experimental network infrastructure being developed as part of the ENNCE project. In particular, a number of experiments intended to demonstrate the effect of the implemented QoS mechanisms are presented, as well as the different implementation approaches taken during the project period. The results from the experiments involving resource reservation (RSVP) and video conferencing (VIC) prove that the infrastructure now works as intended, and can provide a platform for further QoS experiments.

Emneord/Keywords: Next-generation Internet, network infrastructure, QoS, RSVP

Tilgjengelighet/Availability: Open

Prosjektnr./Project no.: 801001

Satsningsfelt/Research field: ENNCE

Antall sider/No. of pages: 15

Norsk Regnesentral / Norwegian Computing Center
Gaustadalléen 23, Postboks 114 Blindern, 0314 Oslo, Norway
Telefon 22 85 25 00, telefax 22 69 76 60

Content

1 Introduction

This document describes the final status of the QoS-controllable experimental network infrastructure being developed as part of the ENNCE project. In particular, a number of experiments intended to demonstrate the effect of the implemented QoS mechanisms are presented, as well as the different implementation approaches taken during the project period.

ENNCE [13] is a fundamental research project funded by NFR for a period of 5 years from 1997 to 2002. Project partners include Norwegian Computing Center (NR), and Department of Informatics at University of Oslo (Ifi/UiO), among others. The main purpose of the ENNCE project is to implement and experiment with next generation networked computer environments supporting Quality of Service (QoS). WP1 focus on QoS negotiation models at application/network level, whereas WP2 ("Multe") focus on QoS support in middleware. This document is only concerned with the QoS-controllable network infrastructure developed in WP1.3.

In the remainder of this introduction the topology of the QoS-controllable infrastructure is presented. Section 2 gives a description of the overall goal of the network infrastructure. In section 3, the different implementation approaches selected during the time of the project, along with the problems and obstacles encountered are discussed. Section 4 describes a number of experiments conducted in order to demonstrate and verify the effect of the QoS-controllable mechanisms. Finally, section 5 summarizes the main conclusions to be drawn from this activity.

1.1 QoS-controllable network infrastructure

In the beginning of the ENNCE project the focus was to establish an infrastructure supporting IPv6 [2], as part of a national effort to construct a high capacity, wide area test network between academic institutions in Norway. IPv6 is the next generation IP protocol intended to gradually replace the current version, IPv4. The new protocol supports a much larger address space, has built in support for security (IPSec) and mobility, as well as improved network administration mechanisms (address reconfiguration) [5, 6].

The result of this national effort was Testnett [14], a network intended for and able to support the most demanding of multimedia application. The experimental network software in routers and hosts in these first implementations use dual stack solutions with co-existing IPv6/IPv4 addresses. A single stack IPv6 approach has been thoroughly tested at University of Tromsø in the Pasta project [15]. Since the summer of 2000, IPv6 production addresses have been possible to obtain from the various regional network address registrars e.g. RIPE. However, the experimental address plan proposed in [2] still has to be used in ENNCE and Testnett, as IPv6 production support is still not widespread.

In addition to IPv6, another important element in next generation networks is believed to be network resource reservation, and in particular network throughput reservation, i.e. bandwidth/traffic capacity. Controllable QoS at IP level requires some kind of control/management over these resources. Thus, the next phase of the ENNCE project focused on local experiments with RSVP between NR and IfI/UiO. RSVP (Resource ReserVation Protocol) is the next generation protocol for signalling the amount of bandwidth to be reserved at routers along a network path. The reserved traffic is described using different token bucket models [7].

RSVP is only a signalling protocol, which, contrary to popular belief, will not differentiate traffic alone. Additionally, more advanced traffic management methods (queuing mechanisms) than standard first-in, first-out queuing are needed in the routers. A typical example is Weighted Fair Queuing (WFQ), which places incoming packets in different queues/classes, and serves the output interface in a weighted round-robin fashion without starving any of the queues. Together, the two mechanisms combined can provide new service classes in addition to best-effort, such as *controllable load*, which emulates a lightly loaded network [8], and *guaranteed*, which provide firm delay boundaries [9].

Both resource reservation protocols and traffic management methods have to be supported in all network nodes to have any effect on QoS control. Thus, local experiments with complete control over all nodes are advantageous, which is why only the local part at NR/Ifi/UiO, and not the wide area Testnett is used for demonstration purposes. The network topology of the QoS-controllable infrastructure consists of four machines, two at each location, and two intermediate routers to provide a two-hop connection. See figure 1 below.

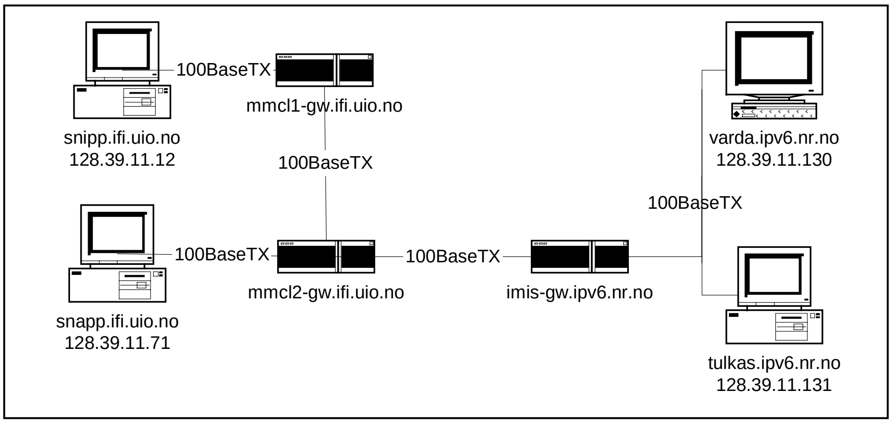


Figure 1: QoS-controllable network topology

All the details regarding the QoS-controllable network infrastructure are given in the forthcoming chapters.

2 Goal

The overall goal of the ENNCE WP1.3 activity was to have the QoS-controllable infrastructure in place, and be able to demonstrate networked QoS-support between two hosts over two intermediate routers. In order to demonstrate any QoS control effect, the network (or rather the link) between the routers has to be *constrained*. This means, too much traffic has to be sent over that link resulting in network congestion, and packets being dropped from the router output queue. If not, the network will have sufficient resources, and there will be no effect and no point of resource reservation in the first place.

When fulfilling the overall goal, a number of problems of mostly engineering nature, were encountered. The most challenging were:

- How to configure the network? This includes selecting the right hardware components with interoperable router and host software across the infrastructure.
- How to constrain the link between the intermediate routers using which technology? Alternatives discussed were Ethernet traffic shaping (see Figure 2 below), and ATM permanent virtual circuits (PVCs).
- How to demonstrate the effect of QoS control? Both *quantitatively*, by calculating the difference between one reserved flow (with accompanying unreserved noise) and one unreserved flow (again with accompanying unreserved noise). And *qualitatively*, by visually observing the difference when using e.g. a video conference application.

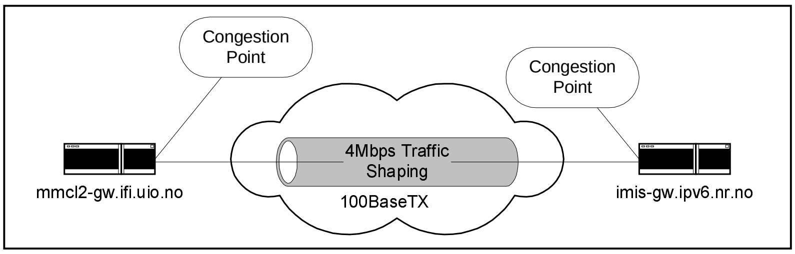


Figure 2: Constraining network conditions

Preferably, the QoS-controllable infrastructure should use IPv6 addressing and software. The drawback is that this requires all application network software to be rewritten using new IPv6 socket API. The obvious advantage is that this will provide a genuine next-generation network demonstration, not "only" demonstration the use of RSVP over IPv4. To be able to rewrite application software, access to source code is a mandatory requirement. This is one of the reasons, the MInT multimedia conferencing framework was originally selected as the host application software for ENNCE WP1 [3].

3 Implementation approaches

During the ENNCE WP1.3 activity, three different implementation approaches have been tried in order to realize the QoS-controllable infrastructure. The approaches are presented below. Unfortunately, only the last approach proved successful, due to hardware and software interoperability problems with QoS support.

3.1 First approach (late 1998)

The first attempt was made in late 1998 using RSVP, experimental IPv6 router and host software, and ATM permanent virtual circuit between the routers. The approach revealed serious software incompatibilities between the components in the network configuration. The main problem detected was no support for advanced traffic management methods on the two ATM router interfaces in either IPv4 IOS 11.3 or experimental IPv6 IOS. Also, there was lack of support for RSVP in IPv6 IOS. Therefore, instead it was decided to use Ethernet interfaces and 4Mbps traffic shaping between the intermediate routers.

The details of this approach are presented in [1].

3.2 Second approach (summer 2000)

In September 1999 Cisco released a new version of experimental IPv6 IOS. A second attempt to realize the infrastructure using the new router software was made in summer 2000. This time the approach included Ethernet traffic shaping between intermediate routers, in addition to RSVP and IPv6 software. The hosts were Sun machines running Solaris, as that was the preferred platform for the MInT multimedia conferencing framework, which by then had been selected as the host demonstration application. Unfortunately, this second approach was also unsuccessful. The main problem was lack of interoperable RSVP support between the network nodes.

As it was a failure, the details of this approach are only presented as a short note in Appendix A.

3.3 Final approach (late 2000)

The first two attempts had highlighted the lack of RSVP support in experimental IPv6 IOS. In order to have any kind of QoS-controllable infrastructure in place at all, it was decided in a final approach, to use IPv4 IOS 12.1 router software instead, with RSVP and Ethernet traffic shaping. The hosts were desktop PCs running Linux. Now it finally worked as it should!

The approach is not next generation in every aspect, since IPv4 addressing is used, but at least network QoS control is easily demonstrated. Also, there is actually very few differences between IPv4 and IPv6 addressing, when it comes to how resource reservation (RSVP) is applied. To our knowledge, a similar infrastructure has previously only been demonstrated once in Norway, and then using a serial line between the intermediate routers [11], not Ethernet traffic shaping.

Details of all the conducted QoS demonstration experiments are presented in chapter 4.

In October 2000 Cisco released another new version of experimental IPv6 IOS. According to Cisco, support for RSVP is still lacking, and will not be included for IPv6 addressing until 2001, when the experimental IPv6 IOS is merged with the production IOS 12.2. This merger was initially supposed to happen two years ago.

Immature and incompatible router software has been the main reason for the delay of ENNCE WP1.3. In hindsight, changing to alternative router software, and implementing on Linux would probably have been a better approach. Another source of delay has been the MInT multimedia framework. Despite claims of the opposite, the software wouldn't run properly on Linux, only Solaris. In hindsight, the selection of MInT as the host demonstration application was an unsuitable one, as the source code was not as platform independent as expected.

4 Demonstration experiments

A number of experiments to demonstrate the effect of the QoS-controllable infrastructure have been conducted. This chapter presents the results. The experiments focused on bandwidth reservation using RSVP, and involved both automatically generated traffic (by MGEN) and compressed video (by VIC). Unfortunately, some additional experiments had to be postponed, since too much project time had already been spent trying to set up the infrastructure properly.

4.1 Test environment

The network topology used in the experiments is the same as given in figure 1. Ethernet traffic shaping is employed on the router interfaces to constrain the intermediate link to 4Mbps as illustrated in figure 2.

Operating systems

Host	*Platform*	*Operating System*
snapp.ifi.uio.no	x86	Red Hat 6.2
snipp.ifi.uio.no	x86	Red Hat 6.2
tulkas.ipv6.nr.no	x86	Red Hat 6.2
varda.ipv6.nr.no	Sun Sparc	Solaris 2.5.1

Router	*Platform*	*Operating System*
imis-gw.ipv6.no	Cisco 7206	IOS 12.1
mmcl2-gw.ifi.uio.no	Cisco 7206	IOS 12.1

RSVP software

On the hosts, ISI's Linux distribution of RSVP daemon is used:
http://www.isi.edu/div7/rsvp/release.html

The reservations are made using the included rtap application.

On the routers, RSVP software is part of IOS.

Multi-Generator MGEN

The MGEN Toolset is used to automatically generate and receive UDP packets of different sizes between the hosts: http://manimac.itd.nrl.navy.mil/MGEN/

Video conferencing tool

A modified version of VIC having support for Linux, using the Video4Linux driver, is used as the video conferencing tool. VIC version 2.8 ucl-1.1.3:
http://www-mice.cs.ucl.ac.uk/multimedia/software/vic/

Original version of VIC: http://www-nrg.ee.lbl.gov/vic/

Calculation of video bandwidth

As there is no way of directly measuring the bandwidth generated by VIC over a specific time period, this is calculated by subtracting throughput reported from MGEN from the max throughput. All the test scenarios are run in an isolated environment with no other traffic consuming bandwidth on the 4Mbps link, so it is safe to assume that bandwidth not used by MGEN is used by VIC. Due to some network overhead we have measured the maximum throughput to be ~3700Kbps.

The VIC-throughput can then be calculated by the following equation:

VIC-throughput ~= 3700Kbps – MGEN-throughput

A note on VIC's compression algorithms

All the tests are done with the H263+ [10] video compression algorithm included with VIC. This algorithm splits the video image into a grid, and only transfers cells which have changed since last transfer, or that has not been updated for a certain time. Thus, a video flow with lots of motion will generate up to 3Mbps, which is VIC's maximum transfer rate, and a video flow with little or no motion can generate down to only a few KBps.

This means that even if 3Mbps are reserved, VIC might not generate this amount of traffic. To compensate for this we are moving the camera while measuring bandwidth to generate as much traffic as possible.

4.2 RSVP and MGEN experiment

The purpose of the tests in this experiment is to examine how the QoS-controllable infrastructure handles multiple reservations and admission control. Six different scenarios with varying combinations of reserved and unreserved MGEN flows are investigated. The reservation service class is *controllable load* as specified in the Tspec and Flowspec below.

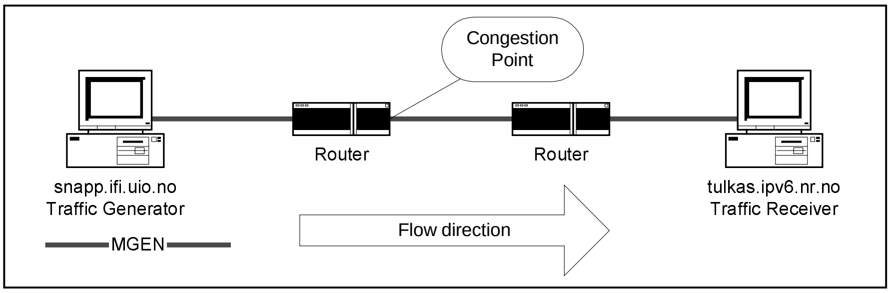


Figure 3: Network topology in RSVP and MGEN experiment

Tspec

data rate (bytes/sec)	128000
token bucket size (bytes)	128000
peak data rate (bytes/sec)	128000
minimum policed unit size (bytes)	1
maximum policed unit size (bytes)	1500

Flowspec

average data rate (bytes/sec)	128000
token bucket size (bytes)	128000
peak data rate (bytes/sec)	128000
minimum policed unit size (bytes)	1
maximum policed unit size (bytes)	1500

Results

Scenario 1: No reservations

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	1024	1024	125	0	770.211
0002	MGEN	1024	1024	125	0	769.509
0003	MGEN	1024	512	250	0	740.426
0004	MGEN	1024	256	500	0	688.201
0005	MGEN	1024	128	1000	0	603.182
0001–0005	MGEN	5120				3571.528

Table: Results from MGEN.

Scenario 2: One reservation

```
To              From           Pro DPort Sport Next Hop      I/F   Fi Serv BPS Bytes
128.39.11.131 128.39.11.71  UDP 5001  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
```

Table: Reservations as reported from mmcl2-gw.ifi.uio.no

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	1024	1024	125	1024	1025.476
0002	MGEN	1024	1024	125	0	705.439
0003	MGEN	1024	512	250	0	679.025
0004	MGEN	1024	256	500	0	630.988
0005	MGEN	1024	128	1000	0	553.098
0001–0005	MGEN	5120				3594.027

Table: Results from MGEN.

Scenario 3: Two reservations

```
To              From           Pro DPort Sport Next Hop      I/F   Fi Serv BPS Bytes
128.39.11.131 128.39.11.71  UDP 5001  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5002  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
```

Table: Reservations as reported from mmcl2-gw.ifi.uio.no

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	1024	1024	125	1024	1026.040
0002	MGEN	1024	1024	125	1024	1026.629
0003	MGEN	1024	512	250	0	576.422
0004	MGEN	1024	256	500	0	535.702
0005	MGEN	1024	128	1000	0	469.650
0001–0005	MGEN	5120				3634.444

Table: Results from MGEN.

Scenario 4: Three Reservations

```
To              From           Pro DPort Sport Next Hop      I/F   Fi Serv BPS Bytes
128.39.11.131 128.39.11.71  UDP 5001  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5002  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5003  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
```

Table: Reservations as reported from mmcl2-gw.ifi.uio.no

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	10024	1024	125	1024	1025.387
0002	MGEN	1024	1024	125	1024	1024.530
0003	MGEN	1024	512	250	1024	1024.636
0004	MGEN	1024	256	500	0	326.698
0005	MGEN	1024	128	1000	0	286.165
0001–0005	MGEN	5120				3687.415

Table: Results from MGEN.

Scenario 5: Four reservations

We now make reservations for 100% of max bandwidth.

```
To              From           Pro DPort Sport Next Hop      I/F   Fi Serv BPS Bytes
128.39.11.131 128.39.11.71  UDP 5001  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5002  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5003  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5004  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
```

Table: Reservations as reported from mmcl2-gw.ifi.uio.no

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	1024	1024	125	1024	966.988
0002	MGEN	1024	1024	125	1024	957.091
0003	MGEN	1024	512	250	1024	936.453
0004	MGEN	1024	256	500	1024	846.235
0005	MGEN	1024	128	1000	0	4.172
0001–0005	MGEN	5120				3710.939

Table: Results from MGEN.

Scenario 6: Reservations on all flows

The router lets us make reservations which exceeds its maximum output rate

```
To            From          Pro DPort Sport Next Hop      I/F   Fi Serv BPS Bytes
128.39.11.131 128.39.11.71  UDP 5001  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5002  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5003  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5004  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
128.39.11.131 128.39.11.71  UDP 5005  6000  128.39.11.226 Fa4/0 FF LOAD 1024K 128K
```

Table: Reservations as reported from mmcl2-gw.ifi.uio.no

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	1024	1024	125	1024	881.529
0002	MGEN	1024	1024	125	1024	900.683
0003	MGEN	1024	512	250	1024	768.685
0004	MGEN	1024	256	500	1024	740.822
0005	MGEN	1024	128	1000	1024	418.969
0001–0005	MGEN	5120				3710.688

Table: Results from MGEN.

Evaluation of test results

The main observation is that reservations are handled properly by the infrastructure. As long as the total available reservable bandwidth is not exceeded, the throughput rate equals exactly the reservation rate as seen in Scenario 2 to 4. Some additional observations are given below.

Admission control

The router allows reservations which exceeds its maximum output rate. Above we have a 4Mbps link with reservations for 5Mbps. It has not been tested if the router lets us make reservations for over 100Mbps.

Unfairness due to packet size

As can be seen in Scenario 1, packet size influences throughput. Flows with larger packets gets a higher throughput than flows with smaller packets.

Unfairness in queueing algorithm

In Scenario 6, we have reserved 1Mbps for each flow. One would expect to get the same throughput as in Scenario 1, which has no reservations. However, the unfairness becomes even greater. Weighted Fair Queuing is used in both scenarios, though the difference is that in Scenario 6 we have a Token Bucket filter on each flow. Each reservation is made with an average rate of 1Mbps and a bucket size of 128KB.

The flows are supposed to receive tokens based on byte count, not packet count.

4.3 RSVP and VIC experiment

The purpose of the tests in this experiment is to demonstrate how the QoS-controllable infrastructure handles video flows from VIC, in the presence of MPEG flows. Three cases are investigated: video with no reservation, video with reservation, and video with reservation but no motion. Both the quantitative and qualitative differences are reported. Again, the reservation service class is *controllable load* as specified in the Tspec and Flowspec below.

VIC has a maximum output rate of 3Mbps. At this rate it generates approximately 15-16 frames per second with the H263+ video compression algorithm.

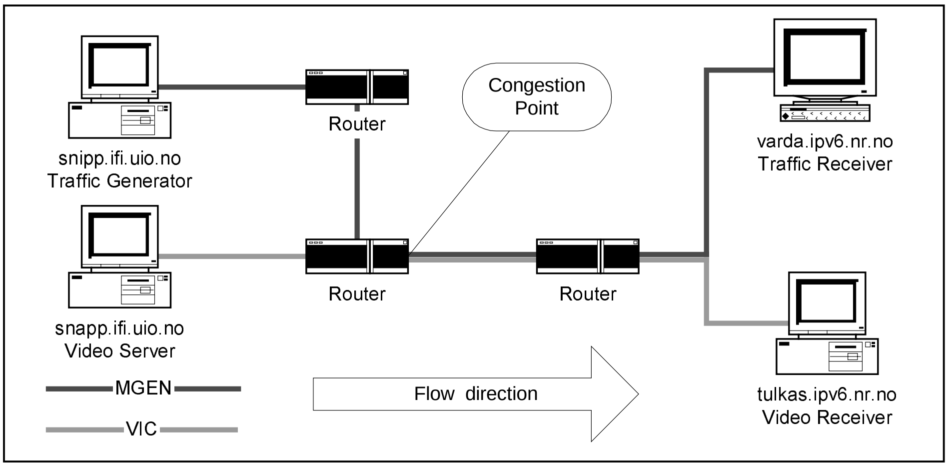


Figure 4: Network topology in RSVP and VIC experiment

Tspec

data rate (bytes/sec)	375000
token bucket size (bytes)	375000
peak data rate (bytes/sec)	375000
minimum policed unit size (bytes)	1
maximum policed unit size (bytes)	1500

Flowspec

average data rate (bytes/sec)	375000
token bucket size (bytes)	375000
peak data rate (bytes/sec)	375000
minimum policed unit size (bytes)	1
maximum policed unit size (bytes)	1500

Video with no reservation

When 5Mbps of MGEN traffic is introduced on the link the video quality is heavily detoriated. VIC's calculated throughput rate is decreased to ~696Kbps.

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	1024	1024	125	0	647.677
0002	MGEN	1024	1024	125	0	647.368
0003	MGEN	1024	512	250	0	622.877
0004	MGEN	1024	256	500	0	578.737
0005	MGEN	1024	128	1000	0	507.303
0001–0005	MGEN	5120				3003.963
	VIC	<=3000			0	~696

Table: Results from MGEN.

Video with reservation

A reservation for the video flow with a data rate of 3Mbps and a token bucket size of 375KB is made. VIC's calculated throughput rate is increased to ~2877Kbps. There is no longer any visual detoriation of the video quality.

```
To             From           Pro DPort Sport Next Hop       I/F   Fi Serv BPS Bytes
128.39.11.131 128.39.11.71   UDP 7000  1026  128.39.11.226 Fa4/0 FF LOAD 3M    375K
```

Table: Reservations as reported from mmcl2-gw.ifi.uio.no

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	1024	1024	125	0	177.457
0002	MGEN	1024	1024	125	0	177.360
0003	MGEN	1024	512	250	0	170.578
0004	MGEN	1024	256	500	0	158.599
0005	MGEN	1024	128	1000	0	139.067
0001–0005	MGEN	5120				823.061
	VIC	<=3000			3000	~2877

Table: Results from MGEN.

Video with reservation but no motion

The video camera is now facing a static object so that little traffic is generated from VIC's compression algorithm. The results prove that even though a reservation for 3Mbps is made for the video flow, other flows can borrow from the reserved bandwidth, when all is not utilized.

```
To             From           Pro DPort Sport Next Hop       I/F   Fi Serv BPS Bytes
128.39.11.131 128.39.11.71   UDP 7000  1026  128.39.11.226 Fa4/0 FF LOAD 3M    375K
```

Table: Reservations as reported from mmcl2-gw.ifi.uio.no

Flow Id	Source Program	Flow Size (Kbps)	Packet Size (bytes)	Packets pr second	Reservation (Kbps)	Throughput (Kbps)
0001	MGEN	1024	1024	125	0	753.759
0002	MGEN	1024	1024	125	0	752.654
0003	MGEN	1024	512	250	0	725.023
0004	MGEN	1024	256	500	0	673.285
0005	MGEN	1024	128	1000	0	589.798
0001–0005	MGEN	5120				3494.519
	VIC	<=3000			3000	~205

Table: Results from MGEN.

Evaluation of test results

Again, the main observation is that reservations are handles properly by the infrastructure. Specifically, the difference between reserved and unreserved video flows can easily be observed visually. Additional observations have already been discussed under each test.

4.4 Other experiments

Originally, three additional experiments were planned, but had to be postponed

RSVP and MInT

The purpose of this experiment was to demonstrate that the QoS-controllable infrastructure worked in combination with the selected host application, the MInT multimedia framework. This was not possible due to problems running the software on Linux, which could not be solved as there was no longer developer support for the framework.

However, the RSVP and VIC experiment in chapter 4.3 essentially demonstrate the same type of multimedia conferencing configuration, as *video* is the most bandwidth demanding media.

RSVP and IPv6

The purpose of this experiment was to demonstrate that the QoS-controllable infrastructure worked with IPv6 addressing. As previously described, this was not possible due to lack of RSVP support in the IPv6 router software.

However, the experiments conducted with IPv4 addressing essentially demonstrate the same type of network infrastructure, as the only additional QoS support in IPv6 is the flow label whose usage has yet to be defined.

RSVP and Service Agent

The purpose of this experiment was to demonstrate the integration between the QoS-controllable infrastructure and the service agent for connection and QoS management developed in ENNCE WP 1.1 [4]. This is a very desirable goal, but judged too time consuming at the present stage of the project (too few resources left).

5 Conclusion

The QoS-controllable experimental network infrastructure is finally in place. The infrastructure supports resource reservation using RSVP. It has proved to be working in a number of demonstration experiments. The infrastructure can also provide a platform for further QoS experiments.

The goal of combining the two next generation protocols, RSVP and IPv6, has not been fulfilled. This is not critical as the QoS support in IPv6 is essentially the same as in IPv4, the new flow label with undefined behaviour notwithstanding. IPv4 has proved to be longer lasting than expected, which means that support for next generation IPv6 has been slower to develop than anticipated at ENNCE project start in 1997.

The main lesson learnt is that when you base the network infrastructure on vendor software, you should never trust software release dates. Specifically, RSVP has been part of production IOS for at least two years, whereas IPv6 was promised to be included two years ago, but is still not present as of February 2001.

In hindsight, a better alternative could have been to base the infrastructure on free source code router software from Linux, and spend time developing the necessary software additions to get RSVP and IPv6 working together. Instead, too much time and resources has been spent testing various software configurations, which in the end prove to be incompatible.

The resource consumption has meant that other next generation QoS mechanisms, such as differentiated service architecture providing lightweight and scalable traffic classification without per-flow state, has not been studied that thoroughly in the ENNCE project as was originally planned. Fortunately, differentiated services have instead been studied and implemented in Testnett within the UNINETT Diffserv project [12].

6 References

1. Aarhus, L. et al., *An Experimental Network Infrastructure Supporting QoS*, NR note IMEDIA/02/99, January 1999
2. Karlsen, T. et al., *Experimental IPv6 Network in Norway*, NR note IMEDIA/02/98, April 1998
3. Leister, W. and Holmes, P., *Characterization and Selection of Applications for ENNCE WP1*, NR note IMEDIA/03/99, January 1999
4. Leister, W. and Spilling, P., *A Service Agent for Connection and QoS Management in Multimedia Systems*, NR note IMEDIA/05/98, April 1998
5. Huitema, C., *IPv6 - The New Internet Protocol*, 2nd ed., Prentice Hall, 1998
6. Deering, S. and Hinden. R, *Internet Protocol, Version 6 (IPv6) Specification*, IETF RFC 2460, December 1998
7. Braden, R. et al., *Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification*, IETF RFC 2205, September 1997
8. Wroclawski, J., *Specification of the Controlled-Load Network Element Service*, IETF RFC 2211, September 1997
9. Shenker, S. et al., *Specification of Guaranteed Quality of Service*, IETF RFC 2212, September 1997
10. *Video coding for low bit rate communication*, ITU-T Recommendation H.263 v2 ("H.263+"), February 1998
11. Danielsen, K. et al, *RSVP Technology for UNINETT Multimedia Services: Pilot Project, Technical Tests*, HiMolde, August 1998
12. UNINETT Diffserv project, `http://www.uninett.no/testnett/differensiertIP.html`
13. ENNCE project, `http://www.ifi.uio.no/~ennce`
14. Testnett, `http://www.uninett.no/testnett/`
15. PASTA project, `http://www.pasta.cs.uit.no/index.html`

Appendix A

RSVP Signalling, a (failed) experiment

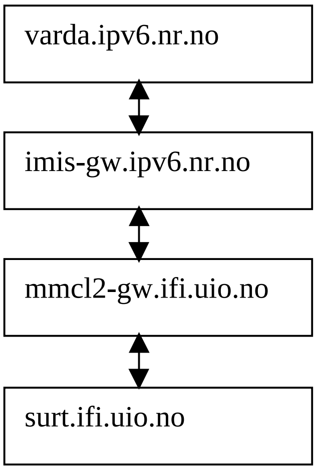
Jan-Olav Eide, Norsk Regnesentral
Jens Thomassen, IFI

Introduction

This experiment was carried out as part of the WP1.3 of the ENNCE project. The intention was to set up a simple 2-node network, which allowed RSVP-signalling [1]
The plan was then to run two applications in parallel on both nodes of this network. One application would be the multimedia tool-set MInT [2], which has RSVP capabilities. This application would use RSVP to set up required QoS parameters before transmitting an audio stream to a receiver. The other application would be a traffic generator/receiver MGEN [3] where no RSVP signalling was to be used. The plan was then to introduce traffic shaping on the communications channel connecting the two routers. We would expect to measure a deteriorating data transmission rate in the MGEN sender/receiver application and (largely) unaffected transmission rates for the MInT application. This would show that the RSVP support in the routers and computers was indeed able to provide an application with the required QoS.

Setup, versions etc.

The following simplified figure shows the network topology:



The routers/Solaris machines were connected via a 100Mb Fast Ethernet

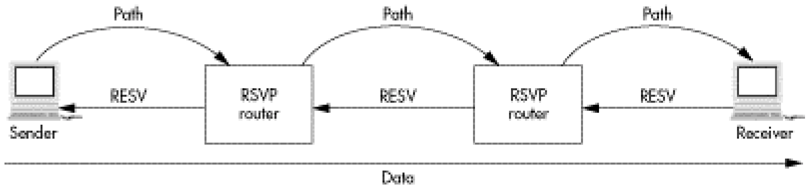
Software OS versions:

Machine	*OS version*	*RSVPD Version*
varda.ipv6.nr.no	Solaris 2.5.1	SunRSVP 0.5.5
surt.ifi.uio.no	Solaris 2.6	SunRSVP 0.5.5

Router	*OS version*
imis-gw.ipv6.nr.no	Cisco 7206 Experimental Version 12.0
mmcl2-gw.ifi.uio.no	Cisco 7206 Experimental Version 12.0

The RSVPD daemon was downloaded from http://playground.sun.com/pub/rsvp.

The following figure shows the signalling that takes place when setting up an RSVP reservation. A Sender sends a PATH message to the receiver, which upon receipt of this message sends a RESV message in the opposite direction, performing the actual reservation. For more details on this, see [4]



We quickly ran into the following problems :

- The router mmcl2-gw.ifi.uio.no (which is nearest the sender) complains that there 'is a non-RSVP capable host between previous hop and me'. We have been unable to determine the cause of this; there are no other hosts between this router and the sender. PATH messages do however appear to come through. They are recorded in the routers, and received by the rsvpd daemon on the receiver side.
- Upon receipt of a PATH message in a receiver, we attempt to send a RESV message. This message never leaves the machine, since the daemon for some reason finds that it has no PATH information available. This is also hard to explain, given the fact that it has just received a PATH message.
- When reversing the roles of sender and receiver, (varda sender, surt receiver), the rsvpd on varda fails with a 'bmptoif failed' message. We have not delved deeply into the source code of the daemon in order to determine the possible cause of this.

The same two problems also appear when we run the ISI rsvpd instead of the one in the SUN package. (downloadable from http://www.isi.edu/div7/rsvp/release.html)

Conclusion

We were thus unable to perform any of the experiments described above. SUN has a package called *Solstice,* which includes a RSVP daemon [5]. As this is a supported package, it could be expected that it would be more stable and generally more solid. This package is not bundled with the OS, but has to be purchased separately. We would recommend looking into this package if this experiment is to be pursued further.

References

[1] Braden, Zhang, Berson, Herzog, Jamin.
Resource ReSerVation Protocol (RSVP) Version 1 Functional Specification
http://www.faqs.org/rfcs/rfc2205.html

[2] Universal Scalable Multimedia in the Internet
http://www.fokus.gmd.de/research/cc/glone/projects/usmint/

[3] The MGEN Toolset
http://manimac.itd.nrl.navy.mil/MGEN/

[4] Metz, General-Purpose Signaling for IP
http://computer.org/internet/v3n3/w3onwire.htm

[5] http://www.sun.com/software/bandwidth/rsvp/docs/RSVPADMIN_1.html