IPv6 QoS Testing on Dual Stack Network

Christos Bouras Dimitris Primpas Kostas Stamos
Research Academic Computer Technology Institute, N.Kazantzaki Str., 26500, University Of Patras Campus, Rio, Patras, Greece &
Department of Computer Engineering and Informatics, University of Patras, 26500 Rion, Patras, Greece
TEL: +30 2610 {960375, 960316, 960316}
FAX: +30 2610 960358
E-MAIL: bouras@cti.gr, primpas@cti.gr, stamos@cti.gr

ABSTRACT

This paper presents our work regarding the testing and evaluation of DiffServ QoS mechanisms over IPv6 and IPv4 in dual stack software based platforms. IPv6 introduces some additional features (like flow label) and the current support for QoS mechanisms in IPv6 implementations approaches the corresponding QoS support in IPv4. Therefore, a number of tests with DiffServ QoS mechanisms applied on IPv6 traffic have been carried out on a testbed created specifically for this purpose, in order to validate the mechanisms and evaluate the router's overall performance. Our evaluations cover the load incurred to the routing devices from the implementation of the mechanisms, the level of support for QoS IPv6 features and the comparison of performance over IPv4 and IPv6.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General – D*ata communications.*

General Terms

Management, Measurement, Performance, Experimentation

Keywords

IPv6, Quality of Service, Differentiated Services

1. INTRODUCTION

A very challenging and demanding issue during the last years for all modern networks, NRENs and ISPs has been the design and management of Quality of Service architectures. As a result, there are a few cases of successful deployment of QoS services, supported over IPv4, using the available techniques. Nowadays, the future of IP networks seems to be the IPv6 protocol [3] that updates the existing IPv4 protocol. Its main benefit is that it solved the problem of the limited IPv4 address space, but it is also designed in order to offer a series of additional improvements over IPv4 in various areas such as autoconfiguration, network management, security, mobility and QoS [6][7]. Our work focuses on the combination of these two promising concepts, QoS and IPv6, as it examines how well they operate together [5][8]. Although many hardware vendors support the IPv6 protocol and many have implemented QoS over IPv6, there is a lack of thorough, real world testing of these implementations and a lack of evaluations for the combination of QoS mechanisms and IPv6.

There are 2 architectures for QoS that have been proposed and standardized by IETF. The first one is called Integrated Services (IntServ) and the second Differentiated Services (DiffServ). They follow different philosophy as they approach the topic of Quality of Service from different points of view. The IntServ architecture tries to provide absolute guarantees via resource reservations across the paths. On the other hand, DiffServ architecture is more flexible and efficient as it tries to provide Quality of Service using a different approach. It classifies all the network traffic into classes and tries to treat each class differently, according to the level of QoS guarantees that every class needs. The DiffServ mechanism for providing QoS guarantees is generally preferred over IntServ, because of its better scalability.

The DiffServ architecture [1] minimizes the number of actions to be performed on every packet at each node and builds a configuration that does not use a signaling protocol. Individual DiffServ mechanisms are applied on traffic aggregates rather than individual flows. The operation of the DiffServ architecture is based on several mechanisms. The first mechanism is the classifier that tries to classify the whole traffic into aggregates of flows (traffic classes), mainly using the DSCP field (Differentiated Service CodePoint). This field exists in both the IPv4 and IPv6 packet headers. In IPv4 it was part of the Type of Service field (ToS) and in IPv6 which is our focus in this paper, it is part of the Traffic Class field. In addition, the IPv6 packet header also has the field Flow label (20 bits) but it is still experimental and its use has only been recently standardized [4]. The operation of services based on DiffServ architecture also uses several additional mechanisms that act on every aggregate of flows. These mechanisms are packet marking, metering and shaping. In addition, in order to provide QoS guarantees it is necessary to properly configure the queue management and the time routing/scheduling mechanism. The most common queue management approaches use the Priority Queue, Weighted Fair Queue or Modified Deficit Round Robin mechanisms.

The paper is organized as follows: Section 2 describes the implemented QoS framework and the dual stack testbed. Section 3 presents the experiments that were conducted, aiming to investigate the proper operation of the QoS mechanisms, the router's performance and the overall QoS performance. Finally, section 4 describes the usage of the IPv6 flow label field and the conducted experiments, while section 6 is dedicated for conclusions and future work.

AAA-Idea'06, October 10, 2006, Pisa, Italy.

2. QUALITY OF SERVICE TESTS

The QoS framework that is applied to the network supports three QoS classes of service [13][8]:

- Premium IP service, based on Expedited Forwarding (EF) [2]. The service aims to provide absolute guarantees to a portion of traffic (absolute priority).
- Best Effort service (BE)
- Less than Best Effort service (LBE)

The network has been previously dimensioned in such a way that a specific quantity of traffic in each access interface can experience absolute guarantees according to the IP Premium service. Packets are identified by a unique DSCP value and mapped to a dedicated queue at each output interface in the backbone links. Table 1 presents the DSCP values that have been used in our experiments for mapping traffic to separate classes.

Table 1. The valid DSCP values

DSCP	Description
46	IP Premium (IPP)
8	Less than Best Effort (LBE)
0	Best Effort (BE)
Other	Best Effort (BE)

The QoS configuration enables policy in all access interfaces for Premium IP service. The policy action defines that the acceptable rate for the traffic is at most 10% of the capacity of the access link and in case that this limitation is violated then exceeding packets can either be dropped, classified to a lower class or the traffic flow can be shaped in order to adhere to the specified traffic profile. On the other hand there is no limitation for the transmission rate of LBE traffic. In addition, in all the access interfaces, there is a check whether there are unauthorised packets with DSCP value 46. Such packets are remarked to DSCP value 0, so that they are not inappropriately treated as IP Premium or LBE traffic. The classification and policing configuration has been applied to all the access interfaces of the network. In the output interfaces, the Class based Weighted fair Queueing scheduling mechanism has been configured properly in order to handle the packets from the 3 supported QoS services. In particular, 2 queues have been defined, one full priority queue and one normal. The packets that belong to EF class are enqueued in the priority queue, unlike all the other packets that are enqueued in the normal queue.

In order to extend the above framework and support IPv6 traffic, a number of tests were conducted. The experiments have been performed on a dedicated testbed (Figure 1) that included a core dual stack router that belongs to the CISCO 7200 series [14] with Gigabit and FastEthernet interfaces, a GigabitEthernet switch (CISCO catalyst 2970) and hardware-based traffic generators Smartbit 600 [16] with Gigabit Ethernet (GigE) interfaces. The SmartFlow ver.3.0 application was used to control the Smartbits and measure the generated test traffic.

The CISCO 7206 router that is part of the testbed, switches IPv4 and IPv6 traffic via software based modules. In addition, the Quality of Service module is based on a software implementation of Class Based Weighted Faired Queuing mechanism.

During our tests, the following parameters (defined by the IETF IP Performance Metric working group [12]) were used to qualify the QoS services provided:

- One-way or round trip delay. One-way delay is defined as the time needed by a packet to be transmitted and fully received by the destination. Round trip delay is the time it takes for a packet to be transmitted in both directions.
- Packet delay variation. Packet delay variation is measured for packets belonging to the same packet stream and shows the difference in the one-way delay that packets experience in the network. Large values for this value usually reveal queuing delays in the network.
- Packet loss. Packet loss is measured as the portion of packets transmitted but not received in the destination compared to the total number or packets transmitted.
- Packet reordering. Packet reordering is measured as the portion of packets that are delivered to the destination in wrong order compared to the total number of packets. There are multiple reasons that lead to packet reordering such as parallel forwarding engines in high performance routers, per packet load balancing on parallel physical links, routing path changes, etc

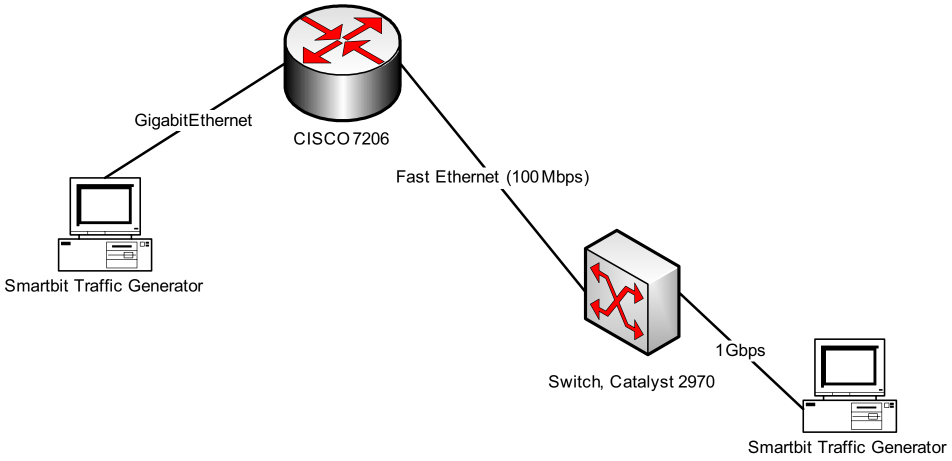

Figure 1. QoS testbed

3. EXPERIMENTS

3.1 IPv4-IPv6 Comparison

The first set of tests focused on classification mechanisms and access lists. A Smartbit traffic generator transmitted IPv6 traffic that was filtered successfully in the network via an IPv6-address based access list. In addition, tests about policing mechanisms were conducted. The traffic generators produced IPv6 traffic marked as Premium IP (EF) traffic that was policed at 50 Mbps while exceeding traffic was remarked to DSCP 0 (best effort).

The next set of tests aimed at investigating the router's traffic switching efficiency and identifying possible differences between IPv4 and IPv6 traffic. In particular, 3 different scenarios were performed, where the router was loaded with IPv4, IPv6, or a mix of IPv4 and IPv6 traffic. The traffic load was increasing, from 70Mbps to 130 Mbps. During those scenarios, the CPU load was measured by collecting info from the router's command line interface (CLI) every 5 seconds. The accuracy of the measurements is not optimal especially when the shell interface of the router freezes. Also, a 4th scenario was also tested, where a mixture of IPv4 and IPv6 Best effort traffic was loaded and standard QoS configuration was activated. This configuration enabled a high priority queue for premium traffic (marked with DSCP 46) and treated all the other traffic as best effort. All the scenarios used packets with size of 512 bytes and the results are presented in Figure 2.

The conclusion from this set of tests is that the CPU load increases (5-9%) when the router has to switch IPv6 traffic. An additional CPU load increase took place when QoS configuration was activated, even when all the traffic was best effort and therefore the required queue management very limited. This can be explained due to the fact that every process (IPv6 operations, QoS operations) is a separate software module that the router has to load each time it needs them.

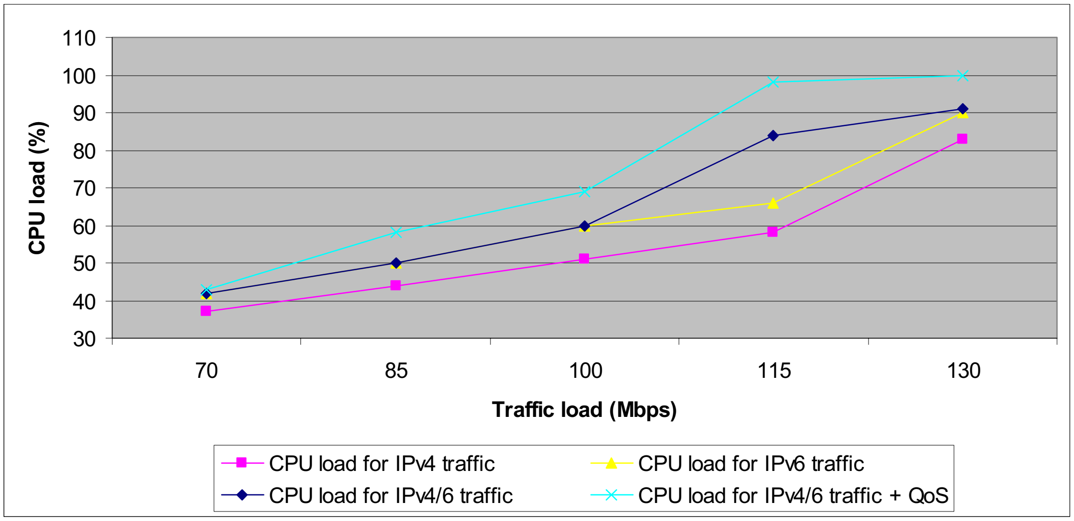

Figure 2. CPU load in software based platforms

Table 2. Packet loss for various packet sizes and traffic loads

packet size (bytes)	Packet loss percentage								
	IPv4 traffic (Mbps)			IPv6 traffic (Mbps)			IPv4and IPv6 traffic (Mbps)		
	85	100	115	85	100	115	85	100	115
128	0	0.17	100	0	14.83	100	0	14.99	100
256	0.01	0	12.55	0.01	0	12.55	0.02	0	12.55
384	0	0	12.62	0	0	12.61	0	0	12.62
512	0	0	12.66	0	0	12.66	0	0	12.66

Additionally, the following scenario was tested: the router was loaded with 85 Mbps for 10 seconds, then the load immediately increased to 100Mbps for a duration of another 10 seconds and finally the load was set to 115 Mbps for another 10 sec. These traffic loads were intentionally large in order to investigate the performance in extreme circumstances. They are not unrealistic though, since many networks may face such conditions for a number of reasons (large increase in network resources demand, malicious attacks, etc.) Our scenario was running in iterations, with each iteration increasing the packet size by 128 bytes. The scenario started with a packet size of 128 bytes and ended with a packet size of 1408 bytes. The scenario was tested for IPv4 only traffic, for IPv6 only traffic and also for a mixture of IPv4 and IPv6 in equal loads (Table 2). Each experiment was run for an adequate time duration so that transient phases due to the load changes do not dominate the results. A very interesting observation arises when the packet size was 128 bytes. In particular, the router showed an unusual behavior when the whole load approached the interface's capacity (100Mbps). For IPv4 only traffic, when the load was 100Mbps, there were a few packet drops and the CPU was high. When the load increased to 115Mbps, the router was extremely unstable and all packets were dropped. The router needed almost 10 sec after the completion of the transmission to resume normal operation. The worst behaviour was noticed when the traffic generator transmitted IPv6 only traffic, as the packet loss was significant high (almost 14.8%) when the load was 100Mbps. This behaviour was also demonstrated when the scenario was repeated with a mixture of IPv4 and IPv6 traffic. During this scenario, all traffic, IPv4 and IPv6, experienced the same packet loss. This situation (heavy traffic load with small 128-byte packets) can be characterized as a Denial of Service attack, and the software based platform is apt to such attacks. We are not able to identify the reason behind the demonstrated difference between IPv4 and IPv6 traffic, where the router's performance with IPv6 was decreased earlier in the experiment. We suspect this is due to the internal design and operation of the software modules for IPv4 and IPv6 switching. Finally, it is worth mentioning that for packet sizes greater or equal to 256 bytes, the router's performance is flawless.

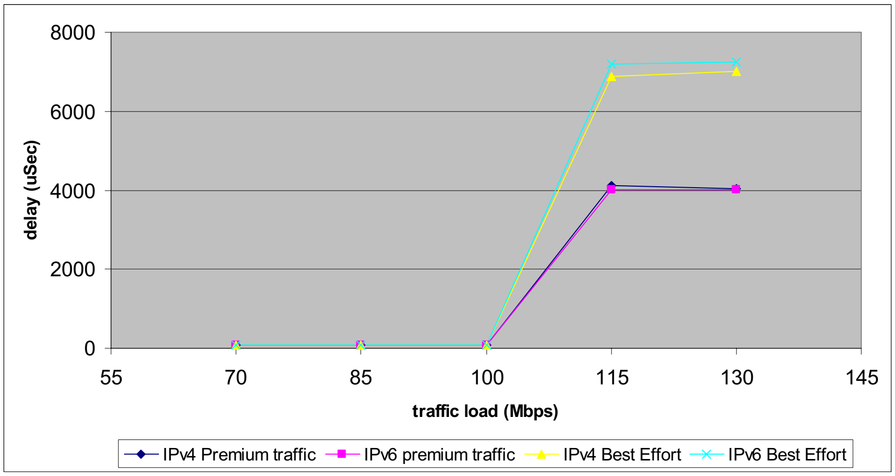

Figure 3. Average delay for Premium and Best Effort Traffic

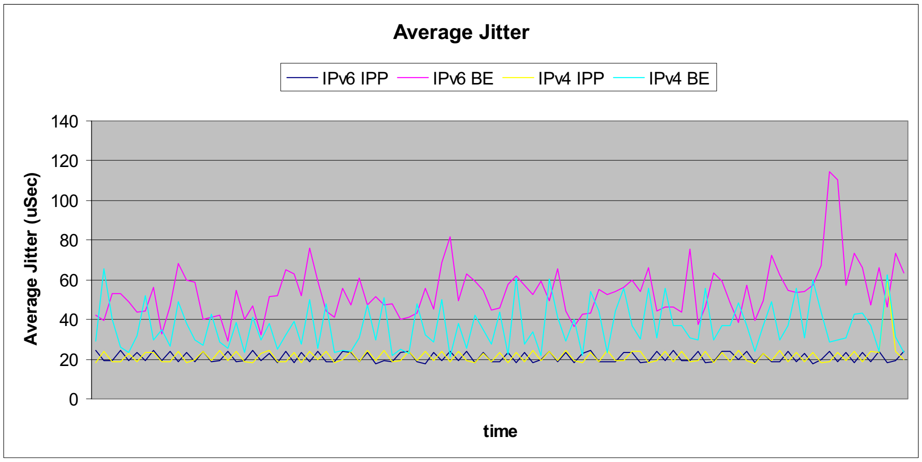

Figure 4. Delay variation for network load equal to 130Mbps

3.2 Queueing tests

Several tests were also conducted that aimed to investigate the basic QoS mechanisms. Therefore, classification based on DSCP, marking on DSCP field and policing mechanisms were successfully enabled. Also, configuration for queue activation on output interfaces was enabled and the queue mechanism operated normally. A typical reasonable result for the achieved delay is presented in Figure 3. This figure is the result of a scenario where the IPv4 and IPv6 premium traffic was 5% of the total traffic load. The experienced delay is identical for premium and best effort traffic as soon as the traffic load is less or equal to the interface's capacity. But, when the load exceeds the interface's capacity (and therefore the corresponding link is congested), premium's traffic delay is significantly less than best effort's delay. Finally, there were no differences between IPv4 and IPv6 traffic. In addition, no packet re-ordering was recorded and the packet loss for the IP premium traffic was also zero, as expected.

Finally, we measured the delay variation (average jitter) for network load equal to 130Mbps (both IPv4 and IPv6 in 50-50 percentage), which is presented in Figure 4. In this scenario, 10% of IPv4 traffic and 10% of IPv6 traffic was marked as IP Premium. We measured the average jitter that Best effort and IP Premium traffic experience during the experiment. In this scenario, there was heavy congestion and the result is expected. The IP Premium traffic experiences low delay variation, in conjunction with quite higher delay variation for Best effort traffic. No significant differences were noticed between IPv4 and IPv6 IP Premium traffic. On the other hand, IPv6 BE traffic seems to experience a little higher jitter than IPv4 BE traffic.

3.3 Tests with Real time traffic

After completing the above-described experimental stages, the QoS mechanisms that can provide QoS guarantees were set up and evaluated. As described above, the Class based Weighted fair Queueing scheduling mechanism had been configured in order to implement a high priority queue (Low Latency Queue). Such implementations are extremely suitable for real-time applications that need low delay, packet loss and jitter. Therefore, we tried a scenario with real-world traffic to investigate mechanism's efficiency. At the QoS testbed, we substituted the traffic generators with PCs that are connected through local LANs (via Cisco Catalyst 2950 switches). In order to simulate realistic conditions of traffic load (network's congestion) and to measure the performance of real-time applications that use the IP Premium QoS service, we generated background traffic using instances of the Iperf software traffic generator [17].

Initially, the network was loaded with background traffic that was a mix of TCP and UDP traffic. At this point we should note that we have started loading the network before inserting the foreground traffic, in order for the TCP to obtain a stable state. The foreground traffic was produced by a live videoconference (using the OpenPhone application based on the OpenH323 library [18]).

This test was performed for more than 5 minutes and we recorded the packets that were exchanged. We noticed that the background traffic had many packet losses because of the TCP protocol that was struggled by UDP. UDP traffic experienced only a few packet losses. On the other hand, the foreground traffic (OpenH323 videoconference) had zero packet loss and excellent quality. Figure 5 shows the achieved throughput by the videoconference.

A similar test was also performed for a second scenario where this time the IP Premium traffic was comprised by the live videoconference traffic and extra UDP traffic, in order to increase the load of priority queues. The overall result was identical, with the IP Premium traffic having zero packet loss and the videoconference taking place with excellent quality.

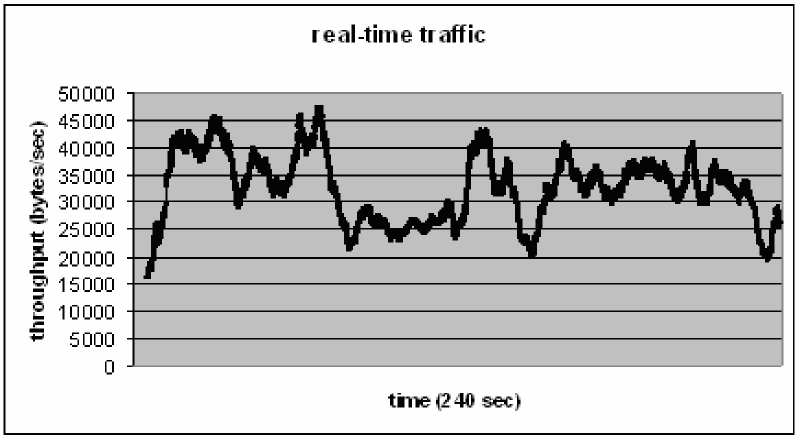

Figure 5. Videoconference's throughput

4. FLOW LABEL USAGE

Flow label is a field in the IPv6 header that has been designed in order to facilitate per flow QoS treatment. General rules for the Flow Label field were recently proposed in RFC 3697 [4], but specific use cases have not been described yet. Actually the flow label tries to integrate the classic Diffserv operation where traffic is aggregated into classes with the flow establishment. Therefore [4] defines that the 20-bit Flow Label field is used by a source to label packets of a flow and the zero value is used to indicate packets that are not part of any flow. Also, packets are processed in a flow-specific manner by the nodes (routers) that have been set up with flow-specific state and in any case the Flow Label value set by the source must be delivered unchanged to the destination node. Each established flow should expire when the flow is idle in order to improve the performance of routers. Therefore the RFC defines that the nodes should not assume that packets arriving 120 seconds or more after the previous packet of a flow still belong to the same flow, unless a flow state establishment method defines a longer flow state lifetime or the flow state has been explicitly refreshed. Finally, to avoid accidental Flow Label value reuse, the source node should select new Flow Label values in a well-defined sequence (e.g., sequential or pseudo-random) and use an initial value that avoids reuse of recently used Flow Label values each time the system restarts. Also special care should be taken to prevent theft of service by spoofing flow label value.

Following the guidelines of [4], we performed a small set of tests focused on the flow label field and its possible usage in an IPv6 QoS service. The only available way to use the flow label field is via classification through IPv6 access lists, as it is not available as a matching criterion in class-maps (in the current versions of CISCO software). Therefore, we performed a successful experiment on the software based platform by activating an IPv6 access list that filtered traffic from source-destination addresses and also filtered packets with specific DSCP and flow label value. The successful execution of this experiment proved that the classification per flow label inside an aggregation (based on DSCP) was possible. This allowed us to test and provide policing per flow at the input interfaces in order to avoid unfairness in the policing function when it is applied in traffic aggregations. The forwarding in the output interfaces and the classification in the queues was performed using only the DSCP value.

The final result (fairness in policing function) is very important especially in critical applications such as VoIP, were the unfairness problems on aggregations can cause overall performance degradation. On the other hand the usage of flow label as an additional criterion for packet classification should be done very carefully. A central mechanism that will assign values for the flow label should be implemented. Also, the range of flow label values that will be used will only be valid in this one domain, since there is no standardized assignment or partitioning of the flow label pool yet. For VoIP applications, as mentioned above, the central call manager or gatekeeper can operate as the flow label distributor, allowing for better classification of traffic, separating it from other simultaneous VoIP calls.

5. CONCLUSIONS AND FUTURE WORK

The performed QoS tests indicated that the software based Cisco routers under test adequately support QoS mechanisms for IPv6 traffic. A noticeable result is that in the case of IPv6 traffic, the CPU load is a little greater than that in IPv4. Also under line card congestion, the software based platform becomes extremely unstable and denies service when it is loaded with small (128-bytes) packets. The latter effect is even more intense when the router switches IPv6 traffic.

In addition, during the tests, we noticed some limitations. In particular the command line interfaces (CLI) for IPv6 and IPv4 traffic were almost identical, but there are a few commands that were either not supported for IPv6 traffic or where different commands existed for IPv6 and IPv4. Additionally, router statistics on interface level do not differentiate IPv6 and IPv4 packets and, thus, it is not easy to count the number of IPv6 packets in a dual stack environment. Finally, the matching flow label value criterion in class-maps was missing and the only possible workaround is to use the flow label inside an IPv6 extended access list.

Those limitations are not critical, but nevertheless impede better management of IPv6 QoS services. Generally, as the portion of IPv6 traffic is currently significantly low compared to IPv4 traffic, an IPv6 QoS schema can be deployed in production networks.

In the future we intend to expand our work in this field by adding IPv6 support to QoS production services that we support on GRNET's network [13]. Also, the usage of flow label is taken under consideration in order to support per flow policing for selective applications such as VoIP and videoconferencing sessions. The latter is very demanding as the whole schema needs a centralized tool for flow label value assignment that will be compliant with the latest RFC guidelines.

6. REFERENCES

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

[2] RFC 2598, “An Expedited Forwarding PHB”, V. Jacobson, K. Nichols, K.Poduri, June 1999

[3] RFC2460, “Intenet Protocol, Version 6 (IPv6)”, S.Deering, R.Hinden, December 1998.

[4] RFC3697, “IPv6 Flow Label Specification”, J.Rayahalme et. al., March 2004.

[5] "QoS experiences in native IPv6 GRNET and 6NET networks" A. Liakopoulos, D. Kalogeras, V. Maglaris, D. Primpas, C. Bouras, The 2005 International Conference on Telecommunication Systems – Modeling and Analysis, Dallas, TX, USA, November 17 - 20 2005

[6] “IPv6 Campus Transition Experiences”, T. Chown, Proceedings of the 2005 International Symposium on Applications and the Internet (SAINT2005), Trento, Italy, 31 January - 4 February 2005.

[7] 6NET Deliverable D2.3.4v2, “IPv4 to IPv6 transition Cookbook for End site networks / universities”, June 2005.

[8] 6NET Deliverable D.4.4.2v2, “Report in QoS Tests, 2nd version”, 6NET project (IST-2001-32603), April 2005.

[9] ‘IP Quality of Service: the complete resource for understanding and deploying IP quality of service for Cisco networks’, S. Vegesna, Cisco Press, 2001

[10] “Deploying Guaranteed Bandwidth Service with MPLS”, Cisco White Paper

[11] SEQUIN: ‘Service Quality across Independently Managed Networks’, IST Project IST-1999-20841, (http://www.dante.net/sequin/)

[12] IP Performance Metrics (IPPM) Working Group, http://www.ietf.org/html.charters/ippm-charter.html, IETF

[13] "Techniques for DiffServ - based QoS in Hierarchically Federated MAN Networks - the GRNET Case" A. Varvitsiotis, V. Siris, D. Primpas, G. Fotiadis, A. Liakopoulos, C. Bouras, The 14th IEEE Workshop on Local and Metropolitan Area Networks (LANMAN 2005), Chania. Island of Crete, Greece, September 18 - 21 2005

[14] http://www.cisco.com

[15] http://www.grnet.gr

[16] Spirent SmartBits 600 Series traffic generators, http://www.spirent.com/.

[17] Iperf homepage, http://dast.nlanr.net/Projects/Iperf/

[18] The OpenH323 project, http://www.openh323.org and http://sourceforge.net/projects/openh323