Quality of Service Mechanism for MANET using Linux

Semra Gulder, Mathieu Déziel
Semra.gulder@crc.ca, mathieu.deziel@crc.ca

Abstract: This paper describes a QoS mechanism suitable for Mobile Ad Hoc Networks (MANET). Routing and Quality of Service domain (Task 5 group) of Interoperable Networks for Secure Communications (INSC) has chosen the Differentiated Services (DiffServ) to be the base mechanism for flow classification in all INSC coalition networks. For the multimedia services (voice and video), several different service classification mappings are proposed by INSC to investigate. In this paper, the most suitable classification and scheduling algorithm for the voice and video in the MANET is presented.

A real IPv6 test-bed is constructed to demonstrate the viability of the proposed approach. In the test-bed the mobile nodes employ the Optimized Link State Routing (OLSR) protocol for routing within the MANET using Linux routers. Results from a performance evaluation on this test-bed are presented.

I. Introduction

Bandwidth is the limiting constraint in most mobile ad hoc networks. Rerouting among mobile nodes causes network topology and traffic load conditions to change dynamically, making it difficult to support real-time applications with appropriate QoS. In military operational environment there is a need for different levels of QoS in order to give more priority to delay, jitter sensitive traffic (e.g. telephony, video, priority messaging). As explained in INSC "QoS Architecture" [1], the DiffServ model is used as a suitable QoS mechanism in all INSC coalition networks including MANET.

In MANET, there are two basic functionalities that the QoS mechanism has to perform at each node: marking of the unmarked packets and applying the QoS policies.

Marking of the packets means to set the "traffic class", TC, field of the packets in the IPv6 header. The marking of the packets can be done by the applications themselves or by an edge router. Since the applications chosen by INSC do not mark the packets, and also because every node in a MANET is a router, the marking may be performed by every node that supports QoS. Each node marks the packets it generates, as well as unmarked packets received for forwarding. Each of these marked packets get treated according to the QoS policies set for the MANET.

The traffic is divided in three classes of services, as defined by IETF [2]: EF (Expedited Forwarding), AF (Assured Forwarding) and BE (Best Effort). Four AF classes are defined (AF1, AF2, AF3 and AF4). Each of these classes is given some forwarding resources. The EF class is meant for low latency - low jitter traffic. The BE class is meant for all unclassified traffic, or for traffic that does not have any special QoS requirement. Minimal amount of resources is allocated to BE.

Beside the typical characteristics of the DiffServ QoS mechanism, additional requirements are imposed onto our design due to the mobility of the MANET nodes and the wireless link. In the wireless link, available data rate is not always known in advance (rate auto fallback might be used). Not only congestion, but also a poor link quality and nodes positions could impact the available rate. As a result, the total throughput can vary substantially. Because of this, it is impossible to guarantee a fixed rate to any of the traffic classes. Therefore, a QoS mechanism that shares the available link rate according to some weights, or according to some weights combined with priorities is a better solution than a mechanism that would guarantee a fixed rate to each class.

The Linux kernel provides number of small modules to perform QoS functionality for our design; Ip6tables for marking the traffic, policing filters for rate limiting, HTB[i] for queuing of outgoing traffic.

The idea of the marking system is to mark the Traffic Class (TC) field of the IPv6 header before it enters the QoS mechanism. Marking is done by using Linux ip6tables functionality. After being marked the packets are queued in their respective HTB queues to be sent out.

As shown in Figure 2, there are two test scenarios considered to demonstrate the viability of our design: server being one hop away and server being two hops away, through MPR (multipoint relay in OLSR) node.

[i] Hierarchical Token Bucket

Both of these test scenarios are tested with and without QoS enabled for audio and video.

II. The Proposed Architecture

In the INSC project, several combinations of per hop behaviour (PHB) are possible for audio and video traffic. They are summarized in the following table:

Solution	Audio	Video
1A	EF	EF
1B	EF	AF31
1C	EF	BE
2A	AF31	AF32
2B	AF31	BE

Table 1 Audio and video solutions

In order to base our "QoS for MANET" design on the most suitable solution for audio and video, all INSC proposed solutions are investigated.

Solutions 1C and 2B are implemented but not tested, since the delay imposed on to video in BE makes it an unpractical solution. As a result only solutions 1A, 1B and 2A are considered for testing in order to determine the best solution(s) for audio and video traffic. In a lightly loaded network conditions, all three gave good results compared to no QoS, both in terms of rate and latency. However, in a heavily congested network, solution 1A and 1B gave better results for both the audio and the video. In solution 1A both the audio and the video go to the EF class which requires the EF class to be split into two queues. Since not all routers currently support two EF classes, this solution is less compatible with existing equipments.

As a result, in our paper, the proposed design and the test results are based on solution 1B .

In the following sections, overview of HTB queuing mechanism, the purpose for rate limiting and the architecture of QoS for MANET by using HTB are presented.

A. HTB

In terms of functionalities, HTB basically provides the following: multiple classes can be created, and they can be assigned priorities and rates. Unused rate by some classes can be borrowed by other classes.

Multiple classes can be created inside a queue. Each of these classes can be further subdivided into two or more classes to create hierarchy. A leaf class is a class that has no child class attached to it. Other queuing disciplines can be attached to a HTB leaf class (typically stochastic fairness queuing (SFQ), token bucket filter (TBF) or first-in-first-out (FIFO)). When the resources assigned to a class are unused by that class, they can be borrowed by other classes.

A solution where a fixed rate is assigned to the classes is not a suitable solution for MANET, because the link data rate is not known in advance, and can vary dynamically. Each HTB class must have a "configured rate". However, in HTB, when the available rate on the link varies, the configured rate of the classes can be seen like a weight in the rate distribution algorithm. The rate is distributed according to the class's configured rate as well as its priority. In case of congestion all classes will suffer but the ones with the lower priorities will suffer more.

B. Limiting the Rate

In our design, it is desired to give a very high priority to EF traffic, to keep the latency to minimum. However, giving EF such a high priority also means that if too much EF traffic is sent by the MANET nodes, all other traffic could starve, including important traffic flows such as ICMP messages or routing protocol control messages. Since there is no admission control in DiffServ mechanism, nothing will prevent such a scenario to occur. One solution is to limit the rate of EF traffic to prevent this from happening. Another solution is to give a high priority to EF traffic up to a certain rate x, and redirect the EF traffic exceeding this rate x to an AF class, or to BE. In this report, the first solution is chosen (limit the rate and drop the rest).

Three different solutions were tested in order to find a suitable rate limiting mechanism for the EF traffic class. The first solution was to configure the HTB EF class so that its configured rate was equal to its ceil[ii] rate. This effectively limited the rate, but the latency of the packets flowing in the EF class becomes very high, which is not desirable for the EF traffic. The second solution was to use a TBF[iii] at the leaf of the HTB EF class to shape the traffic to a desired rate. This also limited the rate, but the latency was still high. The third solution was to limit the rate of EF traffic before it entered the HTB EF class with a policing filter. This solution produced better result than the others in terms of latency. The latency of the EF traffic was reduced significantly. Therefore, a policing filter is used to limit the rate of the EF class in our QoS queuing system design.

[ii] An upper value on the rate.

[iii] Token Bucket Filter

C. QoS for MANET Architecture with HTB

Initially two queuing mechanisms were designed to support QoS in the MANET: one using HTB only and one using PRIO[iv] + HTB. The outcome of the tests performed on these two mechanisms guided us to make recommendation to INSC on a suitable QoS mechanism to be used in the MANET. Results of the HTB only mechanism will be reported since, according to the test results, the HTB only queuing mechanism is more suited to implement QoS in the MANET.

The queuing mechanism designed by using HTB for Solution 1B is shown in Figure 1. In the first level, EF, AF and BE traffic are separated into three queues, the EF having the highest priority.

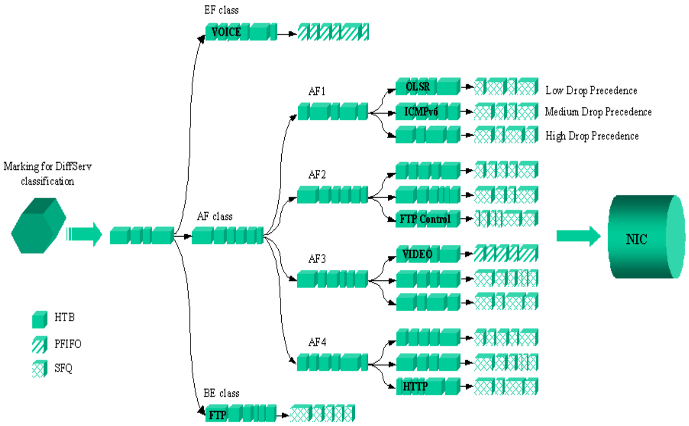


Figure 1 Solution 1B with HTB

Traffic entering the EF class, in our case it is audio, is rate limited with a policing filter in order to prevent the borrowing of all available rate. Contrary to the EF class, the AF class, which has the video traffic in AF31, is not rate limited.

At the end of the audio and video HTB queues a PFIFO (Packet FIFO) is attached. The length of this PFIFO is hard coded to 10 packets. This value was chosen after testing the applications that were going to be used for testing. This value was found to be one where the loss was not suffering, and that was small enough not to increase the latency. However, for bursty audio and video applications, this might not be a suitable value.

After the first HTB stage where EF, AF and BE traffic is divided into three HTB classes, the main AF class divides itself into four HTB classes (AF1, AF2, AF3 and AF4). Each of the AFx classes divide themselves again into three HTB classes of different priorities (the three drop precedence defined by the IETF). Each non audio and video AFxy leaf class has an SFQ queue. The SFQ queue prevents a single flow to dominate others in one leaf class.

On the testbed the EF (audio) class is configured to a rate of 32 Kbps[v]. The leaf AFxy classes are assigned rates of 32 Kbps[v]. AF31 (video) is assigned a rate of 128 Kbps. BE's base rate is 28 Kbps[v]. A SFQ queue is attached to the leaf class of BE to provide fair scheduling among different flows.

The following table explains how the priorities are organized among the leaf HTB queues.

Class	Priority
EF	1
AF11, AF21, AF31, AF41	2
AF12, AF22, AF32, AF42	3
AF13, AF23, AF33, AF43	4
BE	5

Table 2 Priorities in the HTB QoS mechanism

III. Testing

In this section, description of the test-bed and the types of tests conducted will be described.

There are two test scenarios that were considered: server being one hop away and server being two hops away from clients, through MPR (multipoint relay in OLSR) node, as seen in Figure 2. Both of these test scenarios are tested with and without QoS enabled for solution 1B.

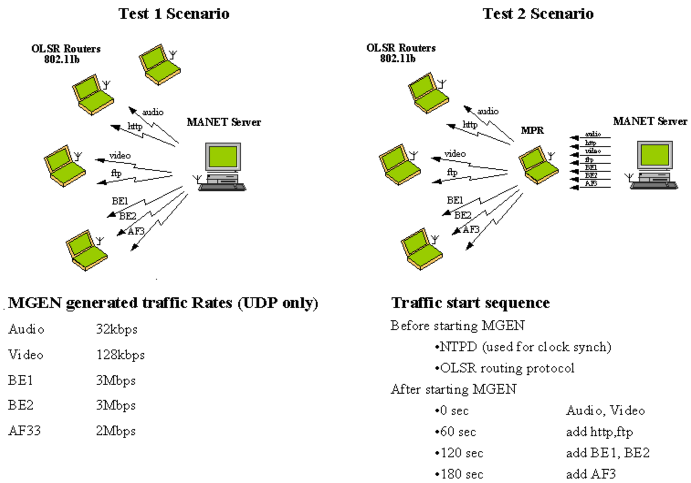
The test-bed consisted of four MANET nodes and a server, as shown in Figure 2. All nodes were equipped with 802.11b Linksys NIC cards operating 11 Mbps and running the IPv6 version of OLSR (developed at CRC) routing protocol. NTP (Network Time Protocol) was also running to synchronize the clocks in the network. Since NTP uses IPv4, it was treated as best effort traffic in the queuing system.

In both tests scenarios, all the flows were identical. Only the physical configuration of the MANET was changed. Below is the list of applications that were used to generate traffic flows:

[iv] Priority Queuing

[v] where K = 1024

- mgen[vi] 4.0x4
- Apache web server 2.0.43
- wget[vii] (an http and ftp client). The version is the one from the production tree (CVS) of January 29, 2003.
- Libra ftp server (1.3-3)



MGEN generated traffic Rates (UDP only)

Audio	32kbps
Video	128kbps
BE1	3Mbps
BE2	3Mbps
AF33	2Mbps

Traffic start sequence

Before starting MGEN
- •NTPD (used for clock synch)
- •OLSR routing protocol

After starting MGEN

•0 sec	Audio, Video
•60 sec	add http,ftp
•120 sec	add BE1, BE2
•180 sec	add AF3

Figure 2 Test Bed and Test Scenarios

The audio, video, BE1, BE2 and AF33 flows were all generated with the mgen software. Tests were run for a duration of 240 sec. As described in Figure 2, the flows were introduced every 60 seconds to incrementally increase the congestion on the link.

Table 3 shows the detailed configuration of mgen generated UDP flows:

Flow	mgen configuration		Flow Characteristic	
	Port Number	Pattern	Packet Size[viii] (Bytes)	Rate[ix] (Kbps)
Audio	5000	Poisson	168	32
Video	5002	Poisson	948	128
BE1	40404	Poisson	1298	3000
BE2	40405	Poisson	1298	3000
AF33	33330	Poisson	1298	2000

Table 3 Flow Characteristics

The audio and video packet sizes are typical values for audio and video applications. The rate values for BE1, BE2 and AF33 were chosen so that enough congestion was created on the link to observe noticeable loss on the audio and video streams without QoS. The audio and video rates exactly match the maximum rates of the EF classes in the configuration of the queuing mechanism.

IV. Analysis

The analysis consists of test results of QoS vs no QoS, Test 1 and Test 2.

A. Comparing QoS vs No-QoS Results

The results show that QoS brings significant improvement to audio, video and any high priority class.

As the network gets congested the rates of the flows get degraded, their latencies increase and there is significant percentage of packet loss. This may be acceptable for TCP or for some non-real time applications, but certainly not acceptable for the audio, video and some other high priority traffic such as control messages for routing protocols in highly mobile networks. In order to bring some quality of service, the flows need to be classified and have to be given priority.

Test results[x] show that, in Test 2 the latency of audio with no QoS is 10.74 sec and with QoS it is 2.9 sec (for video, the results are similar). Since the latency is very important for real time traffic, this is a significant improvement over no QoS. The latency in Test 1 was also significantly improved from 0.28sec to 0.007sec.

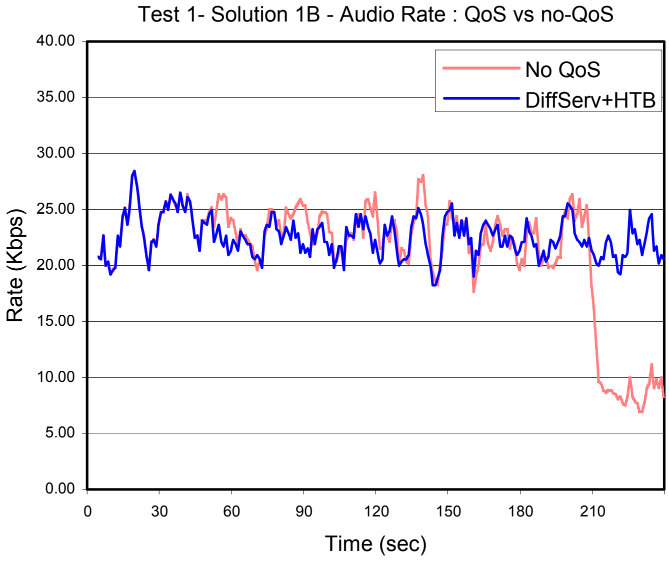


Figure 3 Audio Rate: QoS vs no-QoS

[vi] A version of mgen modified to make the sockets work in "non blocking mode".

[vii] A version of wget modified to dump rate statistics to a file.

[viii] Includes UDP and IPv6 overhead

[ix] Includes UDP and IPv6 overhead, and K=1024

[x] Examples of test results presented here are for the solution 1B and in a congested network.

The rate results also show the advantage of using QoS. With no QoS, in Test 2, the rate of audio falls from 32kbps to 3.4kbps and there is 85% packet loss when the network is congested. With QoS, the rate falls only to 26kbps and the packet loss is only 16% as seen in Figure 3. In Figure 4, the results show similar improvements for video also.

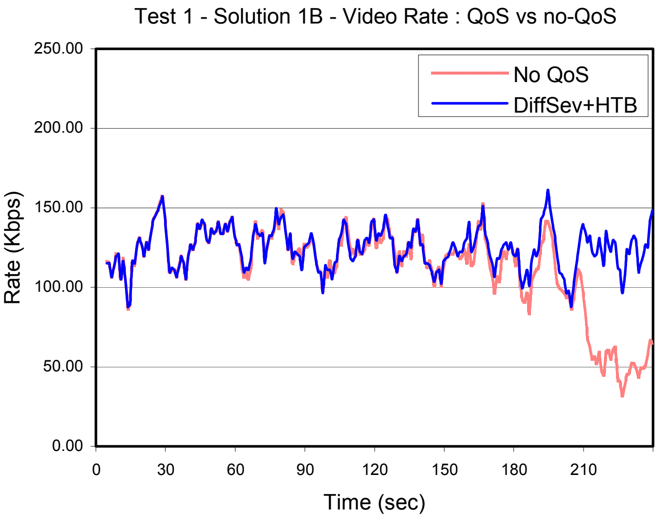


Figure 4 Video Rate: QoS vs no-QoS

Overall test results show that there is a need for a QoS mechanism in the MANET. Figure 5 and Figure 6 show the overall rate performance of all the flows over time. They clearly show that the audio and video rates stayed constant as the network load increased.

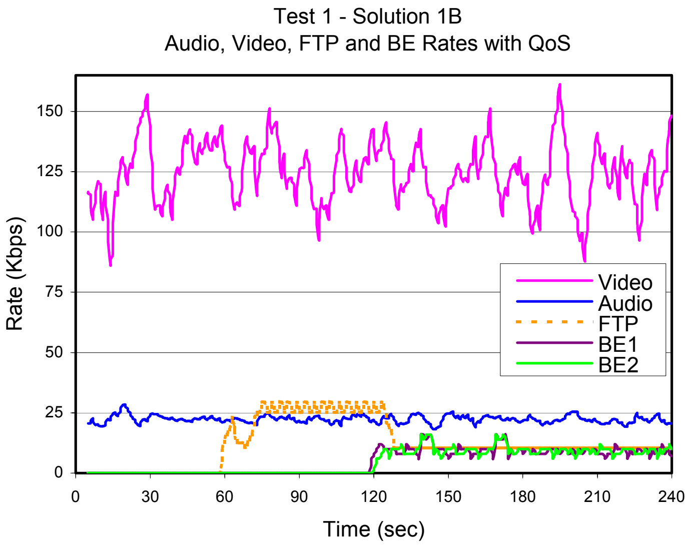


Figure 5 Audio, Video, FTP, BE Rates with QoS

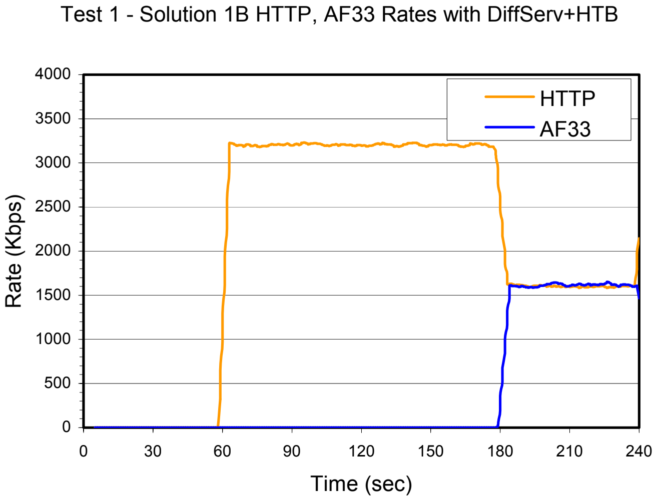


Figure 6 HTTP and AF33 Rates with QoS

In conclusion, the QoS mechanism used in our test bed improves the quality of the real time traffic in a congested network by reducing the latency and the packet loss.

B. Comparing Test 1 and Test 2

The audio and video latency results are more than doubled in Test 2 comparing to Test 1. The additional hop increases the total packet delay. The packets arriving to the ingress side of the MPR can experience larger delays, since our queuing mechanism only operates at the egress side. Therefore the MPR introduces additional sources of delay. Another source of delay could come from 802.11 wireless card, because of buffer size and retransmissions.

By comparing Test 1 and Test 2, the first obvious conclusion that can be derived is that the total available rate is much smaller in Test 2. This is because the MPR, which forwards all the packets, is busy half of the time receiving the packets from the server, and half of the time sending them to their destination. For example, in the last time interval of the test, the sum of the average rates of all the flows is 3970 Kbps for Test 1, and 267 Kbps for Test 2.

This reduction in the total available rate has a significant effect on the borrowing process of all non audio-video AF flows. As a result, the average rate of http, ftp and AF33 are much smaller in Test 2. Since audio and video flows do not attempt to borrow from other classes, they are not much affected, as expected. The effect of MPR is observed only when there is congestion. Due to congestion, audio and video rates are also reduced, regardless of their priority and class. This could be due to the HTB algorithm or loss that occurs at lower layers.

The UDP Best Effort streams (BE1 and BE2), though, get average data rates higher in Test 2 than in Test 1. The explanation for this behaviour is that the tcp protocol (for http and ftp flows) detects the congestion on the link. As a result, tcp back off mechanism reduces the rate more than HTB would. This leaves more bandwidth for BE1 and BE2 flows to borrow.

V. Conclusion

This paper describes the implementation of a DiffServ QoS mechanism in a MANET.

Overall test results show that there is a need for a QoS mechanism in the MANET. By using a QoS mechanism in the MANET, latency, jitter and packet loss are improved for packets with higher priority. MANET overall performance is improved since routing protocol packets suffer from less loss.

The results analysis showed that the HTB queuing mechanism improved the performance of audio and video traffic (both in terms of rate and latency), compared to tests ran without any QoS system in place. Also, the total available rate is shared more fairly among the multiple flows and classes of traffic, as seen in Figure 5 and Figure 6.

The audio and video flows do get better performance when the QoS mechanism is used, but the latency is still in the order of seconds in the worse congestion cases. The QoS mechanism does send packets to lower layers according to the QoS policies. However, when the packets are in the 802.11 MAC layer, additional delay and loss can occur outside the control of the upper layer QoS mechanism.

HTB was not specifically designed for MANET. MANET represent different challenges in terms of QoS compared to fixed network. However in this study HTB was configured so that it is suitable for MANET characteristics. At the time of our study, HTB was the best available queuing mechanism in Linux for our needs.

In conclusion, the queuing mechanism that is being designed improves the performance of the MANET, and does implement the PHB as required. High priority traffic suffers less under heavy congestion scenario if the queuing mechanism is used.

Area of future studies could include adding QoS control at lower layers to eliminate other sources of performance degradation in congested network. In terms of further analysis, the effect of mobility should be investigated. Finally, queuing mechanisms other than HTB, possibly more adapted to the MANET characteristics, could be implemented and compared with HTB to see if improvements are possible.

VI. References

[1] INSC QoS Architecture, INSC/Task5/D/006

[2] IETF, Differentiated Services (DiffServ) working group.