QoS Support for Real-Time Flows in Internet Routers

O. Mirabella S. Fichera, S. Visalli,
Università di Catania, Facoltà di Ingegneria
Dipartimento di Ingegneria Informatica e delle Telecomunicazioni
Viale A. Doria 6, 95125 Catania, Italy
omirabel@diit.unict.it

Abstract

This paper addresses how to offer a specific QoS for soft-real time distributed applications over the Internet. The approach here proposed is entrusting the routers with the task of providing diversified scheduling for real-time and non-real-time flows, keeping the two types of traffic into separate queues and applying a deadline-based scheduling to real-time flows and a FIFO policy to non-real-time traffic. The scheduling techniques proposed for real-time flows are an adaptation of the Earliest Deadline First algorithm. The performances of the approach have been measured on a real test bed and are discussed in the paper.

1. Introduction

Internet Quality of Service (QoS) from the real-time systems viewpoint is currently a research area of great interest for many companies and providers, as it is relevant to several distributed real-time applications, such as collaborative virtual environments, remote system monitoring and control, e-commerce services, etc.

As it is known, real-time traffic flows are sensitive to the delay affecting single packets. However, when the workload is high and the network is close to congestion, an unpredictable and unbounded increase in the packets delay may cause the deadline to be missed (deadline being the length of time for which real-time packets are valid). This means that once a packet reaches its destination, it will be discarded by the application, as it is no more useful. This is a situation to be avoided, because the packet will still have used up resources along the way, in terms of both transmission bandwidth and router buffer storage capacity. Discarding due to a missed deadline is inevitable because a traditional FIFO router is unable to determine whether a real-time packet has expired, and so continues to keep it in its buffer and transmit it to the next routers. Suitable flow regulation mechanisms to avoid, or at least limit, these problems, are needed.

Unfortunately, the service currently offered over the Internet by the standard Internet Protocol (IP) [1] does not provide for diversified management for real-time flows. Traditional routers handle both real-time flows and non real-time flows in the same way, and the currently-used Internet Protocol version 4 *(IPv4)* is unable to efficiently support QoS, a task entrusted to the higher-level protocols. For this reason, the IETF (Internet Engineering Task Force) has been working on standardisation of the new version (v6) of the IP, called *IPv6* [2]. IPv6 offers *native* QoS support, in the sense that it provides fields in the header of IP packets that will make it possible to manage traffic flows in a differentiated fashion, according to their QoS requirements.

This paper proposes a novel approach to provide a diversified service to time-sensitive flows. The basic idea is to entrust routers with flow regulation regardless of the control performed by the higher-level transport and application protocols. Some higher-level protocols (e.g. the TCP [5]) provide for end-to-end flow control, but the mechanisms they implement introduce overheads which are often inappropriate for real-time communication, so many time-constrained applications such as streaming audio-video use the UDP [6], which has no flow control mechanisms.

Intervention at the network level as well, in routers for example, which would make the situation visible at each hop a packet makes on the network, allows for more efficient, faster flow management. Network-level management can discard expired real-time packets before they reach their destination, thus avoiding a waste of resources to deliver information that is no longer useful.

The approach proposed in this paper is therefore entrusting the routers with the task of providing diversified scheduling for real-time and non-real-time flows, keeping the two types of traffic in separate queues and applying a deadline-based scheduling to real-time flows and a FIFO policy to non-real-time traffic. The scheduling techniques proposed for real-time flows are an adaptation of the Earliest Deadline First [7] algorithm .

2. A Deadline-based scheduling algorithm

The scheduling technique proposed is an adaptation of the Earliest Deadline First (EDF) algorithm for use in a packet-switched multi-hop network. Each packet has an “end-to-end deadline”, which is a time interval equal to the packet's period of validity (or lifetime), which has to be distributed over the n hops the packet has to make from the source host to the destination one. Defining the deadline for each single hop as the "hop deadline", the following relation holds:

Σ_i *hop_deadline*$_i$ = *end-to-end deadline* i = 1,2,…n (1).

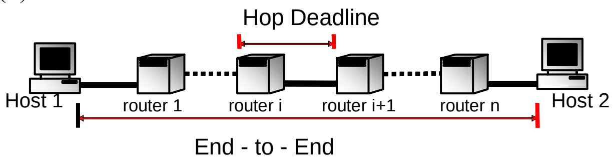

Fig. 1 End-to-end deadline vs. hop_deadline.

Fig. 1 illustrates the concepts of end-to-end deadline and hop deadline. The i-th router has to ensure that the packet reaches the (i+1)-th router before the hop_deadline$_i$ expires, i.e.

arrival_time$_{(i+1)}$< *arrival_time*$_i$ + *hop_deadline*$_i$ *(2)*

If condition (2) holds for each i: 1≤ i ≤ n, where n is the number of hops on the packet's routing path, the packet is guaranteed to be delivered before the end-to-end deadline. If, on the other hand, cond. (2) is not met for every i, there is no guarantee that the end-to-end deadline will be met (even though there is no certainty of a deadline miss because, even if a single hop deadline is missed, the sum of all the hop_deadlines can still be below the end_to_end_ deadline).

In order to meet cond. (2), when a packet arrives, the i-th router reads the hop_deadline$_i$ and calculates the "next_hop_router_expected_arrival_time", i.e. the time at which the packet is expected to reach the next router, by adding to the hop_deadline$_i$ the instant at which the packet arrived at the i-th router.

This time is used by the EDF algorithm to schedule the next packet to be transmitted. In fact, when the output link is free, the packet with the lowest "next_hop_router_expected_arrival_time" will be chosen for transmission. In this way, the most "urgent" packets are transmitted before those which can "survive" in the i-th router for a longer time.

The hop-deadline does not provide a packet with an absolute priority but, in each router, all the hop-deadlines are compared and the packet with the lowest next_hop_router_expected_time will be transmitted first.

3. Adaptive EDF algorithm

The choice of the hop deadlines is a critical point. The simplest way to calculate it is to divide the end-to-end deadline by the number of hops the packet has to make. In this way the maximum amount of time a packet can take to hop from one router to another is always the same. This approach, which we call simply EDF, is easy to implement, but it fixes the hop deadlines statically without taking any account of the workload conditions on the router. If a router is heavily loaded, the hop deadline can expire and the packet will be discarded.

The idea proposed in this paper is to associate an adaptive approach with the EDF algorithm. The mechanism exploits feedback from the network for two-level regulation. At the first level the Hop Deadline (HD) of flows is adapted, detaching a certain percentage from the End-to-End Deadline (EED) and reserving it for slow routers. The second level of regulation consists of reducing the transmission rate for flows, thus reducing the time spent in router queues and consequently the transmission delay.

The first level of regulation consists of taking from the EED a portion which can be seen as an extra budget at the disposal of the routers on the path. This extra budget, here called *reserve_quota* (RQ) can be exploited, when needed, by congested routers. After subtracting the *reserve_quota*, the remaining part of the packet's lifetime can be divided by the number of routers along the path, thus obtaining the HDs. In order to make this mechanism adaptive to the actual load on the network, the *reserve_quota* can be "modulated" by the source router by means of a suitable control mechanism. The approach we propose, called the "Adaptive EDF Algorithm", works as follows:

When a packet arrives at the source router, the latter fixes the amount of the reserve_quota, expressed as a percentage of the EED. Initially, the reserve_quota is 0% of the EED. Once reserve_quota has been set, the source router calculates the hop_deadline as follows:

HD= (EED – RQ)/n_hops (3)

(the hop deadline is the same for all the routers on a path). If a router is unable to transmit a packet within the hop_deadline, it can use part (or the whole) of the reserve_quota. When a router consumes part of the reserve quota, the part consumed is subtracted, leaving a lower RQ for the subsequent routers.

If the i-th router cannot transmit a packet even if it uses up the whole of reserve_quota, i.e. the hop_deadline is insufficient, it discards the packet and sends back to the source router a control packet called a TEP (Time Expired Packet).

When the source router receives a TEP it becomes aware that a packet has missed its deadline, so it increases RQ for that flow by a percentage ΔX of EED. At each step of the algorithm, if a packet does not arrive on time, reserve_quota is increased by the ΔX percentage. This increase proceeds until an upper bound - 100% of EED has been assigned for reserve_quota.

If a router receives too many flows and the available bandwidth is not sufficient, the delay will grow, some packets will be discarded, and TEPs will be sent to the source router regulating the deadline. But if the router is forced to discard packets even after the maximum reserve quota is reached, the second form of regulation has to be activated, reducing the percentage of packets allowed to enter the network.

Noting that Hop Deadline adaptation is unable to guarantee packet delivery in the current workload conditions, the source router limits the flow itself by reducing the percentage of packets allowed to enter the

network, after which it re-applies the algorithm from the beginning, i.e. from a 0% reserveTime. This approach, which may seem rather drastic, is suitable for soft real-time flows, which can tolerate a certain amount of packet loss.
Finally, it should be pointed out that, to apply the EDF and the AEDF algorithms, the source router has to know the length of the routing path the flow has to cover, i.e. the number of hops the packets have to make to reach their destination host.

4. Some implementation notes

To evaluate the algorithms described above, we implemented a test-bed made up of an IPv6 network with three routers and six nodes, as shown in Fig.2. In choosing the routers we avoided commercially available products that only allow partial internal modifications, which are not sufficient for our purposes. We used open source software, which allows a router to be implemented on a PC. The software acts on the kernel of the operating system, directly handling the network interfaces and thus does not excessively penalise performance. Of course, a solution of this kind does not have the speed of a dedicated component like a commercial router, but it allows new approaches to be experimented with.

The operating system used is Linux and the software development environment is Click [8] which was chosen, after careful evaluation, on account of its structure and versatility. Its programmability depends directly on the flexibility of the components of its architecture, which supports easy to use libraries, and a large number of default elements that can be used.
Three scheduling algorithms were implemented in the three routers: FIFO, currently used in commercial routers, and the EDF and AEDF algorithms, so as to be able to make comparative measures.

5. Performance evaluation

Fig.2 shows the network used for the measurements.

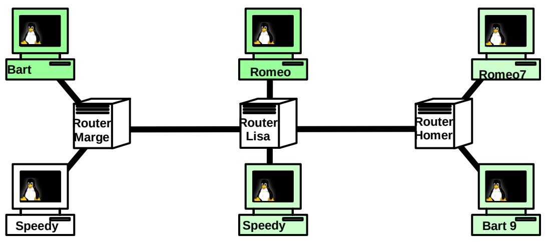

Fig.2: The test-bed.

Transmission occurred between couples of nodes and precisely between the host Bart and Bart9 and from Romeo to Romeo7. For performance evaluation to be meaningful, it was necessary to set the traffic flows accurately in such a way as to create router overload conditions that would show the validity of the approaches being considered.

The measures were made on only two pairs of hosts, the third pair being used to generate TCP "disturbance" traffic. In this scenario a TCP traffic flow was inserted between Speedy and Romeo7. Bart generated a continuous flow of 10 packets/sec. whereas Romeo generated two bursts, each one of 20packets/sec, as shown in Fig.3 . A traffic shaper was inserted in the central router for limiting the bandwidth available to 28 packets/sec. In this way congestion can arise as soon as traffic overcomes this limit.

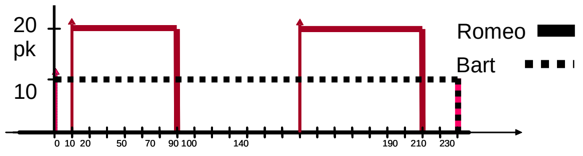

Fig.3: Traffic generated by Bart and Romeo.

In this scenario, the TCP traffic inserts considerable delays. As can be seen in Fig.4 with the FIFO router the transmission delay is almost 8s. In the other two approaches, EDF and AEDF, on the other hand, the packets delivered always meet their deadline but some are discarded.

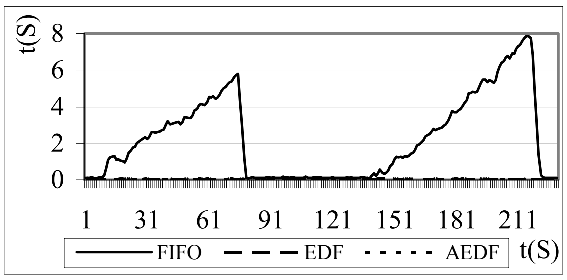

Fig.4: Transmission delay of Bart.

To show better the advantages of the EDF and AEDF algorithms over FIFO we enlarged the graph, considering the packet deadline of 0.2 sec. as the full scale value. In this way we can appreciate better the delays of the EDF and AEDF.

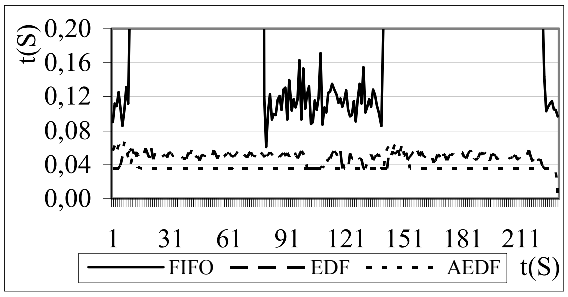

Fig.5: Transmission delay of Bart (in detail).

It is important to point out that the presence of TCP traffic increases delay even in the presence of a single flow, thus increasing the number of packets discarded, especially with the AEDF approach, in which packets are discarded a priori. This can be seen in both Fig.6 and Fig.7.

Fig. 6 shows the average values of delays during the whole test. As we can see the AEDF provides the best results, because it is able to provide a better service to the traffic with more strict time requirements. This behaviour has a cost , i.e the number of packets lost from AEDF is higher than EDF. This must be ascribed to the anticipated packet discard performed by AEDF. In fact, when the network is heavy loaded (and in our simulation the workload reaches very high values during the traffic bursts of Romeo) the first router limits the input traffic following the reception of a TEP, by discarding some packets.

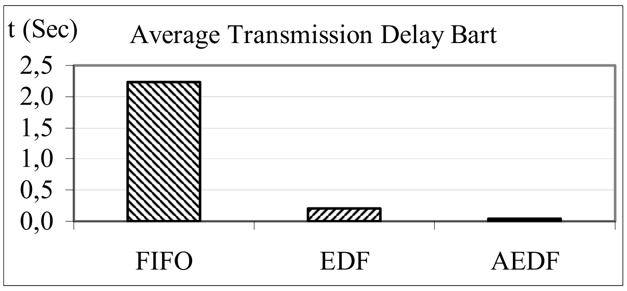

Fig.6: Average delay of Bart.

The trend observed in the Romeo graphs is similar to that of Bart and is shown in Fig.8: deadlines are not met when a FIFO algorithm is used whereas EDF and AEDF behave in a similar fashion, although after an initial transient period AEDF is more stable than EDF. The cost of this improvement is again an increase in the number of packets discarded by the congested router in the case of EDF and the source router in the case of AEDF.

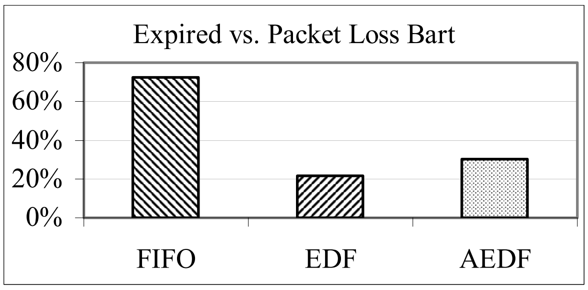

Fig.7: A comparison between expired and discarded packets.

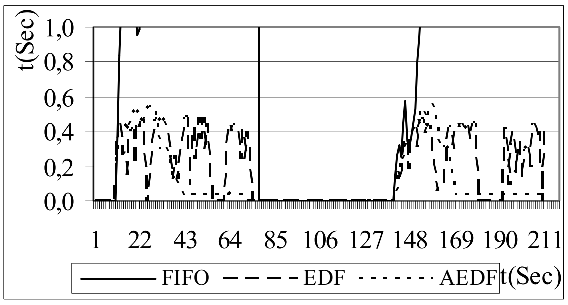

Fig.8: Transmission delay of Romeo (in detail).

Conclusions

The paper has presented and evaluated a new approach to scheduling real-time traffic in routers: this approach, which is deadline-aware, considerably increases the percentage of packets that reach their destination within the deadline as compared with the classical FIFO approach. This allows the design of new networks where the routers can distinguish between traffic with different requirements and provide best support to the most critical one. The results discussed in the paper are only a sample of the many tests which have been performed on an experimental IPv6 network. The limited number of routers in our benchmark network does not allowed to point into evidence the advantages introduced by AEDF with respect to EDF. In fact, the AEDF algorithm can adapt its behaviour to the traffic requirements and to the network conditions this way providing a different distribution of the hop deadline among the various routers according to their different workload.

Instead, EDF uses the same hop deadline into all routers and it is not able to cope efficiently with different internal congestion situations.

The results obtained show the validity of the approach and suggest further study of approaches aiming at moving the management of the network at IP level, thus relieving the user of the need to handle congestion problems by limiting transmission flows.

References

[1] J. Postel, “Internet Protocol”, RFC 791, Sept.1981.

[2] S. Deering, R. Hinden ,“Internet Protocol, Version 6 (IPv6) Specification”, RFC 2460, Dec. 1998.

[3] R. Gilligan, E. Nordmark, “Transition Mechanisms for IPv6 Hosts and Routers”, RFC 1933, Apr.1996.

[4] S. Floyd, V. Jacobson, “Random early detection gateways for congestion avoidance”, *IEEE/ACM Transactions on Networking*, 1(4):397–413, Aug.1993.

[5] J.Postel, “Transmission Control Protocol”, RFC 793, Sept.1981.

[6] J.Postel, “User Datagram Protocol”, RFC 768, Aug.1980.

[7] L. Liu, J. Layland, “Scheduling algorithms for Multiprogramming in a Hard Real-Time Environment,” Journal of ACM, Vol. 20, No.1, 1973, pp.46-61.

[8] Click: http://www.pdos.lcs.mit.edu/click/