A scalable solution for QoS provisioning

Gianmarco Panza *Senior Member IEEE*, and Sara Grilli

Abstract —**Next-Generation Networks will support Quality of Service (QoS) over an IP-based infrastructure. A scalable solution to assure stringent delay requirements for real-time applications is needed.**

In this work, a Proportional Differentiation Model (PDM) in DiffServ architecture is employed to provide even absolute QoS guarantees by a dynamic assignment of service class.

Keywords — **DiffServ, feedbacks, measurement process, Next-Generation Networks, QoS, SLA.**

I. INTRODUCTION

The Next Generation Network (NGN) [1] , is an IP network with Quality of Service (QoS) support. Real-time applications should receive a treatment good enough to fulfill their requirements as for user needs. In this scenario, given quality parameters (e.g. end-to-end delay and loss) are to be guaranteed for all the packets or a high percentage of them.

Different levels of service can be supported in a scalable manner using the DiffServ Architecture [2]. Many algorithms have been proposed ([3]) for differentiating service classes in DiffServ architecture, but the more promising, also for an efficient resource exploitation, is the Proportional Differentiation Model (PDM) [3], in which, the quality spacing between classes of traffic is proportional to given class differentiation parameters that network operator can specify.

After presenting the proportional differentiated service model advantages in terms of controllability, consistency and scalability, Ref. [3] explains how it is possible to use packet forwarding mechanism to implement such a model by a joint scheduling-dropping method to enforce proportional average delay and packet loss. Specifically, Waiting Time Priority (WTP) scheduler can approximate the proportional delay differentiation model in short timescale, because in this scheduler the priority of a packet increases proportionally to its waiting time. Other works based on a PDM are available in literature and proposing other scheduling algorithms, like dynamic Weighted Fair Queuing (WFQ) [4].

However it is often necessary to provide absolute service guarantees to the real-time traffic. Various approaches have been proposed in literature for achieving absolute QoS bounds still relying on a proportional model. They can be divided into two groups: end-to-end and single-hop approaches. In the former, the absolute quality requirement can be addressed with a joint packet scheduling and dropping algorithms, in which WTP is coupled with a dropping mechanism [5]. In the latter, various strategies proposing a run-time change of service class for the real-time traffic are available. In the first solution, the selection of the initial service class and its change during the connection life-time are based on the application requirements and triggered at the user side [6], as needed. The second solution integrates with the admission control process, using a probe packet for verifying the packet loss rate and the presence of congestion [5]. The third one instead, chooses to adaptively select the service class depending on the current performance. Either the end-to-end delay [7] or the loss [8] of the real-time traffic is monitored and if the related Service Level Agreement (SLA) is not met the traffic is moved to higher priority class, otherwise it can be moved to lower one. More specifically, in Ref. [7], the access router sends a probe packet for each flow in order to monitor the end- to-end delay. If the requirement on the delay is not met, the traffic is promoted. While, if the end to end delay of the probe is below the bound and a lower class can sustain the traffic, the traffic is downgraded there. However, in this last and more promising proposal the adaptation is based on the average delay of each class as the reference QoS parameter, which is not often the more relevant in practice. A given (high) percentile of the end-to-end delay is of greater interest for real-time applications. Moreover, it makes the assumption that the performance experienced by probe packets is representative of the critical traffic performance, which is not always true. Moreover, burst of traffic and class change could entail inconsistency in supporting a PDM. Finally, functionality is added in core routers possibly compromising the scalability of the architecture.

The solution proposed in this work to achieve absolute QoS guarantees with a PDM is still based on a dynamic change of the service class for the real-time traffic with the aim of addressing the related SLA. However we don't inject probe packets within the network, avoiding bandwidth overhead, and additional functionality with respect to the base DiffServ architecture [2] is introduced only in Border Routers (BRs), not compromising the scalability of the solution. Specifically, three different measurement processes for the promotion and downgrade of traffic could be included in BRs. All the options were analyzed and compared considering the two critical issues

First author Gianmarco Panza is with CEFRIEL/Politecnico di Milano, IT. Via Fucini 2, 20133 Milan, Italy (phone: +39-0223954326; e-mail: gianmarco.panza@cefriel.it)

Corresponding Sara Grilli is with CEFRIEL/Politecnico di Milano, IT. Via Fucini 2, 20133 Milan, Italy (phone: +39-02239542046; e-mail: sara.grilli@cefriel.it).

of performance stability and adaptation speed to network load dynamics.

Finally, we consider as reference QoS parameter the 99th percentile of the end to end delay for the real-time traffic, giving a better picture, more reliable and consistent with the application requirements, of the granted service. For the purpose, time synchronization between BRs is needed. However, due to the transfer of legacy voice traffic of fixed and mobile users in NGNs, sharing accurate timing information by network nodes in a reliable and cost effective manner is mandatory, and several solutions have been already proposed in literature. Noticeably, IEEE 1588 [9] realizes a scalable synchronization scheme with sub-microseconds accuracy by the exchange of messages between network devices in the same manner as Network Timing Protocol (NTP) [10]. Furthermore, one of the recent major standardization trends in carrier-scale Ethernet is to deliver accurate timing and synchronization information for real-time audio/video streaming and circuit emulation for legacy services[11][12].

The paper is organized as follows. First, we present the proposed solution from an architectural standpoint and the employed measurement processes in Sect. II. Then, in Sect. III we describe the simulation analysis and discuss the collected results. Finally, the main conclusions are drawn out in Sect. IV.

II. ABSOLUTE QoS PROVISIONING

The objective of this work is to provide absolute QoS for real-time flows in an IP network with a PDM in DiffServ architecture. To achieve this, critical traffic can be moved run-time to the lower service class yet addressing its delay requirement as by SLA.

The chosen reference quality parameter for the real-time traffic is the 99th percentile of the end-to-end delay (though, the solution can be easily generalize to a whatever percentile), since the real-time traffic has a very stringent requirement on the end to end delay experienced by a high percentage of its packets (e.g. about 200 ms for voice). Moreover, having peaks on the end-to-end delay badly affects the jitter and therefore, the packet loss (if a packet does not arrive at the receiver soon enough to be played out it is actually discarded).

The end to end delay can be monitored either at the egress of the network (i.e. at the egress BR) or directly at the receiver terminal (as available). The figures of the 99th percentile of the end-to-end delay are sent as feedback to the ingress BR of the network and elaborated by using one out of three measurement processes, either a threshold-based method, moving average or low pass filter. If the calculation results in an exceed of the bound on the 99th percentile of the end-to-end delay for the issued traffic, the ingress BR remarks the DS-field of the packets of the given flow(s) promoting it (them) to the closest upper class of service. If in this new class the requirement on the delay is not respected yet, the flow(s) can be promoted to the subsequent better class, and so on up to the best class of service (or to the highest allowed, as by SLA). While if the end-to-end delay bound is granted by the closest lower class, the flow(s) can be moved back there (up to the initially assigned class, as by SLA).

The two critical issues that must be considered when deciding which method of promotion or downgrade is the most suitable are the system stability and reaction speed to network load changes.

From one hand, the system must be fast in order to promote the critical traffic as soon as the related requirement on the 99th percentile of the end to end delay is not addressed anymore in the currently assigned class or downgrade the flow if such requirement is met also in the closest lower class. From the other hand, the system must be stable enough to avoid frequent changes of the assigned class because this could lead to highly variable performance (e.g. on jitter), also to other flows, not to consider the complexity induced in the ingress BRs that should re-configure too often their marking policy. It is clear that the best solution should be sought in a good tradeoff between the two issues, in order to have a robust and reliable system in providing absolute QOS guarantees in a scalable manner for real-time traffic.

A. Measurement processes

The evaluation of the 99th percentile of the end to end delay for the real-time traffic is to be performed run-time. Our proposal is to store the delay experienced by N_{udp} consecutive packets of the concerned traffic and then to pick up the sample D_{99}^{udp} that better corresponds to the 99th percentile (typically, the real-time traffic is over UDP). For example, if N_{udp} is 100, we pick up the second greatest one. The delay of a packet can be easily calculated if the network nodes are synchronized (e.g. by putting the arrival time of the packet at the ingress of the network into a header option of the packet itself.

When the feedback of the delay reaches the issued ingress BR, it is elaborated in order to decide for promotion. Three measurement processes have been considered.

- **Moving average**: the ingress BR collects and stores K_{up} consecutive values of D_{99}^{udp}; when a new value of the 99th percentile is received, the oldest one of the K_{up} consecutive values is discarded. On this data set the ingress BR calculates the moving average *MA*, that is,

$$MA = \sum_{i=0}^{i=K_{up}} D_{99}^{udp} \Big/ K_{up} \qquad (1)$$

  If *MA* is below the threshold D specified in the issued SLA nothing happens. Otherwise, the SLA is not satisfied and the given traffic needs to be promoted to the closest upper class.

- **Threshold**: the ingress BR checks if the 99th percentile of the end to end delay of the traffic is greater than the threshold D for K_{up} consecutive times. In the case, the traffic needs to be promoted.

- **Low pass filter**: the ingress BR uses the last calculated output for the averaged value of the 99th percentile of the end to end delay of the traffic and the current (last received) one for calculating the new output of the low pass filter *LPF*, that is:

$$LPF = LPF_{old}^{udp} * P_{up} + D_{99}^{udp} * (1 - P_{up}) \qquad (2)$$

  If the result is greater than the threshold D, the traffic needs to be promoted.

The decision for downgrade the issued real-time traffic is based on the monitoring of the end to end delay of the

whole traffic D_{99} (in fact, a portion of it big enough to provide reliable evaluations) for the lower class of service closest to the currently assigned. The ingress BR collects the feedbacks about the delays sent by the egress br (actually, from all the egress BRs, but only one is related to the given traffic) and elaborates them according to one of the following measurement processes.

- **Moving average**: the ingress BR collects and stores K_{down} consecutive values of D_{99} for the closest lower class; when a new value of the 99th percentile is received, the oldest one of the K_{down} consecutive values is discarded. Using this data set the ingress BR calculates the moving average *MA*, that is,

$$MA = \sum_{i=0}^{i=K_{down}} D_{99} \Big/ K_{down} \qquad (3)$$

  If *MA* is below the threshold D specified in the issued SLA, the closest lower class can address the traffic delay requirement and the given traffic can be moved into that class. Otherwise, the traffic remains in the current class.

- **Threshold**: the ingress BR checks if the 99th percentile of the end to end delay of the closest lower class is below the threshold D for K_{down} consecutive times. In the case, the traffic can be downgraded to that class.

- **Low pass filter**: the ingress BR uses the last output of the averaged value of the 99th percentile of the end to end delay and the current (last received) one of the closest lower class for calculating the new output of the low pass filter *LPF*, that is:

$$LPF = LPF_{old} * P_{down} + D_{99} * (1 - P_{down}) \qquad (4)$$

  If the result is below the threshold D, the traffic can be downgraded.

The monitoring of the end to end delay for each class happens independently from the promotion of the issued traffic. This is reasonable because the movement of a small percentage of traffic (i.e. only the traffic with absolute QOS requirements) between classes in a core network with high speed links has no impact on the overall network behaviour. However, the target closest lower class will not decrease its performance after the downgrade in our PDM. The configuration parameters with suitable values are: $K_{up}=2$, $N_{udp}=10$ and $P_{up}=0.9$ for promotion, and $K_{down}=20$, $N_{down}=100$ and $P_{down}=0.9$ for downgrade.

III. SIMULATION ANALYSIS AND RESULTS

The simulation analysis has been carried out in a network scenario with four aggregated sources of traffic and three routers, (see Fig. 1), which deploy the Advanced WTP scheduler [13] on Output links of 100 Mbit/s (the other links are set to a higher capacity in order not to have an impact on the collected results).

The first router can classify and mark the incoming IP packets, as ingress BR in DiffServ (ds) architecture. While all routers route the packets and schedule them according to the assigned class of service. Indeed, at each router the traffic, coming from source(s) or from the uplink router, is divided into four queues by looking at the DS-field, each one corresponding to a different class. The quality factors 1, 2, 3 and 4 are assigned to the four queues, respectively. According to a PDM, the delay experienced in the first queue should be about twice the delay experienced in the second queue and three times the delay in the third queue; while, the delay experienced in the second queue should be about twice the delay experienced in the fourth queue, and so on. Buffers are big enough to avoid losses.
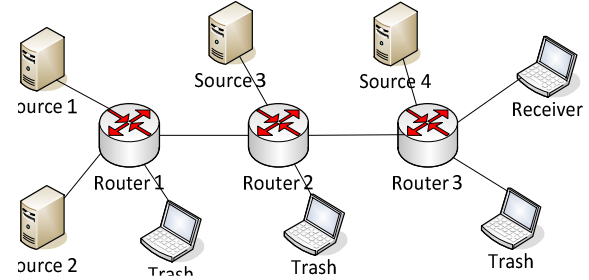

Fig. 1 Network scenario for simulations.

A router sends about 50% of the traffic to a node that acts as a trash, and the remaining downlink. At the receiver, the measurements are made on the real-time traffic generated by *Source 1* and *Source 2*, which is about 4 Mbit/s (the 15-20% of an aggregate).

For each class of service, an aggregate is composed of different types of real backbone traffic traces:

- 1 TCP flow aggregate, with an average traffic rate of 16 Mbit/s,
- 17 MPEG4 video flows, each one with an average traffic rate of 280 kbit/s,
- 3 ON/OFF G.729 audio flows, each one with an average traffic of 10 kbit/s

This leads to about 80 Mbit/s on average of overall traffic in each router output link, as ordinary network conditions, in a simulation period of 60 s. During two time intervals additional traffic is injected in every link to create a higher load, up to almost 100Mbit/s. This further traffic is artificially synthesized and is characterized by a constant packet inter-generation time of 0.001 s and a constant packet size of 4 kbits between 10 and 25 s. whereas, by a constant packet inter-generation time of 0.002 s and a constant packet size of 7 kbits between 40 and 55 s.

The threshold D for the 99th percentile of the end-to-end delay as specified in SLA is set to 10 ms, and in a first analysis, the transmission delay of the feedbacks from the receiver side to the ingress BR has been neglected.

We start analyzing the results for the promotion of real-time traffic with either the three measurement processes. We have to evaluate how fast the system reacts to a load increase (while the downgrade is done employing the threshold-based method, being the most conservative one).

The reported graph depicts the DS Code-Point (DSCP*)* assigned to the packets of the issued traffic. It changes from the initial value 1, when the issued traffic is in the worst class, till the value 4, when the issued traffic is in the best class.

The configuration parameters were as specified at the end of Sect. II, for a good compromise between stability and fast adaptation. It is worthwhile to point out that the number of samples of the 99th percentile of the end to end delay of the issued traffic that are considered for a promotion, is K_{up} equals to 2 only (this avoids negative memory effects on the decision process).

The system can be considered stable when the real-time traffic remains in the lowest class (the one initially

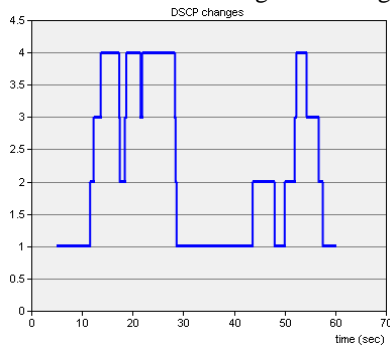assigned at minimum) addressing its delay requirement, when the traffic load does not change on average.

Fig. 2 Promotion by moving average with $K_{up}=2$,
downgrade by threshold with $K_{down}=20$.

The reaction times (time passed from the load increase and the first re-assignment to the subsequent upper class) at each traffic addition of the different promotion methods are: 2 s – 8 s for threshold-based, 1.5 s – 3.5 s for moving average and 2.2 s – 12 s for low pass filter.

We can notice that the moving average (see also Fig. 2) shows the best reaction time in particular at the second load increase, while the low pass filter appears the slowest method. Such a conclusion still holds when analyzing the results collected with different configuration parameters, which still allow for the system stability.

Let us fix the critical traffic promotion method to moving average with $K_{up}=2$ and analyze the different methods for the downgrade. Again, we have to evaluate both how fast the system reacts to a traffic reduction and its stability (i.e. change of class with load dynamics).

We notice how the threshold-based method is very stable (Fig. 2). When the network load increases at 10 s, the real-time traffic changes class, till it arrives in the best one. It remains in that class till the network load reduces slightly at 15 s and then it is promoted again and definitely, into the best class where it stays till the network congestion is finished at 25 s. Considering the second network load increase at 40 s, the issued traffic is first promoted, but then downgraded after some seconds because of a reduction in the network load, while it is promoted when the load increases again at 50 s. The downgrades start at 55 s till the issued traffic returns to the first class. Reducing K_{down} to 10, the reaction time is shortened, but the drawback is that there are more changes of classes.

With the moving average oscillations appear every time there is a promotion. This is due to the fact that the calculation of the moving average made with 20 samples of the 99th percentile of the end to end delay can include a lot of them still below the threshold D. As a consequence, while the delay of the issued traffic is above threshold in the current class and hence a promotion is triggered, the moving average evaluated for the previous class can result in a period of time below threshold (the network load is variable in nature) and the issued traffic is then downgraded. This situation could repeat for some time after the first promotion. Indeed, by reducing the value of K_{down} a slightly lower number of class change happens, but not enough to outperform the threshold-based method.

The low pass filter shows a similar behavior to the moving average, though with fewer oscillations. The filter considers only a sample of the 99th percentile of the end to end delay of the lower class closest to the currently assigned, weighted according to the value of the filter pole, which is a configuration parameter and can be set properly. However the real-time traffic still changes frequently class, possibly jeopardizing the stability of the system, in particular when there is congestion (e.g. in the first load increase).

We can conclude that the downgrade decision for the real-time traffic in order to have a fairly stable system but also fast adaptation to load reduction and dynamic can be better made using the threshold method with K_{down} set to a value between 10 and 20, according to the desired response time of the system.

IV. CONCLUSIONS

The designed system can provide both relative and absolute QoS guarantees in a scalable manner. Indeed, it relies on a PDM over DiffServ architecture with additional functionality at the border routers only. A good trade-off between fast adaptation and stability exist, even for stringent delay requirements. Last but not least, low bandwidth and computational overheads are introduced.

Regarding the future work. a deeper simulation analysis should be performed by considering a larger set of configuration and design options. For example, feedback refreshing period and transmission delay, different traffic types and dynamics and about addressing other QoS requirements (i.e. packet loss).

REFERENCES

[1] ITU Y.2001, "General overview of NGN", December 2004
[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, W. Weiss, "An architecture for Differentiated Services – RFC2475," IETF DiffServ WG, 1998..
[3] C. Dovrolis and P. Ramanathan "A Case for Relative Differentiated Services and the Proportional Differentiation Model" in *IEE Network*, Volume 13, Issue 5, September/October 1999.
[4] A.Martino, G. Panza, A. Pattavina and F. Valente, "QoS guarantees on DiffServ architectures with GPS-like scheduling disciplines", in *2007 Proc. IEEE/GLOBECOM Conf.*, p. 2672 – 2677.
[5] Y. Chen, C. Qiao, M. Hamdi and D. Tsang "Proportional Differentiation: a scalable QoS approach", *Communications Magazine*, IEEE June 2003, Volume 41, Issue 6, pp 52-58
[6] C. Dovrolis and P. Ramanathan, "Dynamic Class Selection: from Relative Differentiation to Absolute QoS", in *2001 Proc. IEEE/Network Protocols Conf.*, p. 120-128.
[7] T. Nandagopal et al., "Delay Differentiation and Adaptation in Core Stateless Networks," in *2000 Proc. INFOCOM, Conf.*, vol. 2, pp. 421–30.
[8] W. Wu et al., "Forwarding a Balance between Absolute and Relative: A New Differentiated Services Model", in *2001 Proc IEEE Wksp. High Perf. Switching and Routing*, pp. 250–54.
[9] IEEE 1588-2002, "IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems," 2002.
[10] D. L. Mills, "Internet Time Synchronization: The Network Time Protocol," IEEE Trans. Comm., vol. 39, no. 10, Oct. 1991, pp. 1482–93.
[11] G. M. Garner et al., "IEEE 802.1 AVB and its Application in Carrier-Grade Ethernet," IEEE Comm. Mag., vol. 45, no. 12, Dec. 2007, pp. 126–34.
[12] J. Ferrant et al., "Synchronous Ethernet: A Method to Transport Synchronization," IEEE Comm. Mag., vol. 46, no. 9, Sept. 2008, pp. 126–34.
[13] Yuan-Cheng Lai, Wei-Hsi Li, "A novel scheduler for proportional delay differentiation by considering packet transmission time," IEEE Comm. Letters, vol. 7, No. 4, pp. 189-181, April 2003.