Admission Control for Inter-domain Real-Time Traffic Originating from Differentiated Services Stub Domains

Stylianos Georgoulas[1], George Pavlou[1], Panos Trimintzios[2], and Kin-Hon Ho[1]

[1] Centre for Communication Systems Research, University of Surrey
Guildford, Surrey, GU2 7XH, United Kingdom
[2] ENISA, EU, PO Box 1309, 71001, Heraklion, Crete, Greece

Abstract. Differentiated Services (DiffServ) are seen as the technology to support Quality of Service (QoS) in IP networks in a scalable manner by allowing traffic aggregation within the engineered traffic classes. In DiffServ domains, admission control additionally needs to be employed in order to control the amount of traffic into the engineered traffic classes so as to prevent overloads that can lead to QoS violations. In this paper we present an admission control scheme for inter-domain real-time traffic originating from DiffServ stub domains; that is real-time traffic originating from end-users connected to a DiffServ stub domain towards destinations outside the geographical scope of that domain. By means of simulations we show that our scheme performs well and that it compares favorably against other schemes found in the literature.

Keywords: Admission Control, Real-time Traffic, Differentiated Services.

1 Introduction

DiffServ offers a scalable approach towards QoS in the Internet by grouping traffic with similar QoS requirements into one of the engineered traffic classes and forwarding it in an aggregate fashion. To provide QoS guarantees, DiffServ domains must additionally deploy admission control in order to control the amount of traffic injected into the traffic classes so as to prevent overloads that can lead to QoS violations.

The various admission control schemes can be classified into three categories: endpoint admission control (EAC), traffic descriptor-based admission control (TDAC), and measurement-based admission control (MBAC). EAC is based on metrics applied to probing packets sent along the transmission path before the flow is established [1]. The probing packets can be sent either at the same priority as flow packets (in-band probing) or at a lower priority (out-of-band probing). One problem of EAC schemes is that simultaneous probing by many sources can lead to a situation known as thrashing [1]. That is, even though the number of admitted flows is small, the cumulative level of probing packets prevents further admissions. TDAC is based on the assumption that traffic descriptors are provided for each flow prior to its establishment. This approach achieves high utilization when the traffic descriptors used by the scheme are appropriate. Nevertheless, in practice, it suffers from several problems [2]. One is the inability to come up with appropriate traffic descriptors before establishing the flow. MBAC tries to avoid this problem by shifting the task of traffic characterization to the network [2].

That means that the network attempts to "learn" the characteristics of existing flows through real-time measurements. This approach has certain advantages. For example, a conservative specification does not result in overallocation of resources for the entire duration of the service session. Also, when traffic from different flows is multiplexed, the QoS experienced depends on their aggregate behavior, the statistics of which are easier to estimate. However, relying on measured quantities raises issues, such as estimation errors and memory related issues [2].

The various admission control schemes can also be classified according to the location where the admission control decision is made; at a centralized server or at various possible points in a network in a distributed manner. The idea of centralized schemes is simple. Signaling messages are exchanged between the sender of the flow and the centralized entity and between routers in the network and the centralized entity. These messages include the requirements of the flow and the resources state at each router, therefore admission control is performed by an entity that has complete and up-to-date knowledge of the network topology and resources, which is an ideal situation. However, in practice, centralized schemes have certain disadvantages. One is that a centralized entity constitutes a single point of failure. Another is the scalability problems that a centralized scheme raises [3]. Distributed schemes avoid these problems, but the existence of multiple admission control decision points means that concurrent admission control decisions may be made by distinct decision points for flows competing for the same resources; this can lead to QoS violations. In order for concurrency to be handled there exist some proposals in the literature [4], such as employing some safety margins to absorb the negative effects of concurrency.

Most of the schemes, to be applicable in practice, explicitly or implicitly make the assumption that the traffic is intra-domain; that is it originates and terminates within the same domain. The schemes that do not make this assumption, in many cases, e.g. see [5], require the cooperation of adjacent domains along the end-to-end paths on a per-flow basis as well as the existence of a commonly understandable signaling protocol end-to-end in order to perform admission control in each domain and propagate downstream the admission control decision and/or the QoS received so far.

Contrary to these schemes, in this paper we present a measurement-based admission control scheme for inter-domain real-time traffic originating from DiffServ stub domains, which when deployed in the context of a cascaded QoS peering model, does not require the cooperation and signaling among adjacent domains on a per-flow basis. In the rest we will first present the assumptions and conditions needed for this scheme to provide end-to-end QoS (section 2). We will then describe in detail our scheme (section 3) and we will evaluate and compare its performance against other schemes found in the literature (section 4) before concluding the paper in section 5.

2 Assumptions and Conditions

2.1 Existence of a Cascaded QoS Peering Model

The main assumption in our scheme is that a cascaded QoS peering model, similar to the one of the MESCAL project [6], is employed in the Internet. Each network provider or Autonomous System (AS) establishes provider service level agreements (pSLAs) with the directly interconnected network providers. This type of peering agreement is used to provide QoS connectivity from a customer to reachable destinations several domains away. Fig. 1 gives an overview of the operations in this model.

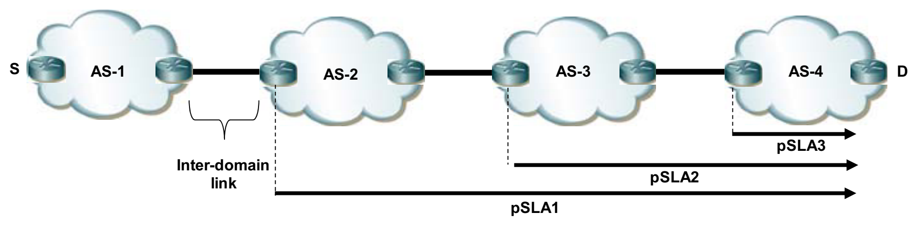

Fig. 1. A cascaded QoS peering model

AS-3 negotiates and establishes a peering agreement with AS-4 (pSLA$_3$) that will allow customers of AS-3 to reach destinations in AS-4 with specific QoS guarantees as long as the total aggregate demand from AS-3 does not exceed the negotiated and agreed bandwidth value in pSLA$_3$. AS-2, in turn, can negotiate with AS-3 a peering agreement (pSLA$_2$) in order to reach destinations in AS-4 with specific QoS guarantees. These guarantees are derived by combining the guarantees specified in pSLA$_3$ and the local QoS capabilities of AS-3. In a similar way, AS-1, which is the DiffServ stub domain, can establish a peering agreement pSLA$_1$ with AS-2 which defines the QoS guarantees that the traffic exiting AS-1 will receive from the ingress nodes of AS-2 till the end-customers connected to AS-4 as long as the aggregate demand from AS-1 does not exceed the negotiated and agreed in pSLA$_1$ bandwidth value.

Since pSLAs are established for aggregate demands, each network provider typically only has to manage a limited number for pSLAs, making the cascaded model scalable. By assuming that such a cascaded QoS peering model exists, DiffServ domain AS-1 does not need to cooperate and signal any of the downstream domains on a per-flow basis for traffic destined to remote destinations. It only needs to ensure that its inter-domain traffic does not exceed the negotiated bandwidth value in the corresponding pSLAs and that the QoS received by this traffic inside AS-1, when combined with the QoS values specified in the pSLAs is adequate to meet the end-to-end QoS requirements. The domains in a QoS chain just need to ensure they enforce the local QoS and that the traffic exiting them towards the next domain in this chain, is mapped to the appropriate class of the downstream domain according to the relevant pSLAs.

In this paper we focus on how the DiffServ stub domain AS-1 will ensure that the inter-domain real-time traffic originating from end-users of AS-1 will receive the required ‘local’ QoS treatment so that when combined with the QoS specified in the corresponding pSLAs it will still meet the end-to-end QoS requirements.

In the rest and for the sake of simplicity we will assume that towards the destinations of interest in a remote domain, AS-1 has one pSLA in place with AS-2 which specifies a bandwidth value C_{pSLA} and the associated packet loss rate PLR_{pSLA}, delay D_{pSLA} and jitter J_{pSLA} guarantees that will be met as long as the real-time traffic demand does not exceed the negotiated and agreed bandwidth value C_{pSLA}.

2.2 Local QoS Versus End-to-End QoS

Given that the delay and jitter parameters are additive and that, for low values, packet loss is also additive [5], and by knowing the end-to-end requirements regarding packet loss $PLR_{end-to-end}$, delay $D_{end-to-end}$ and jitter $J_{end-to-end}$ of the real-time traffic and also the relevant values agreed in the pSLA, it is straightforward to deduce the local QoS values that need to be enforced in the DiffServ domain AS-1. If we denote as PLR_{local} the local PLR requirement, as D_{local} the local delay requirement and as J_{local} the local jitter requirement, then these are given by:

$$\begin{aligned} PLR_{local} &\leq PLR_{end-to-end} - PLR_{pSLA} \\ D_{local} &\leq D_{end-to-end} - D_{pSLA} \\ J_{local} &\leq J_{end-to-end} - J_{pSLA} \end{aligned} \tag{1}$$

2.3 Enforcing Local QoS for Inter-domain Real-Time Traffic

We define as real-time traffic, sources that have strict *delay* and *jitter* requirements and a bounded *packet loss rate* (PLR) requirement. Regarding low delay and jitter, both requirements are likely to be met in a high-speed network core [7]. Furthermore, certain off-line traffic engineering actions can be taken so that delay and jitter are kept within low bounds. For example, the delay requirement can be taken into account by: a) configuring appropriately small queues for the real-time traffic in order to keep the per-hop delay small, and b) controlling routing to choose paths with a constrained number of hops. Jitter can remain controlled as long as the real-time traffic flows are shaped to their nominal peak rate at the network ingress [8]. Also, the deployment of non-work conserving scheduling can be beneficial for controlling jitter [9].

Given that a certain small amount of packet loss can be acceptable [7] without significant quality degradation and that delay and jitter can be controlled by taking the above actions, in this paper we employ the PLR as the QoS metric that needs to be controlled by the admission control scheme employed in the DiffServ stub domain and we focus on keeping it in values lower than the local PLR requirement.

2.4 Measurement/Enforcement Points

A measurement based admission control scheme needs to ensure that it controls the flow of traffic across all possible congestion points (bottlenecks).

As stated in [10], the edge links are currently considered as the most probable congestion points of a domain, whereas backbone links are overprovisioned. Therefore we assume that packets are lost at the DiffServ domain's ingress nodes, whereas in the core of the DiffServ domain, real-time traffic aggregates from different ingress nodes are treated in a peak rate manner. This means that the core is transparent to the real-time traffic sources with respect to packet loss. By assuming that the interior of the DiffServ domain has been engineered in this way and by taking into account the routing behavior, at each ingress node we can have an estimate of the bandwidth available for the inter-domain real-time traffic aggregate from that ingress (to be more precise, from that ingress node output interfaces) to each of the corresponding egress nodes specified in the corresponding pSLA. For inter-domain traffic, however, one also needs to take into account that peering links at the border routers between domains are also bottlenecks [11], therefore they cannot be considered overprovisioned.

Taking the above into account, the proposed scheme applies actions at these bottleneck points (output interfaces of ingress nodes and output interfaces of egress nodes) and aims to ensure that the total PLR incurred at these points is less than the local PLR requirement for the inter-domain real-time traffic. This means that for each pair of ingress-egress nodes output interfaces the following condition is met:

$$\begin{aligned}&\forall(l(i),m(e))\ with\ i\in I, e\in E, l(i)\in L_i, m(e)\in M_e\\&and\ f(l(i),m(e))=1: PLR_{l(i)}+PLR_{m(e)}\le PLR_{local}\end{aligned}\tag{2}$$

where $l(i)$ is the output interface l of ingress node i, $m(e)$ is the output interface m of egress node e, I is the set of ingress nodes with end-customers generating real-time traffic towards the destinations in the pSLA, E is the set of egress nodes that are specified in the pSLA as exit points for inter-domain real-time traffic from the DiffServ domain towards the destinations in the pSLA, L_i is the set of output interfaces of ingress node i, M_e is the set of output interfaces of egress node e, $PLR_{l(i)}$ is the incurred PLR at the output interface l of ingress node i, $PLR_{m(e)}$ is the incurred PLR at the output interface m of egress node e and $f(l(i),m(e))=1$ indicates that the output interface l of ingress node i uses the output interface m of egress node e as the exit point towards the destinations in the pSLA.

We assume that as a result of the provisioning phase, these sets of ingress-egress pairs, as well as the output interfaces pairing and the bandwidth allocated within the domain are already known. We will also denote as $C_{l(i)\to m(e)}$ the available bandwidth for the inter-domain real-time traffic from the output interface l of ingress node i until the output interface m of egress node e, as $C_{m(e),pSLA}$ the available bandwidth from the output interface m of egress node e till the destinations specified in the pSLA, and finally as $C_{e,pSLA}$ the available bandwidth for the inter-domain real-time traffic from egress node e till the destinations specified in the pSLA in place.

We assume that it holds:

$$C_{m(e),pSLA}=UF_{m(e)}\times\sum_{i\in I:f(l(i),m(e))=1}C_{l(i)\to m(e)},\ \forall e\in E\ and\ UF_{m(e)}<1\tag{3}$$

where $UF_{m(e)}$ is the underprovisioning factor, which indicates the extent to which the inter-domain links are underprovisioned with respect to the aggregate bandwidth reservations at the output interfaces of the ingress nodes. $UF_{m(e)}$ needs to be a number with value less than 1, otherwise the inter-domain links would not be bottlenecks.

We also assume that it holds:

$$\sum_{m\in M_e}C_{m(e),pSLA}=C_{e,pSLA}\quad and\quad\sum_{e\in E}C_{e,pSLA=}C_{pSLA}\tag{4}$$

In the next section we will present our scheme, which is distributed and does not require any cooperation between ingress nodes or any per ingress-egress operations or monitoring. It requires per-flow signaling only from the end-users till the ingress node of the DiffServ stub domain they are connected to but not further downstream and it tries to ensure that the local PLR requirement is met by regulating the admission of new flows but not by penalizing or terminating prematurely existing flows.

3 Admission Control Scheme

As stated in [12], in order for an admission control scheme to be successful in practice, it has to fulfill the following requirements.

- *Robustness:* A scheme must ensure that the requested QoS is provided. This is not trivial; for measurement-based schemes, measurement inevitably has some uncertainty, potentially leading to admission errors. The QoS should also be robust to traffic heterogeneity, long-range dependency, and to heavy offered loads.
- *Resource utilization:* The secondary goal for admission control is to maximize resource utilization, subject to the QoS constraints for the admitted flows.
- *Implementation:* The cost of deploying a scheme must be smaller than its benefits. In addition, the traffic characteristics required by the scheme should be easily obtained and the scheme should scale well with the number of flows.

3.1 Admission Control Logic

Our scheme consists of two modules, one module running at each ingress node i serving inter-domain real-time traffic from that node till each one of the egress nodes and one module running at each egress node e. The modules running at the ingress nodes make admission control decisions independently from each other, aiming to regulate the admission of new flows, based on feedback from the egress nodes modules.

The egress nodes modules continuously monitor the state of the egress output interfaces (to be more precise, the status of each of the output queues configured with bandwidth limit $C_{m(e),pSLA}$) and based on their status, at intervals of duration S they communicate PLR information to the ingress nodes that use these egress output interfaces as exit points for their inter-domain real-time traffic. This PLR information is used by the ingress nodes modules to calculate new PLR values to be used -if needed- for the admission of new flows. This means that each egress node only communicates with the ingress nodes that actually use them as exit points for their inter-domain real-time traffic. We need to clarify here that the communicated information relates to the PLR of the aggregate traffic using this egress output interface and not to the PLR of traffic originating from distinct ingresses, therefore the egress nodes do not need to keep any per-ingress state or perform any ingress-specific operations.

3.2 The Ingress Node Module

The functionality of the ingress node module is very similar to the functionality of the module in case of intra-domain traffic described in detail in [13].

We assume that every time an inter-domain real-time flow wants to be established, it signals this to the ingress node i. Then the module, based on a target PLR level $PLR_{l(i),target}$, decides to accept the flow establishment if the bandwidth $C_{l(i)\to m(e)}$ from that output interface l of ingress node i till the egress node e is enough in order to accommodate the existing flows and the new flow requesting admission, while at the same time satisfying this $PLR_{l(i),target}$ value. Since, as stated above, each egress node does not keep any per-ingress state and only communicates one PLR information per egress output interface, all $PLR_{l(i),target}$ values for all ingress nodes that use the same output interface of egress node e as exit point, should be the same. For the rest we will denote the PLR target at interface l of ingress node i, associated with the interface m of egress node e, as $PLR_{l(i),target}^{m(e)}$. This target $PLR_{l(i),target}^{m(e)}$ level is not fixed but is adjusted based on the feedback. Also, in order for the scheme to be able to recover the total locally incurred PLR to values less than the local PLR requirement without having to penalize or terminate existing flows, this $PLR_{l(i),target}^{m(e)}$ should be less than the local PLR requirement, that is:

$$PLR_{l(i),target}^{m(e)} \leq PLR_{local} \times OMF_{l(i)}^{m(e)} \; with \; OMF_{l(i)}^{m(e)} \in (0,1) \tag{5}$$

where $OMF_{l(i)}^{m(e)}$ is an Operational Margin Factor, defining the operational area within which $PLR_{l(i),target}^{m(e)}$ can range. $OMF_{l(i)}^{m(e)}$ should not be given a value close to one and the reason for this is that if, for example, $PLR_{l(i),target}^{m(e)}$ is allowed to get close or become equal to PLR_{local} and an overload situation occurs at the egress node output interface with bandwidth limit $C_{m(e),pSLA}$, then it may not be possible to recover the total locally incurred PLR to values less than the local PLR requirement just by regulating the admission of new flows, because the overload is caused by the existing flows and it will persist until some of the existing flows are terminated. In a similar manner, $OMF_{l(i)}^{m(e)}$ should not be set to very low values, because then the range [0, $PLR_{local} \times OMF_{l(i)}^{m(e)}$] within which $PLR_{l(i),target}^{m(e)}$ can range will be very limited, which will reduce the ability of the ingress nodes modules to react and regulate the admission of new flows, regardless of the feedback information.

3.3 The Egress Node Module

The egress node module passively monitors the output interfaces with bandwidth limit $C_{m(e),pSLA}$ (for the sake of simplicity, we will focus on one egress output interface and refer to it simply as output queue) and every S seconds (we will refer to S as the reporting period) it calculates the packet loss during the past interval of T seconds and depending on its value it reports back to the ingress nodes, which then adjust the target $PLR_{l(i),target}^{m(e)}$ level accordingly.

3.3.1 Egress Node Module Functionality

The desired functionality for the egress node module is to be able to react not abruptly but smoothly (still in a timely fashion) and provide feedback to the ingress nodes modules to regulate the admission of new flows. In order to achieve this smooth but timely operation, the egress node module when first senses a possible congestion situation, it initially tries to correct it by applying a set of 'mild' actions and if this situation is not resolved then it adopts more drastic 'emergency' measures.

In order to achieve this progressive operation, we define two threshold PLR values, named *soft threshold* and *hard threshold* respectively, against which $PLR_{m(e),T}$ is compared and depending on whether it crosses them (upwards or downwards) a specific set of actions is taken. The former threshold is denoted as soft, because it is allowed to be crossed upwards and still the status of the inter-domain link can be considered as not imminently close to becoming congested, whereas the latter is denoted as *hard*, because when it is crossed upwards, it means that the inter-domain link is imminently close to becoming congested. Since by employing the Operational Margin Factor, we have defined an upper value for the PLR allowed at the ingress nodes, both these thresholds should belong to the range $[0, PLR_{local} - PLR_{local} \times OMF_{l(i)}^{m(e)}]$.

3.3.2 Soft and Hard Threshold

The *soft threshold* $PLR_{m(e)}^{soft}$ is a PLR value, which, as long as it is not crossed upwards by $PLR_{m(e),T}$, no action is taken by the ingress node modules and no communication packets are sent. The range $[0, PLR_{m(e)}^{soft}]$ for $PLR_{m(e),T}$, therefore, corresponds to a 'normal operations' range. While in this range, the ingress nodes modules perform admission control using $PLR_{local} \times OMF_{l(i)}^{m(e)}$ as the $PLR_{l(i),t\arg et}^{m(e)}$ level.

The *hard threshold* $PLR_{m(e)}^{hard}$ is a PLR value that defines a range $(PLR_{m(e)}^{soft}, PLR_{m(e)}^{hard}]$, which indicates that a potential congestion situation may arise. While the measured $PLR_{m(e),T}$ is in this range, the egress node sends back to the ingress nodes communication packets that contain as information the difference between $PLR_{m(e),T}$ and $PLR_{m(e)}^{soft}$; that is the $PLR_{m(e),T} - PLR_{m(e)}^{soft}$ value. The ingress nodes receiving this value react to the potential congestion situation by adjusting the $PLR_{l(i),t\arg et}^{m(e)}$ level. In order for the ingress node modules to perform more conservative admission control as $PLR_{m(e),T}$ increases, we set the $PLR_{l(i),t\arg et}^{m(e)}$ level to be:

$$PLR_{l(i),t\arg et}^{m(e)} = PLR_{local} \times OMF_{l(i)}^{m(e)} - (PLR_{m(e),T} - PLR_{m(e)}^{soft}) \tag{6}$$

That is, the more the measured $PLR_{m(e),T}$ deviates from the soft threshold and approaches the hard threshold, the more conservative the admission control becomes. In practice, the ingress node modules attempt to compensate for these deviations by decreasing by the same amount the $PLR_{l(i),t\arg et}^{m(e)}$ value. If, however, despite the regulation of the admission of new flows, $PLR_{m(e),T}$ continues to increase and crosses upwards the hard threshold, then the ingress node modules completely block all incoming admission requests until $PLR_{m(e),T}$ returns to a value lower than the hard threshold. If $PLR_{m(e),T}$ keeps decreasing and becomes lower than the soft threshold, then the $PLR_{l(i),t\arg et}^{m(e)}$ level is set equal to $PLR_{local} \times OMF_{l(i)}^{m(e)}$ and the egress node stops sending communication packets till the soft threshold is crossed upwards again.

This approach minimizes the control overhead, since communication packets are only sent when needed. However, if these packets cannot be guaranteed loss-free delivery, then the ingress node modules may erroneously translate the non-delivery of a communication packet as a recovery to the 'normal operations' range. In such cases, one alternative would be to have communication packets sent continuously every S seconds so that the ingress nodes can detect the loss of a packet.

3.4 On the Selection of the Parameter Values

3.4.1 The Reporting Period *S*

The reporting period S defines how up-to-date with the current egress node output queue status, the ingress node modules are.

The lower the value of S the more up-to-date the information the ingress node modules use when making admission control decisions. However, the lower the value of S, the higher the control overhead. Furthermore, a very low value of S will not allow the traffic contribution of the recently admitted flows to be depicted properly in the measured $PLR_{m(e),T}$ and, therefore, in the reported $PLR_{m(e),T} - PLR_{m(e)}^{soft}$ value.

On the other hand, when an ingress node module performs admission control for flows arriving within two reporting periods, it is not aware of the actual effect that each of these flows will have at the egress nodes output interfaces. Therefore, within a longer S seconds period, the higher the number of arriving flows requesting admission and as a consequence the higher the possibility of making erroneous admission control decisions. In order for this phenomenon to be minimized, egress routers should explicitly perform admission control on a per-flow basis.

Moreover, since the ingress node modules do not cooperate with each other, they may make concurrent admission control decisions. This means that every ingress node is not aware of the traffic contribution from the other ingress nodes towards the same egress node output interface during an S seconds period. And the longer this S seconds period, the higher the number of arriving flows, and, therefore, the higher the possibility for each ingress node to make erroneous admission control decisions. In order for concurrency to be accounted for in our scheme, where competition between ingress nodes takes place only for resources on the inter-domain links, we employ some safety margins when setting the soft and hard threshold values.

As a result of the above discussion we conclude that the value for reporting period S should be a compromise between the above mentioned contradicting requirements.

3.4.2 The Measurement Window *T*

A small value of T will have as an effect the egress node modules to react abruptly to bursts. Moreover, for low values of PLR, a small value of T will mean that the measured $PLR_{m(e),T}$ may not be representative of the real output queue congestion status. On the other hand, a high value of T will reduce the ability of the scheme to react to non-stationarities and will also introduce correlation between successive admission control decisions [2]. Therefore, the value of T should be a compromise between these contradicting requirements.

3.4.3 The Soft and Hard Thresholds

The soft and hard threshold values define three operation ranges, which are:

- [0, $PLR_{m(e)}^{soft}$], normal operation
- ($PLR_{m(e)}^{soft}$, $PLR_{m(e)}^{hard}$], potential congestion
- ($PLR_{m(e)}^{hard}$, $PLR_{local} - PLR_{local} \times OMF_{l(i)}^{m(e)}$], immediate congestion

Therefore, the value $PLR_{m(e)}^{soft}$ determines when the scheme will start reacting to increases in the measured $PLR_{m(e),T}$. The value $PLR_{m(e)}^{hard}$ determines when the scheme will start taking 'emergency actions' to heal immediately impending congestion situations and the difference $PLR_{m(e)}^{hard}$ - $PLR_{m(e)}^{soft}$ determines for how long the scheme will try to recover the system by applying 'mild' actions.

The $PLR_{m(e)}^{soft}$ value setting should take into account the $PLR_{local} \times OMF_{l(i)}^{m(e)}$ value, e.g. to guarantee that eq. 6 does not become negative before $PLR_{m(e),T}$ reaches the $PLR_{m(e)}^{hard}$ value. Also, the $PLR_{m(e)}^{hard}$ value, even though it could go up to $PLR_{local} - PLR_{local} \times OMF_{l(i)}^{m(e)}$, it should be set to lower values than that so as:

- To compensate for the effect of measurement errors.
- To compensate for concurrency-related issues.
- To allow the ingress node modules to react fast enough so that the local PLR requirement is met without having to penalize or terminate existing flows.
- To compensate for the fact that the exact effect of newly admitted flows on the status of the egress node output interfaces cannot be known beforehand. This is especially true, since the egress node modules are not aware of the traffic characteristics of individual flows.

To compensate for all the above, the practical solution we adopt is to set $PLR_{m(e)}^{soft}$ to a relatively low value and leave a margin between $PLR_{m(e)}^{hard}$ and $PLR_{local} - PLR_{local} \times OMF_{l(i)}^{m(e)}$.

4 Performance Evaluation

In order to evaluate the performance of our admission control scheme, we run simulations using the network simulator *ns-2* [14], with the topology of Fig. 2.

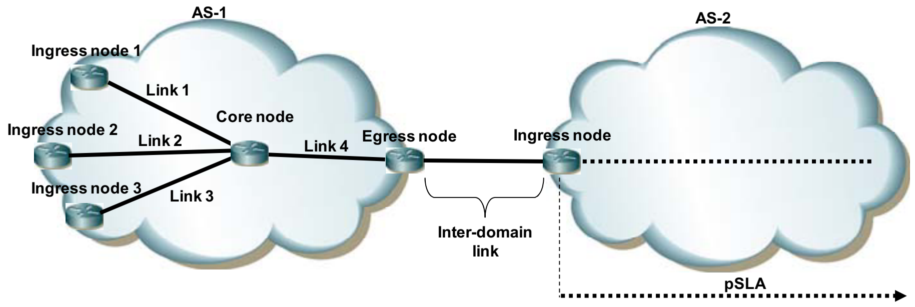

Fig. 2. Simulation topology

We use scenarios with the target local bound on PLR (PLR_{local}) for the inter-domain real-time traffic equal to 0.001. Since the value 0.01 defines a typically acceptable upper value of PLR for the VoIP service and for real-time applications in general [15], this implicitly means that the pSLA has to provide low, but not zero loss guarantees, to keep the end-to-end PLR below 0.01. We set the Operational Margin Factor for the ingress links 1-3 equal to 0.5, which means that the upper value that the target PLR at the ingress nodes output interfaces is allowed to get is equal to the half of the target local PLR, that is 0.0005.

We set the capacities allocated at links 1-3 for the inter-domain real-time traffic ($C_{l(i)\rightarrow m(e)}$) equal to 3.56Mbps. Since we assume that real-time traffic aggregates from different ingress node output interfaces are treated in the core in a peak rate manner, this means that the capacity allocated for the inter-domain real time traffic at link 4 is 10.68Mbps. We assume that the underprovisioning factor ($UF_{m(e)}$) is equal to 0.8, which means that the capacity allocated at the inter-domain link is 8.544Mbps. We also configure the queues at all links for the aggregate inter-domain real-time traffic to hold a maximum of 500bytes and we set the propagation delays at all links to be 5msec. For the sake of simplicity, we do not simulate the communication traffic, we do, however, consider the propagation delays from the instant it is generated at the egress node till the moment it can be used for admission control at the ingress nodes.

Regarding the algorithm's parameters, the employed values are: S = 1sec, T = 3sec, and we set the soft and hard thresholds equal to 40% and 60% of the $PLR_{local} - PLR_{local} \times OMF_{l(i)}^{m(e)}$ margin, which means that the employed value for the *soft* and *hard* threshold pair is (0.0002, 0.0003), meaning that 40% of the range $[0, PLR_{local} - PLR_{local} \times OMF_{l(i)}^{m(e)}]$ is left as safety margin.

In order to test the *robustness* of the scheme with respect to traffic *heterogeneity* and *long-range dependency*, we use a scenario with mixed VoIP and Videoconference traffic sources, the same as in [13]. In order to test the *robustness* of the scheme with respect to *offered load*, as in [13] we test varying loading conditions ranging from 0.5 to 5, where the value 1 (*reference load)* corresponds to the average load that would be incurred by a source activation rate equal to 1000 VoIP sources/hour.

In order to compare the performance of our scheme, which we call inter-MBAC, against other schemes, we implement the EAC scheme described by Karlsson et al in [16]. Since this scheme (we call it EAC-KAR) is an out-of-band probing scheme, we implement a lower priority queue for the probing packets that can store, as in [17], a single probe packet. As in [16], we set the probing rate equal to the peak rate of the source requesting admission and we consider probe durations of 0.5sec up to 5sec. Since the path that needs to be probed includes the inter-domain link, we assume that the probing takes place between the ingress nodes 1-3 of AS-1 and the ingress node of AS-2, which after the end of the probing process signals back to the ingress nodes of AS-1 the PLR that the probing packets experienced. We do not simulate these signaling flows, we do, though, for fairness reasons consider the propagation delays.

As stated in [17], any admission control scheme must address the trade-off between *packet loss* and *utilization.* Therefore, for performance evaluation we use as metrics the locally incurred PLR and the utilization of the inter-domain link, which is the main bottleneck, together with the average blocking rate. For most loading conditions, EAC-KAR is not able to keep the total locally incurred PLR below the 0.001 local PLR target. The results shown are for 5 seconds of probe duration, which gives the lower violation of the local target PLR.

4.1 Simulation Results

Inter-MBAC satisfies the target local PLR for all loading conditions. We observe an increase in the incurred PLR for higher loading conditions, which is anticipated because it relies on measurements, so every new admission request has the potential of being a wrong decision [2]. Furthermore, this is due to concurrency related issues; the higher the load, the more flows arrive within every reporting period S .

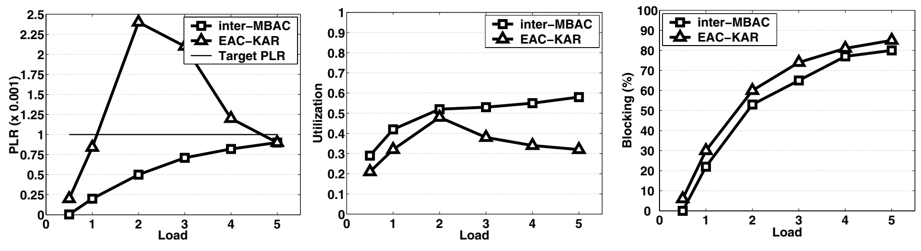

Fig. 3. Incurred PLR (left), inter-domain link utilization (centre) and blocking rate (right)

EAC-KAR violates the target local PLR for loading conditions more than one time the *reference load*. The trend of the incurred PLR for EAC-KAR indicates that it enters very early the thrashing region (for load more than two times the *reference load*) and despite the much higher (compared to inter-MBAC) incurred PLR, the achieved utilization is much lower and the incurred blocking is also higher. This behavior seems to be a consequence of concurrency related issues which exaggerate the thrashing effect and create an oscillation effect. Flows are initially admitted, then because of the amount of probing packets, subsequent flows are rejected, the real-time traffic class is emptied, then a batch of flows is erroneously admitted (which justifies the violations of PLR), then the subsequent flows are rejected (which justifies the high blocking and the low utilization) and so on.

4.2 Further Discussion of the Simulation Results

The simulation results show that inter-MBAC can satisfy the target PLR for all tested loading conditions without requiring reconfiguration of its parameters for individual loading conditions. EAC-KAR fails to satisfy the local target PLR for most loading conditions despite reconfiguring its probe duration. The local target PLR is satisfied for very high load conditions but this is actually due to the thrashing effect.

Regarding the control overhead, it is not straightforward to compare the two schemes using an absolute metric since we have not implemented the communication process or the signaling control process for EAC-KAR. However, we can state that since for EAC-KAR the control overhead is dependent on the number of flows, whereas for inter-MBAC the control overhead is dependent not on the number of flows but on the number of edge nodes, the control overhead of inter-MBAC is expected to be less than that of EAC-KAR in real network situations.

Moreover, for our simulation setup, for inter-MBAC and for low loading conditions (less than load equal to the *reference load*) the simulations show that no communication packets need to be sent back to the ingress nodes because the soft threshold value is not violated at any time. Therefore, there is no control overhead associated with inter-MBAC at very low loading conditions. For higher loading conditions, the control overhead increases and for loading conditions more than one time the *reference load* it stabilizes, since its frequency is determined by the reporting period S and not by the flow arrival dynamics. For EAC-KAR, there is control overhead at all loading conditions and it increases proportionally with the load.

5 Conclusions

In this paper we presented a measurement based admission control for inter-domain real-time traffic originating from DiffServ stub domains.

We showed through simulations that the scheme is *robust* to traffic heterogeneity, time-scale fluctuations and heavy offered loads. The scheme can meet the QoS objectives for a variety of loading conditions without requiring any reconfiguration of its parameters and without incurring significant control overhead. Furthermore, the scheme achieves satisfactory *utilization* and compares well against existing admission control approaches for the same simulation setup.

Our scheme is also easy to *implement*. It is distributed and does not require any cooperation between ingress nodes. Per-flow operations are only performed at the ingress nodes, and egress nodes do not need to keep any per-flow state or perform any per-flow or ingress-specific operations. The scheme requires per-flow signaling only from the end-users till the ingress node of the DiffServ stub domain they are connected to. Also since it is makes the assumption that a hop-by-hop cascaded QoS peering model between adjacent domains exists, it does not require any cooperation of adjacent domains along the end-to-end paths on a per-flow basis or the existence of a commonly understandable signaling protocol end-to-end.

Acknowledgments. This work was undertaken in the context of the IST ENTHRONE phase 2 and IST EMANICS projects, which are partially funded by the Commission of the European Union.

References

1. L. Breslau et al. "Endpoint Admission Control: Architectural Issues and Performance", SIGCOMM 2000.
2. M. Grossglauser et al. "A Framework for Robust Measurement-Based Admission Control", IEEE/ACM Transactions on Networking, June 1999.
3. C. Chuah et al. "Resource Provisioning using a Clearing House Architecture", IEEE IWQoS 2000.
4. S. Lima et al. "Distributed Admission Control in Multiservice IP Networks: Concurrency issues", Journal of Communications, June 2006.
5. S. Lima et al. "Distributed Admission Control for QoS and SLS Management", Journal of Network and Systems Management, September 2004.
6. M. Howarth et al. " Provisioning for Interdomain Quality of Service: the MESCAL Approach", IEEE Communications Magazine, June 2005.
7. G. Schollmeier et al. "Providing Sustainable QoS in Next-Generation Networks", IEEE Communications Magazine, June 2004.
8. T. Bonald et al. "Statistical Performance Guarantees for Streaming Flows using Expedited Forwarding", IEEE INFOCOM 2001.
9. M. Mowbray et al. "Capacity Reservation for Multimedia Traffics", Distr. Syst. Eng., 1998.
10. V. Padmanabhan et al. "Server-based inference of Internet Link Lossiness", IEEE INFOCOM 2003.
11. T. Bressoud et al. "Optimal Configuration for BGP Route Selection", IEEE INFOCOM 2003.
12. M. Grossglauser et al. "A Time-Scale Decomposition Approach to Measurement-Based Admission Control", IEEE/ACM Transactions on Networking, August 2003.
13. S. Georgoulas et al. "Heterogeneous Real-time Traffic Admission Control in Differentiated Services Domains", IEEE GLOBECOM 2005.
14. K. Fall et al. "The *ns* manual" (www.isi.edu/nsnam/ns/ns_doc.pdf).
15. T. Chaded, "IP QoS Parameters", TF-NGN, November 2000.
16. V. Elek et al. "Admission Control based on End-to End Measurements", IEEE INFOCOM 2000.
17. R. Gibbens et al. "Measurement-based connection admission control", 15th International Teletraffic Congress, June 1997.