Macrothink Institute™

**Network Protocols and Algorithms**
ISSN 1943-3581
2013, Vol. 5, No. 3

# A review of the architecture of Admission Control schemes in the Internet

Lluís Fàbrega, Teodor Jové

Institute of Informatics and Applications (IIiA)

Campus Montilivi, University of Girona, 17071 Girona, Spain

Tel: +34 972418889 E-mail: { lluis.fabrega, teodor.jove }@udg.edu

Received: May 20, 2013 Accepted: October 24, 2013 Published: October 31, 2013

DOI: 10.5296/npa.v5i3.3726 URL: http://dx.doi.org/10.5296/npa.v5i3.3726

**Abstract**

Admission Control (AC) is an efficient way of dealing with congestion situations in a network. Using AC, when network resources in a path are not enough for all flows (i.e., during congestion), some of the flows receive the requested service and the rest do not. Congestion situations can be reduced by increasing network resources or by optimizing their use through better routing techniques, but if congestion still occurs, AC achieves efficient use of network resources by maximizing the number of satisfied flows. However, using AC complicates the network scheme, and therefore a major concern is making the AC as simple as possible. In this paper we review the main AC schemes that have been proposed for the Internet, focusing on the simplicity of their architectures in terms of the number of nodes that participate in the AC, the required state, the use of signaling, and others.

**Keywords:** admission control; quality of service; network schemes; network services; network management.

## 1. Introduction

The introduction of Admission Control (AC) in the Internet has been controversial [1, 2, 3, 4]. An AC mechanism is able to assure the desired levels of Quality of Service (QoS) in a network as well as to achieve an efficient use of the existing network resources. However the traditional network architecture of the Internet, based only on First-In-First-Out (FIFO) and Tail Drop queues, is very simple, and the addition of an AC mechanism would make it more complex and costly. The desired levels of QoS can also be assured through over-provisioning the network resources, an option that allows keeping the network simplicity. However over-provisioning can be difficult to achieve, inefficient in the use of resources and expensive. A definitive answer to this question is not easy because it depends on factors that are difficult to predict such as the future cost of network resources and the characteristics of network traffic, but it is clear that the simplicity of the AC mechanism is a very important issue.

There have been many proposals of AC schemes for the Internet [5, 6]. The starting point was a classical hop-by-hop scheme with per-flow signaling and state in nodes, an approach that raised scalability concerns. Further research efforts were devoted to design simpler schemes, where fewer nodes were involved in the procedure, the amount of state maintained in nodes was less, or the required signaling was reduced.

In this paper we review the main AC schemes that have been proposed for the Internet, focusing on the simplicity of their architectures in terms of the number of nodes that participate in the AC, the required state, the use of signaling, and others. The paper is organized as follows. In Section 2 we present and discuss general issues about the AC mechanism and we propose a classification of AC schemes according to which nodes participate in the AC decision. Following this classification, we review specific AC schemes in Section 3. Finally, in Section 4, we present the conclusions.

## 2. The AC mechanism

In this section we cover the conceptual background of the AC mechanism. First we define the concepts of network service and network scheme, and discuss about the use of over-provisioning versus AC. Then we describe how the AC proceeds, i.e., the set of actions to do, and the different aspects that can be taken into account for evaluating an AC scheme. After that, we present our classification of AC schemes, and we deal with several important concepts related to AC, the AC algorithm, reservations based on state or on occupancy, and finally, explicit and implicit AC and its relation to signaling.

### *2.1 AC versus resource over-provisioning in network schemes*

The definition of a network service comprises two aspects, namely, a description of the desired QoS and a description of the input traffic profile of the flow that receives this QoS, through a set of traffic parameters (e.g., average rate, peak rate, etc.) and QoS parameters (e.g., maximum delay, maximum jitter, percentage of loss, throughput, etc.). A service is said to have absolute guarantees when the desired QoS is assured during the flow's lifetime, i.e.,

when it is not possible that the flow receives the desired QoS for a certain time and later the provided QoS gets worse (this is in contrast to services with no absolute guarantees, such as the relative services or the best-effort service, where the QoS is not assured) [1, 7]. The QoS parameters can be expressed in deterministic or "hard" terms (i.e., a set of QoS parameters are "always" met), statistical or "soft" terms (a set of QoS parameters are met during some specified percentage of time), or qualitative terms (a set of "generic" QoS parameters are met, such as "low" delay, "low" loss, "low" jitter, etc., but neither concrete values nor the percentage of time are specified) [1, 8, 9].

A network service is provided by a network scheme, which is composed of a combination of resource provisioning (link's capacity, queues, etc.) and mechanisms of management, routing, AC, traffic conditioning and queue disciplines. For example, the traditional network scheme in the Internet is simply based only on First-In-First-Out (FIFO) and Tail Drop queues, there is neither traffic conditioning nor AC mechanism, and resource provisioning can be whatever. If a "normal" resource provisioning is used, then transient congestion situations may occur, i.e., situations when resources in the followed network path are not enough to satisfy the QoS requirements of all flows (a "traffic overload"). When congestion does not occur, all flows are satisfied, but during congestion situations, none of them is satisfied, because the "few" resources are shared among all flows (it is said that this scheme provides the best-effort service, a service with no absolute guarantees) [2, 3, 4]. Congestion situations can be reduced by increasing resources or by optimizing their use through better routing techniques. If the resources are over-provisioned so that congestion never or rarely occurs, then this scheme always provides the desired QoS to all flows (it provides a service with absolute guarantees to all flows).

Over-provisioning the resources is a common practice in backbone networks, since it allows simple network schemes to be used. However, over-provisioning can be difficult to achieve since unexpected events may happen (inaccurate traffic forecasts, routing changes, link or router failures, etc.), and it can be very inefficient in using resources and expensive [2, 4]. If more efficient ("normal") provisioning is desired, another possible option is using network schemes that include an AC mechanism. By using AC, when resources in the followed network path are enough to satisfy the QoS requirements of all flows, all of them are satisfied; otherwise, i.e., during congestion situations, the "few" resources are allocated to only some of the flows, which receive the desired QoS (they are "accepted"), while the rest of the flows do not receive it (they are "rejected" or "blocked") [2, 3, 6]. Again, congestion situations can be reduced by increasing resources or by optimizing their use through better routing techniques, but if congestion still occurs, AC achieves efficient use of resources by maximizing the number of satisfied flows. The blocking rate depends on the behavior of users' demands, the chosen resource provisioning, the routing techniques used, and the capability of the AC mechanism to maximize the number of satisfied flows. However, using AC complicates the network scheme, and therefore a major concern is making the AC as simple as possible.

*2.2 AC procedure*

A new flow arriving to the network edge (Fig. 1) wishes to receive a given network service with absolute guarantees, expressed in deterministic, statistical or qualitative terms (here the term "network" can refer to either an entire network, where source and destination nodes would be hosts, or a single domain within a network, where source and destination nodes would represent edge nodes of neighboring domains). The requested service, which is indicated to the network through some way, comprises a description of the QoS and a description of the input traffic profile of the flow that receives this QoS. Network routing finds a path through the network towards the flow's destination. AC evaluates if in the chosen path it is possible to provide the service requested by the new flow while maintaining the service already promised to the actual flows in the path (note that these actual flows may travel, in general, through different network paths, which at some point, share one or more links with the path that has been chosen for the new flow). The AC decision must take into account the QoS requirements and traffic characteristics of the new flow and the actual flows in the chosen path, an information that is distributed in the network. Then the AC decision is indicated through some way to the flow, and the so-called AC phase ends. If the flow is rejected it will not receive the requested service. If the flow is accepted, it will receive the requested service during its lifetime, since the necessary resources of each node of the path will have been reserved for the flow. Moreover, in the case of acceptance and in order to isolate (or protect) the rest of accepted flows, the network will have to check the flow's input traffic and enforce the agreed traffic profile (e.g., discarding out-profile packets, degrading them to a lower QoS, etc.), so that if the flow exceeds the traffic profile, it does not damage well-behaved flows. This operation can be done through traffic conditioning mechanisms and/or some specific queue disciplines (e.g., Weighted Fair Queuing –WFQ–), at the ingress node or at each node, and it requires maintaining a list of accepted flows.

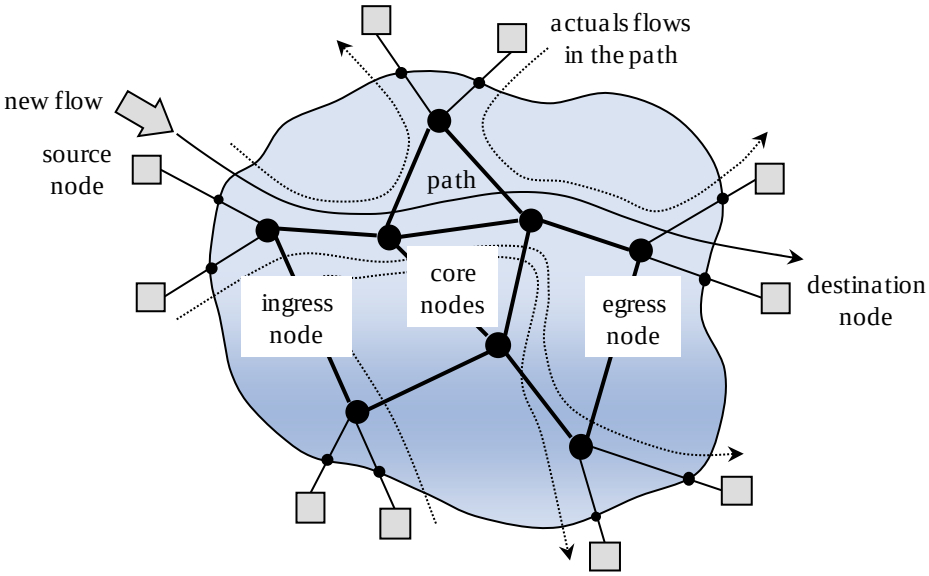


Figure 1. AC evaluates if in the chosen path it is possible to provide the service requested by the new flow while maintaining the service already promised to the actual flows in the path.

*2.3 Evaluation of an AC scheme*

When evaluating an AC scheme, besides the capability of maximizing the number of satisfied flows and the simplicity of the architecture, there are other aspects to study, such as the duration of the AC phase and others. A more detailed list of the various points to consider when evaluating an AC scheme is the following:

- The ability to provide the desired QoS guarantees (e.g., delay, jitter, loss, throughput), by comparing the target value of the QoS parameter with the average or percentile values of the QoS parameter achieved.
- The efficiency in the use of resources, by comparing the satisfied traffic load with the maximum possible traffic load (e.g., the capacity of links, a desired percentage, etc.).
- The simplicity of the scheme regarding how many nodes participate in the AC, the required state, the use of signaling, the intrusive traffic, computational efforts, etc.
- The duration of the AC phase, i.e., the time that is required for the AC decision. This should be short since usually it is the time an accepted flow has to wait before starting to transmit and also in order to improve resource utilization.
- The fairness in the AC decisions, i.e., whether different kinds of flows (e.g., flows requesting different traffic rates, traveling through different paths, etc.) have the same likelihood of being accepted.

Our review of AC schemes in this paper will focus mainly on the simplicity of their architecture.

*2.4 Classification of AC schemes*

We classify the AC schemes mainly according to which nodes are aware of AC, i.e., which nodes participate in the AC decision. A basic classification is the differentiation between centralized and distributed approaches:

- In *centralized AC schemes* there is a single AC entity in the network that makes the AC decisions and exchanges signaling packets with the ingress nodes when new flows arrive.
- In *distributed AC schemes*, AC decisions are made in different nodes. These schemes can be further classified into several types. In *hop-by-hop AC schemes*, the AC decision is split into partial decisions in each node of the path and each one performs a local AC decision. In *one-hop AC schemes on Logical Paths (LPs) with reservation*, a LP from ingress to egress nodes is previously established with a resource reservation, and after that only the ingress node is involved in the AC decision. In *edge-to-edge AC schemes* only the ingress and egress nodes participate in the AC, since the AC decision is taken from measurements (of different kinds) performed at the egress and sent back to the ingress through special packets.

In *centralized AC schemes* the AC requests are processed serially, while in *distributed AC*

*schemes*, they may be processed simultaneously in different nodes. This concurrency may cause false acceptance and rejection decisions. For example, in *hop-by-hop AC schemes*, concurrency does not lead to false acceptances since AC requests are processed serially in each node; however, partial reservations that further along the path are rejected, may lead to false rejections of other flows. Other *distributed AC schemes* also suffer from false AC decisions for similar reasons. False rejections due to many flows arriving simultaneously and causing low utilization are also known as the "thrashing" problem [10, 11].

Our complete classification of AC schemes is shown in Fig. 2. In Section 3 we follow this classification to describe specific AC schemes that have been proposed for the Internet: *centralized AC schemes* in Subsection 3.1, *hop-by-hop AC schemes* in Subsection 3.2, *one-hop AC schemes on LPs with reservation* in Subsection 3.3 and *edge-to-edge AC schemes* in Subsection 3.4.

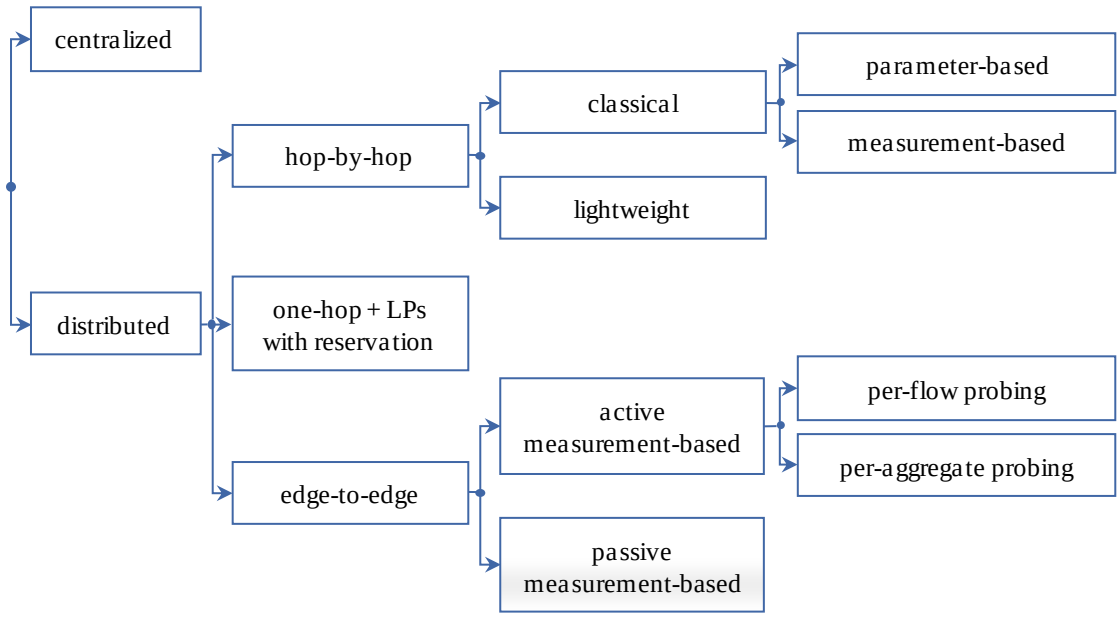


Figure 2. Classification of AC schemes.

### *2.5 The AC algorithm*

The central issue in an AC scheme is how to decide to accept or reject a flow and making the corresponding resource reservation. A false acceptance will cause the violation of the QoS guarantee, and a false rejection a lower utilization. It is a difficult task to understand the interaction between the new flow and the actual flows in the set of queues and links of the path, in order to predict if the QoS that all flows will receive will meet the requested QoS. Moreover, the information about the QoS requirements and traffic profiles of the actual flows in all links of the path is distributed in the network and it changes in time due to the arrival and departure of flows. This interaction can be modeled using complex theoretical principles or simpler ones, but it results in the so-called AC algorithm, which is a test for making the AC decision.

The AC algorithm may be implemented in one or more nodes depending on the type of

AC scheme. It is implemented in all nodes in *hop-by-hop AC schemes*; at the edge nodes in *one-hop AC schemes on LPs with reservation* and in *edge-to-edge AC schemes*; in a single network entity in *centralized AC schemes*.

The scope of the AC algorithm is the portion of the path where the resulting AC decision is valid. It can be a single link or the whole path:

- An AC algorithm with a link scope means that the resulting AC decision is valid in a link. The AC algorithm is used consecutively in each link of the path in order to obtain the total AC decision. For example, in a *hop-by-hop AC scheme*, the AC algorithm is implemented in all nodes, each node maintains the information about the available resources in its links, and usually signaling between nodes propagates the partial decisions in order to obtain the total AC decision. A link-scope AC algorithm may also be used in a *centralized AC scheme*, where a single entity maintains the information about the available resources in each link of the path.

- An AC algorithm with an edge-to-edge (or end-to-end) scope means that the resulting AC decision is valid in an edge-to-edge (or end-to-end) path. This is the case of all *edge-to-edge AC schemes*. An example would be an *active measurement-based edge-to-edge AC scheme with per-flow probing*: a probing flow with similar characteristics to the new flow is generated and sent through the path; its packet loss is measured at the egress of the path; a special packet carries the measurement to the ingress; the AC algorithm implemented at the ingress compares this measurement with the desired packet loss target in order to decide the admission. Another example would be the AC algorithm implemented at the ingress in a *one-hop AC schemes on LPs with reservation*: it takes an AC decision for the whole path as if there were a single link (one hop) from ingress to egress, with resources equal to the LP's reserved resources.

The AC algorithm may use traffic and QoS parameters declared by flows at the admission time, measured traffic and QoS parameters over individual flows or aggregates, resource parameters (queue length, link's capacity), and others. Usually the new flow is characterized by declared parameters at the admission time, but the already accepted flows (or in other words, the aggregated reservation, or reversely, the available resources) can be characterized by its declared parameters or by measurements over the actual traffic. In this way, AC algorithms can also be classified as being parameter-based or measurement-based:

- Parameter-Based AC (PBAC) algorithms use declared parameters for the new and the accepted flows. An example would be a link scope AC algorithm (within a scheme that guarantees a service with no loss) that admits a new flow in a link if the sum of all peak rates of the accepted flows plus the peak rate of the new flow is less than the link's capacity.

- Measurement-Based AC (MBAC) algorithms use declared parameters only for the new flow and characterize the accepted flows through measurements. Measurements can be made over individual flows or over aggregates. They can be passive (i.e., over

data packets) or active (i.e., over a special flow that is generated and sent for measuring purposes). An example would be a link scope AC algorithm (within a scheme that guarantees a service with low jitter and low loss) that admits a new flow in a link if the measured average rate of the accepted flows plus the peak rate of the new flow is below a certain percentage of the link's capacity. Another example is the edge-to-edge scope AC algorithm of the *active measurement-based edge-to-edge AC scheme with per-flow probing* that was described above, which admits a new flow if the measured packet loss of the probing flow at the egress of the path is smaller than the desired packet loss target.

*2.6 Reservations based on state and reservations based on occupancy*

Once a flow has been accepted, the necessary resources in the network path are reserved for the flow. The resource reservation for a flow in any node of the network path can be based on state or on occupancy:

- State-based reservations are maintained independently of whether the flow transmits or not and only depend on the state. The node maintains a list of the flows that have been accepted, which contains the flows' identifiers and service parameters (and maybe others). There is a link scope and parameter-based AC algorithm in the node, which uses the information stored in this list.

- Occupancy-based reservations are maintained as long as the flow transmits during its lifetime. There is no list of the flows that have been accepted in the node, and there is either a link scope AC algorithm in the node or an edge-to-edge scope AC algorithm for the whole path, which uses passive or active measurements over individual flows or aggregates. Usually it is required that the flow always transmits following a certain traffic profile that fills the reservation (although "virtual" measurements – Subsection 3.2.3 – can avoid this requirement).

Note that when flows' reservations in a node are based on state, accepting a flow implies a new entrance in its list of accepted flows. When flows' reservations in a node are based on occupancy, this operation is obviously not necessary. However, in some other node, e.g., in the ingress node of the path, in a centralized entity, etc., a list of the accepted flows should be maintained in order to be able to differentiate between the packets of new flows (i.e., its start and therefore to initiate the AC phase) and the packets of accepted flows, and also in order to isolate (or protect) the accepted flows by enforcing the agreed traffic profile on their input traffic. A flow may be identified by the usual 5-tuple in IPv4 (protocol, source and destination IP addresses and ports) or the more flexible 3-tuple in IPv6 (flow label, source and destination IP addresses).

*2.7 Explicit and implicit AC: the use of signaling*

The distributed nature of any AC scheme implies using some type of communication between the different nodes involved, i.e., the ones where the AC algorithm is implemented, or all the nodes of the path or other, which depends on the type of AC scheme:

- In *centralized AC schemes*, the centralized entity exchanges signaling packets with the ingress nodes and, if traffic measurements are used or maybe for other reasons, also with all the nodes in the path. The centralized entity maintains flows' reservations in nodes either based on state or on occupancy.
- In *hop-by-hop AC schemes* all the nodes of the path communicate with each other and each node maintains its flows' reservations based on state or on occupancy.
- In *one-hop AC schemes on LPs with reservation* only the ingress node is involved and maintains its flows' reservations based on state or on occupancy (although the previous reservation made in the establishment of the LP involves either all the nodes of the path or a centralized entity, and it is a reservation based on state).
- In *edge-to-edge AC schemes* only the ingress and egress nodes exchange special packets (e.g., for carrying measurements) and each node of the path maintains flows' reservations based on occupancy.

Therefore, an important issue is how the source, the destination and all the involved network nodes communicate to each other the information that is required for making the AC decision and (in the case of acceptance) for establishing the reservation, its maintenance and release. Specifically, we refer to information such as the following: the start of a new flow and its requested service parameters (traffic and QoS parameters) in order to begin the AC phase (i.e., the AC request); the acceptance or rejection (i.e., the AC response); the end of (an accepted) flow in order to release the reservation (i.e., the AC release request); or traffic measurements, if used. This communication can be done in two ways, explicitly or implicitly:

- The explicit way uses signaling packets, i.e., extra packets besides data packets. It allows rich and dynamic information to be indicated but it complicates the AC scheme and consumes resources that could be used for data traffic.
- The implicit way only uses data packets and signaling packets are not used. It is simpler and extra traffic is avoided, but the range of information is narrower: either the information is the same for all flows (and does not need to be indicated) or it has very few possibilities that can be indicated or derived through an already existing field in packets' headers.

Several explicit and implicit procedures can be used:

- The start of a new flow can be indicated by an initial signaling packet, or implicitly, by the first data packet of the flow. This initiates the AC phase. Both packets carry the flow's identifier.
- The QoS and traffic parameters can be indicated in different ways: both written in an initial signaling packet; or, for the traffic parameters, indicated by an initial sequence of (signaling or data) packets by means of the number of packets sent or by its rate; or the first data packet indicates the "type" of service simply through the port numbers or other packet fields marked for that purpose (e.g., a "real-time" or "elastic" mark), and the specific QoS and traffic parameters are not indicated since they are the same for

all flows of the same "type".

- The acceptance and rejection decision can be indicated by a signaling packet back, or implicitly, by forwarding the data packet in the case of acceptance, and in the case of rejection, by discarding it or modifying an agreed mark. The acceptance of a flow implies updating the list of accepted flows wherever it is placed.
- The end of a flow can be indicated by a signaling packet, or implicitly, when no packet is received within some defined timeout interval. State-based reservations are maintained independently of whether the flow transmits or not: they are released upon the reception of a release signaling packet ("hard" state), although a timeout procedure for detecting long periods of inactivity can also be used (e.g., for failures); or in case of using a reservation refreshing procedure, they are released when the refreshing signaling packet is not received within a defined timeout interval ("soft" state). Occupancy-based reservations are maintained as long as the flow transmits (following a certain traffic profile) during its lifetime: they are released when the flow ends and is inactive, without needing signaling packets (because the related AC algorithm is based on measurements).

## 3. Review of AC schemes

In this section we review specific AC schemes that have been proposed for the Internet following the classification proposed in Subsection 2.4 (Fig. 2). Our focus is mainly on the simplicity of their architectures in terms of the number of nodes that participate in the AC, the required state, the use of signaling, and others.

### *3.1 Centralized AC schemes*

In *centralized AC schemes* there is a single AC entity (where the AC algorithm is implemented) that maintains the information concerning the network topology and the state of resource usage in the whole network (Fig. 3). When a new flow requires admission, a signaling packet is sent from the ingress node to the centralized entity specifying the flow's identifier and the service parameters, and then the AC decision is made. For example, if a link scope and parameter-based AC algorithm is used, the AC entity obtains the available resources in each of the links of the chosen path from its state information database, and using the AC algorithm in each link, makes the AC decision. Then the ingress node is notified through signaling packets. If the flow is accepted, the AC entity updates its state information database and the ingress node updates its list of accepted flows; if necessary, signaling packets are also sent to the rest of nodes (recall that edge and core nodes might require this signaling information for classification, scheduling or traffic conditioning). If measurements were used in the AC algorithm, signaling packets would carry this information from the nodes to the AC entity.

The early proposals of Bandwidth Brokers (BBs) in Differentiated Services (Diffserv) networks were an example of *centralized AC schemes* [12, 13]. Diffserv networks are based

on packet classes, i.e., flows' packets are assigned to a small number of classes at the ingress (a mark that identifies the class is written in the packet's header), and queue disciplines in the core apply a different treatment to packets belonging to different classes [14]. BBs were responsible for authenticating and authorizing the service request, implementing domain policies about service usage, controlling domain agreements, making the AC decision, and the consequent edge nodes configuration (for classification and traffic conditioning). In addition, if the flow's destination was outside the domain, BB was responsible for negotiating with the downstream domain's BB (through interdomain signaling) in order to achieve the end-to-end service. A centralized approach was chosen due to its simplicity and because it does not require maintaining state in the core, a major concern in Diffserv.

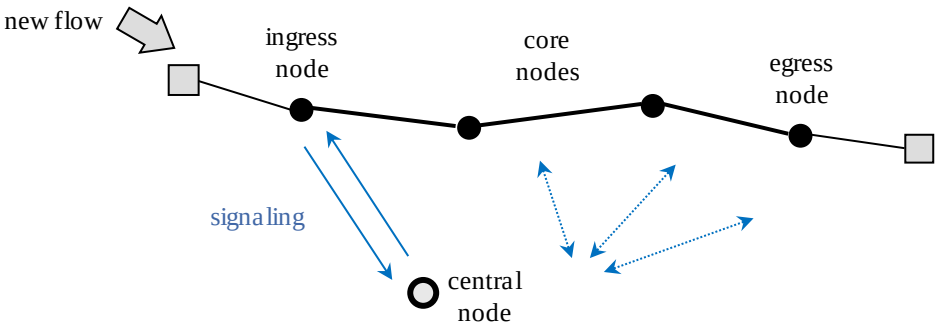


Figure 3. The procedure of *centralized AC schemes*.

*Centralized AC schemes* allow AC to be performed without complicating the control plane inside the network and eliminating the need for maintaining state in all nodes. Moreover, service requests can be processed serially and AC decisions are never made simultaneously. However, the centralized entity has to store the complete network state and process signaling packets in order to update it. This so demanding processing can be difficult to achieve in networks with a large number of flows and highly dynamic behavior (i.e., with many service requests and reservation release events). Moreover, the centralized entity can become a sending and receiving point for a lot of signaling traffic and suffer network congestion. Therefore, the *centralized AC scheme* is more appropriate in networks in which most flows are long, and service requests and reservation release events are not frequent. Finally, its dependence on a single entity makes it highly vulnerable to failures.

The next subsections are devoted to *distributed AC schemes*, in which AC decisions are made in different nodes and the state information is distributed in the network. These schemes are able to cope with large and highly dynamic networks and are more robust. However, decisions are not made serially as in *centralized AC schemes*, and therefore *distributed AC schemes* have to face the problems associated with simultaneous AC decisions.

### *3.2. Distributed hop-by-hop AC schemes*

In *hop-by hop AC schemes* the total AC decision is split into partial decisions in each node of the path, and each node maintains the actual aggregated reservation and performs a

local AC decision (through a link scope AC algorithm). The procedure is the following: if the flow is accepted in a node, a local reservation is made; then this partial acceptance decision is propagated to the next node in the path, and so on; if it reaches the end, the procedure stops and the total AC decision is acceptance; if the flow is rejected in a node, the procedure stops and the total AC decision is rejection. Usually signaling between nodes is used to propagate the partial decisions and to communicate the response: an AC request packet carrying the service parameters is sent from node to node and then an AC response packet carrying the total acceptance or rejection decision is sent back. The duration of the AC phase is about one Round-Trip Time (RTT). Traditional AC schemes are hop-by-hop and use per-flow signaling. The Intserv architecture [1] adopted the hop-by-hop approach and the Resource ReSerVation Protocol (RSVP) [15] as the signaling protocol.

Being a distributed scheme, the hop-by-hop approach can handle several AC decisions simultaneously, which may lead to concurrency problems. Since flows' requests in each node are processed serially, and in the case of acceptance, a local reservation is made, concurrency does not lead to false acceptance. However, "thrashing" may occur [10]: partial reservations in a hop, made for flows that later are rejected in other hops, may prevent other flows from being accepted in this hop, leading to false rejection.

Another feature of *hop-by-hop AC schemes* is that they exhibit multihop and multirate unfairness, i.e., they discriminate against flows crossing longer paths (in terms of the number of hops) and flows demanding larger traffic rates [9, 16, 17]. This problem, which is also a problem of other types of AC schemes, happens because admitting a flow is only based on the criterion of whether there are enough available resources. Multihop and multirate unfairness is usually considered to be a question that is orthogonal to the original AC problem, and that could be solved through adding other criteria to the AC decision (see, e.g., the "trunk reservation" mechanism in [17]).

As we explained in Subsection 2.5, the AC algorithm that is implemented at each node can characterize the accepted flows by its declared parameters at the admission time (PBAC) or through measurements (MBAC). In the next subsections we study in more detail *hop-by-hop AC schemes with PBAC algorithms* and *with MBAC algorithms*.

### 3.2.1 Classical parameter-based hop-by-hop AC schemes

*Hop-by-hop AC schemes with PBAC algorithms* use declared parameters at the admission time for the new flow and also for the accepted flows (for the aggregated reservation). PBAC algorithms allow building services with deterministic or statistical guarantees:

- One example of a deterministic guarantee is the peak rate allocation algorithm that guarantees zero packet loss if the sum of the peak rates of all accepted flows plus the peak rate of the new flow is less than the link's capacity (note that "real" peak rates vary as flows travel along the path, and therefore it is better to allocate slightly higher peak rates). A second example is the algorithm in [18], which guarantees no loss and a maximum delay $D$ (as the definition of the Guaranteed Service of Intserv [19]), by using a WFQ scheduler and characterizing flows with the average rate $r$ and burst size

*b* of a token bucket traffic profile: the flow is accepted if it can be assigned an unreserved portion *R* of the link's capacity (through the appropriate WFQ's weight), so that $D = b/R$ and $R > r$. An important point here is that both algorithms, due to the deterministic guarantee, are based on considering the worst-case scenario; therefore, if the flows are bursty, the utilization is low.

- Using strong mathematical models, PBAC algorithms can also provide statistical guarantees. By relaxing the deterministic guarantee to a statistical one, these algorithms consider scenarios that are likely to happen in most cases (with certain probability), instead of the worst-case, and thus they achieve better utilization. There are many examples based on this idea: in [17] several AC algorithms (for real-time traffic) are discussed, and classified as either based on rate envelope ("bufferless") multiplexing or on rate sharing ("buffered") multiplexing; in [20] there is a comparison of the performance of several sets of PBAC algorithms (average and peak rate combinatorics, additive effective bandwidths, engineering the "loss curve", maximum variance approaches, and refinements of effective bandwidths using large deviations theory); similar work is carried out in [21] for ATM networks.

A problem in PBAC algorithms is the need to accurately describe the flow's traffic parameters. It is difficult for a user to tightly characterize its traffic in advance; therefore, the traffic parameters declared at the admission time are inevitably loose upper bounds. This can result in low utilization.

*Classical hop-by-hop AC schemes with PBAC algorithms* use reservations based on state (with per-flow state in each node) and per-flow signaling between nodes. The ingress node (or all nodes) isolates the accepted flows by enforcing the agreed traffic profiles to their input traffic. State-based reservations are maintained, no matter whether the flow is active or not, until a release signaling packet is received, or in the case of using a reservation refreshing procedure, until the refreshing signaling packet is not received within some defined timeout interval. The signaling protocol includes an AC request packet that carries the flow's identifier and the service parameters, an AC response packet that carries the total acceptance or rejection decision in the path, and either an AC release packet or an AC refreshing packet. The procedure is the following (Fig. 4):

- Upon receiving the AC request packet, each node performs a partial AC decision.
- If the node accepts the flow, it makes a local reservation, it forwards the AC request packet to the next node, and it waits for the AC response packet that carries the total AC decision; if the destination accepts the flow, an AC response packet of acceptance is sent back to the source through the same path, and partial reservations are confirmed. The duration of the AC phase is about one RTT, after which an accepted flow is allowed to transmit.
- If the flow is rejected in one node, an AC response packet of rejection is sent back to the source through the same path, and partial reservations are released.
- When the flow ends, either the AC release packet is sent through the path or the

periodic sending of the AC refreshing packet is stopped, and the reservations in the path are released.

Note that these schemes require the per-flow state to be maintained in each node. In principle, it is possible to envision a simple solution without needing identifying individual flows, where each node maintains only the actual aggregated reservation, which is updated for each flow's acceptance or departure indicated by the signaling packets. However, if signaling packets suffer a loss, these packets are then retransmitted, and this may generate duplicates that nodes have to be able to detect by maintaining the per-flow state (to avoid duplicated reservations or releases). Moreover, per-flow state is also required in order that each node knows the previous node in the path followed by the flow, and signaling packets can be sent back through this path. The need to maintain the per-flow state in each node and to process per-flow signaling packets to update it, results in scalability limitations; therefore, this is one of the main disadvantages of these AC schemes.

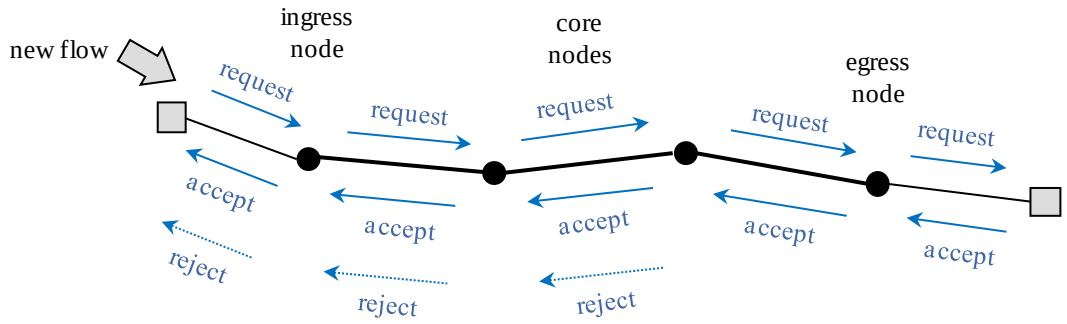


Figure 4. The procedure of *classical parameter-based hop-by-hop AC schemes*.

### 3.2.2 Classical measurement-based hop-by-hop AC schemes

*Hop-by-hop AC schemes with MBAC algorithms* use declared parameters at the admission time for the new flow and characterize the accepted flows (the aggregated reservation) through measurements. Measurements are (passively) taken over aggregates during a certain time interval, being the average rate (or traffic load) the usually used parameter, although others are also possible (e.g., the average delay, loss percentage, etc.). The estimation of the aggregated reservation obtained from measurements in a given time interval is used to make AC decisions in future time intervals. Usually, when a flow is accepted (during the "current" time interval), the estimation (which was obtained from the "past" time interval) is updated at once (artificially) to take into account the decrease in available resources, because measurements won't reflect this until the next time interval.

In MBAC algorithms, besides the AC algorithm itself, there is a second component, the measurement process, which can be carried out in different ways. As an example, we describe the Measured Sum AC algorithm together with the Time Window measurement algorithm, proposed in [22] for a low jitter and low loss service:

- The Measured Sum algorithm admits a new flow if the sum of the rate requested by the new flow plus the estimated aggregated reservation of accepted flows $R_T$, is less

than some defined percentage of the link's capacity (the percentage is required to bound jitter and loss).

- In the Time Window algorithm there is a measurement window $T$ divided into several time intervals $S$. At the beginning of $T$ there is an estimate $R_T$. For each interval $S$, the traffic load $R_S$ (i.e., the aggregated average rate in $S$) is obtained. If this new $R_S$ is above $R_T$, then $R_T$ is immediately increased until this new $R_S$. If a flow is admitted, $R_T$ is also immediately increased artificially with the requested rate of the new flow. At the end of $T$, the highest value of the set of obtained $R_S$ during the $T$ that has just ended is used as $R_T$ for the next $T$.

MBAC algorithms have several advantages over PBAC algorithms. Measurements can better estimate the aggregate behavior in flow's multiplexing and improve resource utilization (especially when the number of flows is high). It also eliminates the problem of tightly characterizing the traffic parameters of the new flow in advance. A very conservative and simple traffic specification (such as the peak rate) does not imply a waste of resources, because after some time, measurements in the next intervals will reflect the actual aggregated reservation. However, using measurements is not easy. Due to the dynamic behavior of flows, measurements taken in past time intervals cannot be accurate predictors for the future (especially in the presence of long-range dependence), and this can cause eventual QoS degradations and even false acceptances. In order to avoid false acceptances, measurements are used carefully, adopting very conservative methods. Due to eventual QoS degradations, MBAC algorithms cannot provide deterministic or even statistical guarantees, but they can provide qualitative guarantees (e.g., a low jitter and low loss service for real-time traffic, similar to the definition of the Controlled Load Service of Intserv [23]).

Examples of MBAC algorithms can be found in [9, 24, 25, 26] and a comparison in [16, 22, 27] (and also in [28] for ATM). More specifically, the works in [22, 27] describe and compare several AC algorithms (Measured Sum, Hoeffding Bounds, Tangent at Peak, Aggregate Traffic Envelopes, etc.) and several measurement algorithms of the traffic load (Time Window, Exponential Averaging, and Point Sample), to provide a low jitter, low loss service. One of the conclusions in [27] is that choosing one or another algorithm is not very important, since all achieve nearly identical performance in terms of the loss-load curve (i.e., the loss rate that occurs at a given utilization). The authors suggest that what is more important in the performance is the way the departure and arrival of flows is treated, and the responsiveness of the traffic load measurement to traffic fluctuations:

- When a new flow is admitted, the actual traffic load measurement does not reflect the presence of the new flow yet. In order to prevent the risk of admitting too many flows before the measurement is updated in the following time intervals, all these algorithms take the worst-case approach of increasing the actual measurement artificially with the declared flow's traffic parameters (e.g., the peak rate). This can reduce the utilization. When a flow ends, the algorithm does not know how much the departing flow was contributing to the actual measurement. The measurement is not explicitly adjusted, and it will not reflect the new situation until the next intervals.

This again may reduce the utilization.

- Although the number of flows does not change, the traffic fluctuates in the sort-term and the traffic load measurements reflect this. A high fluctuation can be confused by having a lot of admitted flows, which leads to too many flows being rejected; a low fluctuation can be confused by having few admitted flows, which leads to too many flows being accepted.

The length of the measurement interval can control the above two factors, but with conflicting goals: a longer measurement interval reduces the effect of the short-term traffic fluctuations but it also slows down the reaction of the algorithm to the departure and arrival of flows. Moreover, another important conclusion in [27] is that none of the studied algorithms is able to reliably achieve a loss rate close to the targeted loss rates.

In *hop-by-hop AC schemes with MBAC algorithms* reservations are based on occupancy due to the measurements being used. Classical schemes use per-flow signaling between nodes but per-flow state in the core is not required. Per-flow state is only kept at the ingress in order to detect new flows and to isolate the accepted flows by enforcing the agreed traffic profile on their input traffic. Occupancy-based reservations are maintained as long as the flow transmits (following a certain agreed traffic profile), and they are released when the flow ends and is inactive, without needing an AC release packet. The signaling protocol includes an AC request packet that carries the flow's identifier and the service parameters, and an AC response packet that carries the total acceptance or rejection decision in the path. The procedure is the following (Fig. 5):

- Upon receiving the AC request packet, each node performs a partial AC decision.
- If the node accepts the flow, it makes a local reservation (i.e., the measurement is artificially increased, e.g., with the flow's peak rate), and it forwards the AC request packet to the next node (the node does not need to wait for the AC response packet since in any case the aggregated reservation is updated through measurements); if the destination accepts the flow, an AC response packet of acceptance is sent back to the ingress node (and then to the source) to confirm the flow in the list of accepted flows. The duration of the AC phase is about one RTT, after which an accepted flow is allowed to transmit.
- If the flow is rejected in one node, an AC response packet of rejection is sent back to the ingress node (and then to the source), and the flow is erased from the list of accepted flows. Partial reservations are released in future measurements.
- When the flow ends, the reservations in the path are released in future measurements.

Note that the effect of possible duplicated AC request packets will also be corrected by future measurements.

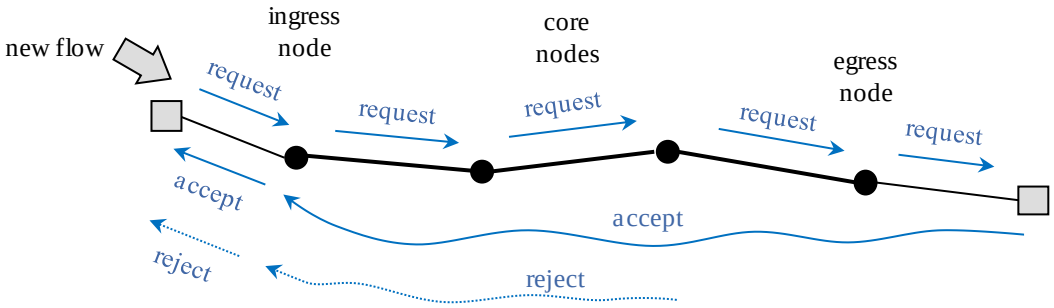


Figure 5. The procedure of *classical measurement-based hop-by-hop AC schemes*.

### 3.2.3 Lightweight hop-by-hop AC schemes

These schemes are hop-by-hop since each node maintains the actual aggregated reservation and performs a local AC decision. However, they differ from the classical schemes studied in the two previous subsections in one or more aspects, with the common goal of being simpler. These aspects are the following:

- Some schemes use less per-flow signaling between nodes, or even no signaling.
- The majority of schemes do not require a per-flow state in the core (only at the edge).
- All schemes use (passive) measurements over aggregates. However, some use "virtual" measurements instead of the "real" measurements used by classical MBAC algorithms. "Virtual" measurements can be seen as a way of using PBAC algorithms but without maintaining per-flow state in the node.

By "virtual" measurements we mean that the traffic load is not directly measured from ordinary data packets but from some "special" (data or signaling) packets that reflect the flow's reserved rate (and not the actual one, which can be smaller). In consequence, the aggregated reservation provided through "virtual" measurements aims to be equal to the sum of the reserved rates of the flows. This is in contrast to the one provided through "real" measurements, which aims to be the actual aggregated rate taking into account the multiplexing gain. Reservations maintained through "real" measurements require that the flow always transmits following a certain agreed traffic profile that fills the reservation, while through "virtual" measurements this is not required. "Real" measurements may increase resource utilization but run the risk that traffic fluctuations may cause eventual QoS degradations; therefore, they provide qualitative guarantees. "Virtual" measurements may provide deterministic or statistical guarantees but probably at the cost of reducing resource utilization. Therefore, "virtual" measurements obtain a similar behavior to PBAC algorithms without requiring per-flow state.

There have been many proposals of *lightweight hop-by-hop AC schemes*, and hereafter we describe some of them. For each one, firstly we present the service provided by the complete network scheme including the data path mechanisms, and then the AC scheme (how it works and how it fits into the three aspects explained above).

The *Scalable resource Reservation Protocol (SRP) network scheme* [29] provides a low

jitter and low loss service for real-time traffic (qualitative guarantees). It does not provide isolation to accepted flows and the scheme relies on the cooperation between sources implementing the same algorithms. These are the basic features of the data path:

- There are three types of packets: "requested", "reserved" and "best-effort" (the real-time service coexists with the best-effort service). All of these packets are flow data packets. In each node, "reserved" and "requested" packets are scheduled with priority over "best-effort" packets.
- When a flow is accepted, the source sends data packets marked as "reserved" at the accepted rate.

The AC scheme uses "real" measurements, and per-flow state is not required in the network core or at the edge. There is no AC request packet and the only signaling used is an AC response packet from the destination to the source (e.g., using RTCP):

- A source that wishes to make a reservation starts by sending its first data packets marked as "requested". The rate of these packets is the requested rate of the service. The source estimates the requested rate by measuring the sent "requested" packets.
- Each "requested" packet is forwarded unchanged by network nodes if it is accepted, or degraded to "best-effort" if it is rejected.
- The destination estimates the accepted rate by measuring the received "requested" packets, and then it sends this information back to the source through a signaling packet (the final accepted rate is the minimum between the source's measured requested rate and the destination's measured accepted rate).
- The local AC decision tries to keep the load of "reserved" and "requested" packets below a dedicated portion of the link's capacity so that they are not discarded and therefore have a low delay. At the beginning of a certain time period, each node knows the actual reserved load for this period (and therefore the available resources). An arriving "requested" packet is accepted only if enough resources are available, and if so, the amount of available resources is accordingly reduced. During the same time period the node monitors the arriving "reserved" and "requested" (and accepted) packets to determine the new reserved load for the future intervals.

The *Load Control network scheme* [30, 31] provides a low jitter and low loss service for real-time traffic (qualitative guarantees). It provides isolation to accepted flows. The basic features of the data path are the following:

- There are four types of packets: "probe", "marked", "refresh" and "ordinary". In each node, "probe", "refresh" and "ordinary" packets together receive a higher priority in queues than "marked" packets.
- When a flow is accepted, the ingress node marks (in-profile) flow's packets as "ordinary". It also refreshes the reservation periodically by marking some of these packets (or generating packets, if necessary) as "refresh". One "refresh" packet in a

given refreshment period indicates the reservation of one unit of resources for the time of this period.

The AC scheme uses "virtual" measurements, per-flow state is not required in the core and signaling is not reduced:

- A basic requested rate is defined in the domain, which requires one "unit of resources" (e.g., based on averages or effective bandwidths). When a flow desires a given number of unitary resources, the ingress node sends the same number of "probe" (signaling) packets through the path.
- A "probe" packet is forwarded unchanged by network nodes if it is accepted, or changed to a "marked" packet if it is rejected.
- When the "probe" or "marked" packets reach the egress node, they are sent back to the ingress node, which makes the AC decision (these backward packets can also be used to probe the backward path if necessary).
- For the local AC decision, each node knows, at the beginning of a given refreshment period, the amount of reserved resources (and the ones that are available). An arriving "probe" packet is accepted only if one unit of resources is available, and if so, the amount of available resources is reduced by one unit. During this refreshment period the node estimates the actual load to determine the amount of reserved resources for the next refreshment period. The measurement takes into account the accepted "probe" packets and the "refresh" packets (instead of the "ordinary" packets). Finally, some variations of this scheme were proposed in [31], such as replacing the initial sequence of "probe" (signaling) packets by a single signaling packet carrying the desired resources, using an AC release packet and improving the measurement algorithm.

The *network scheme based on Dynamic Packet State* (DPS) [32] provides a service with deterministic delay bound and no loss for real-time traffic. It provides isolation to accepted flows. A basic point of the scheme is the queue discipline:

- It uses the Core Jitter Virtual Clock scheduler, which provides the same deterministic delay bound as a set of WFQ schedulers. Per-flow state is not maintained in nodes, it is carried by data packets using the technique called DPS: the ingress node initializes several state variables encoded in the packet's header; in all nodes the scheduler processes each incoming packet based on this state, and then updates both its internal state and the state in the packet's header before forwarding it to the next node; the egress node extracts the state.
- When a flow is accepted, the ingress node writes each packet's "virtual" length in its header, i.e., the number of bits that the flow was supposed to transmit at its reserved rate $r$ since the previous packet was transmitted. It also inserts in the packet's header the initial values of the state variables required for the scheduling.

The AC scheme uses "virtual" measurements, per-flow state is not required in the core

and signaling is not reduced:

- When a new flow arrives to the network edge, the ingress node sends a request signaling packet for a reservation of peak rate *r* through the path. If a node in the path accepts the flow, it forwards the request packet to the next node; otherwise it sends back a reject signaling packet to the ingress.
- For the local AC decision, each node knows the actual aggregated reservation at the beginning of each measurement period, *Rbound*. A new flow is accepted if *Rbound* plus the peak rate *r* is below the link's capacity, and then *Rbound* is increased in *r*. During each measurement period, the node also calculates *Rcal*: it measures the actual aggregated rate using the "virtual" lengths instead of the real ones; and moreover, each time a flow is accepted, *Rcal* is also increased in *r*. At the end of the measurement period, *Rbound* is updated to the minimum value between *Rbound* and *Rcal*.

The *Corelite network scheme* [33, 34] provides a minimum throughput service with deterministic guarantees for elastic traffic. It provides isolation to accepted flows. The basic features of the data path are the following:

- When a flow is accepted, the ingress node turns some (in-profile) flow's data packets into "markers". The "marker" carries a number of data packets (or bytes) that it represents. The rate of "markers" indicates the accepted minimum throughput *r*. In each node, ordinary and "marker" packets are scheduled together using FIFO.

The AC scheme uses "virtual" measurements, per-flow state is not required in the core and signaling is not reduced:

- When a new flow arrives to the network edge, the ingress node sends a request signaling packet for a reservation of rate *r* through the path. If a node in the path accepts the flow, it forwards the request packet to the next node; otherwise, it sends back a reject signaling packet to the ingress.
- For the local AC decision, each node knows, at the beginning of a given time period, the available bandwidth $B_{av}$ for this period. A request is accepted if r is available ($r < B_{av}$), and then $B_{av}$ is reduced by *r*. During this time period, the node estimates the value of $B_{av}$ to be used for the next time period from the number of "markers" received over that time (taking into account the number of packets that the "markers" represent).

The *implicit network scheme for TCP* [35, 36] provides a minimum throughput service with qualitative guarantees for elastic traffic. The minimum throughput's value is the same for all flows and a flow is a TCP connection. It does not provide isolation to accepted flows and the scheme relies on traditional cooperation between TCP sources implementing the same algorithms. The data path is simply based on FIFO queues. The AC scheme uses "real" measurements, per-flow state is not required in the core or at the edge and there is no signaling:

- The start of the flow is indicated to the node through the TCP connection establishment packets (SYN or SYN/ACK). In the case of acceptance, the connection establishment is allowed to proceed by forwarding these packets; otherwise, the connection establishment is aborted.

- For the local AC decision, the node measures a particular parameter (the occupancy of the link in [35] and the incoming traffic to the link's queue in [36]), and when it exceeds a given threshold, new connections are rejected. The relation between the threshold and the minimum throughput comes from analytical models of TCP connections sharing a single link.

The *Flow-Aware Networking network scheme* [37, 38] provides two services with qualitative guarantees, a low jitter and low loss service for real-time flows and a minimum throughput service for elastic flows. The minimum throughput's value is the same for all elastic flows while the peak rate of real-time flows should be smaller than a given value. It provides isolation to accepted flows. The basic features of the data path are the following:

- The queue discipline uses the Priority Fair Queuing (PFQ) algorithm, which requires per-flow state in each node. It shares the link's capacity fairly between all flows and also gives scheduling priority to flows whose peak rate is less than the current link's fair rate. In this way, the requested flow's QoS (real-time or elastic) can be implicitly indicated: a flow whose peak rate is smaller than the fair rate is considered a real-time flow; otherwise it is considered an elastic flow.

The AC scheme uses "real" measurements, per-flow state is required in each node and there is no signaling:

- Per-flow state in each node is required to detect new flows and is maintained using an implicit method. A new flow is indicated by the arrival of its first packet. The node indicates a local acceptance decision of the flow by forwarding this packet or a local rejection decision by discarding it. The end of the flow is detected when no packet is received within a defined timeout interval.

- The AC algorithm does not distinguish between elastic and real-time flows. Moreover, the traffic parameter of the new arriving flow is supposed to be a given value (the maximum of the minimum throughput of elastic flows and the possible peak rates of real-time flows). The AC algorithm ensures that the current priority traffic load is smaller than a given percentage of the link's capacity, and that the fair rate is higher than a given threshold, which is chosen higher than the peak rate of the envisaged real-time flows (so that they receive scheduling priority in PFQ queues).

### *3.3 Distributed one-hop AC schemes on Logical Paths (LPs) with reservation*

Consider a network using LPs from ingress to egress points, established with a resource reservation (e.g., a given capacity) as in Fig. 6. The view is as if there were a single link from ingress to egress with these assigned resources, where an aggregation of flows travels through. In this way, AC only has to be done locally, only at the ingress, using any ("link scope") AC

algorithm in only one hop and only taking into account the LP's reserved resources. MultiProtocol Label Switching (MPLS) is an example of a network architecture supporting LPs with reservation [39].

Obviously, previously establishing the LP's reservation requires performing AC in the chosen path as if it were a "flow", with its traffic and QoS parameters. Note that unlike the per-flow AC we are considering in this paper, this AC is usually applied at a higher timescale than the flow's timescale, and with higher traffic requests. A *centralized AC scheme* or a *hop-by-hop AC scheme* can be used, with a link scope and parameter-based AC algorithm and state-based reservations (which are maintained although there is no traffic). The requested traffic and QoS parameters correspond to the aggregated traffic of the future accepted flows and their QoS requirements. For example, if a network provides a low jitter, low loss service for real-time traffic, implemented using priority queuing plus a limitation of the real-time traffic's load to a percentage of the links' capacity, it must be assured that the sum of the LP's capacity carrying this traffic in any link is below this limitation.

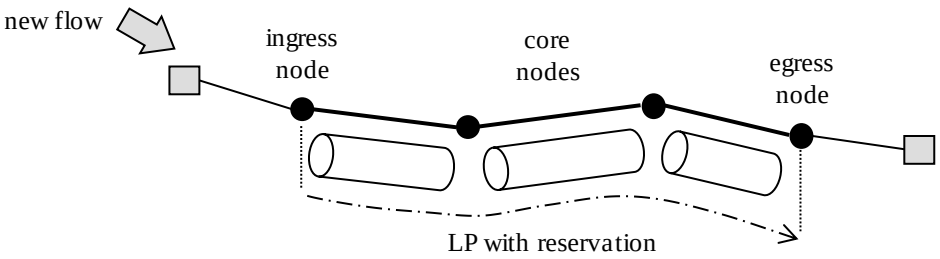


Figure 6. The procedure of *one-hop AC schemes on LPs with reservation*.

The main advantage of these schemes is their simplicity. It is a local decision, and it is not required to know about flows in other paths. Moreover, per-flow signaling is not necessary between nodes, since no per-flow reservation has to be made in transit nodes. The AC decision is fast since it is made as soon as the flow arrives. Simultaneous AC decisions in different ingress nodes cannot cause any false acceptance since resources in links have been reserved in advance. Thrashing cannot occur since in each LP, decisions are made serially. However, the main disadvantage of these schemes is that in some situations false rejections may occur: it is possible that the LPs passing through the same link share resources in a non-appropriate way, i.e., some might be congested and rejecting flows while others might be underutilized. In this situation the underutilized LP should decrease its capacity and the congested LP should increase it. The modification of the LPs is carried out by other network management functions [40] (which also consider other goals such as optimizing resource usage, fairness, etc.). Another disadvantage is that a significant number of LPs can divide the link's capacity into small blocks, therefore limiting the possible statistical multiplexing gain.

### *3.4. Distributed edge-to-edge AC schemes*

In *edge-to-edge AC schemes* only the ingress and egress nodes participate in the AC decision. There is an edge-to-edge scope AC algorithm implemented at the ingress or egress node, which is based on measurements performed at the egress and sent back to the ingress

through special packets (usually signaling packets). The nodes of the path do not maintain either per-flow or aggregate reservation state (they are not aware of AC), do not exchange signaling packets and maintain flow's reservations based on occupancy. Per-flow state is only kept at the ingress in order to detect new flows and to isolate the accepted flows by enforcing the agreed traffic profile to their input traffic. According to whether the measurements are active or passive, these schemes can be classified as:

- In *active measurement-based edge-to-edge AC schemes*, a special probing flow is generated and sent through the path, and its QoS is measured at the egress to be used in the AC decision. Moreover, probing can be per-flow, when there is a probing flow for each new flow, or be per-aggregate, when there is a single and continuous probing flow in relation to the aggregation of accepted flows.
- In *passive measurement-based edge-to-edge AC schemes*, the QoS of the aggregation of accepted flows is continuously measured at the egress to be used in the AC decision.

3.4.1 Active measurement-based edge-to-edge AC schemes with per-flow probing

These schemes, also known as *endpoint AC* or *end-to-end measurement-based AC* (*EMBAC*), are the majority of the *active measurement-based edge-to-edge AC schemes*. In some of them, the source and destination hosts, instead of the edge nodes, participate in the AC. They have in common that the AC phase consists in generating and sending a probing flow through the path, and then measuring its QoS at the egress to be used in the AC decision. The general procedure is the following (Fig. 7):

- For each new flow, a special probing flow travels from the sender to the receiver through the network path. The probing flow is a sequence of extra (i.e., signaling) packets with similar characteristics to the data packet flow.
- The receiver measures the QoS experienced by the probing flow during a defined time interval. The measurement can be simply the average rate received during the time interval, or counting the number of received packets with congestion marks (using Explicit Congestion Notification –ECN– [41]), or even be based on more complex jitter statistics.
- The receiver reports the measured QoS through a signaling packet to the sender, which performs the AC decision, or alternatively, the receiver decides and then informs the sender. There is a timeout mechanism in the sender associated with the start of the AC phase to deal with the loss of feedback signaling packets.
- According to the AC decision, the source sends packets or not. The duration of the AC phase is about one RTT plus the measuring interval, after which an accepted flow is allowed to transmit.

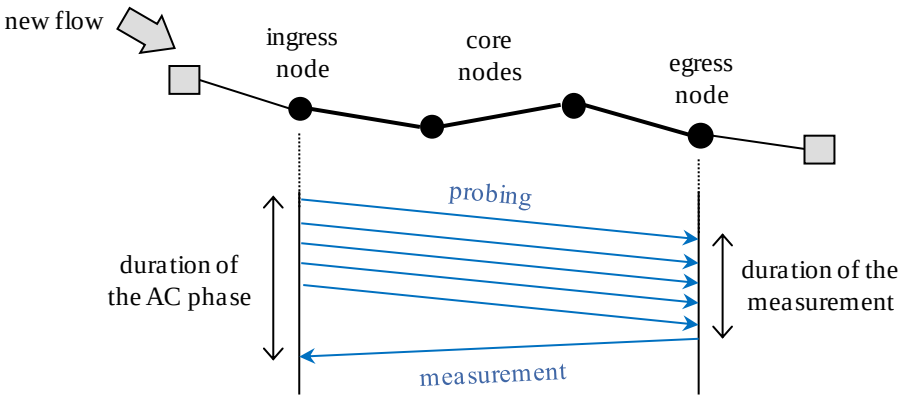


Figure 7. The procedure of *active measurement-based edge-to-edge AC schemes with per-flow probing*.

In some schemes the packets from probing flows are treated inside the network in the same way as packets from accepted flows (in-band probing). In others they are treated differently (out-of-band probing). In out-of-band probing, two packet classes are used: one for the packets of accepted flows and another for the packets from probing flows; in core nodes, accepted packets receive priority when using resources in order to be protected from the effect of probing packets.

Flow's reservations in nodes are based on occupancy. The probing flow establishes the reservation in the node if resources are available just by transmitting; the reservation is maintained as long as the flow transmits (following a certain traffic profile) during its lifetime; it is released when the flow ends and is inactive, without needing signaling packets. As they are based on traffic measurements, these schemes provide qualitative guarantees (e.g., a low jitter, low loss service for real-time traffic) because the dynamic behavior of traffic may cause eventual QoS degradations.

Examples of these schemes can be found in [11, 42, 43, 44], where they are used to build a low jitter, low loss service for real-time traffic. The Phantom Circuit Protocol [42, 43] uses a Constant Bit Rate (CBR) probing flow with a rate at 20% higher than the peak-rate of the new flow, and its jitter is measured and compared to a threshold to make the AC decision. It is an out-of-band probing scheme, with two packet classes, the 1st one (low priority) for the probing packets, and the 2nd one (high priority) for packets of accepted flows, and queues in nodes use two-priority scheduling. In [44] an analytical model is developed to evaluate the performance of a similar scheme using a throughput measurement (counting the number of received bytes within the measurement interval) instead of measuring the jitter. In [11] the performance of several schemes based on throughput measurements is discussed in detail through simulation. The schemes differ in using out-of-band or in-band probing, and a dropping or a marking mechanism (through ECN). Again two packet classes with different priorities are used, and nodes use priority queuing with a rate limited to a percentage of the link's capacity. Using marking and out-of-band probing has been shown to achieve better performance than the other options.

The main advantage of these schemes compared to *hop-by-hop AC schemes* is the edge-to-edge architecture. They do not require changes in the network core, where

reservation state is not maintained (neither per-flow nor per-aggregate), and only edge nodes are AC aware. However, the duration of the AC phase (and also the time an accepted flow waits for transmitting) may be quite long, because the measurement interval should be long enough to take into account a reasonable number of packets (e.g., if one considers between 50 to 100 samples [43], then a 32 kbps CBR flow with 640 bit packets would result into measurement intervals between 1 and 2 seconds). Reducing the measurement interval of the individual (probing) flow may make the QoS estimation very dependent on the traffic fluctuations and therefore reduce the performance. Moreover, the probing traffic in the network can be considerable because a probing flow is sent for each new flow. Thrashing may occur in out-of-band probing [11], since even though the number of accepted flows is small, simultaneous probing by too many flows can lead to false rejections (in the case of in-band probing, this situation can lead to a collapse, since accepted flows would lose their QoS guarantees). Finally these schemes also exhibit unfairness for multirate and multihop flows [11].

3.4.2 Active measurement-based edge-to-edge AC schemes with per-aggregate probing

In these schemes, a single probing flow, related to an aggregation of accepted flows, is generated and sent continuously through the path and its QoS is measured in order to be used in the AC decision. The procedure is the following:

- For an aggregation of accepted flows in a path, a single probing flow travels continuously from the ingress to the egress through the path.
- The egress node continuously measures the QoS experienced by the probing flow (i.e., in each particular time interval).
- The egress node reports the measured QoS through a signaling packet to the ingress node (periodically, or when a defined threshold is exceeded, etc.), which performs the AC decision, or alternatively, it decides and then informs the ingress node.
- According to the AC decision, a new flow is allowed to transmit or not. The AC decision is fast as it is made as soon as the new flow arrives.

One example of these schemes is [45], in which it is used to build an edge-to-edge AC for a multiservice network based on packet classes [46]. A probing flow, which contains timestamping and sequencing information, is sent for each packet class and path, and multiple QoS parameters are measured at the egress, such as delay, jitter and loss, although this depends on the packet class characteristics. Each measured QoS parameter (updated in each time interval) is then compared to a threshold to decide whether new flows can be admitted or rejected. The threshold is based on a target value of the packet class and also on a safety margin. This is necessary since the AC algorithms do not use the traffic parameters of the new flow. Probing is in-band but with a low rate (to not interfere with the data traffic), and uses a pattern that has been especially designed to better capture the QoS parameters of packet classes. In comparison with *active measurement-based edge-to-edge AC schemes with per-flow probing*, this scheme avoids per-flow intrusive probing traffic; the AC is fast, made at once since measurements are available online; and it increases the confidence level of

measurements since they are achieved using many samples instead of just a few. However, the AC algorithms do not immediately consider the effect of recently accepted flows until future measurements take them into account, but this takes some time. Therefore, a high rate of new arriving flows to an ingress-egress pair may cause false acceptances. The same problem holds for concurrently accepted flows in other ingress nodes. Some proposed solutions, which may lead to lower utilization, are increasing safety margins in the AC algorithms, or using some degree of over-provisioning, or a rate-based credit system controlled by egress nodes [47].

3.4.3 Passive measurement-based edge-to-edge AC schemes

In these schemes, the QoS of the aggregation of accepted flows is continuously measured at the egress to be used in the AC decision. No probing flow is generated. The procedure is the following:

- The egress node continuously measures the QoS experienced by the aggregation of accepted flows in a path (i.e., in each certain time interval).
- The egress node reports the measured QoS through a signaling packet to the ingress node (periodically, when a defined threshold is exceeded, etc.), which performs the AC decision, or alternatively, it decides and then informs the ingress node.
- According to the AC decision, a new flow is allowed to transmit or not. The AC decision is fast as it is made as soon as the new flow arrives.

One example of these schemes is [48], in which it is used to build a service with statistical guarantees on the maximum delay. It uses the theory of traffic envelopes to describe the aggregated rate of traffic at the ingress (“arrival envelope”), the available service in the path (“service envelope”) and the traffic characteristics of the new flow (“flow envelope”). Packet’s arrival times at the ingress (which are used to obtain the arrival envelope) are written on packets in order to measure their delay at the egress (which is used to obtain the service envelope). Besides continuously measuring, the egress node makes the AC decision. The new flow specifies the required service to the ingress node through RSVP packets, which are forwarded to the egress node. The AC algorithm considers the maximum new flow envelope (e.g., the peak rate), the desired delay bound, the desired maximum violation probability, the mean and the variance of the measured maximum arrival envelope, and the mean and the variance of the measured minimum service envelope. Besides inserting timestamping information, sequencing and ingress identification is required.

Another example is [49], in which it is used to build a service with statistical guarantees on the loss rate. This scheme uses the concept of “achievable” capacity of the path, i.e., an equivalent capacity of the path, and assumes that the aggregated traffic rate follows a Gaussian distribution. With this model it is easy to relate the aggregated traffic rate at the ingress, the “achievable” capacity and the loss rate in the path. Then the actual loss rate in the path is measured at the egress by counting the lost packets in a defined time interval (packets carry a sequence number). This information is periodically reported to the corresponding ingress node. The aggregated rate of accepted traffic is measured periodically at the ingress

(characterized with its mean and variance), and, together with the actual loss rate, is used to obtain the actual "achievable" capacity. The mean and variance of the new flow's rate, the mean and variance of the actual aggregated rate, and the actual "achievable" capacity are used to estimate the future loss rate if the flow is accepted, which is compared to the target loss rate to make the AC decision.

The main advantage of these schemes compared to *hop-by-hop AC schemes* is the edge-to-edge architecture. They do not require changes in the network core, where reservation state is not maintained (neither per-flow nor per-aggregate), and only edge nodes are AC aware. In comparison with *active measurement-based edge-to-edge AC schemes with per-flow probing*, these schemes avoid any intrusive probing traffic, the AC is fast as it is made at once since measurements are available online, and the confidence level of measurements is increased since they use many samples instead of just a few. However, packets are required to carry information such as sequencing, timing, or ingress identification. Moreover, the AC algorithms do not immediately consider the effect of a recently accepted flow until future measurements take it into account, but this takes some time. Therefore, a high rate of new arriving flows to an ingress-egress pair may cause false acceptances. The same problem holds for concurrently accepted flows in other ingress nodes.

## 4. Conclusions

In this paper we have studied the main AC schemes that have been proposed in the Internet, focusing on the simplicity of their architectures in terms of the number of nodes that participate in the AC, the required state, the use of signaling, and others.

The majority of AC schemes we have studied deal with real-time traffic and only a few with elastic traffic (and very few for both traffic types). We have classified them mainly according to the nodes that participate in the AC decisions, i.e., *centralized AC schemes*, *hop-by-hop AC schemes*, *one-hop AC schemes on LPs with reservation* and *edge-to-edge AC schemes*:

- In *centralized AC schemes* there is a single AC entity in the network (where the AC algorithm is implemented) that maintains the state of resource usage and exchanges signaling packets with the ingress nodes when new flows arrive. Flow's reservations can be based on state or on occupancy. The network remains simple since the state is not distributed. Service requests are processed serially and unlike distributed schemes, they do not have the problem of simultaneous AC decisions. However, in large and highly dynamic networks the centralized entity would have to process too many signaling packets and might become a traffic bottleneck. Moreover, a centralized approach is highly vulnerable to failures.

- In *hop-by-hop AC schemes*, each node maintains the actual aggregated reservation and performs a local AC decision through a link-scope AC algorithm based on measurements or parameters. Flows' reservations can be based on state or on occupancy. Usually the duration of the AC phase (and also the time an accepted flow

has to wait for starting to transmit) is about RTT. As they are distributed, these schemes are more robust and able to cope with highly dynamic networks. However, all nodes are aware of AC. Moreover, some of these schemes maintain the per-flow state in nodes and many often use per-flow signaling packets to communicate between nodes (although some schemes use a lightweight signaling or even no signaling).

- In *one-hop AC schemes on LPs with reservation* only the ingress node is involved and maintains its flows' reservations based on state or on occupancy (although the previous reservation in the establishment of the LP can involve either all the nodes of the path or a centralized entity, and the reservation is based on state). The AC decision is simple, only at the ingress node and without per-flow signaling. It is also fast, as it is made at once. However, per-path resource reservations require more complex management to avoid that the LPs that pass through the same link share resources in a non-appropriate way and consequently false rejections occur.

- In *edge-to-edge AC schemes* only the ingress and egress nodes participate in the AC decision. There is an edge-to-edge scope AC algorithm implemented at the ingress or egress node, which is based on measurements performed at the egress and sent back to the ingress through special packets (usually signaling packets). Flows' reservations are based on occupancy. These schemes do not require changes in the core, since the nodes of the path do not maintain either a per-flow or aggregate reservation state (they are not aware of AC) and do not exchange signaling packets. However, some of them generate intrusive (signaling) traffic that can be considerable and the duration of the AC phase (and also the time an accepted flow waits before transmitting) may be quite long. In others the AC algorithms do not consider the effect of a recently accepted flow until future measurements take it into account. Since this takes some time, a high rate of new arriving flows to an ingress-egress pair can cause false acceptances. Other schemes require that data packets carry information such as sequencing, timing information, or ingress identification.

Using implicit ways of communication between nodes and traffic measurements in the AC algorithms can considerably reduce the number of nodes that are aware of AC, the state they maintain and the signaling required. The implicit way (e.g., through the port numbers in data packets, marks in other packet fields, predefined values of parameters, etc.) has the advantage of not requiring signaling packets. Using active or passive measurements results in flow's reservations based on occupancy, which do not require a per-flow state, are maintained as long as the flow transmits (following a certain traffic profile) and are released when the flow is inactive, without needing signaling packets. Moreover, measurements may increase resource utilization, although they run the risk that traffic fluctuations may cause eventual QoS degradations; therefore, they provide qualitative guarantees. Also note that although per-flow state is not required, in some other nodes (e.g., in the ingress node of the path, in a centralized entity, etc.) a list of the accepted flows should be maintained to differentiate between the packets of new flows from the ones of accepted flows, and also to be able to isolate (or protect) the accepted flows by enforcing the agreed traffic profile on their input traffic.

## References

[1] Braden R., Clark D., Shenker S., "Integrated services in the Internet architecture: an overview", RFC 1633, 1994.
[2] Shenker S., "Fundamental design issues for the future Internet", IEEE Journal on Selected Areas in Communications, vol.13, no. 7, 2006.
http://dx.doi.org/10.1109/49.414637
[3] Breslau L., Shenker S., "Best-effort versus reservations: a simple comparative analysis", ACM SIGCOMM Computer Communication Review, vol. 28, no. 4, 1998.
http://dx.doi.org/10.1145/285243.285248
[4] Bonald T., Oueslati-Boulahia S., Roberts J., "IP traffic and QoS control: the need for a flow-aware architecture", Proceedings of the World Telecommunications Congress, 2002.
[5] Lima S.R., Carvalho P., Freitas V., "Admission control in multiservice IP networks: architectural issues and trends", IEEE Communications Magazine, vol. 45, no. 4, 2007.
http://dx.doi.org/10.1109/MCOM.2007.343620
[6] Wright S., "Admission control in multi-service IP networks: a tutorial", IEEE Communications Surveys & Tutorials, vol.9, no. 2, 2007.
http://dx.doi.org/10.1109/COMST.2007.382408
[7] Dovrolis C., Ramanathan P., "A case for relative differentiated services and the proportional differentiation model", IEEE Network, vol. 3, no. 5, 1999.
http://dx.doi.org/10.1109/65.793688
[8] Kurose J.F., "Open issues and challenges in proving quality of service guarantees in high-speed networks", ACM SIGCOMM Computer Communication Review, vol. 23, no. 1, 1993. http://dx.doi.org/10.1145/173942.173943.
[9] Jamin S., Danzig P.B., Shenker S.J., Zhang L., "A measurement-based admission control algorithm for integrated services packet networks", IEEE/ACM Transactions on Networking, vol. 5, no. 1, 1997. http://dx.doi.org/10.1109/90.554722
[10] Mitzel D. J., Estrin D., Shenker S., Zhang L., "A study of reservation dynamics in integrated services packet networks", Proceedings of the IEEE Infocom, 1996.
http://dx.doi.org/10.1109/INFCOM.1996.493386
[11] Breslau L., Knightly E., Shenker S., Stoica I., Zhang H., "Endpoint admission control: architectural issues and performance", ACM SIGCOMM Computer Communication Review, vol. 30, no. 4, 2000. http://dx.doi.org/10.1145/347057.347400
[12] Nichols K., Jacobson V., Zhang L., "A two-bit Differentiated Services architecture for the Internet", RFC 2638, 1999.
[13] Teitelbaum B., Hares S., Dunn L., Neilson R., Narayan V., Reichmeyer F., "Internet 2 Qbone: building a testbed for Differentiated Services", IEEE Network, vol. 13, no. 5, 1999. http://dx.doi.org/10.1109/65.793686
[14] Blake S., Black D., Carlson M., Davies E., Wang Z., Weiss W., "An architecture for Differentiated Services", RFC 2475, 1998.
[15] Braden R., Zhang L., Berson S., Herzog S., Jamin S., "Resource ReSerVation protocol (RSVP) -- Version 1 functional specification", RFC 2205, 1997.
[16] Breslau L., Jamin S., Shenker S., "Measurement-based admission control: what is the research agenda?", Proceedings of IEEE International Workshop on Quality of Service, 1999.

[17] Roberts J.W., Mocci U., Virtamo J. (Eds.), "Broadband network teletraffic. Performance evaluation and design of broadband multiservice networks. Final report of Action COST 242", Lecture Notes in Computer Science (LNCS), vol. 1155, Springer-Verlag, ISBN 3-540-61815-5, 1996.
[18] Parekh A.K., Gallager R.G., "A generalized processor sharing approach to flow control in integrated services networks: the single-node case", IEEE/ACM Transactions on Networking, vol. 1, no. 3, 1993. http://dx.doi.org/10.1109/90.234856
[19] Shenker S., Partridge C., Guerin R., "Specification of Guaranteed Quality of Service", RFC 2212, 1997.
[20] Knightly E.W., Shroff N.B., "Admission control for statistical QoS: theory and practice", IEEE Network, vol. 13, no. 2, 1999. http://dx.doi.org/10.1109/65.768485
[21] Perros H.G., Elsayed K.M., "Call admission control schemes: a review", IEEE Communications Magazine, vol. 34, no. 11, 1996. http://dx.doi.org/10.1109/35.544197
[22] Jamin S., Shenker S.J., Danzig P.B., "Comparison of measurement-based admission control algorithms for controlled-load service", Proceedings of the IEEE Infocom, 1997. http://dx.doi.org/10.1109/INFCOM.1997.631035
[23] Wroclawski J., "Specification of the Controlled-Load Network Element Service", RFC 2211, 1997.
[24] Qiu J., Knightly E.W., "Measurement-based admission control using aggregate traffic envelopes", IEEE/ACM Transactions on Networking, vol. 9, no. 2, 2001. http://dx.doi.org/10.1109/90.917076
[25] Grossglauser M., Tse D.N.C., "A time-scale decomposition approach to measurement-based admission control", IEEE/ACM Transactions on Networking, vol. 11, no 4, 2003. http://dx.doi.org/10.1109/TNET.2003.815289
[26] Jamalipour A., Kim J., "Measurement-based admission control scheme with priority and service classes for application in wireless IP networks", Wiley International Journal of Communication Systems, vol. 16, no. 6, 2003. http://dx.doi.org/10.1002/dac.602
[27] Breslau L., Jamin S., Shenker S., "Comments on the performance of measurement-based admission control algorithms", Proceedings of the IEEE Infocom, 2000. http://dx.doi.org/10.1109/INFCOM.2000.832506
[28] Shiomoto K., Yamanaka N., Takahashi T., "Overview of measurement-based connection admission control methods in ATM networks", IEEE Communications Surveys & Tutorials, vol. 2, no. 1, 1999. http://dx.doi.org/10.1109/COMST.1999.5340510
[29] Almesberger W., Ferrari T., Boudec J.L., "SRP: a scalable resource reservation protocol for the Internet", Elsevier Computer Communications, vol. 21, no. 14, 1998. http://dx.doi.org/10.1016/S0140-3664(98)00183-2
[30] Turányi Z.R., Westberg L., "Load Control: congestion notifications for real-time traffic", Proceedings of the IFIP Working Conference on Performance Modeling and Evaluation of ATM & IP Networks, 2001.
[31] Marquetant Á., Pop O., Szabó R., Dinnyés G., Turányi Z., "Novel enhancements to load control - a soft-state, lightweight admission control protocol", Proceedings of the 2nd COST263 International Workshop on Quality of Future Internet Services, 2001.
[32] Stoica I., Zhang H., "Providing guaranteed services without per flow management",

ACM SIGCOMM Computer Communication Review, vol. 29, no. 4, 1999. http://dx.doi.org/10.1145/316194.316208

[33] Sivakumar R., Kim T., Venkitaraman N., Bharghavan V., "Achieving per-flow weighted rate fairness in a core stateless network", Proceedings of the IEEE Conference on Distributed Computing Systems, 2000. http://dx.doi.org/10.1109/ICDCS.2000.840929

[34] Sivakumar R., Venkitaraman N., Kim T., Lu S., Nandagopal T., Bharghavan V., "The Corelite QoS architecture: providing a flexible service model with a stateless core", Research report, Illinois Mobile Environments Laboratory (TIMELY) research group at the University of Illinois at Urbana Champaign, 1999.

[35] Kumar A., Hegde M., Anand S.V.R., Bindu B.N., Thirumurthy D., Kherani A.A., "Nonintrusive TCP connection admission control for bandwidth management of an Internet access link", IEEE Communications Magazine, vol. 38, no. 5, 2000. http://dx.doi.org/10.1109/35.841841

[36] Mortier R., Pratt I., Clark C., Crosby S., "Implicit admission control", IEEE Journal on Selected Areas in Communications, vol. 18, no. 12, 2000. http://dx.doi.org/10.1109/49.898743

[37] Roberts J.W., "Internet traffic, QoS, and pricing", Proceedings of the IEEE, vol. 92, no. 9, 2004. http://dx.doi.org/10.1109/JPROC.2004.832959

[38] Kortebi A., Oueslati S., Roberts J.W., "Cross-protect: implicit service differentiation and admission control", Proceedings of the IEEE Workshop on High Performance Switching and Routing, 2004.

[39] Rosen E., Viswanathan A., Callon R., "Multiprotocol Label Switching architecture", RFC 3031, 2001.

[40] Vilà P., "Dynamic management and restoration of virtual paths in broadband networks based on distributed software agents", PhD thesis, University of Girona, 2004.

[41] Ramakrishnan K., Floyd S., Black D., "The addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, 2001.

[42] Borgonovo F., Capone A., Fratta L., Petrioli C., "VBR bandwidth guaranteed services over DiffServ networks", Proceedings of the IEEE Real-Time Technology and Applications Symposium, 1999.

[43] Borgonovo F., Capone A., Fratta L., Marchese M., Petrioli C., "PCP: a bandwidth and delay guaranteed transport service for IP networks", Proceedings of the IEEE International Conference on Communications, 1999.

[44] Bianchi G., Capone A., Petrioli C., "Throughput analysis of end-to-end measurement-based admission control in IP", Proceedings of the IEEE Infocom, 2000. http://dx.doi.org/10.1109/INFCOM.2000.832544

[45] Lima S., Carvalho P., Freitas V., "Measuring QoS in class-based IP networks using multi-purpose colored probing patterns", Proceedings of SPIE ITCom – Performance, Quality of Service, and Control of Next-Generation Communication Networks, 2004.

[46] Lima S., Carvalho P., Santos A., Freitas V., "A distributed admission control model for CoS networks using QoS and SLS monitoring", Proceedings of the IEEE International Conference on Communications, 2003.

[47] Lima S., Carvalho P., Freitas V., "Distributed admission control in multiservice IP

networks: concurrency issues", Academy Publisher Journal of Communications (JCM), vol. 1, no. 3, 2006. http://dx.doi.org/10.4304/jcm.1.3.1-9

[48] Cetinkaya C., Kanodia V., Knightly E.W., "Scalable services via egress admission control", IEEE Transactions on Multimedia, vol. 3, no. 1, 2001. http://dx.doi.org/10.1109/6046.909595

[49] Zhu H., Li V.O.K., Ma Z., Zhao M., "Statistical connection admission control framework based on achievable capacity estimation", Proceedings of the IEEE International Conference on Communications, 2006. http://dx.doi.org/10.1109/ICC.2006.254797

## Copyright Disclaimer