Implementing QoS in IP networks

Adam Przybyłek

http://przybylek.wzr.pl

University of Gdańsk, Department of Business Informatics

Piaskowa 9, 81-824 Sopot, Poland

Abstract

With the increasing number of real-time Internet applications, "the best-effort service" has become insufficient to satisfy the needs of end users. The provision of predictable service levels for different types of application in the IP networks has been a subject of active research over the past decade. As a result, IETF has defined two quality of service (QoS) architectures: Integrated Services (IntServ) and Differentiated Services (DiffServ). The aim of this paper is to analyse the QoS aspects in the context of these architectures as well as to discuss the possibilities of integration between them.

1. Introduction

IP networks were originally designed to provide only best-effort service. This minimalist service allows the Internet to be stateless, and routers do not need to maintain any fine-grained information about traffic. As a result of this stateless architecture the Internet is both highly scalable and robust [19].

The best-effort service proved successful during the early Internet years, when data applications such as FTP or HTTP constituted the bulk of Internet traffic. Generally, these applications used TCP and therefore adapted gracefully to variations in bandwidth, latency, and loss. The amount of interactive traffic was minimal, and other applications requiring stricter guarantees were at an experimental stage. However, the increasing popularity and capacity of the Internet has made it an attractive infrastructure for delivering real-time content.

The Internet has evolved and is continuing to evolve into a global communications infrastructure, supporting significant economic, educational and social activities. In this context, the best-effort service is no longer sufficient and a demand for quality of service (QoS) has emerged. QoS is a set of service requirements to be met by the network when transporting any network traffic. Service requirements are expressed in measurable parameters such as bandwidth, delay, jitter, and packet loss. The QoS recommendations for common applications are shown in Table 1.

Table 1. QoS recommendations [5], [20]

	bandwidth [kbps]	delay [ms]	jitter [ms]	packet loss ratio [%]
FTP	adequate to user needs	< 15 000		0
Web-browsing	64	< 2 000		0
TELNET	8	< 200		0
Video on Demand	4096[1]	< 10 000		< 1
VoIP	64	<150	< 30	< 3
Videophone	384	<150	< 30	< 1

The efforts to provide QoS in IP networks gave birth to two architectures: Integrated Services (IntServ), which requires a new stateful architecture, and Differentiated Services (DiffServ), which maintains the stateless property of the original Internet. While stateful architecture can provide more powerful and flexible services, it is less scalable than stateless architecture. On the other hand, while stateless architecture is much more scalable, it offers weaker services [19].

In the following sections, the paper first covers the principles and some of the details of IntServ and DiffServ, then compares these architectures in terms of the QoS they provide and their complexity and finally discusses how they could be integrated to provide end-to-end QoS in the Internet.

2. Integrated Services

2.1. Overview

IntServ is an architecture requiring per-flow traffic handling, which means that reservations are made separately for each traffic flow and at each router along the flow path [6], [8]. In a per-flow model an application running on the end-host uses a signalling protocol to make a request for special treatment of its datagrams through the network. The request is distributed to every hop along the path to the destination [4], [6], [21]. At each hop the request is accepted or rejected on the basis of resource availability. Once all the hops have accepted the request, the application gets the level of service it has requested.

2.2. Classes of service

IntServ provides three classes of service that an application can request:

- **Guaranteed service** guarantees that a datagram will be delivered strictly according to the specified service level agreement (SLA) [10]. An SLA is a formal service contract between customers and service providers, consisting of QoS parameters. It is intended for real-time applications which have firm bandwidth and delay requirements.
- **Controlled-load service** provides soft QoS, which means that there is no quantitative measure to express the minimum acceptable level of service. Soft QoS approximates to

[1] depends on the encoding format

the service that the same flow would obtain from an unloaded best-effort network. It allows applications to have low delay and high throughput even during times of congestion [9]. This service is intended for adaptive real-time applications, such as playback of a recorded conference [2], [20].

- **Best-effort service** provides no guarantees of any type.

2.3. Resource ReSerVation Protocol (RSVP)

RSVP is a resource reservation setup protocol designed for the IntServ architecture. It provides correct protocol operation even when two RSVP-capable routers are joined by an arbitrary "cloud" of non-RSVP routers. RSVP messages are sent hop by hop between RSVP-capable routers as raw IP datagrams with protocol number 46 [7], [8]. The two primary messages are PATH (path establishment) and RESV (reservation). These are used to request specific QoS from the network for a particular session. An RSVP session defines a simplex unicast or multicast data flow specified by the triple {destination IP address, IP protocol ID, destination port}. Reserving resources for bi-directional traffic flow requires two independent RSVP sessions, one in each direction [1].

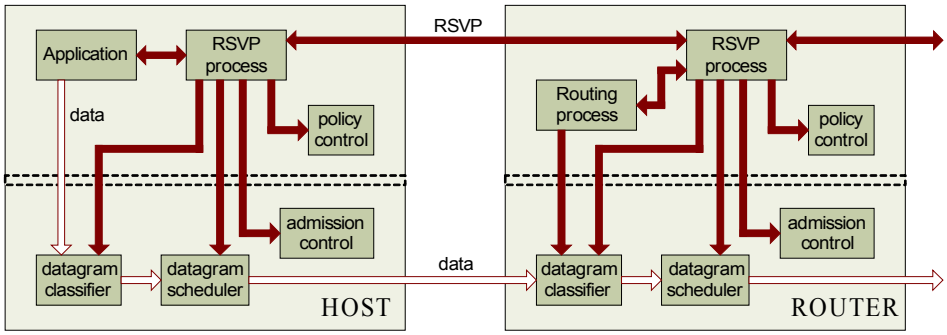
An elementary RSVP reservation request consists of a "FlowSpec" together with a "FilterSpec"; this pairing is called a "flow descriptor". The FlowSpec in a reservation request will generally include a service class and two sets of numerical parameters: (1) a "ReserveSpec," which defines the desired QoS, and (2) a "TrafficSpec," which describes the data flow [2], [7], [8]. In the most general approach FilterSpec may select arbitrary subsets of the datagrams in a given session. Such subsets are defined in terms of sender IP address and source port.

RSVP uses soft-state signalling. Routers along the traffic's path must periodically be refreshed with information about active RSVP sessions; otherwise, the router removes all state associated with the session [1], [8]. State may also be deleted by an explicit "teardown" message.

2.4. The RSVP model

The entire set of machinery in the node that supplies requested QoS to data streams is called a traffic controller. A traffic controller includes a policy controller, datagram classifier, datagram scheduler, and admission controller, as shown on Figure 1. RSVP on a router has interfaces to routing and to traffic control. RSVP on a host has an interface to applications (i.e. an API) and also an interface to traffic control, if this exists on the host.

Figure 1. The RSVP model [7], [21]



The **datagram classifier** determines a service class for each datagram in accordance with the reservation state setup by RSVP. The FilterSpec is used to set parameters in the datagram classifier. Data datagrams that are addressed to a particular session but do not match any of the FilterSpec for that session are handled as best-effort traffic.

The **datagram scheduler** applies the particular mechanisms to the datagrams in order to provide the requested QoS. The FlowSpec is used to set parameters in the datagram scheduler.

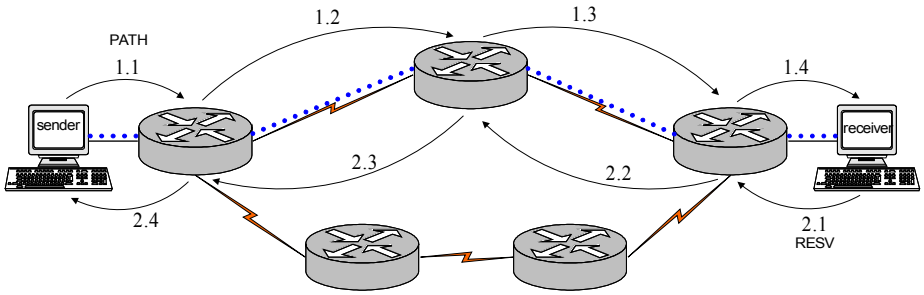
The **admission controller** determines whether enough resources exist in the network node to support the requested QoS.

The **policy controller** determines whether a new request for QoS has administrative permission to make the reservation. (Certain types of user, for example, may be excluded at certain times of day or from certain classes of request at all times). The input to policy control is carried by the PATH message in a POLICY_DATA object.

2.5. RSVP protocol operation

A basic RSVP protocol operation is shown in Figure 2. The endpoint sends the PATH message downstream along the route provided by the routing protocol. The PATH can also originate from a router (such as one on behalf of an endpoint which is not capable of RSVP signalling).

Figure 2. Basic RSVP Protocol Operation



The PATH message collects information about the QoS capabilities and stores the "path state" in each node along the way. The state includes at least the IP address of the previous hop node.

The receiver then processes the PATH information and generates the RESV message. This message is sent upstream to make the actual reservation in each router along the path [2], [4]. When the sender gets this RESV, the session is established. The sender does not have to wait for the RESV, but in that case it receives best-effort service.

A PATH message contains the following information [7], [8]:

- RSVP_HOP – carries the IP address of the previous RSVP-capable router for downstream messages.
- SENDER_TEMPLATE – defines the same structure as the FilterSpec; contains the sender IP address and the source port; can be used to select this sender's datagrams from other datagrams in the same session.
- SENDER_TrafficSpec – defines the traffic characteristics of the data flow that the sender will generate; is not modified by any intermediate nodes; is an estimation of the data stream the sender will generate.
- POLICY_DATA – includes credentials identifying users and their quotas; is used by the policy controller.
- ADSPEC – carries advertising information that is modified at each hop to reflect network characteristics (such as available link and router capacities) between sender and receiver.

The receiver uses the SENDER_TrafficSpec and the ADSPEC to compute the FlowSpec, which is sent upstream in the RESV message. A FlowSpec together with a FilterSpec represent a flow descriptor [2]. A session may have multiple flow descriptors. In this way different QoS levels may be provided to session traffic originating from different sources. Datagrams matching the session description receive best-effort service if they do not match any FilterSpec [1].

RESV messages carry reservation requests hop by hop from receivers to senders along the reverse paths of data flows for the session. The IP destination address of a RESV message is the IP address of a previous-hop node, obtained from the path state. The IP source address is the address of the node that sent the message. At each intermediate node a reservation request triggers two general actions as follows [1]:

1. Make a reservation on a link – the request is passed to admission control and policy control; if both tests are successful, the node sets the datagram classifier to select the data datagrams defined by the FilterSpec and configures scheduler parameters.
2. Forward the request upstream – the request is propagated upstream towards the appropriate sender.

A RESV message contains the following information [7], [8]:

- RSVP_HOP – contains the IP address of the interface through which the RESV message was sent.
- FlowSpec – defines the QoS to be provided for a flow; is used to configure the datagram scheduler.
- FilterSpec – contains the sender IP address and the source port; together with the session description defines the set of data datagrams to receive the QoS specified by the FlowSpec; is used to set parameters in the datagram classifier.

3. Differentiated Services

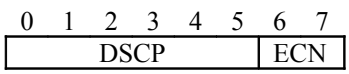
3.1. Overview

DiffServ removes the per-flow scalability issues by aggregating traffic flows into different classes, also called behaviour aggregates, and allocating network resources on a per-class basis. Each behaviour aggregate (BA) is identified by a Differentiated Services Code Point (DSCP) [12], [21].

Routers select a particular datagram handling mechanism by matching against the DSCP value. The externally observable forwarding behaviour of a DiffServ-capable router is called per-hop behaviour (PHB) [17]. When the effects of the individual PHBs are concatenated, this results in an end-to-end service. PHBs are implemented by employing a range of queue services and queue management [11].

DiffServ is designed to be a scalable architecture where the sophisticated classification and conditioning operations need only be implemented at network boundaries [2], [12]. A datagram classifier selects datagrams in a traffic stream which should receive a differentiated service and then steers them to an element of a traffic conditioner for further processing [12], [16]. A datagram entering a DiffServ domain is classified on the basis of a variety of Layer 3 through Layer 7 characteristics, and is then assigned to a particular BA. The selected BA is marked directly on the datagram by setting the DS field. The DSCP field consists of the first 6 bits in the IP type of service (ToS) field (Figure 3).

Figure 3. The DS field structure [11]

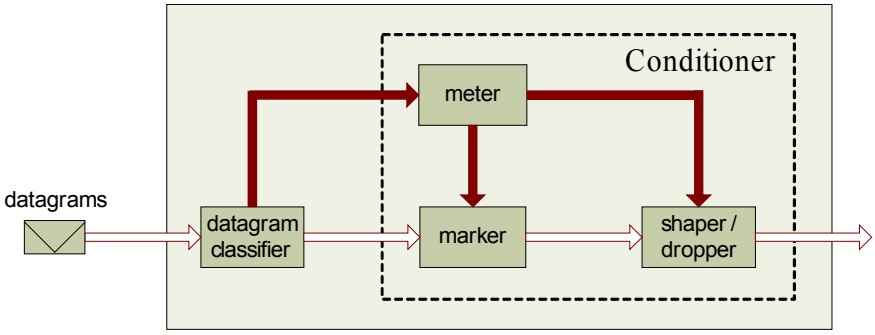


After the datagram has been marked at network boundaries, it is possible to treat it on the basis of the marking at each hop throughout the DiffServ domain [4]. Hence within the DiffServ domain classification is simplified and may be based only on the content of the DS field.

3.2. DiffServ-compliant node

The traffic conditioner forces the traffic entering the DiffServ domain to conform to the rules specified in the traffic conditioning agreement (TCA). This specifies the classifier rules and any corresponding (1) traffic profiles, and (2) metering, marking, discarding and shaping rules which are to apply to the traffic streams selected by the classifier [12]. A traffic conditioner may contain the following elements: meter, marker, shaper and dropper. Figure 4 shows a block diagram of a DiffServ-compliant node.

Figure 4. Logical View of a DiffServ-compliant node [12]



A **traffic meter** measures the temporal properties (e.g. the rate) of the stream of datagrams selected by a classifier against a traffic profile specified in a TCA. A traffic profile provides rules for determining whether a particular datagram is in-profile or out-of-profile. Out-of-profile datagrams may be queued until they are in-profile (shaped), discarded (policed), marked with a new codepoint, or forwarded unchanged, while triggering some accounting procedure [12].

A **datagram marker** sets the DS field of datagrams based on their classification, so that they may be distinguished more easily at the next routers.

A **shaper** delays some datagrams in a traffic stream in order to bring the stream into compliance with a traffic profile. A shaper usually has a finite-size buffer, and datagrams may be discarded if there is not sufficient buffer space to hold the delayed datagrams [12]. A shaper can also use the explicit congestion notification (ECN) bits to avoid congestion [15], [18].

A **dropper** discards some datagrams in a traffic stream in order to bring the stream into compliance with a traffic profile. This process is known as "policing" the stream [12].

Marking, shaping and policing are performed in accordance with the state of a corresponding meter.

3.3. Classes of service

DiffServ describes four main classes of service that an application can receive:

- **Best-Effort** – codepoint '000000'; the default forwarding behaviour (the network will deliver as many of these packets as possible and as soon as possible) [11]; is intended for sending "normal internet traffic" across a DiffServ domain [17].
- **Class selector** – codepoints xxx000; is reserved for backward compatibility with IP precedence [11]; datagrams are handled as if they are marked with IP precedence.
- **Assured Forwarding** – contains four independent classes of service, in each of which there are three drop precedences; in the event of congestion, the drop precedence of a packet determines the relative importance of the packet within the AF class [13]; each AF class is totally independent of the other three, and no assumptions can be made regarding the treatment of datagrams belonging to one class when compared with the treatment of datagrams belonging to another; Table 2 lists the recommended AF codepoint values.
- **Expedited Forwarding** – codepoint 101110; can be used to build a low-delay, low-jitter and low-loss service [17]; is intended for real-time traffic.

Table 2. Codepoints recommended by RFC 2597 [13]

		Class AF1	Class AF2	Class AF3	Class AF4
Drop precedence	Low	001|01|0	010|01|0	011|01|0	100|01|0
	Medium	001|10|0	010|10|0	011|10|0	100|10|0
	High	001|11|0	010|11|0	011|11|0	100|11|0

4. Comparison between IntServ and DiffServ

IntServ requires each router to maintain and manage per-flow state information. With millions of simultaneous flows traversing backbone routers, this stateful solution proved to be unscalable. To address this problem, the DiffServ architecture was proposed. DiffServ achieves scalability by pushing the complexity towards the boundary nodes. The boundary nodes perform complex datagram classification and traffic conditioning functions. The interior nodes simply apply per-hop behaviours based on the DSCP [12]. This results in a lighter processing load on the interior nodes. Table 3 summarises the main differences between IntServ and DiffServ. It should be noticed that these architectures provide complementary approaches to the problem of providing end-to-end QoS in the Internet.

Table 3. A Comparison of the IntServ and DiffServ architectures

	IntServ	DiffServ
granularity of service	individual flow	aggregate of flows
admission control	required	not required
signalling protocol	required	not required
co-ordination for service	end-to-end	per-hop
scalability	limited by the number of flows	limited by the number of service classes
network accounting	based on FlowSpec and QoS requirement	based on class usage
recommended deployment	at the edges of the network or in intranets	in the backbone network

IntServ and DiffServ have their own benefits and drawbacks as listed in Table 4. With no clear advantage to either architecture, they must be able to coexist and effectively inter-operate.

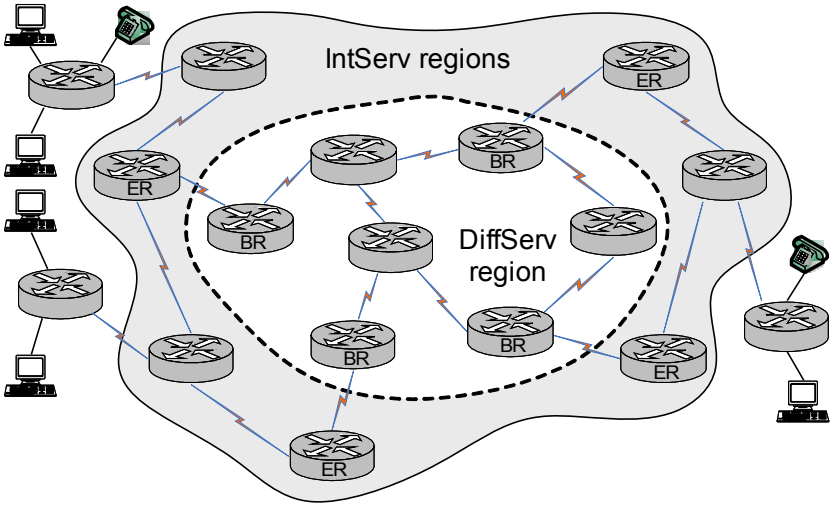
Table 4. The main benefits and drawbacks of IntServ and DiffServ

	drawbacks	benefits
IntServ	all routers must maintain state information on a per-flow basis	provides strict QoS guarantees to individual flows
	periodic RSVP refresh messages require an additional amount of bandwidth	allows a network to reject new sessions if all reservable bandwidth is booked
	not scalable to large networks	
DiffServ	lack of a per-session strict QoS guarantees	scalability – no state or flow information needs to be maintained
		performance – complex operations are carried out only at network boundary

5. IntServ-DiffServ integration

This section gives an overview of the way in which IntServ and DiffServ can be used in combination to achieve end-to-end QoS, taking advantage of the strengths of each approach. The architecture for supporting the IntServ-DiffServ co-operation includes IntServ regions at the periphery of the network and a DiffServ region at the core of the network. The interface network elements between these regions are called the edge router (ER) and the border router (BR). Figure 5 shows this architecture. From the perspective of IntServ, the DiffServ region is treated as virtual links connecting ERs [1], [14]. ERs act as admission control agents to the DiffServ region. They process signalling messages from hosts and apply admission control based on local resource availability and on customer-defined policy.

Figure 5. The IntServ-DiffServ architecture



An application uses RSVP to request admission to the network. If the request is accepted by the IntServ regions, the ERs must map this request into a corresponding BA. ERs can then approve or reject resource requests on the basis of the capacity available in the DiffServ region at the mapped BA. The availability of resources is determined by the capacity provisioned in the SLA. An ER may also apply a policy decision such that the resource request may be rejected on the basis of the customer's specific policy criteria, even though it has been determined that the aggregate resources are available per the SLA [14].

6. Summary

This paper has reviewed the fundamental concepts of QoS, such as IntServ and DiffServ. IntServ enables application to request per-flow quantifiable resources along an end-to-end data path and to obtain feedback regarding the admissibility of this request. DiffServ, on the other hand, enables scalability in backbone networks. However, neither IntServ nor DiffServ is adapted to support end-to-end QoS across large networks. The paper presents a way of integrating the two architectures using IntServ at the edge of the network and DiffServ within the core network. This approach combines the strength of both architectures, while at the same time avoiding their drawbacks, and can facilitate deployment of real-time applications.

References

1. Armitage G. (2002), Quality of Service in IP Networks, New Riders, USA
2. Chimento P.F. (1998), Tutorial on QoS support for IP, CTIT Technical Report 23
3. Dovrolis, C., Ramanathan, P. (1999), Case for Relative Differentiated Services and the Proportional Differentiation Model. In IEEE Network, vol 13(5), pp. 26 - 34

4. Flannagan M. (2003), Cisco Catalyst QoS: Quality of Service in Campus Networks, Cisco Press, Indianapolis
5. ITU-T Recommendation G.1010 (2001), End-user multimedia QoS categories
6. RFC 1633 (1994), Integrated Services in the Internet Architecture: an Overview
7. RFC 2205 (1997), Resource ReSerVation Protocol (RSVP)
8. RFC 2210 (1997), The Use of RSVP with IETF Integrated Services
9. RFC 2211 (1997), Specification of the Controlled-Load Network Element Service
10. RFC 2212 (1997), Specification of Guaranteed Quality of Service
11. RFC 2474 (1998), Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers
12. RFC 2475 (1998), An Architecture for Differentiated Service
13. RFC 2597 (1999), Assured Forwarding PHB Group
14. RFC 2998 (2000), A Framework for Integrated Services Operation over Diffserv Networks
15. RFC 3168 (2001), The Addition of Explicit Congestion Notification (ECN) to IP
16. RFC 3260 (2002), New Terminology and Clarifications for Diffserv
17. RFC 3246 (2002), An Expedited Forwarding PHB (Per-Hop Behavior)
18. RFC 3540 (2003), Robust Explicit Congestion Notification (ECN) Signaling with Nonces
19. Stoica I. (2000), Stateless Core: A Scalable Approach for Quality of Service in the Internet, PhD. Dissertation, Carnegie Mellon University
20. Szigeti T. (2004), End-to-End QoS Network Design, Cisco Press, Indianapolis
21. Vegesna S. (2001), IP Quality of Service, Cisco Press, Indianapolis