Boomerang – A Simple Protocol for Resource Reservation in IP Networks

Gábor Fehér, Krisztián Németh,
Markosz Maliosz

*High Speed Network Laboratory,
Department of Telecommunication and
Telematics, Technical University of Budapest
Pázmány Péter sétány 1/D, H-1117 Budapest,
Hungary,Phone: +36 1 4633119
E-mail: {feher, nemeth_k, maliosz}@
ttt-atm.ttt.bme.hu*

István Cselényi, Joakim Bergkvist,
David Ahlard, Tomas Engborg

*Department of Network Services,
Telia Research AB
Vitsandsgatan 9, S-12386 Farsta,
Sweden,Phone: +46 8 713 8173
E-mail: {cse, jobe, davahl, toeng}@trab.se*

Abstract

This paper describes a resource reservation concept for providing quantitative IP services with guaranteed QoS. Although the proposed reservation approach is based on a new, lightweight signaling protocol, it suits to other IP QoS techniques, such as Diff-Serv and Int-Serv. Distinguishing features of the Boomerang protocol are the following. Only the Initiating Node generates signaling messages, therefore complexity and intelligence is concentrated in one point enabling simple implementation. Flow state aggregation can be suggested to subsequent nodes by appending it to the Boomerang message, thus good scalability can be achieved. Bi-directional reservation can be made independently of the path, which yields a fast and effective reservation process. The only requirement on the Far-End Node is that it should be able to bounce the Boomerang message back, ensuring quick penetration into the current Internet.

This paper introduces the reservation concept and highlights the simplicity by presenting the results of a performance study, which compares the Boomerang protocol to Resource ReSerVation Protocol (RSVP).

1. Terminology

The terminology given in [1] is used in this document extended with the following notions:

Initiating Node: This is the node that initiates the resource reservation. It can be the sender or the receiver host or any Boomerang-aware network node.

Far-End Node: This is the other end-point of the data flow, for which reservations are being made.

Micro-Flow: A data flow from one end-point to another, defined uniquely by multi-field classification.

Boomerang Node: A Boomerang-aware node that performs admission control and reservation for single or aggregated micro-flows.

Aggregating Node: A Boomerang node that associates several micro-flows with a common, aggregated QoS specification and appends an aggregation field to the Boomerang messages of associated micro-flows, in order to suggest an aggregated reservation state for subsequent Boomerang nodes.

2. Introduction

Besides Differentiated Services [2] network operators may also be interested in offering guaranteed services to ambitious customers, and quantitative QoS applications [1] in an end-to-end manner. However, best effort network domains or bottleneck segments within a Diff-Serv domain may compromise the end-to-end QoS during for example the peak hours. Therefore a simple signaling protocol is needed to signal per micro-flow QoS requirements to the network and to reserve resources end-to-end in a guaranteed manner. Currently RSVP is the only accepted signaling protocol for resource reservation setup in IP networks [3], although it has several limitations:

1. It relies on per micro-flow state that results in a scalability problem in terms of memory, capacity and processing time.

2. It is complex to implement both in nodes and hosts due to separation of reservation and path finding messages, receiver diversity.
3. It requires modification in the far end host.
4. The intelligence of signaling processing is spread over the network. Each node along a reserved path contains a flow state and a signaling state. Therefore, each reservation session increases the load on network nodes.
5. It requires multiple interactions between sender and receiver for a successful reservation setup.

We propose a new reservation protocol for IP networks, called Boomerang, which meets the following challenges:

1. Simple processing of signaling messages at network nodes resulting in simple implementation.
2. No special requirements on the far end.
3. Concentrating the intelligence into the Initiating Node. The Initiating Node is responsible for signaling and handling flow state for the setup reservation. Network nodes along reserved path keep only flow states resulting in low load and processing time on network nodes.
4. Quick reservation setup thanks to the single signaling loop.

The Boomerang protocol has been designed to be quick and simple, yet powerful. It aims for extending the QoS mechanisms offered by Diff-Serv, RSVP or Int-Serv [4] rather than replacing them.

This paper is organized as follows: Section 3. describes the Boomerang protocol and the reservation concept. The performance of Boomerang and RSVP is compared based on numerical analysis and a simulation study in Section 4. And finally, conclusions are drawn in Section 5.

3. Reservation concept

3.1. The Boomerang protocol

The Boomerang protocol contains only one protocol primitive for setting up bi-directional, soft-state resource reservations in IP networks. This message is forwarded hop-by-hop in the network to the Far-End Node, then it turn back and returns to the Initiating Node that is responsible for following the life cycle of the reservation.

The Boomerang reservation message contains the following parts for IPv4 traffic[5]: message header, flow specification and QoS specification.

3.1.1. Message header

This part contains the refresh interval and several flags in which Boomerang-aware nodes can indicate the result of the Boomerang message processing. The *refresh interval* in the message specifies how often the reservation has to be refreshed in order to keep the soft-state information in network nodes. Although the Initiating Node chooses the refresh interval, any node that requires a higher refresh rate can decrease that field while the message passes at the node.

3.1.2. Flow specification

A Boomerang node identifies a micro-flow uniquely by five fields, given in the flow specification part of the Boomerang message. These fields are the source IP address, the source port number, the destination IP address, the destination port number and the protocol identifier.

3.1.3. QoS specification

In the Boomerang reservation message, the Initiating Node indicates the type of QoS that it requires for the specified flow. The QoS parameter specifies the required bit rate both for the forward and reverse directions (they may of course be different) and the expected forwarding service in network nodes. Bit rates can be given by a single value or in vector form [6]. The Boomerang protocol can differentiate forwarding services according to the Diff-Serv [2] or the IPv4 Precedence Marking models [7].

3.1.4. Transport Mechanism

There are several ways for transporting the Boomerang protocol messages.

- In the current prototype implementation the Boomerang protocol messages are wrapped into ICMP ECHO / ECHO REPLY messages, i.e. into the PING message. This conveniently ensures the correct node behavior on most existing nodes: both passive routers and hosts.
- A cleaner implementation would be to define a new ICMP message for the Boomerang protocol primitives, or to assign an entirely new protocol for Boomerang protocol signaling. In both these later cases however, we would not have the correct end-node behavior for free. We would probably have to configure each unaware network node to accept these messages. The router alert option should be set for each Boomerang message in order to indicate that the routers should investigate this message [8].
- A slightly different approach is to transport the reservation messages as an integral part of the transport protocol. In this case a transport mechanism must be defined for each transport protocol where we wish to do resource reservation. This approach is

chosen in YESSIR [9], which is an inbound signaling protocol for RTP connections.

3.2. Operation

As the first step, the Initiating Node sends a Boomerang message to the Far-End Node containing the desired forward and reverse bit rates in the QoS specification. It is worth to mention that either the sender or the receiver host or even a network node can act as an Initiating Node. When this message reaches the Far-End Node, it is echoed back to the Initiating Node (see Figure 1).

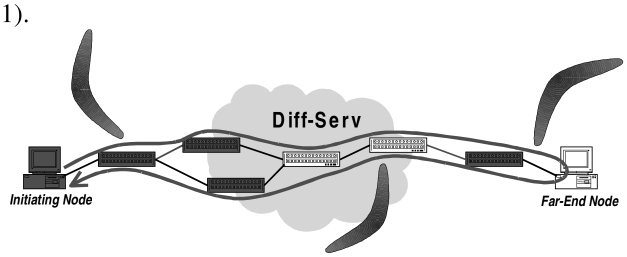

Figure 1. Reservation setup with the Boomerang protocol

The Boomerang message follows standard routing protocols and allocates resources hop-by-hop in every Boomerang-aware node en route. This ensures that the reservation will be made along the correct path for both upstream and downstream traffic. When the Initiating Node gets back the Boomerang message, it verifies the completion of the reservation by examining the appropriate message flags have been set by the Boomerang-aware nodes along the way. Reservation messages are then sent periodically from the Initiating Node to the Far-End Node to keep the reservation alive in all of the nodes along the upstream and downstream paths. Thanks to this refresh procedure, the reservation can easily adapt to route changes.

If a Boomerang-aware node has to deny a reservation request, it takes the following actions before passing over the reservation message:

a) Sets the NACK flag in the message header indicating that the reservation has failed.
b) If the requested resources are greater than the available resources at that node, the QoS Specification part of the Boomerang message is updated to reflect the maximum of currently available resources.
c) If the Refresh Interval in the Boomerang exceeds the refresh interval required by the node, it is also updated.

If a message arrives at the network node with the NACK flag already set, only actions (b) and (c) are taken. When the reservation arrives back at the Initiating Node, it checks the message to see if the reservation was successful. If the NACK flag is NOT set, then the reservation was successful, however the Refresh Interval field might still have changed. If this is the case, the Initiating Node continues to send Boomerang messages according to the new the refresh rate.

If the NACK flag is set, the request was denied somewhere along the message path. This is where the Initiating Node has to take some action. A new resource maximum value might have been specified in the returned message, so the Initiating Node may accept it and start using that guaranties, it also can retry the old request and finally it might release the whole request as well.

If the Boomerang message does not return to the Initiating Node within a certain time, it is considered to be lost. Thus the Initiating Node can repeat the original request again with the same resource demand or it can perhaps downgrade the demanded resource amount and request less.

3.3. Interoperation with Diff-Serv

Let us consider a scenario (see Figure 2) where we have two hosts (A, F) attached to a Diff-Serv domain (B-C-D-E). The Diff-Serv network is non-blocking for high priority traffic, but there is a highly utilized link in the middle (C-D) where over-provisioning is difficult or very expensive (e.g. the Atlantic cable). This is a potential bottleneck link for the guaranteed service.

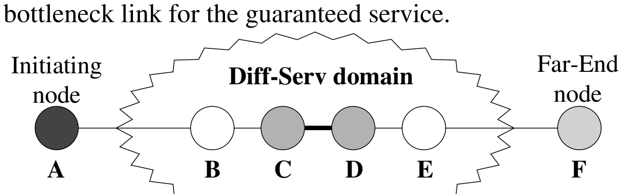

Figure 2. Single domain with a bottleneck

The traffic between hosts (A) and (F) has to pass through the Diff-Serv compliant network including the bottleneck link. Therefore one of the hosts should send a Boomerang protocol reservation request, in order to get resources on the bottleneck link in a guaranteed manner. Let us assume that host (A) sends a Boomerang protocol message, which is forwarded hop-by-hop through the network and echoed back by host (F). Since (B) and (E) are not Boomerang-aware nodes, they just simply forward the Boomerang message to the next hop. Node (C) as the ingress node of the bottleneck link performs admission control for the requested resources and indicates the result in the Boomerang message. Similarly, node (D) performs admission control for the reverse direction. It is possible to specify different amount of resources for the traffic going in the two directions, and forward and reverse paths can also differ. Host (A) can then see the result of the reservation from the returned Boomerang protocol message. In case of a successful reservation, host (A) can start to send (or receive) traffic with guaranteed QoS on the bottleneck links and Diff-Serv-based QoS in the rest of the Diff-Serv network. For the latter, data packets shall be marked with proper DSCP[3]. The initiating host should refresh the reservation by sending Boomerang protocol messages periodically to prevent the soft states in the nodes from timing out.

3.4. Flow-state aggregation

The Boomerang protocol does not require state machine for handling signaling in network nodes. However, other kind of states may be needed for several other reasons such as policing of flows and keeping track of the amount of booked resources in a robust way. This, however, is inevitable when someone wishes to have hard QoS guarantees, the emphasis here is on that the signaling protocol does not itself consume large memory spaces in routers, which is a key issue in terms of scalability.

To take full benefit of the Boomerang protocol in network nodes forwarding a large amount of traffic, admission control should be made without keeping flow-state information per micro-flow. There are different ways for making flow state aggregation in Boomerang protocol.

3.4.1. Aggregation suggested by earlier node

This is an extension to the Boomerang protocol, in which a so-called Aggregating Node suggests a destination subnet-based aggregation to subsequent network nodes. When the Aggregating node discovers that it has a large number of reservations between two subnets it can decide to append aggregation information for these subnets to each signaling message within this aggregated flow. This is done by appending a special message field to Boomerang containing the class, IP protocol, source / destination subnet mask for this aggregation, an aggregation key (e.g. the Aggregating Node's own IP number) and the total resource demand of this aggregated flow. Nodes inside the network may now use this information to create states for these aggregates of micro-flows and can perform policing on these. The later node can then in turn suggest further aggregation for following nodes.

3.4.2. Measurement based admission control

This group covers any method where the amount of allocated resources is not explicitly stored. On the contrary, admission control is performed by measuring the current amount of traffic per service class. Experiments indicate [11] that measurement based techniques can be made to work even with long-range dependant flows for a reasonably large aggregation.

3.4.3. Refresh message based admission control

Instead of storing hard states for keeping track of the used resources, the refresh request messages can be used. By summing the requested rates weighted by the refresh interval over a sliding window, the node can estimate the amount of resources already reserved and use this knowledge to do admission control. A similar approach is proposed in [12]. The accuracy of the estimate will depend on the number of refresh messages within the sliding window, and the node should therefore request a rather frequent small interval. As with the measurement-based scheme, dynamic policing of individual flows is not possible.

4. Performance analysis

In this chapter we compare the Boomerang protocol to RSVP[3], the well-known resource reservation protocol. We examined the *signaling overhead* caused by the protocols on the links and in the routers with numerical analysis and made also simulations to characterize the *reservation blocking probability*. We focus on comparing the direct performance measures of the protocols and neglect indirect circumstances, such as scheduling, classification delays and traffic conditioning tools. We use Telia's prototype implementation for Boomerang and the source code of the public ISI implementation for RSVP.

During the examination we investigate an Internet telephony application setting up a bi-directional, unicast flow over IPv4. To be fair we do not use any advantages that can not be performed by the other protocol. According to this, in case of Boomerang protocol we omit the possibility of flow aggregation and user authentication, while in the RSVP protocol we do not use policing and assume that the protocol messages are in one piece, i.e. there is no fragmentation.

In case of RSVP protocol we use a Fixed Filter style reservation [3] that with the two-two sources and destinations and the RSVP Path messages carried state advertisement that gather resource information along the flow.

4.1. Signaling overhead analysis

This performance measure expresses the bandwidth overhead caused by signaling messages during the life cycle of the reservation. Both protocols use soft state, which means that after the successful reservation setup they send refresh messages periodically along the flow to maintain flow-states in the routers. We went behind the length and the frequency of protocol messages and calculated a signaling bandwidth for both protocols from these values.

In case of Boomerang there is a universal reservation message that can be used for reservation set up, refresh and tear down. This message is 44 bytes long and travels from the Initiating Node to Far-End Node and back. The designers suggested 30 seconds for refresh time that means that in every 30 seconds the same message goes on the same way until the initiator tears down the reservation. [Equation 1] shows the calculated bandwidth that is consumed by the Boomerang protocol during a reservation that exists for T seconds. The reservation setup, refresh and reservation hang-up messages are counted 88 bytes long each as they travel the route from source to destination back and forth. The refresh message appears once in every T/30 seconds. We can assume that the connection uptime (T) is big, so the expression converges to 2.93 bytes/sec.

$$\frac{88 \cdot \left\lfloor \frac{T}{30} \right\rfloor + 88}{T} \left[\frac{byte}{\sec}\right]$$

Equation 1. Signaling overhead for boomerang

In case of RSVP protocol the reservation setup requires two different message types, the *Path* and the *Resv* messages. The first one is sent by the source and creates the path states in the routers while the second is sent by the destination and makes the resource reservation along the created path. The refresh time is a uniform distribution between 15 and 45 seconds with a 30 seconds mean value with almost the same messages as before, except that the *Path* refresh message is shorter, as it does not contain path state advertisement. For reservation hang-up there are also two kind of messages, the *PathTear* and the *ResvTear* messages that resolve the corresponding states in the routers. There is an additional *ResvConf* message that can assure the destination about the success of the reservation. As the presumed connection is bi-directional we need two sets of these messages. Table 1. shows the lengths of the messages.

Table 1. Size of signaling primitives for resource reservation in bytes

	RSVP		Boomerang
Reservation setup (*Path, Resv*)	168	116	44
State Refresh (*Path, Resv*)	88	116	44
Reservation hang up (*PathTear, ResvTear*)	44	52	44
ResvConf	108		N/A

We calculated the signaling bandwidth in the same way as previously for a connection that is reserved for T seconds. We needed two pairs of every message as the connection is reserved for bi-directional traffic and each end-point is a source and destination in one. Equation 2 shows the result of the calculation and if we also assume that the T is big then this converges to a 13.6 bytes/sec signaling traffic.

$$\frac{784 + 408 \cdot \left\lfloor \frac{T}{30} \right\rfloor + 192}{T} \left[\frac{byte}{\sec}\right]$$

Equation 2. Signaling overhead for RSVP

4.2. Resource demand analysis

In this measurement we tried to estimate how much memory and processing time is required in a router to handle the flow states. We looked up the data structures in the source codes that stores state information and computed their lengths. In case of processing time we tried to give estimation by counting how much line of C code is executed when a message is processed in the router.

For the memory allocation Boomerang consumes 26 bytes to keep state information for a flow while RSVP consumes 436 bytes for each flow!

Table 2 shows the number of lines and the consumed processing time that occur when a certain type of message is processed. The time estimation based on measurements that an average strength SUN workstation runs 1500 lines C code in 1 ms.

Table 2. Size of source code and amount of CPU time for Boomerang and RSVP

	RSVP		Boomerang	
Reservation setup	570 l	0.38 ms	160 l	0.11 ms
State refresh	460 l	0.31 ms	110 l	0.07 ms
Reservation hang up	370 l	0.25 ms	160 l	0.11 ms

4.3. Simulation environment

The modeled network scenario can be seen on Figure 3. We have an Initiating Node and a Far-End Node and 10 routers along the data path between them. Each link has got a capacity of 100 Mbps. The Initiating Node tries to make a resource reservation along the path for a 2 Mbps Internet telephony session. If it is successful then it holds the reservation for a time that is computed from an exponential distribution with a mean value of 10 min. If the reservation setup fails then the source releases the already reserved resources and makes a tries again after a certain time, that is computed from a uniform distribution between 5 and 7 until it is succeed. The time of the next reservation is calculated from an exponential distribution with a mean value of 30 min, this way a Poisson process is used as the source model. The routers also have background load that is caused by various number of sources that behave exactly same as the previously described source except that their traffic go through on the correspondent router only. To make the situation even more realistic, at the Initiating Node altogether 50 sources are competing for the resources. These sources however have not got only local significance: they are trying to build end-to-end connections using the signaling protocol (RSVP or Boomerang). We are examining one of them, to see how do they perform under different conditions.

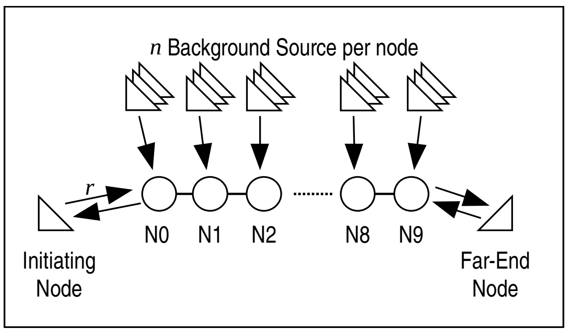

Figure 3. Network topology for the simulation study Reservation Setup Time

4.4. Number of reservation trials

According to the behavior of the source it retries the reservation if it was unsuccessful in the previous try until it will be succeed. Figure 4 shows the number of retries depending on the background load. If the number of retries is zero it means that the first try was successful and no retries are necessary. The amount of background load is represented by the number of background sources.

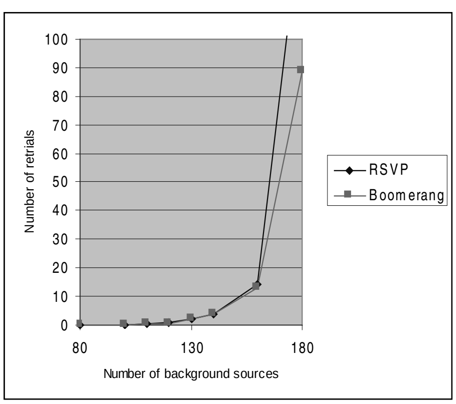

Figure 4. Number of reservation request retrials vs. background load

4.5. Reservation blocking probability

The functional behavior of the two protocols is characterized by comparing the reservation blocking probability in a network where routers make reservations using the mentioned protocols. The collected information is presented in Figure 5. showing the blocking probability if we scale the background load increasing the number of background sources.

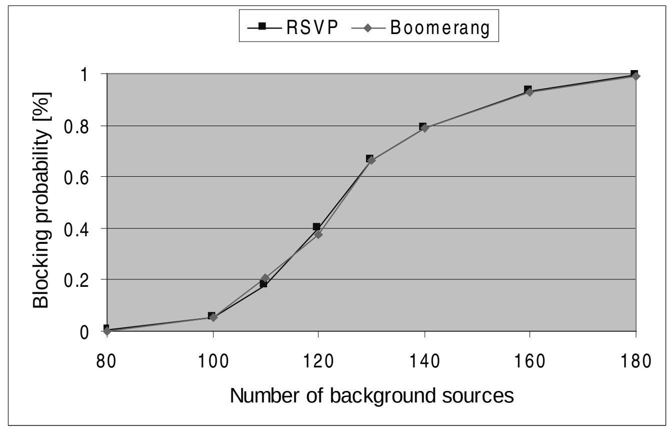

Figure 5. Reservation blocking probability vs. background load

The figure shows only little difference between the protocols, which we consider to be due to the over-simplified model. At the time this paper is being written, more realistic scenarios are developed, to pinpoint the different behaviors of the different protocols.

5. Conclusions

This paper proposes a new resource reservation protocol for IP networks. The main properties of the Boomerang protocol are grouped according to the main benefits they yield:

- Simple Implementation:

In Boomerang protocol, signaling states (meaning intelligence) are needed only in the Initiating Node. Other Boomerang nodes are "passive" and the only requirement is that they need to be able to interpret the Boomerang message. Therefore the most complex part of the Boomerang is implemented locally in Initiating Node and it is a reasonable belief that Boomerang protocol can be simply implemented in other nodes.

- Good Scalability:

In order to decrease the context made for keeping track of the reservation state of micro-flows, Aggregating Nodes can propose an aggregation of these flow states. Consistency of micro-flow aggregation is maintained by appending an aggregation field in the repeatedly circulated Boomerang protocol refresh messages. All control logic related to an aggregation is concentrated in one Aggregating Node, while other nodes can simply rely on the information steadily coming in Boomerang messages or handle micro-flow states.

- Quick Penetration:

The following features could result in an incremental deployment of Boomerang protocol in current IP networks. *Transparency* is ensured by fact that the only requirement for the Boomerang-unaware nodes is that they must be able to forward the Boomerang protocol messages. Another benefit is that there are *no requirements on the Far-End Node*, so the other end-user can be quite unaware of all Boomerang protocol implementation in Initiating Node and network nodes and still profit from its functionality.

At the time this paper is written, there are simulations and measurements in test-networks in progress. So far none of them is at the stage that can show final performance comparison results, but some simulation results could already been included in this paper.

The numerical analysis, however, has proven that the signaling overhead of Boomerang is significantly (4-5 times) less than for RSVP and its resource demand is considerable small, which shows that Boomerang protocol is really a lightweight protocol that can be easily implemented in backbone routers.

6. References

[1] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, K. Nichols, M. Speer, "A Framework for End-to-End QoS Combining RSVP/Intserv and Differentiated Services", IETF <draft-ietf-DiffServ-rsvp-00.txt>, March 1998.

[2] D. Black, S. Blake, M. Carlson, E. Davies, Z. Wang and W. Weiss, "An Architecture for Differentiated Services", Internet Draft, May 1998.

[3] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource Reservation Protocol (RSVP) Version 1 Functional Specification", IETF RFC 2205, Proposed Standard, September 1997.

[4] R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An Overview", Internet RFC 1633, July 1994.

[5] D. Ahlard, J. Bergkvist, I. Cselényi, T. Engborg, "Boomerang - A Simple Resource Reservation Framework for IP, Internet Draft, February 1999; http://www.ietf.org/internet-drafts/draft-ahlard-boomerang-framework-00.txt

[6] I. Cselényi, R. Szabó, I. Szabó, N. Björkman, A. Latour-Henner, "Experimental Platform for Telecommunication Resource Management" Computer Communications (21)17 (1998) pp.1624-1640

[7] Internet Protocol - DARPA Internet Program Protocol Specification, Information Sciences Institute University of Southern California, September 1981

[8] D. Katz, "IP router alert option", RFC 2113, IETF, February 1997

[9] P. Pan, H. Schulzrinne, "YESSIR: A Simple Reservation Mechnism for the Internet", http://www.ctr.columbia.edu/~pan/work/yessir/yessir.ps

[10] K. Nichols and S. Blake, "Definition of the Differentiated Services Field (Diff-Serv Byte) in the IPv4 and IPv6 Headers", Internet Draft, May 1998.

[11] J.T. Lewis, R. Russell, F. Toomey, B. McGurk, S. Crosby, I. Leslie, "Practical connection admission control for ATM networks based on on-line measurements", Computer Communications (21)17 (1998) pp. 1585-1596

[12] A. Eriksson, C. Gehrmann, "Robust and Secure Light-weight Resource Reservation for Unicast IP Traffic", International WS on QoS'98, IWQoS'98, May 18-20, 1998