RMD – a lightweight application of NSIS

G. Karagiannis[1], A. Báder[2], G. Pongrácz[2], A. Császár[2], A. Takács[2], R. Szabó[3], L. Westberg[4]

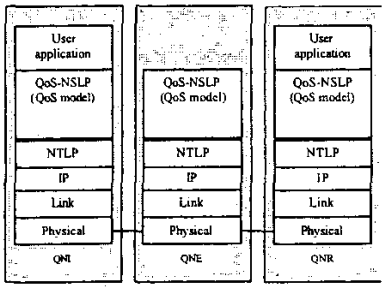
[1] University of Twente, P.O. BOX 217, 7500 AE Enschede, The Netherlands, {email:karagian@cs.utwente.nl}
[2] Traffic Laboratory, Ericsson Research Hungary, Irinyi u. 4-20, P.O.Box 107, H-1300 Budapest, Hungary
[3] Budapest University of Technology and Economics, Magyart Tudosok krt. 2, H-1117, Budapest, Hungary
[4] Ericsson Research, Törshamnsgatan 23, SE-164 80 Stockholm, Sweden

Abstract

Resource Management in DiffServ (RMD) framework is a simple, effective and scalable signaling method for resource reservation within a DiffServ domain. In the present paper we describe the way of integrating RMD in the future NSIS QoS application protocol that is under standardization in IETF. We demonstrate that the RMD edge-to-edge signaling is able to inter-work with end-to-end resource reservation protocols e.g. RSVP. Basic performance measurements and evaluation, comparing RMD and RSVP, are also shown.

1 Introduction

A number of different QoS solutions have been developed by the Internet Engineering Task Force (IETF), among them Integrated Services (IntServ) [1] and its signaling protocol, resource ReSerVation Protocol (RSVP), defined in [2]. RSVP is a resource reservation signaling protocol that is designed to be applied in an end-to-end communication path. It can be used by an application to make its quality of service (QoS) requirements known and reserve resources in all the network nodes in the path. RSVP has not enjoyed the level of deployment that might have been expected. This is argued with its design and complexity (e.g. optimized for multicast). The other main reason is that for many applications a simple traffic differentiation, i.e. DiffServ architecture, and simple over-provisioning is able to provide the required QoS.



QNE is a NSIS Entity (NE), supporting QoS-NSLP
QNI is the first node in the sequence of QNEs that issues a reservation request.
QNR is the last node in the sequence of QNEs that receives a reservation request.

Figure 1 Basic QoS-NSLP scenario

A new IETF working group, denoted as Next Steps in Signaling (NSIS) has been initiated to solve the deployment constraints of the RSVP protocol. The NSIS working group is considering a generic signaling framework for the control of IP datagram delivery. It provides a model for the network entities that takes part in such signaling and for the relationship between signaling and the rest of the network operation. The NSIS overall protocol suite is decomposed into a generic (lower) layer named NSIS Transport Layer Protocol (NTLP), and a separate upper layer for each signaling application known as NSIS Signaling Layer Protocol (NSLP). The core functionality of the NTLP is efficient upstream and downstream peer-to-peer message delivery, in a wide variety of network scenarios. The NSLP includes signaling for different services or resources, such as QoS resources, i.e., QoS-NSLP, and network address translation (NAT) and firewall traversal, i.e., NAT/Firewall-NSLP (see [3]). The rest of the article will only focus on the QoS-NSLP [4].

The QoS-NSLP protocol establishes and maintains reservation states at nodes along the path of a data flow for providing forwarding resources for that flow. It is intended to satisfy the QoS related requirements as described in [5]. The design of QoS-NSLP is conceptually similar to RSVP [2], and uses soft-state peer-to-peer refresh messages as the primary state management mechanism. However, QoS-NSLP extends the set of reservation mechanisms to support sender or receiver initiated reservations, bi-directional reservations or reservations between arbitrary nodes, e.g. edge-to-edge, end-to-access, etc. On the other hand, there is no support for IP multicast as it was considered as a major complexity issue with RSVP. The reference model of QoS-NSLP protocol stacks is shown in Figure 1.

Within the NSIS signaling framework different QoS models can be used. A QoS model is a mechanism for achieving QoS as a whole. Its specification includes a description of its QoS parameter information, as well as how that information should be treated or interpreted in the network. Examples of QoS models are Integrated Services [1], Differentiated Services [6] and Resource Management in DiffServ (RMD) [7], [8].

2 RMD and NSIS

RMD represents a QoS framework that extends the DiffServ architecture with new admission control and resource reservation concepts in a scalable way. It is based on standardized DiffServ principles for traffic differentiation and as such, it is a single domain, edge-to-edge resource management scheme. The RMD QoS model in NSIS describes a lightweight signaling method for making resource reservation within DiffServ routers and providing QoS in a DiffServ domain [7][12].

Scalability in RMD is achieved by separating a complex reservation mechanism used in edge nodes from a much simpler reservation mechanism needed in interior nodes. It is assumed that edge nodes support "per-flow states" in order to initiate or terminate aggregated signaling. On the other hand interior nodes use only aggregated reservation state per traffic class. Compared to the RSVP aggregation method [9], the major difference is that in interior nodes the aggregated reservation states are not identified based on the combination of the used DSCP, and the source-destination edge pairs, but they are identified based only on the used DSCP.

Corresponding to these principles, two types of protocols are defined in RMD: the Per Domain Reservation (PDR) protocol and the Per Hop Reservation (PHR) protocol. The PDR protocol is used for resource management in the whole DiffServ domain, while the PHR protocol is used for managing resources for each node in the domain, on per hop basis, and per DSCP. The PDR protocol can either be a newly defined protocol or an existing one such as RSVP [2] or RSVP aggregation [9], while the PHR protocol is a newly defined protocol. So far, there are two types of PHR protocols specified, the RMD On-demand (RODA) PHR protocol and the RMD Measurement-based Admission Control (RIMA) (see [7], [8]).

In RODA, each node in the communication path from an ingress node to an egress node keeps only one reservation state per traffic class. The reservation is done in terms of resource units, which may be based on a single parameter, such as bandwidth, or on a combination of parameters.

These resources are requested dynamically per PHB (i.e., per DSCP) and reserved on demand on all nodes

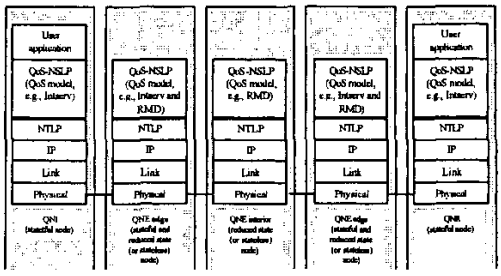


Figure 2: QoS-NSLP scenario that includes stateful, reduced state (or stateless) nodes

in the communication path from an ingress node to an egress node. In addition to this, RODA has to maintain a threshold for each PHB that specifies the maximum number of reserved resource units. This threshold can, for example, be statically configured.

RIMA is used to check the availability of resources before flows are admitted and without installing any reservation state. The measurements are done on the real average traffic (user) data load and the available resources are compared to the requested resources in each node. The main advantage of RIMA is that interior (core) nodes do not have to maintain any reservation states.

The RMD concept can be introduced in NSIS as a local QoS-NSLP model (or architecture) using a stateless/reduced state operation mode [4]. A measurement based operation, such as RIMA, is conceivable if QNEs interior (core) to a domain store neither NSLP nor NTLP state (stateless operation). They rather e.g. send and receive messages querying available resources, and react upon overload detection. Also a reduced-state operation, such as RSVP aggregation [9] or RODA is conceivable, in which QNEs do not store full per session (per-flow or per-aggregate) state in NSLP, but rather, e.g. one per-class state in NSLP and no state in NTLP (reduced state operation).

A stateless/reduced state QoS-NSLP operation makes the most sense when the underlying NTLP is able to operate in a stateless way as well. Such nodes should not worry about keeping reverse path state, message fragmentation and reassembly (at the NTLP), congestion control or security associations. A stateless/reduced state QNE will be able to inform the underlying NTLP of this situation.

Stateless/reduced state operation is only applicable under certain circumstances. This operation prohibits receiver-initiated reservation, message bundling, fragmentation and reassembly or per-flow security associations within the RMD domain, which we argue not be a limitation when proper engineering and domain policies are used. They are not able to establish and maintain security associations with their stateless neighbours, which means, they can only be applied in a trusted environment. For these reasons, a typical

application of the stateless operation mode is the QoS signaling within a single domain where the edges of the domain are statefull QNEs and the interior nodes are stateless QNEs.

The protocol model of RMD used as a local NSIS QoS model is shown in Figure 2. The QNI and QNR are statefull nodes, the QNE edges are statefull and reduced state (or stateless) nodes and the QNE interior (core) is reduced state (or stateless) node. The QNE edges are able to support the PDR and PHR type of RMD protocol associations, while the QNE interior (core) node(s) is (are) able to support only the PHR type of RMD protocol associations.

In edge nodes the end-to-end QoS-NSLP messages are initiating/terminating local QoS-NSLP messages. The e2e messages are transmitted to the egress edge node, possibly using the reliable transport option of NTLP. In NSIS, the PHR and PDR RMD messages are encapsulated into the NSLP objects that specify the local RMD QoS model [12], i.e., the PHR and PDR objects, respectively. The local NSLP messages used by the RMD QoS model are using the simple NTLP datagram mode. The protocol operation within the domain is very similar to the one described in the original RMD concept [8].

The QoS-NSLP protocol and integration of RMD and NSIS are under development. However, a similar method of integrating the RMD concept into RSVP as an e2e signaling protocol has been specified and implemented, which is described in the following section.

3 RSVP and RMD inter-working

RMD is not designed to be an end-to-end signaling protocol, but it can inherently inter-work with any e2e signaling protocols. As an example RSVP-RMD-RSVP inter-working is shown in Figure 3. RMD statefull edges provide the inter-working functionality between the RSVP and RMD protocols. Within the domain, stateless QNE's are used. Note that the e2e signaling will consider the RMD domain as one single QoS aware hop.

When RSVP PATH messages arrive at the ingress edge, the RSVP requested reservation bandwidth is mapped into a RMD requested reservation bandwidth. This will also initiate the RMD signaling procedures. The RSVP messages are sent transparently over the RMD domain with ingress and egress edges set as RSVP hops. If either reservations fail, i.e., inside or outside the RMD domain, then the corresponding RSVP user agent must be notified or a RMD tear down must be issued respectively. See Figure 3 for operational details. Even though both protocols operate with soft state refresh mechanisms the timing of these refreshes are stricter for RMD. Therefore only a loose coupling is maintained, i.e., as long as RSVP reservation is maintained around the RMD domain, RMD refreshes are issued.

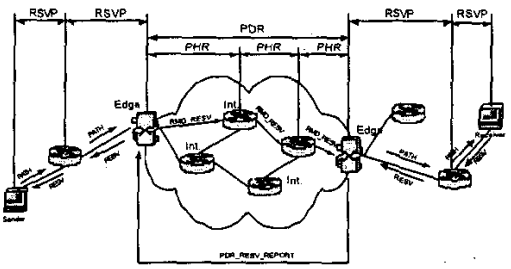


Figure 3 RSVP RMD inter-working, protocols within and outside an RMD domain

In many applications a bi-directional reservation procedure is needed. In RSVP, a bidirectional reservation can only be performed by two independent reservations in the two directions, i.e., forward and reverse. In RMD, if a bi-directional reservation is initiated by one edge node towards another edge node, then two sender-initiated reservations have to be initiated by these two edge nodes, one in the forward direction and another one in the reverse direction. The more detailed description of the RMD and RSVP inter-working operation and message sequence can be found in [10].

4 RMD performance

4.1 Response times

This section describes the performance comparison between the RSVP and RMD reservation establishment in terms of their response times. The signaling details of both protocols for a successful and unsuccessful reservation are shown in Figure 4 and Figure 5, respectively. Let's denote the downstream and upstream signaling delays with d_d and d_u, respectively, in accordance with the data flow (note RSVP's receiver initiated reservation). Applying the notations it takes RSVP $d_d + d_u + d_d$ time to establish and inform both parties of the successful reservation while only $d_d + d_u$ for RMD. Note that $d_d + d_u$ is equal to one Round Trip Time (RTT). If only the response times for the reservation requests are compared then for both protocols the reservation establishment is accomplished in *1* RTT.

In case of a reservation failure, RSVP feeds back the error as soon as it occurs. Therefore, for the sender side it may take a timeout interval (> RTT) to detect the failure, unless a Path Error is signalled earlier. On the other hand RMD takes the complete signaling loop regardless of its success or failure with the reservation. Table 1 summarises the response times for both protocols with the assumptions that $d_d = d_u$ and $d_d + d_u$ = *1* RTT. One can conclude that RMD is in all cases at least as fast as RSVP.

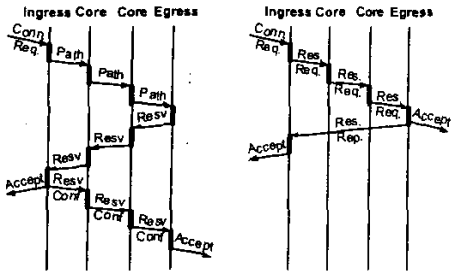


Figure 4 Successful reservation of RSVP (left) and RMD (right)

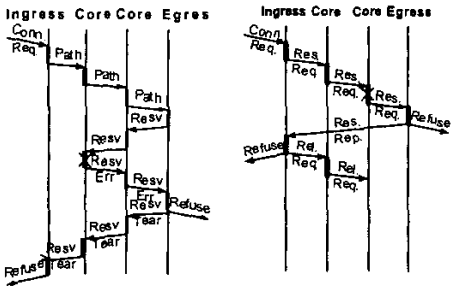


Figure 5 Refused reservation of RSVP (left) and RMD (right)

While regular connection release requires the same notification time for both protocols, the release of the refused connections on the successfully reserved part of the path requires different notification times. With RMD, a superfluous link reservation always needs *1 RTT* to be released. With RSVP, the closer the connection is refused to the receiver, the smaller this partial release time is.

	RMD		RSVP	
	Admitted	Refused	Admitted	Refused
Sender	1 RTT	1 RTT	1 RTT	1RTT or time-out
Receiver	0.5 RTT	0.5 RTT	1.5 RTT	0.5 RTT < = 1.5 RTT

Table 1: Response times

4.2 Protocol overhead

Protocol overhead can be divided into computational, storage and bandwidth overhead (see [11]).

	Edge node	Core node
RMD/RODA	O(f)	O(1)
RMD/Hybrid	O(f)	O(1)
RMD/MBAC	O(f)	O(1)
RSVP	O(f)	O(f)

Table 2: Memory requirements

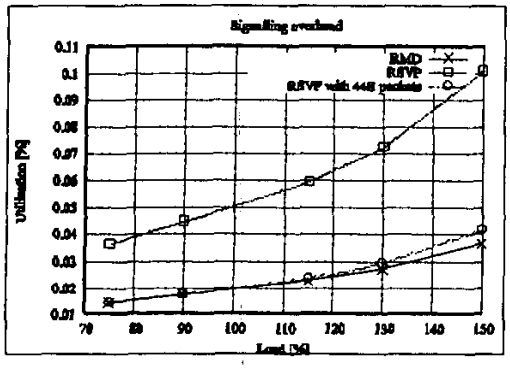


Figure 6 Signaling overhead

Computational and storage overheads are strongly correlated through the complexity of searching algorithms. The memory requirements are shown in Table 2, where f represents the number of flows. In high-speed networks, where the number of flows can be many thousands, DiffServ and RMD have a clear advantage over IntServ and RSVP. Moreover, RMD does not need to create a new signaling message, on each passing core node like RSVP does.

Considering as performance measure the bandwidth overhead, we traced the bandwidth consumption of signaling messages in a test scenario. While RSVP packet sizes followed [2], the packet sizes with RMD were over-estimated with 44 bytes (see [10]). We calculated the maximal signaling utilisation on any of the links in the network, the values are shown in Figure 6. We can conclude that the signaling overhead is not significant with either protocol. Though, RSVP occupies around twice more bandwidth for signaling messages. This is due to the fact that average signaling packet size of RSVP is bigger than 44 bytes. To see the pure difference in the number of signaling messages, we ran the RSVP simulations again with fixed 44 bytes simulated packet sizes. The results show that RSVP still consumes a little higher signaling bandwidth because it involves more packets for refusing a connection than RMD. Hence, RMD performs better under high loads than RSVP because the higher the load is, the more flows must be refused.

4.3 Severe congestion handling

When link or node failure is detected by dynamic routing protocols, they propagate this information into the network so that other routers can bypass the failed part. At first such a failure seems to be a routing issue, but it has influence on resource reservation because some flows will traverse a (partially) new path after re-routing, where no bandwidth has been reserved. If the new path has already been highly utilised, the quality of both the original flows of that path and the new ones will degrade, even packet drops may occur. We identify this undesirable situation as "severe congestion". The situation should be detected and handled to reduce the performance degradation introduced by the excess traffic.

RSVP resolves the situation with the help of a feature called "local repair", where the re-routing node trig-

gers the re-routed flows to refresh their reservation on the new path if possible. The excess traffic that cannot reserve bandwidth on the new path can be terminated. Please note that this procedure can be accomplished only because RSVP stores per-flow state information, and it can identify every single flow that was re-routed. RMD has no such possibility, therefore, severe congestion handling with RMD is triggered from the node that detects the overload, e.g., by user data measurement. The interior (core) node measures the ratio of overload, notifies this information to the egress nodes, which will terminate the appropriate amount of flows to solve the overload and the severe congestion situation.

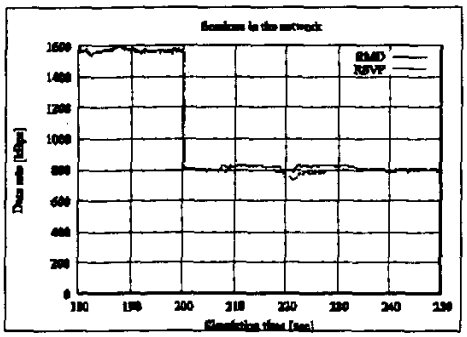


Figure 7 Severe congestion handling with RMD and RSVP

Figure 7 shows the simulation results of severe congestion handling for an RMD and an RSVP example network scenario. In both cases, the results are approximately the same. The figure shows that before severe congestion the studied network scenario could support 1600 kBps traffic on two paths. At 200 seconds simulation time a link on one of the paths goes down, forcing the dynamic routing protocol to divert all traffic onto one path, where only 800 kBps traffic can be forwarded, so the re-routed 800 kBps causes overload. Both RSVP and RMD protocols have to consider that around half of the flows get terminated so that the other half can be served without any more QoS degradation.

4.4 Measurements

Basic performance measurement tests were carried out on the RMD prototype developed at Ericsson Research using a simple measurement setup. The test network consisted of 3 PCs (ingress: 500MHz P3, core: 333 MHz P2, egress: 266 MHz P2) with Mandrake Linux 9.0 installed on them. The RMD prototype was implemented in Python code extended with shell scripts in the edge nodes, and in C source in the core node. ISI RSVP v4.2a4 was used as an opponent in the tests. In the core server RMD is implemented as kernel and Linux traffic control modules. In each test only signaling load was present in the network, no traffic was generated on these links.

The first measurements were focused on the interior (core) performance for estimating the mean processing delays of the RODA PHR signaling messages. In the interior (core) node, all PHR messages are handled with a very high speed. The average handling times (see Figure 8) of an average ping packet and a PHR packet are approximately the same, i.e., 60±10 μsec and are independent of the number of flows.

The processing time of RODA PHR and RSVP messages were measured also at the signal processing/IP forwarding interface as well. The mean processing delay of traffic forwarding through the IP packet forwarder and Network Interface Card were 12 μsec. In these performance measurements only one flow (session) was running on the interior (core) node, therefore, these results refer to unloaded interior (core) nodes.

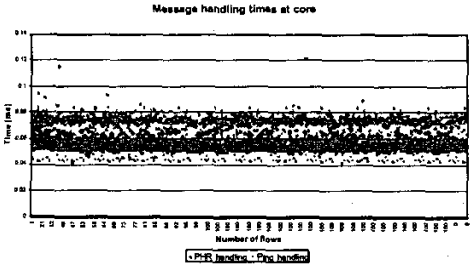


Figure 8 Message handling time in core routers measured at the network interface cards

The mean processing delays of the RODA PHR and RSVP RESV signaling messages are listed in Table 3. The measured mean processing delay of the RSVP PATH message was 273 μsec.

	RODA PHR (delays)	RSVP (delays and their 95% confidence intervals)
Reservation	< 2	790 [762, 793]
Refresh	< 2	67 [64, 69]

Table 3: Mean processing delays in *μs*.

As the results show, the mean processing delays of the RODA PHR reservation messages are significantly smaller than the mean processing delays of the RSVP reservation messages. The reason is that, unlike RSVP, the RODA PHR does not maintain per-flow traffic classifiers, and it does not require per-flow maintenance and lookup in a reservation table.

In Figure 9 the call setup times of RSVP and RMD can be seen. The absolute values depend highly on the performance of the used computers, therefore the relative measured values (between PHR and RSVP) are of more interest.

Refresh messages are handled very fast. The message handling times for a typical refresh message processing time measured at the network interface cards is depicted in Figure 10.

It is expected, that in large networks, where the flows pass several core routers, end-to-end setup times leads to different results in case of RMD and RSVP. Using RMD, the setup time remains nearly constant, as it is caused by the scripts that are running on the edges (about 0.5 s), while each core node processes the setup message fast (below 100 µsec). Using RSVP, the setup time depends heavily on the number of nodes along the path but also on the number of flows. Thus, in terms of scalability, RMD has clear advantages over RSVP.

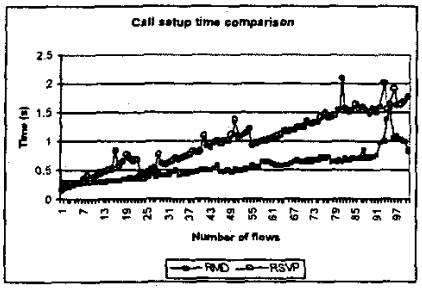


Figure 9 Call setup time in the function of the number of flows

As far as the CPU load is concerned, it remains below 1% up to 100 flows in the RMD core modules. The edge nodes do the most of the job, which makes their CPU moderately loaded, the CPU reached 40% load when the node builds up and keeps alive 100 flows in our test network. RSVP nodes are loaded heavier when building up the flows (about 55% CPU utilization), while they produce smaller load when it comes to keep the flows alive (about 7-10%).

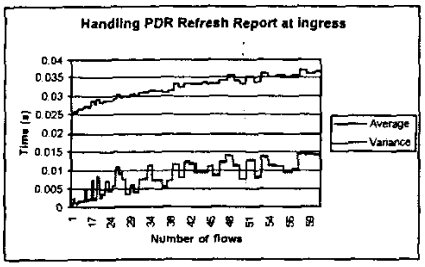


Figure 10 Refresh message handling time in edge nodes as a function of number of flows

The measurement results show that RMD does not cause any significant performance degradation in the interior nodes, while RSVP does. Knowing that the number of flows is usually much higher in the interior nodes than in the edges, the advantage of RMD is clear. At the network edge, the performances of the current RMD and RSVP implementations are similar.

5 Conclusion

The authors believe that the RMD framework, because of its simplicity and good scalability, is suitable for resource management in future IP based networks. RMD can be integrated into the NSIS protocol framework in a natural way, and used for providing QoS signaling to DiffServ routers. RMD as an edge-to-edge signaling protocol is able to inter-work with other end-to-end resource reservation signaling protocols as well.

Preliminary prototype measurements and performance evaluation show that from many aspects RMD performance is superior to that of RSVP. In interior nodes minimum functionality is needed. These nodes do not necessarily need to be powerful routers, and they still can handle a large number of flows and have a fast forwarding capability. The RMD edge node performance behaviour is similar to the RSVP node performance behaviour when handling per-flow operations.

From the deployment point of view we expect that the introduction of the RMD framework will require limited initial investment costs and limited operating expenses because of its simple implementation of the PHR protocol in the interior (core) routers.

6 Acknowledgement

This work has been partly funded by the EU 5FP project Seacorn.

7 Literature

[1] Braden, R., et al., "Integrated services in the internet architecture: an overview", RFC1633, IETF, 1994.

[2] Braden, R., et al. "Resource reservation protocol (RSVP) functional specification", RFC2205, IETF, 1997.

[3] Hancock, R., Freytsis, I., Karagiannis, G., Loughney, J.,Van den Bosch, S, "Next Steps in Signaling: Framework", IETF Internet Draft, 2003, Work in progress

[4] Van den Bosch, S., Karagiannis, G., McDonald, A., "NSLP for Quality-of-Service signaling", IETF Internet Draft, 2004, Work in progress.

[5] Brunner, M., Ed. "Requirements for QoS signaling protocols", IETF Internet Draft, 2003, Work in progress.

[6] Blake, S., et al. "An Architecture for Differentiated Services", RFC 2475, IETF, 1998.

[7] Westberg, L., et al. "Resource Management in DiffServ Framework", IETF Int. Draft, 2003, Work in Progress.

[8] Westberg, L., et al., "Resource Management in DiffServ (RMD): A Functionality and Performance Behavior Overview", IFIP PfHSN'02, 2002, Berlin.

[9] Baker, F., et al., "Aggregation of RSVP for IPv4 and IPv6 Reservations", RFC 3175, IETF, 2001.

[10] Westberg, L., Bader, A., Partain D., Rexhepi, V., Karagiannis, G., "Proposal for RSVPv2-NSLP", IETF Internet Draft, work in progress, 2003.

[11] Császár, A., Takács, A., "Comparative Performance Analysis of RSVP and RMD", QoFIS2003, pp. 41-51, Oct. 2003, Stockholm, Sweden.

[12] Bader, A., Karagiannis, G. Westberg, L. "RMD (Resource Management in DiffServ) QoS-NSLP model" IETF Internet Draft, 2004, Work in Progress