Resource Management in Diffserv (RMD): A Functionality and Performance Behavior Overview

Lars Westberg[1], András Császár[2], Georgios Karagiannis[3], Ádám Marquetant[2], David Partain[4], Octavian Pop[2], Vlora Rexhepi[3], Róbert Szabó[2,5], and Attila Takács[2]

[1] Ericsson Research
Torshamnsgatan 23 SE-164 80, Stockholm, Sweden
lars.westberg@era-t.ericsson.se
[2] Net Lab, Ericsson Research Hungary
Laborc u. 1., Budapest H-1037, Hungary
{robert.szabo,andras.csaszar,adam.marquetant,octavian.pop,
attila.takacs}@eth.ericsson.se
[3] Ericsson EuroLab Netherlands,
P.O. Box 645, 7500 AP Enschede, The Netherlands
{vlora.rexhepi,georgios.karagiannis}@eln.ericsson.se
[4] Ericsson Radio Systems AB
P.O. Box 1248 SE-581 12 Linkoping, Sweden
david.partain@ericsson.com
[5]High Speed Networks Laboratory, Department of Telecommunications and Telematics,
Budapest University of Technology and Economics
Stoczek u. 2, Budapest H-1111, Hungary
robert.szabo@ttt.bme.hu

Abstract. The flexibility and the wide deployment of IP technologies have driven the development of IP-based solutions for wireless networks, like IP-based Radio Access Networks (RAN). These networks have different characteristics when compared to traditional IP networks, imposing very strict requirements on Quality of Service (QoS) solutions, such as fast dynamic resource reservation, simplicity, scalability, low cost, severe congestion handling and easy implementation. A new QoS framework, called Resource Management in Differentiated Services (RMD), aims to satisfy these requirements. RMD has been introduced in recent publications. It extends the IETF Differentiated Services (Diffserv) architecture with new admission control and resource reservation concepts in a scalable way. This paper gives an overview of the RMD functionality and its performance behavior. Furthermore, it shows that the mean processing delay of RMD signaling reservation messages is more than 1330 times smaller then the mean processing delay of RSVP signaling reservation messages.

Introduction

Internet QoS has been one of the most challenging topics of networking research for several years now. This is due to the diversity of current Internet applications, ranging from simple ones like e-mail and World Wide Web (WWW) up to demanding real-

G. Carle and M. Zitterbart (Eds.): PfHSN 2002, LNCS 2334, pp. 17–34, 2002.

time applications, like IP telephony, which are increasing the demand for better performance on the Internet.

Currently, due to flexibility and wide deployment of IP technologies, IP-based solutions have been proposed for wireless networks, like IP-based Radio Access Networks (RAN). These networks have different characteristics when compared to traditional IP networks, imposing very strict requirements on QoS solutions [19]. These QoS requirements are fast dynamic resource reservation, simplicity, low costs, severe congestion handling and easy implementation along with good scalability properties.

The Internet Engineering Task Force (IETF) standardization body is starting a new Working Group (WG), called Next Steps In Signaling (NSIS) [5] to specify and develop new types of QoS signaling solutions that will meet the real time application requirements and the QoS requirements imposed by the IP-based wireless networks. Several resource reservation mechanisms defined in the context of IP networks might be used as input to this WG. The most promising are RSVP (Resource reSerVation Protocol) [6], RSVP aggregation [7], Boomerang [8], YESSIR (YEt another Sender Session Internet Reservation) [9], Feedback control extension to differentiated services [10], Dynamic packet states [11], and Dynamic Reservation Protocol (DRP) [12].

While these schemes are able to satisfy the expectations that users of demanding real-time applications have, they fail to meet the strict QoS requirements imposed by IP-based wireless networks.

This paper presents a new QoS framework, called Resource Management in Differentiated Services (RMD), which aims to correct this situation. RMD is introduced in [13], [14], [15], [16], [17] and [18], and it represents a QoS framework that extends the Diffserv architecture with new admission control and resource reservation concepts in a scalable way. Even though it is optimized for networks with fast and highly dynamic resource reservation requirements, such as IP-based cellular radio access networks [19], it can be applied in any type of Diffserv networks. This paper presents a general overview of the RMD concept, functionality and its performance behavior.

The organization of this paper is as follows: Section 2 gives an introduction of the RMD fundamental concepts. The two RMD operation types, i.e., normal operation and fault handling operation are described in Section 3. Section 4 describes the performance evaluation experiments that were used to observe the performance behavior of the RMD framework. Finally, the conclusion and future work is given in Section 5.

RMD Fundamental Concepts

The RMD proposal described in this paper is based on standardized Diffserv principles for traffic differentiation and as such it is a single domain, edge-to-edge resource management scheme. RMD extends the Diffserv principles with new ones necessary to provide dynamic resource management and admission control in Diffserv domains. In general, RMD is a rather simple scheme.

Basic Idea behind RMD

The development of RMD was initially based on two main design goals. The first one was that RMD should be stateless on some nodes (e.g. interior routers), i.e., no per-flow state is used, unlike RSVP [6] , which installs one state for each flow on all the nodes in the communication path. The second goal was that RMD even though stateless should associate each reservation to each flow and therefore should provide certain QoS guarantees for each flow.

These two goals are met by separating a complex reservation mechanism used in some nodes from a much simpler reservation mechanism needed in other nodes.

In particular, it is assumed that some nodes will support “per-flow states”, i.e., are stateful. In RMD these nodes are denoted as “edge nodes”. However, any nodes that maintain reservation states could fulfill this requirement.

The second assumption is that the nodes between these stateful nodes can have a simpler execution by using only one aggregated reservation state per traffic class. In RMD these nodes are denoted as interior nodes.

The edges will generate reservation requests for each flow, similar to RSVP, but in order to achieve simplicity in interior nodes, a measurement-based approach on the number of the requested resources per traffic class is applied. In practice, this means that the aggregated reservation state per traffic class in the interior nodes is updated by a measurement-based algorithm that uses the requested and available resources as input. Unlike typical measurement based admission control (MBAC) algorithms, that apply admission control using data traffic measurements and available resources as input, RMD applies admission control on resource parameter values included in the reservation requests, i.e. signaling messages and available resources per traffic class.

RMD Protocols

The scalability of the Diffserv architecture is achieved by offering services on an aggregated basis rather than per flow and by forcing the per-flow state as much as possible to the edges of the network. The Differentiated Services (DS) field in the IP header and the Per-Hop Behavior (PHB) are the main building blocks used for service differentiation. Packets are handled at each node according to the PHB indicated by the DS field (i.e., Differentiated Service Code Point (DSCP) [4]) in the message header. However, the Diffserv architecture currently does not have a standardized solution for dynamic resource reservation.

The RMD framework is a dynamic resource management developed on Diffserv principles, which does not affect its scalability. This is achieved by separation of the complex per domain reservation mechanism from the simple reservation mechanism needed for a node. Accordingly, in the RMD framework, there are two types of protocols defined: 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, on per hop basis, i.e., per DSCP. The PDR protocol can either be a newly defined protocol or an existing one such as RSVP [6] or RSVP aggregation [7],

while the PHR protocol is a newly defined protocol. So far there is only one PHR protocol specified, the RMD On-demand (RODA) PHR [14] protocol.
This is made possible by definition of the Per Domain Reservation (PDR) protocol and the Per Hop Reservation (PHR) protocol. The signaling messages used by each protocol are given as well.

Per Domain Reservation – PDR Protocol

The PDR protocol manages the reservation of the resources in the entire Diffserv domain and is only implemented in the edge nodes of the domain. This protocol handles the interoperation with external resource reservation protocols and PHR protocol. The PDR protocol thus can be seen as a link between the external resource reservation scheme and the PHR.

The linkage is done at the edge nodes by associating the external reservation request flow identifier (ID) with the internal PHR resource reservation request. This flow ID, depending on the external reservation request, can be of different formats. For example, a flow specification ID can be a combination of source IP address, destination IP address and the DSCP field.

A PDR protocol has a set of functions associated, regardless of whether PDR protocol is an existing protocol or a newly-defined protocol. But, depending on the type of network where RMD is applied, it may have also a specific set of functions.

A PDR protocol implements all or a subset of the following functions:

- Mapping of external QoS requests to a Diffserv Code Point (DSCP);
- Admission control and/or resource reservation within a domain;
- Maintenance of flow identifier and reservation state per flow (or aggregated flows), e.g. by using soft state refresh;
- Notification of the ingress node IP address to the egress node;
- Notification that signaling messages (PHR and PDR) were lost in the communication path from the ingress to the egress nodes;
- Notification of resource availability in all the nodes located in the communication path from the ingress to the egress nodes;
- Severe congestion handling. Due to a route change or a link failure a severe congestion situation may occur. The egress node is notified by PHR when such a severe congestion situation occurs. Using PDR, the egress node notifies the ingress node about this severe congestion situation. The ingress node solves this situation by using a predefined policy, e.g., refuses new incoming flows and terminates a portion of the affected flows.

These functions are described in detail in [13].

Per Hop Reservation – PHR Protocol

The PHR protocol extends the PHB in Diffserv with resource reservation, enabling reservation of resources per traffic class in each node within a Diffserv domain. This protocol is not able to differentiate between individual traffic flows, as for example RSVP [10], as there is no per-flow information stored and no per-flow packet scheduling. Therefore, it scales very well.

The RMD framework defines two different PHR groups:

- **The Reservation-based PHR** group enables dynamic resource reservation per PHB in each node in the communication path. All the nodes maintain one state per PHB and no per-flow information. The reservation is done in terms of resource units, which may be based on a single parameter, such as bandwidth, or on more sophisticated parameters.
- **The Measurement-based Admission Control (MBAC) PHR** group is defined such that the availability of resources is checked by means of measurements before any reservation requests are admitted, without maintaining any reservation state in the nodes in the communication path. These measurements are done on the average real traffic (user) data load.

Only one PHR protocol has been specified thus far, the RMD On-demand (RODA) PHR [14] protocol. RODA is a reservation-based unicast edge-to-edge protocol designed for a single DiffServ domain, aiming at extreme simplicity and low cost of implementation along with good scaling properties. The RODA PHR protocol is implemented on hop-by-hop basis on all the nodes in a single Diffserv domain. The resource reservation request signaled by RODA is based on a resource unit. A resource unit is a bandwidth parameter that must be reserved by all nodes in the communication path between ingress and egress. The edge nodes, i.e., ingress and egress, of the Diffserv domain use certain message types to request and maintain the reserved resources for the flows going through the Diffserv domain. Each flow can occupy a certain number of resource units assigned to a particular Diffserv class.

The RODA PHR protocol implements all or a subset of the following functions:

- Admission control and/or resource reservation within a node;
- Management of one reservation state per PHB by using a combination of the reservation soft state and explicit release principles;
- Stores a pre-configured threshold value on maximum allowable resource units per PHB;
- Adaptation to load sharing. Load sharing allows interior nodes to take advantage of multiple routes to the same destination by sending via some or all of these available routes. The RODA PHR protocol has to adapt to load sharing when it is used;
- Severe congestion notification. This situation occurs as a result of route changes or a link failure. The PHR has to notify the edges when this occurs;
- Transport of transparent PDR messages. The PHR protocol may encapsulate and transport PDR messages from an ingress node to an egress node.

The sets of functions performed by the RODA protocol are described in [13], [14].

RMD Signaling Messages

The RMD signaling messages are categorized into RODA PHR and PDR protocol messages. These signaling messages and their description are depicted in Table 1.

The PDR signaling messages such as the “PDR_Reservation_Request”, “PDR_Refresh_Request” or “PDR_Release_Request” messages may be encapsulated into a RODA PHR message or sent as separate messages. The PDR messages encapsulated into RODA PHR messages will contain the information that is required by the egress node to associate this RODA PHR signaling message with, for example, the PDR flow ID and/or the IP address of the ingress node.

Table 1. RODA PHR and PDR signaling messages

Protocol	Signaling Message	Signaling Message Description
RODA PHR	**"PHR_Resource_Request"**	Initiate the PHB reservation state on all nodes located on the communication path between the ingress and egress nodes according to the external reservation request.
	"PHR_Refresh_Update"	Refreshes the PHB reservation soft state on all nodes located on the communication path between the ingress and egress nodes according to the resource reservation request that was successfully processed by the PHR functionality during a previous refresh period. If this reservation state does not receive a "PHR_Refresh_Update" message within a refresh period, reserved resources associated to this PHR message will be released automatically.
	"PHR_Release_Request"	Explicitly releases the reserved resources for a particular flow from a PHB reservation state. Any node that receives this message will release the requested resources associated with it, by subtracting the amount of PHR requested resources from the total reserved amount of resources stored in the PHB reservation state
PDR	**“PDR_Reservation_Request”**	Initiates or updates the PDR state in the egress. It is generated by ingress node.
	“PDR_Refresh_Request”	Refreshes the PDR states located in the egress. It is generated by the ingress node.
	“PDR_Release_Request”	Explicitly release the PDR state. It is generated by the ingress node. Applied only when the PDR state does not use a reservation soft state principle.
	“PDR_Reservation_Report"	Reports that a "PHR_Resource_Request"/"PDR_Reservation_Request" has been received and that the request has been admitted or rejected. ”. It is sent by the egress to the ingress node.
	“PDR_Refresh_Report”	Reports that a "PHR_Refresh_Update"/"PDR_Refresh_Request" message has been received and has been processed. It is sent by the egress to the ingress node.
	“PDR_Congestion_Report”	Used for severe congestion notification and is sent by egress to ingress. These PDR report messages are only used when either the "greedy marking" or "proportional marking" severe congestion notification procedures are used.
	“PDR_Request_Info”	Contains the information that is required by the egress node to associate the PHR signaling message that encapsulated this PDR message to for example the PDR flow ID and/or the IP address of the ingress node. It is generated by the ingress node.

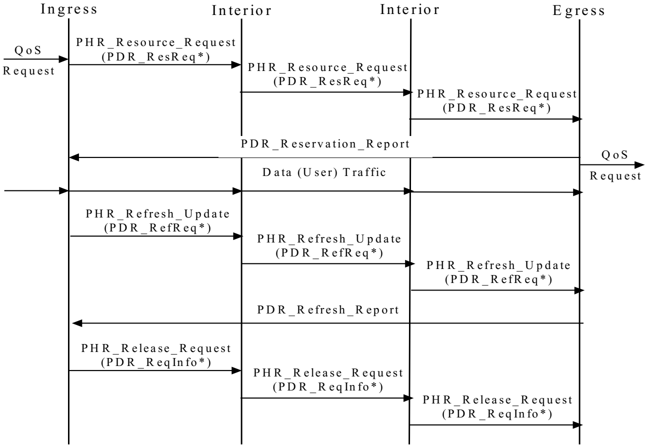
RMD Functional Operation

The functional operation of the RMD framework given here is based on the interoperation between the RODA PHR and PDR protocol functions, assuming that PDR is a newly defined protocol. Two illustrative functional operation examples are presented:

- Normal Operation that describes the scenario for successful reservation between edges of a Diffserv domain
- Fault handling that describes the severe congestion handling between edges of a Diffserv domain

Normal Operation

When an external “QoS Request” arrives at the ingress node (see Fig. 1), the PDR protocol, after classifying it into the appropriate PHB, will calculate the requested resource unit and create the PDR state. The PDR state will be associated with a flow specification ID. If the request is satisfied locally, then the ingress node will generate the “PHR_Resource_Request” and the “PDR_Reservation_Request” signaling message, which will be encapsulated in the “PHR_Resource_Request” signaling message. This PDR signaling message may contain information such as the IP address of the ingress node and the per-flow specification ID. This message will be decapsulated and processed by the egress node only.



* PDR_ResReq, PDR_RefReq are the encapsulated PDR messages
PDR_ResReq: PDR_Reservation_Request;
PDR_RefReq: PDR_Refresh_Request
PDR_ReqInfo: PDR_Request_Info

Fig. 1. RMD functional operation for a successful reservation

The intermediate interior nodes receiving the “PHR_Resource_Request” must identify the Diffserv class PHB (the DSCP type of the PHR signaling message) and, if possible, reserve the requested resources. The node reserves the requested resources by adding the requested amount to the total amount of reserved resources for that Diffserv class PHB.

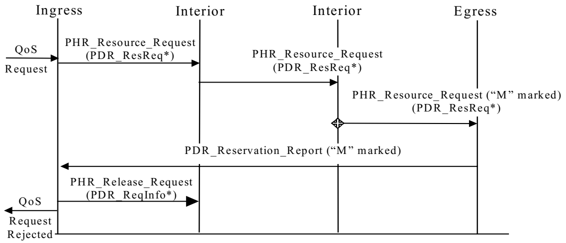
The egress node, after processing the “PHR_Resource_Request” message, decapsulates the PDR signaling message and creates/identifies the flow specification ID and the state associated with it. In order to report the successful reservation to the ingress node, the egress node will send the “PDR_Reservation_Report” back to the ingress node. After receiving this report message, the ingress node will inform the external source of the successful reservation, which will in turn send traffic (user data).

If the reserved resources need to be refreshed (updated), the ingress node will generate a “PHR_Refresh_Update” to refresh the RODA PHR state and

"PDR_Refresh_Request" message to refresh the PDR soft state in the egress node. The PDR refresh message will be encapsulated in the "PHR_Refresh_Update".

Apart from the soft state principle, the reserved resources in any node can also be released explicitly by means of explicit release signaling messages. In this case, the ingress node will create a "PHR_Release_Request" message, and it will include the amount of the PHR requested resources to be released. This message will also encapsulate a PDR information message. The amount of the resources to be released will be subtracted from the total amount of reserved resources in the RODA PHR state.

If there were no resources available in one of the interior nodes (see Fig.2), the "PHR_Resource_Request" will be "M" marked and, as a result, the reservation request will be rejected. The ingress node will be notified of the lack of the resources by means of the "M" marked "PDR_Reservation_Report" message. Furthermore, in addition to marking the "PHR_Resource_Request" message, the interior node will also include the number of the interior nodes that successfully processed this message. This number can be derived from the TTL value in the IP header of the packet received. Using this information the ingress node will generate a "PHR_Release_Request" that will release the resources in the interior nodes that have reserved PHR resources for the rejected resource reservation request. In this way, the degradation of link utilization will be minimized.



* PDR_ResReq is the encapsulated PDR message
PDR_ResReq: PDR_Reservation_Request;
PDR_Refreq: PDR_Refresh_Request
PDR_ReqInfo: PDR_Request_Info

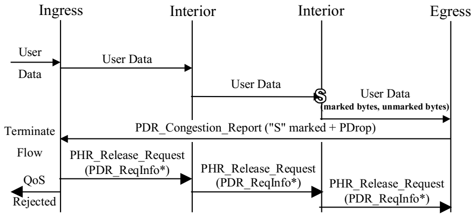
Fig. 2. RMD functional operation for a failed reservation

Fault Handling

Fault handling functional operation refers to handling undesired events in the network, such as a route changes, link failures, etc. This can lead to the loss of PHR signaling messages or to severe congestion.

Severe Congestion Handling

Routing algorithms in networks will adapt to severe congestion by changing the routing decisions to reflect changes in the topology and traffic volume. As a result the re-routed traffic will follow a new path, which may result in overloaded nodes as they need to support more traffic than their capacity allows. This is a severe congestion occurrence in the communication path, and interior nodes need to send notifications to the ingress nodes by means of PHR and PDR signaling messages. The ingress node has to resolve this situation by using a predefined policy. The interior node first detects the severe congestion occurrence, after which it will first notify the egress node and subsequently the ingress node. One can think of various detection and notification methods for the interior nodes, such as marking of all the data packets passing through a severe congested node or marking the PHR signaling messages only. In this paper only one method is considered. This is the “proportional marking” method, where the number of the remarked packets is proportional to the detected overload. The severely congested interior node will remark the user data packets with a domain specific DSCP (see [4]), proportionally to the detected overload. Once the marked packets arrive at the egress node, the egress node will generate a “PDR_Congestion_Report” message that will be sent to the ingress node. This message will contain the over-allocation volume of the flow in question, e.g., a blocking probability.



* PDR_ResReq is the encapsulated PDR message
PDR_ResReq: PDR_Reservation_Request;
PDR_Refreq: PDR_Refresh_Request
PDR_ReqInfo: PDR_Request_Info

Fig. 3. RMD functional operation in case of severe congestion

For each flow ID, the egress node will count the number of marked bytes and the number of unmarked bytes, and it will calculate the blocking probability using the formula (1):

$$P_{\text{drop}} = \frac{\mathrm{B}_{\mathrm{m}}}{\mathrm{B}_{\mathrm{m}} + \mathrm{B}_{\mathrm{u}}} \qquad \textbf{(1)}$$

where $\mathrm{B}_{\mathrm{m}=}$ number of marked bytes and $\mathrm{B}_{\mathrm{u}=}$ number of unmarked bytes.

The ingress node, based on this blocking probability, might terminate the flow. That is, for a higher blocking probability there is a higher chance that the flow will be terminated. If a flow needs to be terminated, the ingress node will generate a “PHR_Release_Request” message for this flow.

RMD Performance Evaluation

The performance behavior of the RMD framework has been studied through performance evaluation. For this evaluation, simulation and performance measurement techniques were used. These experiments focus on the two main RMD operation scenarios, the normal operation and the fault handling, described in Sections 0, 0 respectively.

Normal Operation

The normal operation of RMD (see Section 0) was simulated using a network simulator (ns) [20] environment. The goal was to study the RODA PHR protocol efficiency on bandwidth utilization. The experiments were performed on the RODA PHR both with and without using explicit release signaling messages ("PHR_Resource_Release" message - see Section 0). The RMD normal operation with the explicit release of resources is depicted in Fig.1, while the RMD normal operation without explicit release of resources is described in [13][14]. In the case of the RODA PHR without explicit release, the resources will be released based on the soft state principle after a refresh timeout. This has an impact on resource utilization. In order to improve the resource utilization, a sliding window is used. A sliding window enables faster release of the unused reservations compared to the refresh timeout. This algorithm is explained in detail in [13][14].

This section presents the simulation model, the performance experiments and their results for RMD normal operation, for both types of RODA PHR, with and without explicit release.

Simulation Model

The network topology used for performance evaluation of RMD normal operation is shown in Fig. 4.. The link capacities, i.e. bandwidth, between the routers are set to C, 2xC and 3xC, where C=620 units. A single resource unit is set to 2000 bytes/second rate allocation. This particular value was chosen as it represent the rate required by an encoded voice communication, e.g., GSM coding. There are 3 sources generating the same type of traffic to a single destination. Note that in this model there was only signaling traffic generated, thus there was no actual data load. As all three sources generate the same type of traffic, only one traffic class is used. That is, the resource requests are related to a single DSCP.

The resource requests ("PHR_Resource_Request"s – see Fig. 4) are generated according to Poisson process, with intensity of 1.356 requests/second. The requests holding times are distributed exponentially with mean set to $\frac{1}{\mu} = 90$ seconds. The resource unit request is distributed evenly between 1 and 20 units. With these traffic and bandwidth parameters the theoretical request blocking probability is 50%.

The length of the soft state refresh period is set to 30sec. Using the sliding window (see [15], [17]), in case of RODA PHR without explicit release, the length of the refresh timeout can be decreased on average, from 1.5 times to (1 + 1/N) times of the refresh period, where N denotes the number of cells in the sliding window. In the simulation model, the number of cells in a refresh window is set to 30, i.e. N = 30.

The expected output of the performance experiments with this simulation model were the results on link reservation and link utilization. The link reservation represents the number of the reserved resources per traffic class, i.e. Diffserv class on a link. The link utilization represents the amount of resource that can be actually used by traffic sources. For example, the link 2-3 (see Fig. 4) utilization is the sum of the resources that can be used by source 0, 1 and 2.

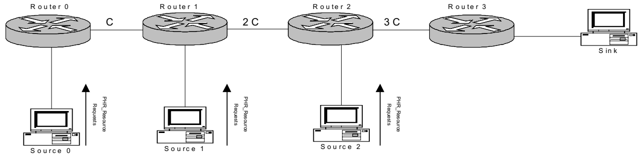


Fig. 4. Network simulation layout

Numerical Evaluation

Fig.5 and Fig. 6 present the reservation and utilization between routers 0 and 1 using the RODA PHR with and without explicit resource release. The [%] in the Fig.5 and Fig. 6 is the percentage of maximum reserved units available that are reserved/utilized. The maximum number of units is 620 on the first link, 2*620 on the second link, and 3*620 on the third link.

It can be seen that link reservation in both cases is slightly below 100 percent of link capacity, 94% and 91% respectively. However, the link utilization is improved by an average of 18% (from 62% up to 80%) when the explicit release messages are used when compared to the situation where there is no explicit release of resources. This is because, in the latter case, the resources are released based on a sliding window in proportion to the refresh period, while explicit release of resources is done in the order of roundtrip times of signaling messages. This round trip time is of course shorter than the length of the refresh period.

Fig.7 and Fig.8 show the results obtained for the link between router 2 and 3. Here again, link reservation is around 100 percent (97% and 98%) and link utilization is improved from 83% to 97% due to explicit resource release.

The average link utilization is higher on link between routers 2 and 3 than on the link between routers 0 and 1, because there are no downstream routers to reject a request already accepted by router 3. This is, however, not the case for the link between routers 0 and 1 as, for example, router 2 may reject resource request admitted by router 1. For the RODA PHR without explicit release of resources, it gets worse as the resources are kept reserved on average for the length of an additional 1.5 refresh periods. This means that the resource requests admitted by router 0 but rejected by routers 1 or 2 install reservation state for the length of 1.5 refresh periods, on average, even though the resource request is ultimately rejected.

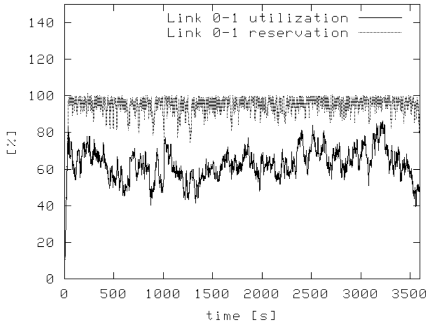


Fig. 5. Link utilization and reservation on link Router0-Router1 (RODA without release)

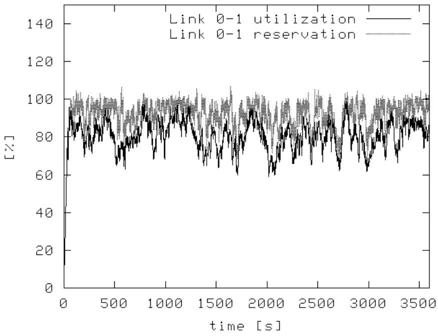


Fig. 6. Link utilization and reservation on link Router0-Router1 (RODA with release)

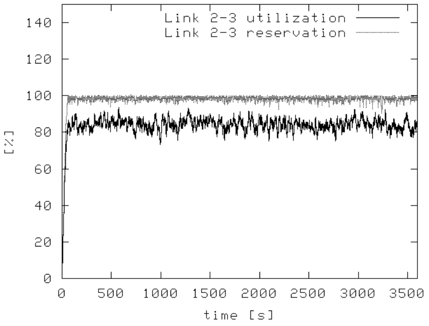


Fig. 7. Link utilization and reservation on link Router2-Router3 (RODA without release)

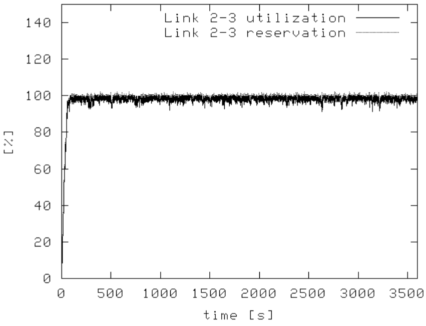


Fig. 8. Link utilization and reservation on link Router2-Router3 (RODA with release)

Performance Measurements

This section presents the performance measurements that were used to estimate the mean processing delays of the RODA PHR signaling messages. In order to compare the mean processing delays of the RODA PHR signaling messages with the mean processing delays of standardized IETF resource reservation protocols, performance measurements on the mean processing delays of the RSVP signaling messages are also done. For a 'fair' comparison between the two protocols, all experiments were performed using the same hardware.

The measurement topology consisted of two edge nodes interconnected by one interior node. The edge and interior nodes were PCs with 400Mhz Intel Pentium-II processors and 64 MBytes of RAM (Random Access Memory) running the Linux operating system. Furthermore, all nodes were running a preliminary prototype implementation of the RODA PHR protocol and a public implementation of the RSVP protocol [22]. Note that this preliminary prototype implementation of the

RODA PHR protocol does not support the processing of the PHR_Resource_Release message.

The ingress node sends either RODA PHR or RSVP signaling messages to the egress node in order to reserve and/or refresh resources for different flows in the interior node. During the performance measurements only one flow (session) was running on the interior node.

Fig. 9 depicts the measurement points located within the interior node (see also [21].). The measured processing delays are the processing delays in the (unloaded) interior node of the RODA PHR and RSVP protocol messages, excluding the delays necessary for the forwarding of these protocol messages through the IP protocol stack available in the node. The traffic is forwarded through the IP packet forwarder and Network Interface Card (see Fig. 9). The measured mean processing delay of the traffic forwarder is 12 μsec.

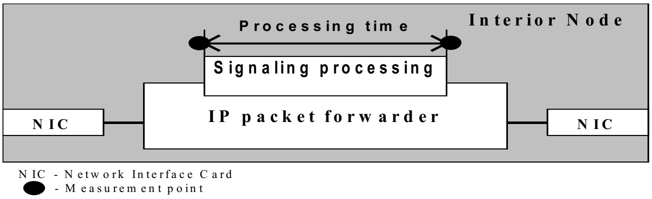


Fig. 9. Interior Node Measurement Points

The mean processing delays and their 95% confidence intervals of the RODA PHR and RSVP RESV signaling messages are listed in Table 2.

RSVP, which is a receiver-initiated reservation protocol, uses the RSVP PATH message to store backward routing information. This is used by the associated RSVP RESV message traveling back to the sender. The RMD protocol, which is sender-initiated, has no corresponding message. Therefore, the mean processing delay of the RSVP PATH message is not shown Table 2. However, its mean processing delay has been measured and is 273 [μsec].

Table 2. Mean processing delays and their 95% confidence intervals

Protocol Messages	RODA PHR processing delay (μsec)	RSVP processing delay (μsec)
Reservation message	PHR_Resource_Request 0.58 [0.0 ; 2.0]	RESV 790 [762.8 ; 793.2]
Refresh message	PHR_Refresh_Update 0.5 [0.0 ; 2.0]	RESV 67.02 [64.03;69.23]

From Table 2, it can be seen that the mean processing delays of the RODA PHR reservation messages are approximately 1338 times 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.

Fault Handling

This section presents the performance behavior of the severe congestion handling described in Section 0. For this purpose, the RMD was simulated using the network simulator (ns) [20] environment. This section describes the simulation model used, the experiments performed and the results.

Simulation Model

Based on the operational description of the RMD protocol, resources are requested in bandwidth units, which are also used for the description of the traffic models. As in Section 0, a single resource unit was set to represent 2000 bytes/second rate allocation. There were three different scenarios examined:

i) resource requests for only 1 unit
ii) resource requests from {1,2,…,20} units
iii) resource requests selected from {1,2,…,100} units.

The resource requests generation model is a Poisson process. The average resource request call holding time was set to $\frac{1}{\mu} = 90$ seconds.

The resource requests were generated in such a way that the requested bandwidth for each reservation unit class was balanced, i.e.,

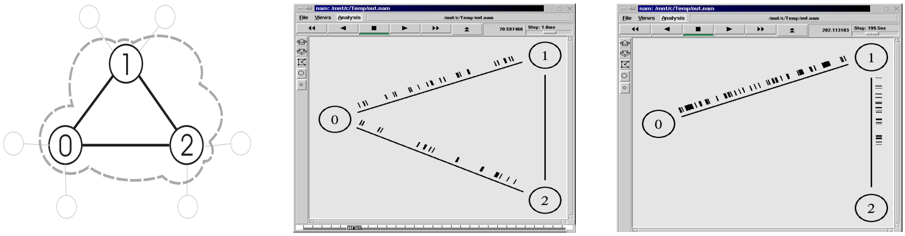
$$\frac{\lambda_i}{\mu} BW_i = \frac{\lambda_j}{\mu} BW_j \qquad \textbf{(2)}$$

where $BW_i = i$ U [unit] is the bandwidth request, U = 2000 bytes/sec and $\frac{1}{\lambda_i} = 0.9$ sec as per default. Hence, higher bandwidth requests arrived less frequently than smaller ones. Packet sizes (L) of the connections were determined according to their reservations:

$$L_i = L_1 * BW_i \qquad \textbf{(3)}$$

where, L_1 = 40 bytes, is the packet size for a single unit connection BW_1 = 1*U (assuming the packet inter-arrival time remains constant, which is L_i/BW_i = const)
For the sake of simplicity, packet inter-arrival times were kept constant (constant bit rate - CBR), hence every flow sent one single packet in each 20 msec interval.

For the evaluation of the methods, a simple delta network topology (see Fig. 10) was used.



a) domain b) normal c) link failed

Fig. 10. Network Layouts

For the sake of simplicity, node 0, 1 and 2 represent the Diffserv domain using the RMD protocols. These nodes act as interior nodes as well as network edge nodes whenever necessary. In order to have effective multiplexing of flows, the bandwidth of the links (*C*) (see (4)) was set to be able to accommodate at least 100 flows of the highest bandwidth demands, i.e., to 100, 2000 and 100000 units.

$$C = 100 * BW_{maxi} \qquad \textbf{(4)}$$

As discussed in Section 0, the severe congestion detection can be based on packet drop ratio measurements. Hence, it was important to find the proper dimensioning for the network buffers. As this traffic model was based on CBR traffic with 20 msec packet inter-arrival times, the queue lengths were set such that no packet loss can occur during normal operation. The buffer sizes (*B*) were determined using the following formula:

$$B = C * \frac{L_i}{BW_i} * C_{threshold} \qquad \textbf{(5)}$$

where, $C_{threshold}$ is the amount of maximum amount of resources available for a single traffic class.

The dimensioning was done for a target load level of 80% link capacity, hence buffer size is set to *B*=*C**0.02*0.8 [bytes], assuming the 20 msec packet inter-arrival time.

In order to model a failure event in the network in **Fig. 10**, after the system achieves stability, the link between nodes 0 and 1 goes down at 350 sec of simulation time. Afterwards, the dynamic routing protocol (OSPF) updates its routing table at 352.0 sec and all flows previously taking the 0-2 path will be re-routed to the 0-1-2 alternate path. Due to this, node 0 will suffer a severe overload resulting from the re-route event.

Numerical Evaluation

The detailed simulation results obtained for severe congestion handling described in Section 0 are presented below.

After detecting the severe congestion situation, the severe congestion handling algorithm immediately drops some of the flows in proportion to the detected overload. Thereby, the load of the link between nodes 0 and 1 almost instantly returns to the target load level of 80 percent. (see Fig. 11). From Fig. 12 one can see the short-term transients caused after severe congestion.

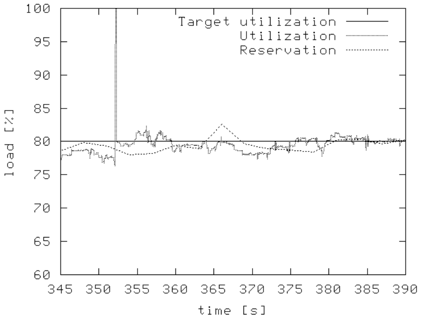


Fig. 11. Utilization and reservation after severe congestion

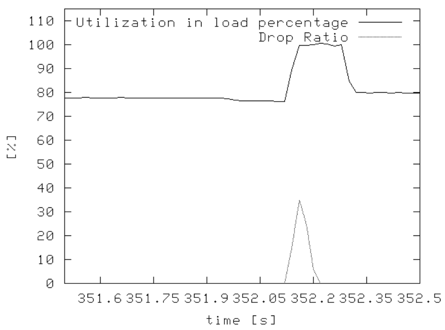


Fig. 12. Short term transients in packet drops and utilization during severe congestion

It is a natural side effect that during congestion the highly saturated queues will induce shifted round trip times for the connections (see Fig. 13). It can be seen that it takes about 2 seconds for the dynamic routing protocol[1] to update its link state and re-direct the affected connection. It can also be seen that the retained round trip time doubles for connections now traversing along 2 hops (every link's delay is 5 msec). Note that these times are extremely low (under 26 msec). That is, they correspond to one or two voice frames, which is quite acceptable.

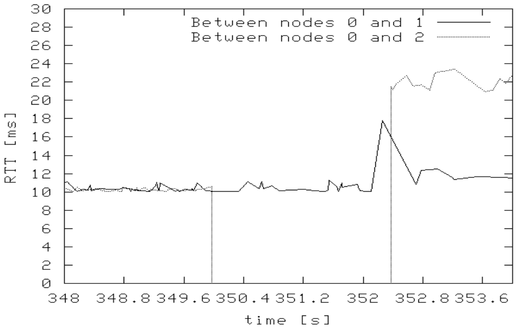


Fig. 13. The transient in Round Trip Times

Conclusions

In this paper an overview of the RMD functionality and its performance behavior is given. RMD is a simple, fast, low-cost and scalable dynamic resource reservation scheme. As such, it enhances the already scalable Diffserv architecture with dynamic

[1] We added the Open Shortest Path First (OSPF) routing extension to ns.

resource reservation and admission control. In order to observe the overall RMD functionality behavior and to prove the RMD concepts, extensive performance evaluation experiments have been performed.

When operating under normal conditions, the results show that network utilization is very close to maximum achievable even though the protocol does not use per-flow state maintenance in core routers. The performance measurement experiments show that the mean processing delays of RODA PHR reservation messages are more than 1338 times smaller than the mean processing delays of RSVP reservation messages. The performance evaluation of the severe congestion procedure has shown that the RMD reaction and recovery time on such events is negligible. Furthermore, using the RMD concepts, the utilization of the links is only slightly affected by severe congestion occurrence.

The performance behavior of RMD will be further investigated by experimenting with network topologies consisting of a larger number of nodes. Moreover, comparison with other types of RSVP implementation distributions such as e.g. KOM RSVP engine [23] could be accomplished.

Also there are still open issues with RMD that need to be studied, such as for example, extending the RMD applicability in a multi-domain, extending RMD with policy control and the Measurement-based PHR that still needs to be specified and evaluated.

Acknowledgements

Special thanks to our colleagues Simon Oosthoek, Pontus Wallentin, Martin Jacobsson, Marcel de Kogel, Geert Heijenk and Gábor Fodor for their useful inputs and comments.

References

1. Braden, R., Clark, D., Shenker, S., “Integrated Services in the Internet Architecture: an Overview”, IETF RFC-1633, Jun. 1994
2. Wroclawski, J., “ The use of RSVP with IETF integrated Services”, IETF RFC 2210, 1997.
3. Blake, S., Black, D., Carlson, M., Davies, E., Wang, Zh., Weiss, W., “An Architecture for Differentiated Services”, IETF RFC 2475, 1998.
4. Nichols, K., Blake, S., Baker, F. and D. Black, “Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers”, RFC 2474, December 1998.
5. Preliminary WWW site of the IETF NSIS WG located at: http://www1.ietf.org/mailman/listinfo/nsis
6. Braden, R., Zhang, L., Berson, S., Herzog, A., Jamin, S., “Resource ReSerVation Protocol (RSVP)-- Version 1 Functional Specification”, IETF RFC 2205, 1997.
7. Baker, F., Iturralde, C. Le Faucher, F., Davie, B., “Aggregation of RSVP for IPv4 and IPv6 Reservations”, IETF RFC 3175, 2001.
8. Bergkvist, J., Cselényi, I., Ahlard, D., “Boomerang – A simple Resource Reservation Framework for IP”, Internet Draft, draft-bergkvist-boomerang-framework-01.txt, Work in Progress, November 2000.

9. Pan, P., Schulzrinne, H., “YESSIR: A Simple Reservation Mechanism for the Internet”, Proceedings NOSSDAV'98, 1998.
10. Chow, H., Leon-Garcia, A., “A Feedback Control Extension to Differentiated Services”, Internet Draft, draft-chow-diffserv-fbctrl-01.txt, Work in Progress, March 1999.
11. Stoica, I., Zhang, H., Shenker, S., Yavatkar, R., Stephens, D., Malis, A., Bernet, Y., Wang, Z., Baker, F., Wroclawski, J., Song, C., Wilder, R., “Per Hop Behaviours Based on Dynamic Packet States”, Internet Draft draft-stoica-diffserv-dps-00.txt, Work in Progress, February 1999.
12. White, P.P., Crowcroft, J., “A Dynamic Sender-Initiated Reservation Protocol for the Internet”, Proceedings, HPN’98, 1998.
13. Westberg, L., Jacobsson, M., Karagiannis, G., Oosthoek, S., Partain, D., Rexhepi, V., Szabo, R., Wallentin, P., "Resource Management in Diffserv Framework", Internet Draft, Work in Progress, 2001.
14. Westberg, L., Karagiannis, G., Partain, D., Oosthoek, S., Jacobsson, M., Rexhepi, V., "Resource Management in Diffserv On DemAnd (RODA) PHR", Internet Draft, Work in progress.
15. Jacobsson, M., “Resource Management in Differentiated Services – A Prototype Implementation”, M.Sc. Thesis, Computer Science/TSS, University of Twente, June 2001.
16. Heijenk, G., Karagiannis, G., Rexhepi, Westberg, L., "DiffServ Resource Management in IP-based Radio Access Networks", Wireless Personal Multimedia Communications (WPMC'01), Aalborg, Denmark, September 2001.
17. Marquetant, A., Pop, O., Szabo, R., Dinnyes, G., Turanyi, Z., "Novel enhancements to load control a soft state, lightweight admission control protocol", QofIS'2000 - 2nd International Workshop on Quality of future Internet Services, September 2001.
18. Csaszar, A., Takacs, A., Szabo, R., Rexhepi, V., Karagiannis, G., "Severe Congestion Handling with Resource Management in Diffserv On Demand", submitted to Networking 2002, May 19-24 2002, Pisa - Italy.
19. Partain, D., Karagiannis, G., Westberg, L., "Resource Reservation Issues in Cellular Access Networks", Internet Draft, Work in progress.
20. The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/
21. Feher, G., Korn, A., "Performance Profiling of Resource Reservation Protocols", IFIP 2001, 27th - 29th June, 2001,Budapest- Hungary.
22. ISI public source code for the RSVP protocol, located at (www.isi.edu/div7/rsvp/)
23. KOM RSVP engine located at (http://www.kom.e-technik.tu-darmstadt.de/rsvp/)