Advanced Quality-of-Service Signaling for IP Multicast

Roland Bless and Martin Röhricht
Institute of Telematics
Karlsruhe Institute of Technology (KIT)
P.O. Box 6980, D-76049 Karlsruhe, Germany
Email: {bless, roehricht}@kit.edu

***Abstract*—Supporting Quality-of-Service resource reservations for IP multicast flows is especially advantageous for distributed multimedia applications like video conferencing, 3D tele-immersion, or multi-player online gaming. In response to various limitations of RSVP the IETF developed more flexible signaling protocols within the Next Steps in Signaling (NSIS) framework. But unlike RSVP, the NSIS protocols were designed to consider unicast flows only in order to reduce protocol complexity. This paper presents an extension of the NSIS signaling protocols that allow for QoS resource reservations of IP multicast data flows. We describe the main challenges and discuss the resulting design decisions in detail. Enhancements of an existing NSIS implementation show that the required changes are minimal and do neither affect the unicast protocol operation, nor increase the protocol's complexity significantly. Instead, all of the advanced features introduced by NSIS, like reliable signaling message transport or support for sender- and receiver-initiated reservations can also be used with IP multicast flows. Evaluation results confirm that the overhead introduced by supporting IP multicast in NSIS compared to unicast reservations is negligible and that the presented solutions also offers scalable sender-initiated reservations.**

I. Introduction

Examples of upcoming interactive multimedia applications are 3D tele-immersion environments, tele-presence environments as well as multi-player gaming spaces [1]. Such environments combine new 3D sensory, new actuating devices and high-definition displays with very high resolutions. These application's demands for a very high bandwidth and real-time interactivity are therefore calling for an efficient transfer of high volume streams across the Internet. Since IP multicast [2] is often the only way of transporting such high bandwidth streams across larger networks and in order to fully unleash the power of such distributed interactive multimedia environments, quality-of-service guarantees for IP multicast should be employed.

Twenty years ago, early QoS research already considered multimedia group communication applications in the Internet and came up with solutions like the Resource ReSerVation Protocol (RSVP) [3] with its Integrated Services architecture. RSVP offers receiver-initiated reservations and provides inherent support for IP multicast. Its shortcomings however, like its missing support for mobile end-users or support for sender-initiated reservations, led to the creation of the *Next Steps in Signaling* (NSIS) working group in the IETF. The recently finished NSIS signaling framework [4] provides also an advanced QoS signaling solution covering mobility support, reliable and/or secure messaging transport, transfer of large messages as well as sender- and receiver-initiated reservations.

One main objective of the NSIS framework was to support further signaling applications besides QoS signaling and to provide an extensible architecture. Therefore, it is based on a two-layered architecture (cf. Figure 1), separating signaling applications from signaling message transport and routing.

Unlike RSVP, the NSIS protocols considered unicast flows only since it was felt that unicast applications prevail and multicast may add unnecessary complexity, so it should not be the main focus of the design. The overall NSIS protocol design, however, does not preclude multicast support. When considering the earlier mentioned distributed interactive multimedia environments, we felt that support for multicast reservations should be added, providing all the advanced NSIS functions, too. For example, NSIS offers support for *scalable sender-initiated reservations*, whereas RSVP offers support for receiver-initiated reservations only.

In this paper we show how the NSIS protocols can be extended to support IP multicast by relying on the existing protocol functionality and its protocol data units. The results underline the extensibility and flexibility of the NSIS protocols. In Section II we give a short overview of relevant parts of the NSIS framework and related work. In Sections III and IV we provide an analysis of the GIST and QoS NSLP protocols regarding necessary adaptations for proper multicast support and explain design principles used in order to accomplish these adaptations. In Section V we provide an evaluation of the chosen approach before we conclude in Section VI.

II. Background and Related Work

The NSIS protocol suite was once designed in response to some shortcomings of the already existing RSVP protocol. Its architecture follows a two-layered approach as depicted in Figure 1. The NSIS *Transport* Layer Protocol (NTLP) is responsible for the transport and routing of signaling messages and is decoupled from the actual signaling application, which is realized by a dedicated NSIS *Signaling* Layer Protocol (NSLP).

978-1-4577-0103-0/11/$26.00

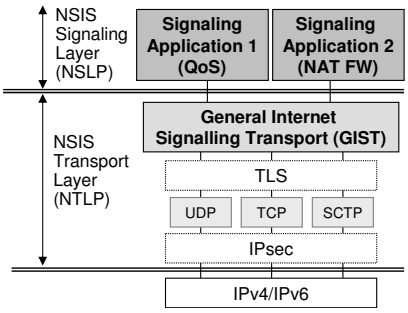

Figure 1. Architecture of the NSIS protocol suite

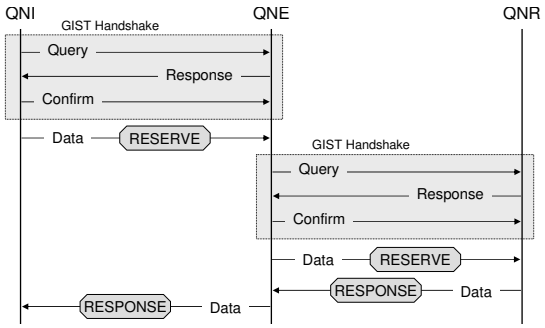

Figure 2. Protocol operation between three QoS NSLP capable nodes for sender-initiated reservations

The *General Internet Signaling Transport* (GIST) protocol [5] fulfills the requirements of an NTLP. It works on a hop-by-hop basis and uses already existing underlying transport protocols like UDP, SCTP, TCP or TCP with TLS. A connection between two adjacent peers using connection-oriented transport protocols is referred to as a *Messaging Association* (MA) and can be re-used by different flows, once established. GIST's default routing behavior for signaling messages, called *path-coupled signaling*, strictly follows the data path in the same manner as with RSVP.

GIST uses a three-way handshake, consisting of QUERY, RESPONSE, and CONFIRM messages, to set up signaling routing state between two peers (cf. Figure 2), before exchanging NSLP signaling application messages via DATA PDUs. During a handshake each peer is uniquely identified by a *Peer-ID* that forms together with the peer's network interface IP address and a routing state validity time the *Network Layer Information* (NLI) object. The *Source Identification Information* (SII) handle is a node internal number that can be used as referral to a specific signaling peer. This SII handle is locally exchanged between GIST and any given NSLP application and can be also used to detect a change of an adjacent peer, e.g., due to route changes.

Signaling for Quality-of-Service reservations is realized by a dedicated QoS NSLP [6]. This protocol was designed to install and modify state for resource reservations on intermediate nodes of a given path from end to end. Similar to RSVP, it follows a soft-state approach that uses refresh messages between adjacent peers. QoS NSLP abstracts from the actual QoS model that is to be used (e.g., IntServ or DiffServ) by carrying a separate object, called QSPEC, which encapsulates all QoS specific information. Unlike RSVP, it supports receiver- and sender-initiated reservations, reliable transfer, and transfer of large signaling messages. These features should also be available in the multicast case.

QoS NSLP mainly uses QUERY, RESERVE, RESPONSE, and NOTIFY PDUs. The first entity that issues a reservation request is called *QoS NSLP Initiator* (QNI), each intermediate node participating in the QoS NSLP session is denoted as *QoS NSLP Entity* (QNE) and the last node that receives the reservation request is called *QoS NSLP Responder* (QNR). By using a sender-initiated reservation, as depicted in Figure 2, the QNI issues a RESERVE to the QNR along the data path in downstream direction which is then followed by a RESPONSE from the QNR. In case of a receiver-initiated reservation the RESERVE messages travel in upstream direction, originating at the receiver of the data flow. Therefore, signaling message routing state must be installed on all intermediate QNEs by a prior QUERY in downstream direction upon which the QNI can issue a RESERVE in upstream direction, usually followed by a RESPONSE message.

As already mentioned above, RSVP can be seen as the predecessor of the NSIS protocols and is probably one of the most prominent QoS signaling protocols for IP-based networks. RFC 4094 [7] reviews some of the existing QoS signaling protocols and puts special emphasis on the RSVP operation and existing RSVP extensions. But even though there exist a wide variety of different QoS signaling protocols that have been developed in the past, only few of them provide support for IP multicast. For example, Boomerang [8] or INSIGNIA [9] were designed to support IP unicast only. Inter-domain signaling protocols, like BGRP [10], DARIS [11], or SICAP [12] work on a per-aggregate reservation granularity and do therefore not support IP multicast-based resource reservations. A comparison of different Internet QoS signaling protocols is provided by [13] with regard to different reservation initiation schemes, the reservation granularity, or the scope of the signaling operation.

One of the earliest signaling protocols for the Internet that does also provide support for IP multicast is ST-II [14]. However, ST-II does not follow a soft-state approach, does not provide any security mechanisms, and only supports sender-initiated reservations. YESSIR [15] is another prominent example of a resource reservation protocol that supports multicast by introducing individual and shared reservation styles. Despite the fact, that it adapts useful features from RSVP and can still be considered a light-weight and simple protocol, it is designed as an extension to the Real-Time Transport Control Protocol (RTCP) and therefore requires direct support of applications and support from the network routers with regard to RTCP.

III. Analysis and Design of Multicast Extensions for GIST

Due to the modular NSIS design, some of the multicast functionality needs to be placed into GIST and some into the QoS NSLP. The changes and enhancements for the GIST layer are described in this section, whereas Section IV describes the changes related to the QoS NSLP layer.

IP multicast routing protocols (e.g., PIM-SM, PIM-DM) establish a distribution tree along the data path in order to efficiently convey messages to a group of receivers. GIST's default signaling message transport was designed to use path-coupled signaling where signaling messages follow strictly the data path. Applying IP multicast to GIST signaling messages therefore implies that a GIST node may discover more than one peer in downstream direction for a single IP multicast flow. Consequently, a multicast-aware GIST node must be able to handle signaling messages and states for several peers in relation to a single (multicast) data flow.

Each NSIS signaling session starts with an initial GIST handshake where not only connection related parameters are negotiated, but also routing of signaling messages is performed by determining the next GIST node in downstream direction that participates in this signaling session. The scope of an analysis of how GIST must be extended to support IP multicast is therefore limited to the segment between two adjacent GIST nodes.

Any GIST handshake starts with a QUERY message emitted by a Querying Node (QN) towards a Responding Node (RN) thereby containing the signaling application's destination address in its IP header and the related *Message Routing Information* (MRI), which contains all relevant addressing information with respect to the data flow. In case of signaling for an IP multicast user data flow the GIST IP headers as well as the MRI contain a multicast destination address.

By using IP multicast GIST QUERY messages from a QN can reach multiple RNs basically in different ways. A multicast packet can be replicated within the network itself, either on link layer (consider several RNs attached to the same subnet) or at the IP layer via intermediate GIST-unaware multicast routers. GIST-aware multicast routers, however, have to actively replicate GIST multicast packets for all attached interfaces, because the NLI information in the QUERY message must reflect the related outgoing interface address.

In either way, a single emitted QUERY message will be received by more than one GIST RN and each of them replies with a corresponding individual RESPONSE. The QN must treat the latter separately for each individual RN and must confirm each RESPONSE by a dedicated CONFIRM if requested. A multicast-unaware GIST implementation would interpret the responses of the different peers as re-routing events and report them to the NSLP thereby overwriting the routing table information for the next GIST hop.

A further issue is that DATA messages must be also actively replicated per GIST peer, since GIST addresses peers directly via unicast and may also use different MAs for different peers, so IP multicast cannot be used for sending such signaling messages.

Moreover, IP multicast group memberships are dynamic, i.e., new peers can join or leave the group anytime. Thus a multicast-aware GIST must cope with dynamic group memberships and must provide means for detecting new peers as well as for removing state for peers that left the multicast group.

Another issue occurs at the QN since all RNs receive the QUERY at nearly the same time due to multicast replication, so they will respond nearly simultaneously. If the number of RNs is large, the QN may be overwhelmed by the number of RESPONSE messages.

In summary, we need to address the following problems:

- At the Querying Node
  - Replication of GIST QUERY messages
  - Handling reception of several RESPONSE messages from different peers
  - Maintaining signaling routing state for a set of peers
  - Replication of DATA messages
- At the Responding Node: avoid sender/QN flooding caused by simultaneous RESPONSE messages

In the following, we discuss the solutions to each of the problems.

Whenever an explicit replication is necessary on a branching GIST node, the intercepted QUERY message cannot simply be forwarded and replicated by IP multicast routing as mentioned earlier. Instead, all following QUERY messages must be slightly adapted for each of the interfaces due to the different NLI. Furthermore, similar to the unicast case, as soon as the GIST entity notices a *routing change*, a QUERY should be generated in order to update the GIST routing state. Such a routing change may be caused by a joining receiver (or branch) that creates a new multicast routing entry.

Adaptation 1— Replicate QUERY messages on multicast-aware GIST nodes

A branching GIST node is responsible to replicate incoming QUERY messages with destination multicast addresses according to the IP multicast routing table. The *Multicast Forwarding Cache* provides information about which outgoing network interface must be used. This data structure sets outgoing interfaces of a multicast path in relation to triples of incoming interfaces, source IP addresses, and multicast group addresses. The incoming interface is stored in GIST's routing table upon reception of the incoming QUERY (or in case of delayed-state installation upon the reception of the final CONFIRM). Only in case of a signaling session's very first originated QUERY an incoming interface is not available. In this case the information about the signaling path must be delivered by the signaling application or by choosing the default's kernel entry automatically. Once the outgoing interface is known, the following QUERY messages must reflect the corresponding interface address in their NLI object.

As outlined above, a replicated QUERY message may be received by multiple GIST RNs and each of these messages

causes an individual RESPONSE to be sent back to the QN. Therefore, all RESPONSE messages of a corresponding multicast flow must be processed individually by a QN.

Adaptation 2— Allow for reception of multiple RESPONSE messages upon the transmission of a single QUERY

The QN must collect and process all incoming RESPONSE messages from different peers. So after reception of the first valid RESPONSE the QN must accept further RESPONSE messages for a certain period of time and update the set of downstream GIST peers accordingly. The duration of this period has to be defined though. Each multicast peer should have the same time to answer a QUERY as a single peer in the unicast case. We consider that the querier state (cf. Figure 3) is *Established* if *at least one* valid RESPONSE was received. If no RESPONSE is received in *Awaiting Response* the QUERY will be retransmitted following an exponential back-off strategy as in the unicast case.

Adaptation 3— Manage signaling routing state for *a set of* peers

As mentioned earlier, GIST must be able to cope with group membership dynamics and it has to provide means for detecting routing changes as well as for removing state for peers that left the group. GIST already probes the network periodically to detect routing changes (so called 'GIST probing') that may have occurred in the meantime. A route change is detected by getting a response from a different peer (as indicated by the different NLI). As in the unicast case, detection of a new multicast peer generates a `NetworkNotification()` indication via the GIST API to the NSLP. This is necessary since the NSLP may have to take action upon discovering a new signaling peer.

GIST probing can also be used in case of multicast, where absent RESPONSE messages can be used as an indication that peers left the multicast group. An entry should then be removed from the set of multicast peers if refreshing GIST probes are not acknowledged by RESPONSE messages within a given time frame.

However, a missing RESPONSE can also result from packet loss in the network. To address this problem, the GIST specification provides a retransmission timer that uses a binary exponential backoff in order to retransmit QUERY messages that did not get a corresponding RESPONSE. The initial timeout T_1—as recommended by the specification—is set to $500\,\mathrm{ms}$ and can be increased up to a value of T_2 (with $T_2 = 64 \cdot T_1$ by default).

RESPONSE messages from a multicast peer should therefore be accepted at least until expiration of the initial timeout. In order to prevent a multicast peer being removed from the set too early, each multicast peer should get more than one chance to respond to a QUERY. The number n of opportunities per peer should be chosen in a way that the maximum response time T_2 is not exceeded by the sum of all opportunities. The $RoutingStateValidity$ time describes the time a GIST routing entry should be considered valid and is also an upper bound for emitting refreshing GIST probes. The number n of opportunities to respond can then be calculated with given times T_1, T_2, and $RoutingStateValidity$ to:

$$n = \left\lfloor \frac{T_2}{RoutingStateValidity + T_1} \right\rfloor$$

Once a multicast peer does not respond to this number n of QUERY messages, its entry should be removed from the set of active peers.

To properly maintain state about each multicast peer in downstream direction we defined a hash table that uses the NLI as the hash table's key. By following this approach a new entry is inserted in case an NLI is yet unknown, otherwise the existing table entry is updated.

RESPONSE messages of different peers can be distinguished by their NLI object that contains the peer ID and the IP address of the outgoing network interface. In order to prevent the *NoResponse* timer being stopped upon the reception of the first RESPONSE, we extended the Querying Node's GIST state machine by introducing a new action rule 9 on *to_No_Response*.

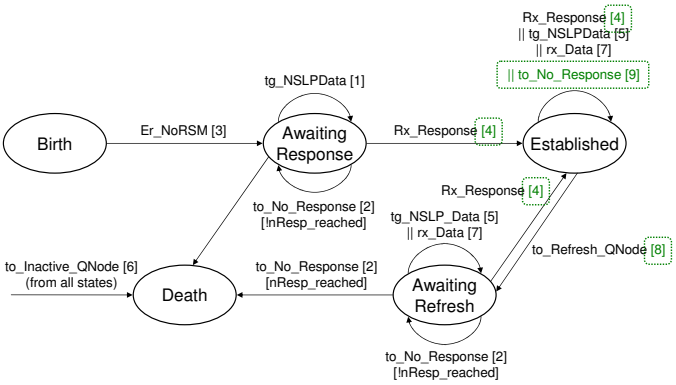

Figure 3. Extension of the Querying Node's GIST state machine to be multicast-aware

The differences of the Querying Node's state machine compared to the one specified in [5] are highlighted in Figure 3. The newly introduced action rule maintains the set of active multicast peers. The corresponding processing rule 9 therefore reads: Remove multicast peers with too many missing responses. Furthermore, rule 4 should be extended to reset the counter for missing responses in case of multicast signaling and rule 8 should be extended to increase the counter for missing responses for each multicast peer.

GIST's three-way handshake is also used to establish and negotiate MAs between two adjacent GIST peers. The use of multicast requires that a GIST node needs to maintain MA state for every single multicast GIST neighbor per multicast flow. As described above, RESPONSE messages that arrive later are treated as if corresponding CONFIRM messages got lost and are processed again. However, this should not lead to an update of a GIST peer's identity and its corresponding MA. Therefore, a GIST node must distinguish RESPONSE messages in case of multicast based on their NLI and must keep state of the MAs to be used for each of its GIST neighbors separately.

Adaptation 4— Maintain state for each single peer and its corresponding Messaging Association

Another consequence of this decision is that a multicast neighbor must receive a dedicated copy of all currently queued DATA messages, that have been passed to GIST by the signaling application since the initialization of the three-way handshake.

Once a GIST connection is established between adjacent peers the signaling application's DATA messages must be exchanged subsequently. Transmitting DATA messages via IP multicast is different from the unicast operation. First of all, the IP destination address of each single DATA message must use the IP address of this particular adjacent GIST peer and cannot simply use the group's multicast address. Furthermore, an MA may not be able to use multicast anyway. For these reasons GIST is also responsible for the replication and transmission of DATA messages towards each individual next peer.

By using IP multicast we must take care that all DATA messages are delivered to all group members even if a GIST connection to one peer is not yet fully established. That means, the corresponding queue for all DATA messages must not be emptied after the first successfully established connection to an adjacent peer, which is the default behavior in the unicast case. Furthermore, a QN should keep track about the DATA messages that were delivered to each multicast peer.

Adaptation 5— Maintain state about which signaling data packet was sent to which peer

DATA messages that are queued for transmission at a node should not be transmitted before the GIST handshake has been completed with every (new) peer. The above mentioned issues are related to Querying Nodes only. Due to the fact that a Responding Node has only one upstream peer—no matter if multicast or unicast communication is used—the QN's GIST protocol behavior is only affected by some minor changes.

Consider the QUERY messages sent by one QN are all received within a short fraction of time by multiple RNs. In case the Responding Nodes reply instantaneously, the subsequently sent RESPONSE messages are all targeted at a single QN and may therefore lead to a flood at the QN.

Thus, we infer:

Adaptation 6— Avoid flooding the sender caused by multiple simultaneous RESPONSE messages

In order to protect the QN from being flooded by too many simultaneously arriving RESPONSE messages, Responding Nodes should add a random delay before transmitting their RESPONSE. The delay should be chosen from a uniformly distributed interval that ranges from 0 to *MaxMulticastResponseDelay* in order to avoid any potential synchronization effects [16]. This maximum value should not exceed T_1, i.e., the retransmission timeout for the first QUERY message. In fact it should even be smaller than T_1 in order to take the transmission time into account. Therefore, the maximum response delay should be chosen from an interval as follows:

$$MaxMulticastResponseDelay = rand(0, \alpha \cdot T_1)$$

The value of α should be chosen to reflect the fraction of T_1, e.g., for a maximum response delay that should not exceed 80% of the retransmission timeout $\alpha = 0.8$. T_1 can actually vary due to an exponential back-off mechanism when QUERY retransmissions happen. But since an RN does not know whether it receives the first or a retransmitted QUERY, it should assume T_1 to be 500 ms, since this is the worst case limit for a responding node.

IV. ANALYSIS AND DESIGN OF MULTICAST EXTENSIONS FOR QOS NSLP

The Quality-of-Service NSIS Signaling Layer Protocol (QoS NSLP) was designed to support sender- and receiver-initiated reservations. In case of sender-initiated reservations, the sender of the data flow is also the initiator of a reservation (QoS NSLP Initiator, QNI) and the data flow receivers act as QoS NSLP Responders (QNR) as depicted in Figure 4.

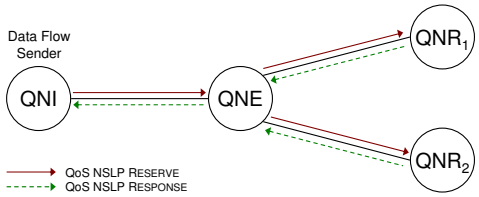

Figure 4. Sender-initiated reservation for a multicast flow

Whenever receiver-initiated reservations are used, the data flow receiver triggers reservations by QoS NSLP QUERY messages and acts as a QNR, whereas the data flow receivers act as QNIs by emitting the initial QoS NSLP RESERVE as shown in Figure 5.

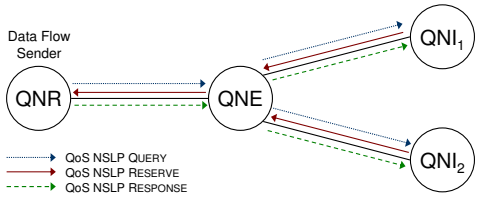

Figure 5. Receiver-initiated reservation for a multicast flow

In order to extend QoS NSLP to support IP multicast we assume an underlying multicast-aware GIST.

A. Identification of the last signaling hop

Unlike unicast addresses, a multicast address does not identify one particular host. Therefore, by using IP multicast, the destination address contained in the MRI can no longer be compared with locally configured IP addresses in order to let a host determine that it is actually the last signaling hop. This decision is, however, important as it affects all the nodes at the leaves of a multicast distribution tree. In case of

sender-initiated reservations these nodes must emit RESPONSE messages once they received a RESERVE; in case of receiver-initiated reservations they must process QUERY messages and send a RESERVE towards the data flow sender.

There exist different possibilities on how to solve this problem. For instance, the multicast routing table could be looked up for entries matching the data flow. In case no entry is found, this can be used as an indication that the data flow does not have to be forwarded and hence, we reached a signaling end point. Another option would be an explicit list of addresses specified by the signaling application that uses QoS NSLP.

We decided to introduce a configuration option that contains a list of local multicast addresses. In case of multicast signaling, a QoS NSLP instance can then compare the destination address contained in the MRI with the ones configured in the list and decide if it acts as a last signaling hop based on this comparison.

B. Rerouting in case of multicast

The NSIS protocols were designed to automatically detect re-routing of IP data flows in order to properly adapt the corresponding signaling flows. Whenever the underlying GIST transport protocol detects a new signaling neighbor for a particular signaling flow, the new neighbor's SII-Handle is passed to the NSLP instance. A QoS NSLP instance is then able to initiate a reservation towards the new peer and may also tear down a reservation on the old branch.

However, the situation is different in case multicast communication is used. A newly reported SII-Handle of a signaling flow does not necessarily correspond to a re-routing event in a multicast environment but can be the result of a recently joined peer. Hence, whereas a new reservation should still be actively initiated, existing reservations should not simply be torn down as in the unicast case once a new SII-Handle is reported by the underlying GIST instance, as this may negatively affect peers that are still actively participating in the signaling session.

Therefore, the QoS NSLP instance should maintain a list of all SII-Handles that were reported by the GIST instance. This list should then be used whenever GIST passes an SII-Handle to the QoS NSLP instance in order to decide whether a reservation was already established towards this peer or if a new reservation needs to be set up to this peer in which case its corresponding SII-Handle also needs to be added to the list.

C. Initial QUERY *or* RESERVE *messages for new neighbors*

IP multicast allows peers to join or leave a group anytime, even during an active signaling session. The periodically sent refreshing RESERVE messages of a QoS NSLP instance permit such peers to join the signaling session sooner or later. However, refreshing RESERVE messages are only emitted within a fixed negotiated time interval, which leads to an unnecessary delay for new multicast peers. A multicast-aware QoS NSLP should therefore avoid such delays and rather emit a RESERVE as soon as an SII-Handle of a new peer is reported by the underlying GIST instance; this can be also triggered by routing changes.

Note that, in case of sender-initiated reservations, the new peer does not yet have any information about the actual resource reservation request, because refreshing RESERVE messages do not have to carry QSPEC objects. Hence, the initial RESERVE must contain the complete information, i.e., it must carry a QSPEC object and a PACKET_CLASSIFIER object.

Moreover, an entire branch of new nodes may be located behind a new peer. Therefore, the RESERVE should not only reach the new peer, but also be forwarded by this peer if necessary. For this reason, the RESERVE should also include an RII object.

In case receiver-initiated reservations are used, a QUERY message must be sent towards the new peer in order to trigger a subsequent RESERVE. Just like the QUERY that was used to initially trigger the receiver-initiated reservation, this one must have the RESERVE-flag set, carry a QSPEC object, and possibly a PACKET_CLASSIFIER object.

D. Forwarding of Signaling Messages

In order to provide scalable (with respect to the group size) signaling mechanisms, the forwarding of signaling messages must be limited whenever possible. For instance, consider the case where RESPONSE messages are sent within a sender-initiated reservation. In this case the initial RESPONSE from one of the QNRs should be forwarded by a QNE towards the QNI, but all subsequently received RESPONSE messages from different QNRs should not be propagated any further as they do not contain any more relevant information. Finally, we note that a joining receiver limits the signaling messages to the newly grafted branch, especially the sender is not involved.

In case receiver-initiated reservations are used, this rule applies to RESERVE messages that are sent in upstream direction. However, RESERVE messages of a different QNI belonging to the same signaling session may be forwarded by a QNE if the resource reservation request in upstream direction can no longer be fulfilled and more resources must be possibly allocated.

Signaling messages in downstream direction should only be forwarded by branching nodes if they are considered useful for the entire multicast group. RESPONSE messages should therefore only be forwarded towards this particular peer that originally initiated the corresponding RESERVE. Nodes acting as QNRs, which generate a RESPONSE message directly in response to a preceding RESERVE can simply use the SII-Handle of the corresponding peer.

The situation becomes more difficult for peers acting as QNEs. Such peers need to check whether the RESERVE message contains an RII object. In this case the peer can definitely expect a RESPONSE to be returned, which then needs to be forwarded towards the peer that originated the RESERVE. In order to forward this RESPONSE correctly, a QNE should keep state about each RII object and the SII-Handle of the corresponding peer.

E. Merging of reservations belonging to different branches

By using receiver-initiated reservations, resource reservation requests of different branches arrive at branching nodes of the multicast tree. These branching nodes are therefore responsible to merge these independent reservations that were initiated by different QNIs. Figure 6 illustrates how reservations from QNI_1 and QNI_2 towards one QNR join at an intermediate QNE (without requiring a RESPONSE).

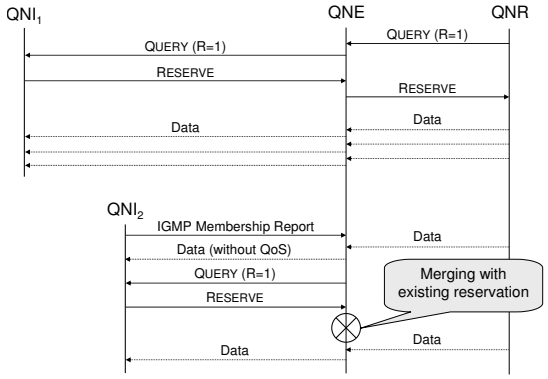

Figure 6. Merging of reservations belonging to different branches of a multicast flow by using receiver-initiated reservations

The reservations used by both QNIs are not synchronized regarding the *Reservation Sequence Numbers* (RSN). A multicast-aware QoS NSLP instance at the QNE should therefore keep state for each multicast peer's RSN.

Furthermore, we cannot assume the individually negotiated RESERVE's validity time to be synchronized between nodes belonging to different branches. As the QoS NSLP is responsible to tear down reservations that have not been refreshed in time, the QoS NSLP instance must maintain individual RESERVE validity timers for each of its multicast peers.

Note that merging different resource reservation requests in terms of allocating the right amount of resources is the responsibility of the *Resource Management Function* (RMF). In case the amount of reserved resources needs to be increased or decreased in upstream direction, the RMF triggers new RESERVE messages to be sent by the QoS NSLP instance in order to update the resource reservation along the path.

V. IMPLEMENTATION AND EVALUATION

In this section we discuss some important implementation details and evaluate the signaling performance of our proposed design. The implementation is based on our existing open source NSIS-ka suite [17] and runs under Linux. Therefore, details of our multicast extensions, such as the multicast routing tables, the multicast routing daemon, or the corresponding multicast APIs had to be also Linux specific. However, the protocol mechanisms outlined in Section III and Section IV are not affected by these implementation-specific design decisions. Note, that we do not evaluate any QoS-related metrics of user data flow packets such as latency or jitter, since the NSIS protocols act only as plain signaling protocols.

One of the most important design decisions related to the existing GIST implementation is the extension of its internally used routing table. The routing table is structured as a hash table for unicast communication only, i.e., a routing key serves as an index for a routing entry of one specific GIST peer. However, in case of multicast communication, a routing entry should be related to a group of multicast peers. We therefore extended the routing entry of the routing table with a `multicast_peers` hash table where information about all multicast peers is stored. Figure 7 illustrates the extended routing table.

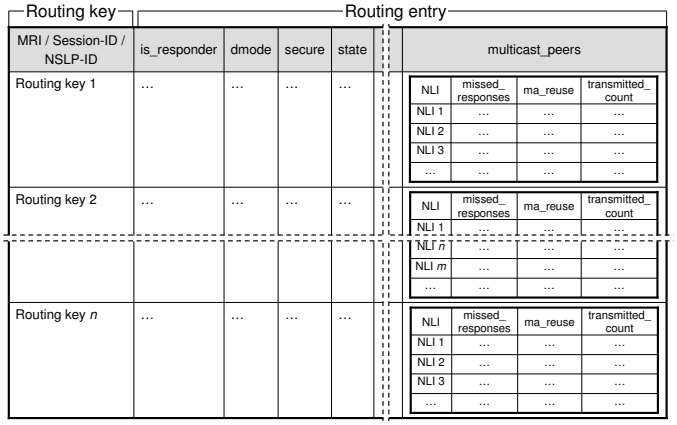

Figure 7. Extended routing table where the new field *multicast_peers* stores relevant information about each multicast peer

Within the newly introduced multicast_peers hash table, each peer's NLI object is used as the hash table's key so that the following attributes can easily be maintained and quickly accessed:

- A counter specifies the number of QUERY messages that a peer did not respond to in a row, called `missed_responses`
- A boolean value, called `ma_reuse`, indicates whether a peer requested to reuse an existing MA
- A field `transmitted_count` keeps track about all data messages of the send queue that were already delivered to this peer

Furthermore, we extended the routing table with entries for the following fields:

- A boolean flag, called `is_multicast_querier` that is set to true in case an incoming QUERY used an IP multicast address
- A dedicated slot for the NoResponse timer
- A value that keeps track of the minimum of all routing state validity timers of all peers

We evaluated the proposed multicast extensions in a testbed environment following the setup depicted in Figure 8. Each node consists of Intel Xeon X3430 quadcore CPUs running at 2.40 GHz, 4 GB RAM, and four Intel 82580 Gigabit Ethernet network interfaces, interconnected by a Cisco Catalyst Switch 6500 running CatOS. All nodes used an Ubuntu 10.10 server installation with a 2.6.35 Linux kernel. The latency between

the endpoints was small (approximately 1.235 ms on average between tb6 and all multicast destinations measured by 100 ping tests) in order to concentrate measurements on the pure protocol and processing overhead. Fine-grained measurements for the artificial delays for GIST RESPONSE messages were performed by putting reference points into specific places within the code. Once such a reference point is executed, the value of the calculated delay is stored in memory. After the entire experiment is finished, the recorded values are written into a file. This avoids the measurements to be affected from file I/O operations. Although the implementation also supports IPv6 multicast, we concentrate in this Section on evaluations for IPv4 multicast only.

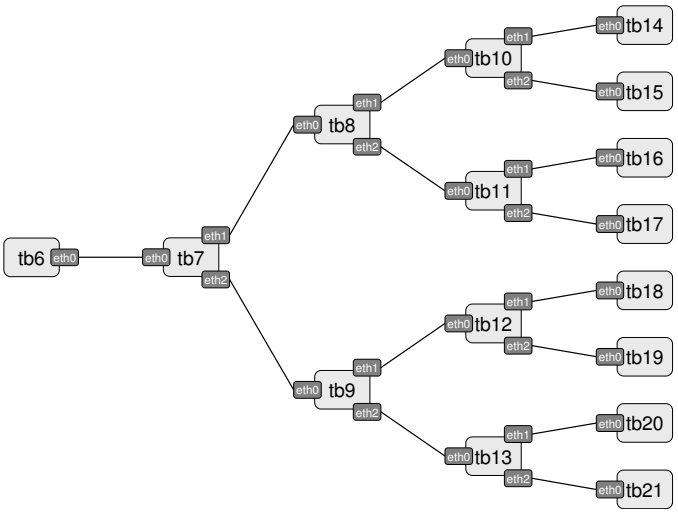

Figure 8. Network topology used for evaluation tests. Testbed routers tb14 to tb21 were configured to act as multicast destinations.

Measurements were generally performed with tb6 acting as the data flow sender, testbed routers tb14 to tb21 acting as multicast receivers, and all intermediate testbed routers acting as NSIS capable IP multicast routers. Therefore, all nodes actively participated in our NSIS multicast evaluations.

For the sake of brevity, we concentrate in this section on signaling performance evaluations of sender- and receiver-initiated reservations. We have, however, verified the functionality of earlier mentioned features like mixed messaging associations, peers dynamically joining or leaving the multicast group, or merging of reservations, in our testbed environment.

A. Sender-initiated multicast reservations

We performed measurements of 50 runs in total for sender-initiated multicast reservations to evaluate the signaling performance of our multicast extensions. The 50 runs consist of 10 series of measurements, each containing 5 single sender-initiated reservations that were subsequently torn down within an interval of 5 and 2 seconds, respectively. We configured the GIST multicast response delay for each node to be in the range of [0, 50] ms and increased each node's GIST state lifetime parameter to not interfere the measurements with refreshing QUERY messages.

Figure 9 shows the results for the sender-initiated multicast reservations. The total duration of sender-initiated reservations, originating from a GIST QUERY until the QoS NSLP RESPONSE is received by tb6, is illustrated by the red bullet points. The rather alternating behavior stems from the artificial randomly chosen GIST response delay that is added by each intermediate node. We traced these artificial delays by the aforementioned fine-grained measurements and calculated the resulting delay of the path with the lowest cumulative GIST delays (not shown in the figure). The difference between the total signaling time and the cumulative artificial delay is the actual plain signaling overhead, shown by the blue line at the bottom.

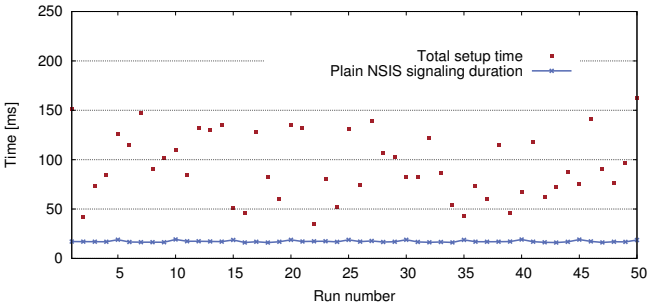

Figure 9. Duration of sender-initiated reservations and the calculated artificial delay for GIST RESPONSE messages

The plain NSIS signaling duration to setup a sender-initiated multicast reservation takes about 17.5 ms on average with a standard deviation of [16.87, 18.11] ms within our setup. Figure 10 shows the setup time (dashed blue line) in a higher resolution together with the time required to tear down a sender-initiated multicast reservation (red line). The small peaks of the setup time result from the aforementioned 10 separate measurement series where the very first run takes a bit longer due to the instantiation of state and caches.

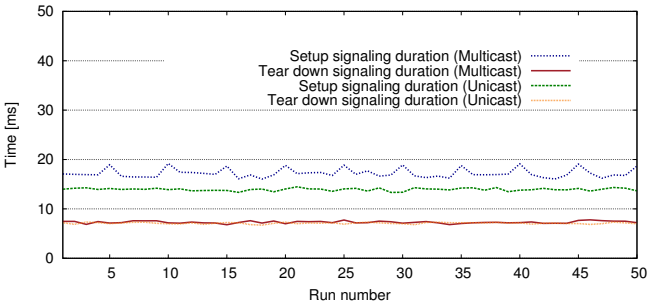

Figure 10. Plain signaling overhead of sender-initiated reservations and tear down overhead

We also performed measurements for sender-initiated unicast reservations from tb6 to tb14 for comparison. The results for the setup time and the tear down time are also depicted in Figure 10. We can see that the time required to tear down a sender-initiated multicast reservation (7.31 ms on average with a standard deviation of [7.26, 7.36] ms) is almost identical to

the one for a sender-initiated unicast reservation (7.11 ms on average with a standard deviation of [7.09, 7.14] ms), whereas a unicast reservation is slightly faster instantiated (13.97 ms on average with a standard deviation of [13.90, 14.03] ms).

B. Receiver-initiated multicast reservations

As NSIS supports both, sender- and receiver-initiated reservations, we also performed evaluations for receiver-initiated reservations. The results for 50 consecutive runs are depicted in Figure 11. This time, tb6 initiates a QoS NSLP QUERY toward the multicast destination upon which all multicast receivers initiate a RESERVE. The top red bullet points again correspond to the entire signaling duration as seen by tb6 ranging from the initial GIST QUERY until a QoS NSLP RESPONSE is emitted. By subtracting the artificially added GIST multicast response delays of the path with the lowest cumulative delays (not shown in this figure), we retrieve the plain NSIS signaling overhead for receiver-initiated multicast reservations. The time to setup such reservations takes about 26.5 ms on average with a standard deviation of [22.68, 30.34] ms.

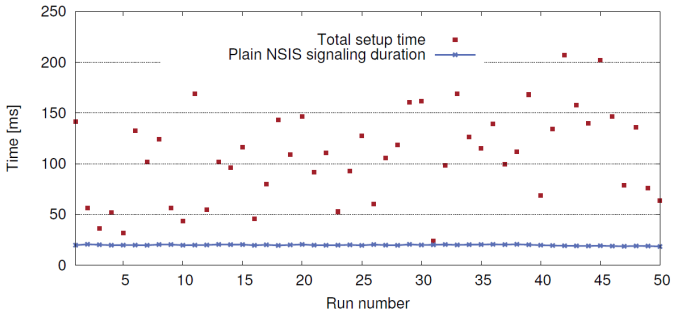

Figure 11. Duration of receiver-initiated reservations and the calculated artificial delay for GIST RESPONSE messages

VI. CONCLUSIONS

The Next Steps in Signaling Framework was designed to support IP unicast communications only, in order to reduce the protocol's complexity at the beginning. In this paper we showed that the NSIS protocols can be extended to support IP multicast without the need to change the protocol's behavior regarding its unicast capabilities or the introduction of any new protocol data units. This opens up new opportunities for signaling applications. Unlike RSVP, a multicast-aware NSIS protocol suite now even allows for *scalable sender-initiated reservations* in multicast environments: as soon as a new receiver joins the multicast tree, a GIST Query will be generated at the branching node for the new branch and a GIST handshake will take place. Afterwards, the QoS NSLP will be notified and send a RESERVE message downstream along the new branch, thereby repeating the GIST handshake with each new peer. Furthermore, it is now possible to send signaling messages reliably, which may be of use for some new NSLPs. The evaluation of the proposed design principles shows that NSIS can be extended to be used with IP multicast. Future work will study the integration of mobile multicast users more closely.

ACKNOWLEDGMENT

We thank Micha Lenk for his contributions to the design and the resulting implementation. Part of this work was supported by the G-Lab project, funded by the Federal Ministry of Education and Research of the Federal Republic of Germany (support code 01 BK 0809, G-Lab, http://www.german-lab.de/).

REFERENCES

[1] K. Nahrstedt, A. Arefin, R. Rivas, P. Agarwal, Z. Huang, W. Wu, and Z. Yang, "QoS and resource management in distributed interactive multimedia environments," *Multimedia Tools Appl.*, vol. 51, pp. 99–132, Jan. 2011.

[2] S. Deering, "Host extensions for IP multicasting," RFC 1112 (Standard), Internet Engineering Task Force, Aug. 1989, updated by RFC 2236. [Online]. Available: http://www.ietf.org/rfc/rfc1112.txt

[3] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification," RFC 2205 (Proposed Standard), Internet Engineering Task Force, Sep. 1997, updated by RFCs 2750, 3936, 4495, 5946. [Online]. Available: http://www.ietf.org/rfc/rfc2205.txt

[4] X. Fu, H. Schulzrinne, A. Bader, D. Hogrefe, C. Kappler, G. Karagiannis, H. Tschofenig, and S. V. den Bosch, "NSIS: A New Extensible IP Signaling Protocol Suite," *Communications Magazine, IEEE*, vol. 43, no. 10, pp. 133–141, October 2005.

[5] H. Schulzrinne and R. Hancock, "GIST: General Internet Signalling Transport," RFC 5971 (Experimental), Internet Engineering Task Force, Oct. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5971.txt

[6] J. Manner, G. Karagiannis, and A. McDonald, "NSIS Signaling Layer Protocol (NSLP) for Quality-of-Service Signaling," RFC 5974 (Experimental), Internet Engineering Task Force, Oct. 2010. [Online]. Available: http://www.ietf.org/rfc/rfc5974.txt

[7] J. Manner and X. Fu, "Analysis of Existing Quality-of-Service Signaling Protocols," RFC 4094 (Informational), Internet Engineering Task Force, May 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4094.txt

[8] G. Fehér, K. Németh, M. Maliosz, I. Cselényi, J. Bergkvist, D. Ahlard, and T. Engborg, "Boomerang – A Simple Protocol for Resource Reservation in IP Networks," in *Proceedings of IEEE Workshop on QoS Support for Real-Time Internet Applications*, Jun. 1999.

[9] S.-B. Lee, G.-S. Ahn, X. Zhang, and A. T. Campbell, "INSIGNIA: An IP-Based Quality of Service Framework for Mobile ad Hoc Networks," *Journal of Parallel and Distributed Computing*, vol. 60, no. 4, pp. 374 – 406, 2000.

[10] P. P. Pan, E. L. Hahne, and H. G. Schulzrinne, "BGRP: Sink-Tree-Based Aggregation for Inter-Domain Reservations," *Journal of Communications and Networks*, vol. 2, pp. 157–167, Jun. 2000.

[11] R. Bless, "Towards Scalable Management of QoS-based End-to-End Services," in *Network Operations and Management Symposium, 2004. NOMS 2004. IEEE/IFIP*, vol. 1, Apr. 2004, pp. 293–306.

[12] R. Sofia, R. Guérin, and P. Veiga, "SICAP, A Shared-segment Inter-domain Control Aggregation Protocol," in *High Performance Switching and Routing, 2003, HPSR. Workshop on*, Jun. 2003, pp. 73–78.

[13] D. Vali, S. Paskalis, L. Merakos, and A. Kaloxylos, "A Survey of Internet QoS Signaling," *Communications Surveys Tutorials, IEEE*, vol. 6, no. 4, pp. 32–43, 2004.

[14] L. Delgrossi and L. Berger, "Internet Stream Protocol Version 2 (ST2) Protocol Specification - Version ST2+," RFC 1819 (Experimental), Internet Engineering Task Force, Aug. 1995. [Online]. Available: http://www.ietf.org/rfc/rfc1819.txt

[15] P. Pan and H. Schulzrinne, "YESSIR: A Simple Reservation Mechanism for the Internet," *SIGCOMM Comput. Commun. Rev.*, vol. 29, pp. 89–101, Apr. 1999.

[16] S. Floyd and V. Jacobson, "The Synchronization of Periodic Routing Messages," *SIGCOMM Comput. Commun. Rev.*, vol. 23, no. 4, pp. 33–44, 1993.

[17] Institute of Telematics, "NSIS-ka – A free C++ implementation of NSIS protocols," May 2011. [Online]. Available: http://nsis-ka.org/