Policy-based End-to-End QoS Guarantee Using On-Path Signaling for Both QoS Request and Feedback

Yasusi Kanada
Central Research Laboratory, Hitachi, Ltd.
Higashi-Koigakubo 1-280, Kokubunji, Tokyo 185-8601, Japan
Yasusi.Kanada.yq@hitachi.com

***Abstract**: To guarantee network QoS, mechanisms and protocols for traffic resource reservation and those for measurement and feedback have conventionally been developed separately. The QoS parameters in reservation and feedback messages should coincide, and the mechanisms for these two can be similar. Therefore, we designed a protocol called SNSLP for both QoS request and feedback messages. SNSLP has a unified format for both and reduces the protocol complexity. SNSLP was implemented on the top of RTCP for both an experimental network with policy-based QoS control and a voice application called voiscape. In addition, an implementation method using routers without an SNSLP stack for policy-based routing is described and the result of implementation is reported.*

1. Introduction

The Internet was not initially designed to guarantee end-to-end QoS. However, IP networks are widely used for IP telephony or other real-time communications, so building an architecture that enables an end-to-end QoS guarantee on IP networks is now required. The Next-Generation Network (NGN), which is being developed by the standardization organizations including the ITU-T, 3GPP, and ETSI, is a type of IP network, and the intention is to guarantee end-to-end QoS in the NGN. The NGN will replace the conventional telephone network, so there is a strong requirement to guarantee the same level of QoS as that of the conventional telephone network in the NGN. The IP Multimedia Subsystem (IMS), which was developed by the 3GPP, is the central concept that enables an end-to-end QoS guarantee in the NGN.

To achieve end-to-end QoS, applications must declare QoS parameters to be guaranteed or send a request resource reservation to the network. The Resource Reservation Protocol (RSVP) [Bra 97] was developed for this purpose. RSVP allows applications to communicate implicitly with the network (nodes) for reserving communication bandwidth and several other network resources. To meet the request, the network nodes or servers must receive and forward RSVP messages and configure the nodes to reserve resources for the flow.

Recently, a new protocol called QoS NSLP [Man 06] has been developed by the NSIS (Next Step In Signaling) Working Group in the IETF for similar purposes. Applications can state QoS classes of their flows and demand bandwidth, delay bound, jitter bound, and other properties by specifying QSPECs (QoS specifications) [Ash 06].

However, resource reservation or a QoS request alone is not sufficient for guaranteeing QoS. The state of a network is continuously changing and unexpected phenomena may occur. Even if the required resources are successfully reserved, the delay, jitter, and other QoS parameter values may be out of range because of the interference of other traffic. In particular, delay and jitter cannot be configured directly at the network nodes or servers, in contrast to bandwidth, which can be specified directly, so the parameter values can be easily outside the specified ranges. To reduce that possibility, the QoS should be measured, and if the conditions are not met, that fact should be fed back to the network.

In this paper, a method of guaranteeing end-to-end QoS using a protocol that is used both for QoS request and feedback (R & F) is proposed. Mostly the same mechanism is used for both R & F, so the complexity of the protocol and implementation can be reduced.

2. Related Work

Pan and Schulzrinne [Pan 99] developed an on-path protocol for resource reservation called YESSIR. That was implemented on top of RTCP but measurement and feedback were outside the focus. Fukumoto et al. [Fuk 06] proposed the Multi-RTCP Scheme for measuring and reporting the quality of an RTP stream at a middle point by using extra RTCP packets. Although they needed extra RTCP packets, they used normal RR and SR, and their method is not reservation based. Xu, et al. [Xu 06] also proposed an extension of RTCP called SubRTCP that enabled middle nodes to monitor an RTP session and to report the result using the same RTCP session.

3. Architecture for End-to-End QoS

The architecture for an end-to-end QoS guarantee, which is outlined in **Figure 1**, is described in this section.

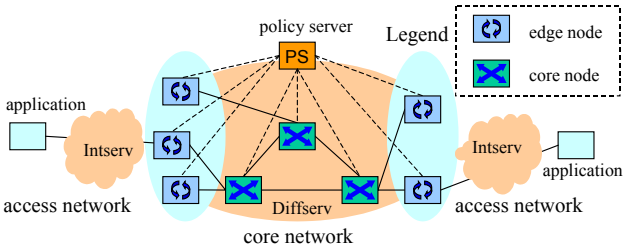

Figure 1. Outline of end-to-end QoS guarantee

3.1 QoS requests through on-path signaling

If RSVP is used for reserving resources for a flow, `Path` and `Resv` messages pass through the same network nodes as the target communication flow. This type of signaling is called *on-path signaling*. When the receiver initiates the reservation sequence, that type of messaging is called *receiver-initiated signaling*. QoS NSLP allows on-path signaling and *off-path signaling*. In the latter, the reservation message goes through a different path than that of the target communication flow. In QoS NSLP, a reservation sequence can be either receiver-initiated or *sender-initiated*.

In our method, on-path signaling is used, and sender-initiated signaling is used in the current implementation. QoS NSLP can be used for this purpose, but a simplified protocol called SNSLP (Simplified NSLP) has been developed. SNSLP has the following types of messages.

- A `RESERVE` message requests to reserve resources.
- A `TEAR` message requests to release resources reserved by the corresponding `RESERVE` message.
- A `RESPONSE_TO_RESERVE` message is sent by the receiver of a `RESERVE` message, and that message contains the result, i.e., success or failure, of the request.
- A `RESPONSE_TO_TEAR` message is sent by the re-

ceiver of a TEAR message as the response.

- A NOTIFY message is sent by the ingress edge node instead of a RESPONSE_TO_RESERVE or RESPONSE_TO_TEAR message when the response message is intentionally delayed by the network node as the response.[1]

An SNSLP signaling sequence reserves resources for one-way communication, which is similar to RSVP signaling, so two sequences are required for two-way communication. A typical QoS request sequence using SNSLP is shown in **Figure 2**. In this figure, the QoS parameters specified in the RESERVE message are passed to a policy server by a protocol called SCOPS, and the policy server deploys policies to the network nodes. The details are explained in the next section. A NOTIFY message is sent to the sender before forwarding the RESPONSE_TO_RESERVE message.

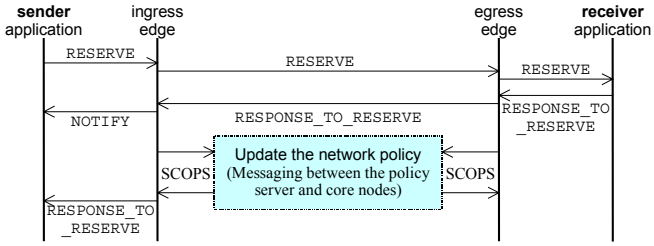

Figure 2. Typical QoS request sequence using SNSLP

An SNSLP message may contain a simplified version of the QSPEC that can contain the following parameters.

- *Bandwidth*: The maximum bandwidth of the flow.
- *Y.1541 QoS class*: One of the eight QoS classes specified in the ITU-T Y.1541 recommendation. For example, one of the following classes can be specified: Value 0 means "conversation" and value 1 means "streaming".
- *Path latency*: The maximum packet-delivery latency allowed for the end-to-end path.
- *Path loss ratio*: The maximum packet-loss ratio allowed at the end of the path.
- *Path jitter*: The maximum jitter allowed at the end of the path.
- *Path error ratio*: The maximum packet-error ratio allowed at the end of the path.

SNSLP is a soft-state protocol, which is similar to RSVP and QoS NSLP. This means QoS requests must be sent periodically before the time-out time specified in the request message. If a request is timed out, the resources reserved for the flow are released. To specify the time-out time, an SNSLP packet has a field named "refresh period".

3.2 Outsourcing policy decisions

A network may contain several access networks and a core network as components. In the access networks, an IntServ [Bra 94] model may be suitable for resource reservation. However, in the core network, IntServ models are not scalable, so a DiffServ [Nic 98] [Car 98] model should be used. This type of combination of IntServ and DiffServ has been proposed by many researchers (e.g., [Bal 00] [Det 99] [Wes 02]). This paper focuses on DiffServ-based QoS in the core network.

In our method, the services are designed as follows (See Figure 1). The ingress edge node of the core network processes the RESERVE and TEAR messages. Then, the ingress edge node outsources requests to a policy server, or so-called bandwidth broker [Soh 02], of the core network by using a simplified protocol called SCOPS ("Simplified COPS-like" protocol, but actually SCOPS is not very similar to COPS [Dur 00]). The policy server deploys an edge policy decision to the ingress edge node. This decision contains decisions on policing and marking. A decision on policing specifies the maximum rate and burst size for the flow, and a decision on marking specifies the DSCP (DiffServ Code Point) value, which corresponds to the per-hop behavior (PHB) [Nic 98], to be marked. The QoS class in the reservation message is mapped to PHB and DSCP by the policy server. For example, the policy server may have policies that assign the expedited forwarding (EF) PHB [Dav 02] to a conversation-class stream of low bandwidth (which may be a voice stream) and assign an assured forwarding (AF) PHB [Hei 99] to a streaming-class stream. There can be multiple policy servers, so this mechanism can be scaled up to cases with many edge routers.

The policy server determines the path between the ingress and egress edge routers by using the following method. The policy server can obtain the ingress edge IP address because it receives the outsourcing message. The policy server also receives the same type of message from the egress edge, so it can obtain the egress edge IP address. The policy server is assumed to know the topology and paths (routing information) of the core network, so it can find the path between the ingress and egress routers. The policy server records resource usage, i.e., computes the total bandwidth per path and per QoS class (PHB). This bandwidth can be regarded to be assigned to the aggregated flow of the QoS class through the path. The policy server must record the following data.

- *Edge IP addresses*: The addresses of the ingress and egress edge nodes. This pair is used for identifying the path between the ingress and egress edges.
- *DSCP*: The DSCP value to be marked. This value represents the QoS class in the core network.
- *Allocated bandwidth*: The bandwidth allocated to the aggregated flow.
- *Total bandwidth*: The total bandwidth required by the microflows in the aggregated flow.

A method for managing bandwidth and another one for managing other parameters are explained. First, the bandwidth is handled in the following method. The policy server adds the requested bandwidth of a flow to the total bandwidth when the new request comes. If the total bandwidth exceeds the allocated bandwidth or if the bandwidths between the QoS classes become unbalanced, the policy server changes the core router configurations (and the egress network interface configuration of the ingress router). If no such configuration that allows the increase in total bandwidth without potential violation of the QoS requests exists, the policy server denies the request. This means, a RESPONSE_TO_RESERVE message that contains a failure result is returned to the requester. The frequency of the core policy update is much lower than that of QoS requests, so this mechanism is scalable.

Second, a method for handling QoS parameters other than bandwidth is outlined. There is no established method for handling them. However, admission control, queuing and scheduling control such as weight control for WFQ (Weighted Fair Queuing) [Kan 08], and scheduling priority control can be combined to solve the problem.

3.3 QoS measurement and feedback

There are three types of QoS measurement approaches. The first is the *end-to-end approach*. A typical method is to use the Real-Time Control Protocol (RTCP) [Sch 03]. A receiver report (RR) message of RTCP contains a set of measurement results of an RTP (Real-time Transport Protocol) [Sch 03] stream, including the latency, loss ratio, and jitter. In this method, the QoS parameters of an RTP

[1] If a resource allocation or deallocation caused by a RESERVE or TEAR message takes much time, the success or failure of the server operation may be unknown when the response message comes to the ingress edge node. If the response message is delayed until the server operation finishes, the retransmission timer may be timed out and the original message may be sent again.

stream, such as a voice or video stream, are measured by the receiver application, and the results are sent to the sender by using RR messages. The sender can change the parameters by modifying the coding of the voice or video.

The second approach is the *network-based approach*. In this approach, some network nodes measure the traffic, or a combination of a measuring device and a network node can be used; the latter copies packet content and sends them to the former. If the measuring device finds a problem, the network configuration is changed to solve the problem, or an alert for the operator to change the configuration manually is generated.

Our approach belongs to the third type, the *mixed approach*; the receiver application measures the QoS parameters and feeds back the results to the network. The network can take appropriate actions to improve QoS. In our method, the same protocol as the QoS request, i.e., SNSLP, is used for end-to-end QoS measurement. Therefore, the following message types were added to SNSLP.

- A `REPORT` message is used for sending the measurement results from the receiver.
- A `RESPONSE_TO_REPORT` message is sent by the receiver of a `REPORT` message as the response.

A typical QoS-reporting sequence using SNSLP is shown in **Figure 3**. This is very close to the QoS request sequence. The policy server receives the measurement results and may update policies deployed to the network nodes when the required QoS parameters are not satisfied. The main purpose of this messaging is to inform the network about the measurement results, but of course, the sender can see and use the results as well.

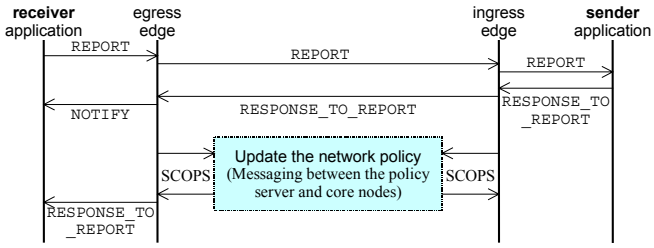

Figure 3. Typical QoS reporting sequence using SNSLP

A QoS-reporting SNSLP message contains a QSPEC in the same format as that of request messages. The SNSLP message should contain the measured path latency, loss ratio, and jitter. The measured values used for an RR message can be used for a report message. If the throughput can be measured in the receiver application, that should be sent as bandwidth.

QoS-reporting messages are sent periodically. The period is specified in the refresh-period field of an SNSLP packet. Therefore, the packet format for feedback is the same as that for request. The measured values in the messages are used by the policy server. The policy server can update the policy decisions and the configurations of the edge and core nodes.

4. Implementation

SNSLP and SCOPS were implemented in an experimental network with a voice application called *voiscape* [Kan 05].

4.1 Network structure and application

The structure of the experimental network is shown in **Figure 4**. The core network consisted of two edge routers, the Hitachi GR2000B, and two core switches, the Hitachi GS4000.

A prototype of a spatialized-voice-based application called *voiscape* was used for the test application. In this prototype, VPIIQ (Voiscape Prototype 2 Q), RTP was used between user agents (communication terminals) and media servers. There were two types of media servers, i.e., a

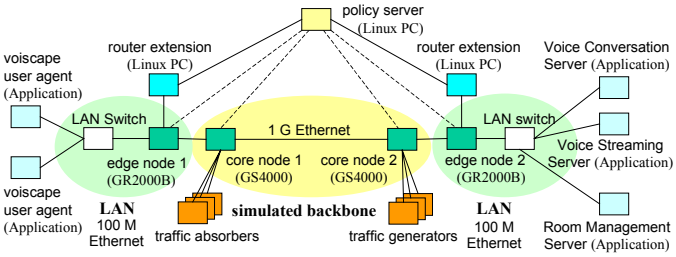

Figure 4. Structure of experimental network

voice conversation server and a voice streaming server. The former collected voices from user agents and spatialized and mixed them. For each user agent, there were three RTP streams, upstream and downstream conversation voices, and streaming voice that was directed downstream. SNSLP was handled in the user agents, conversation and streaming servers, and router extensions (on Linux PCs).

Edge nodes were commercial routers, so SNSLP should not be handled by them. We used the following method to solve this problem. We configured each edge router to forward SNSLP messages to a router extension. The detailed method of handling SNSLP messages at the edges is described in Section 4.3. The core switches passed through the SNSLP messages with no processing.

4.2 SIP messaging and QoS request

SNSLP was implemented on the top of RTCP. Implementing SNSLP directly on UDP was possible, but RTCP was used for simplicity. This protocol stack structure is similar to that of YESSIR. RTCP can be extended by adding an application-specific packet format. The first byte of the RTCP packet contains the SNSLP message type.

The QoS-request mechanism works as follows. VPIIQ is a SIP (Session Initiation Protocol)-based [Ros 02] application. When the user enters into a (conference) room, an `INVITE` request is sent to the server. The request initiates the QoS-request sequences according to the IMS standard [3GP 06]. The conversation is two-way and the streaming is one-way, so three SNSLP sequences run. These sequences are outlined in **Figure 5**. The `INVITE` request contains an SDP (Session Description Protocol) [Han 98] message with the following parameters.

- *User IP address*: The IP address (*Au*) of the user agent.
- *Conversation UDP port of the user*: The port number of the user agent to receive the conversation voice stream from the conversation server. This port, *Puc*, is used for

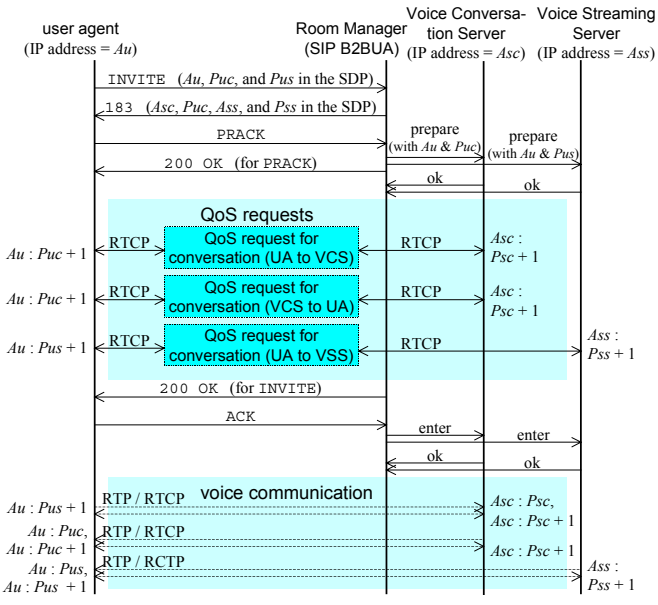

Figure 5. Whole sequence for session initiation in VPIIQ

RTP and port *Puc* + 1 is used for RTCP.

- *Streaming UDP port of the user*: The port number of the user agent to receive the streaming voice stream from the streaming server. This port, *Pus*, is used for RTP and port *Pus* + 1 is used for RTCP.

A temporary reply to the `INVITE` request, i.e., "`183 Session Progress`" response, contains an SDP message with the following parameters.

- *Conversation server IP address*: The IP address (*Asc*) of the conversation server.
- *Conversation UDP port of the server*: The port number of the conversation server to receive the conversation voice stream from the user agent. This port, *Psc*, is used for RTP and port *Psc* + 1 is used for RTCP.
- *Streaming server IP address*: The IP address (*Ass*) of the streaming server.
- *Streaming UDP port of the server*: The port number of the streaming server. This port, *Pss*, is not actually used because the streaming RTP is one-way. However, this must be specified because RTCP communication is two-way and port *Pss* + 1 is, thus, required.

When the above SIP `INVITE` and `183` messages are exchanged, the SIP proxy (P-CSCF in the term of the IMS) of the network should look at the SDP messages that the SIP messages contain and permit use of the RTCP ports to allow QoS-request messaging.[1] Port $p + 1$ is used for negotiating the resources for the RTP stream with port p. (See Figure 5.) This convention makes the QoS-request messaging simpler. The IP address and ports included in the `INVITE` request are passed from the SIP proxy to the media servers using the "prepare" message in Figure 5.

4.3 Outsourcing using policy-based routing

Because SNSLP is a newly-developed protocol, preexisting network nodes cannot process them. SNSLP messaging is on-path, i.e., messages pass through the same network nodes as the target communication flow. Therefore, receiving and forwarding an SNSLP message by an off-path network element such as network servers is basically impossible. If the IP Router Alert Option [Kat 97] is available on the edge router, that can be used for processing SNSLP messages, but that was not available on the GR2000B. However, policy-based routing was used, so an off-path node could receive and forward SNSLP messages. We call this off-path element a *router extension* (RX).

In routers, the policy-based routing means a function that forward packets that match a specific condition to a specific destination. A policy-based-routing configuration (policy) specifies the following parameters:

- *Ingress interface*: If and only if the traffic that comes from this network interface of the router is forwarded by the policy.
- *Flow filter*: Only the traffic that matches this filter is forwarded by the policy. A filter example is {FromIP: 192.168.2.*, ToIP: 192.168.3.2, ToUDP port: 10001}.
- *Destination interface*: The traffic is forwarded out of this network interface of the router.

The edge router, GR2000B, has this function. An RX is connected to a router interface, which was specified as the destination interface for the policy-based routing. The RX was implemented on a Linux PC using C and received Ethernet packets by promiscuous mode, i.e., it received all the packets on the cable without restricting the packets with the IP or MAC address of the RX. The RX detected SNSLP packets and resent the received packets (including packets except SNSLP) to the router. The resent packets went through the network as if they were the original packets. The SNSLP packets, thus, were on-path.

QoS feedback messages are handled in the same manner as the request messages. In the current method, the request is sender-initiated and the feedback is receiver-initiated. Therefore, the directions of the messages are opposite each other. These two types of messages should be caught by the same edge routers. Therefore, if the IP routes are asymmetric, a mechanism that follows the reverse path is necessary. This mechanism is also required by RSVP and NSLP.

4.4 Policy request and decision

The policy server, implemented on a Linux PC using Perl, received QoS request and feedback messages from the ingress and egress edge routers through the RXs. The policy server sent a configuration command (in the CLI) to the ingress edge router for policing and marking.

The ingress and egress edge routers generated a SCOPS packet for outsourcing the request from an SNSLP packet. The SCOPS packet contains the following fields.

- *Sender IP and port*: The IP address and port number of the sender. The IP address is copied from the corresponding SNSLP packet. If the destination port of the SNSLP packet is p, the port number is $p - 1$.
- *Receiver IP and port*: The IP address and port number of the receiver. These values are taken from the corresponding SNSLP packet too.
- *Edge IP address*: The edge node inserts its IP address.
- *QSPEC*: The requested QoS parameters, i.e., the QSPEC in the `RESERVE` or `REPORT` packet is copied.
- *Interface*: The identifier of the network interface that the stream comes in.[2]
- *Refresh period*: The refresh period (time-out time) in the SNSLP packet is copied.

SCOPS messages were sent by using TCP (Transmission Control Protocol) as same as COPS.

If necessary for the core routers, a command for changing the outbound queue configurations on the path was sent. In this implementation, the queues of the GS4000 were configured as follows. The GS4000 has the LLQ (low latency queuing)-based queues. The LLQ and the queue usage in this implementation are described by Kanada [Kan 08]. The policy server changed the queue weights when the weights become unbalanced.

The Y.1541 QoS classes were mapped to DiffServ PHBs. For example, the Conversation class was mapped to the EF PHB, and the Streaming class was mapped to an AF PHB. The QoS classes, PHBs, and the mapping method used were explained by Kanada [Kan 08].

4.5 QoS measurement and feedback

QoS parameters were measured in the voiscape application and reported by SNSLP `REPORT` messages. The measurement and feedback methods are explained here.

To measure the QoS parameters, the following two functions must be built into the sender and the receiver.

1. *NTP* (Network Time Protocol): The sender and the receiver computers must synchronize their internal clocks. They can be synchronized by communicating with a time server using NTP.
2. *SR* (Sender Report): SR is a message type of RTCP. The sender application must send the SR periodically. RTP packets do not contain sufficient information for computing the delay. When an RTP packet arrives, the receiver application obtains the sender's timestamp using both SR and RTP packets and computes the delay.

The receiver application can average delay values and

[1] However, in the current implementation, we do not use a SIP proxy, and all the RTCP ports are always available.

[2] It always contains zero in the current implementation because the packet forwarded by policy-based routing does not contain the interface identifier. It should be added to the packet when the packet is forwarded. For example, it can be added as a VLAN ID.

compute jitter from the delay values. Loss ratios can be computed by using the sequence numbers in the RTP packets. If a packet is dropped, the receiver application can detect that by the absence of the sequence number.

There may be skew between the clocks of the sender and receiver computers, and the clocks may be set by the NTP clients, so delay and jitter values must be adjusted properly. A method for skew adjustment similar to that of Moon, et al. [Moo 99] or Anagnostakis, et al. [Ana 03] may be used for this purpose.

The QoS-feedback mechanism works as follows. `REPORT` messages with the measured delay, loss ratio, jitter, and other QoS parameters are sent to the corresponding RTCP port. If the receiver's RTP port is *Psc* (so the receiver's RTCP port is *Psc* + 1), the corresponding sender's port *Puc* + 1 is used for this feedback, and if the RTP port is *Puc*, the port *Psc* + 1 is used (See Section 4.2).

As explained in Section 3.3, RR messages contain measured values of latency, jitter, and loss ratio. However, `REPORT` messages contain partially duplicated values because using the same format as QoS request messages is better; the `REPORT` message could contain all the values that correspond to the requested values. Measured values were recorded in the log of the policy server.

4.6 Observations

We operated the implemented system and observed the log that the policy server generated. Although the performance was not yet measured because the system was not yet tuned, the following issues were found.

- A pipelined processing of QoS request and report messages is very important for improving performance because if those are processed one by one, the messages are stacked and the delay becomes intolerable.
- RTP packets need to be sent accurately, i.e., the difference between the logical and the actual sending time must be small because timestamps in RTP packets are usually computed logically from the number of media data but not from the clock.
- An RTP packet needs to be processed as soon as it arrives, and the receiving time needs to be obtained from the clock. Some receiver applications leave RTP-packet input buffering to the operating system and obtain packets from the buffer on demand. In such a case, if the applications obtain the receiving time when demanding the data, obtaining an accurate time is impossible.

5. Conclusion

A method of policy-based end-to-end QoS guarantee using on-path signaling has been developed. The major contributions of this paper are as follows.

- This paper showed that the same protocol can be used for both QoS request and feedback (R & F). This method improves the correspondence between requested and measured QoS parameter values and reduces the protocol complexity.
- This paper described a method for handling on-path signaling for QoS R & F by policy-based routing using network nodes without implementing the signaling protocol in the router.
- This paper described a design and implementation of signaling for QoS R & F using a protocol on top of RTCP and described a measurement method and required conditions for senders and receivers.

The developed protocols, i.e., SNSLP and SCOPS, are significantly simplified; they only have the most important functions of QoS R & F. Hence, future work is to develop protocols for actual use or to propose new functions to standardized protocols. More important future work is to develop a method for updating policies using the QoS reports.

Acknowledgments

Part of this project was funded by the Ministry of Internal Affairs and Communications of the Japanese Government. The author thanks Yasushi Fukuda from NSD, Hitachi and Takeki Yazaki and Daisuke Matsubara from CRL, Hitachi for discussing the architecture for the end-to-end QoS guarantee, the plans for method development. The author also thanks Hideki Taira, Tomio Fujiwara, and other members of the Environment and Facilities Unit, CRL, Hitachi, and Takeshi Shibata from CRL, Hitachi, for help in developing the experimental network.

References

[3GP 06] 3rd Generation Partnership Project (3GPP), "Technical Specification Group Core Network and Terminals; Signalling Flows for the IP Multimedia Call Control Based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3 (Release 5)", 3GPP TS 24.228 V5.14.0, December 2005.

[Ana 03] Anagnostakis, K. G., Greenwald, M., and Ryger, R. S., "cing: Measuring Network-Internal Delays Using Only Existing Infrastructure", *IEEE Infocom 2003*, pp. 2112–2121, 2003.

[Ash 06] Ash, J., Bader, A., and Kappler, C., "QoS NSLP QSPEC Template", draft-ietf-nsis-qspec-10, Internet Draft, IETF, June 2006.

[Bal 00] Balmer, R., Baumgarter, F., Braun, T., and Gunter, M., "A Concept for RSVP over DiffServ", *ICCCN 2000*, pp. 412–417, October 2000.

[Bra 94] Braden, R., Clark, D., and Shenker, S., "Integrated Services in the Internet Architecture: an Overview", RFC 1633, IETF, June 1994.

[Bra 97] Braden, R., Ed., Zhang, L., Berson, S., Herzog, S., and Jamin, S., "Resource ReSerVation Protocol (RSVP) – Version 1 Functional Specification", RFC 2205, IETF, September 1997.

[Car 98] Carlson, M., Weiss, W., Blake, S., Wang, Z., Black, D., and Davies, E., "An Architecture for Differentiated Services", RFC 2475, IETF, December 1998.

[Dav 02] Davie, B., Charny, A., Bennett, J. C. R., Benson, K., Le Boudec, J. Y., Courtney, W., Davari, S., Firoiu, V., and Stiliadis, D., "An Expedited Forwarding PHB (Per-Hop Behavior)", RFC 3246, IETF, March 2002.

[Det 99] Detti, A., Listanti, M., and Veltri, L., "Supporting RSVP in a Differentiated Service Domain: an Architectural Framework and a Scalability Analysis", *ICC'99*, Vol. 1, pp. 204–210, June 1999.

[Dur 00] Durham, D., ed., Boyle, J., Cohen, R., Herzog, S., Rajan, R., and Sastry, A., "The COPS (Common Open Policy Service) Protocol", RFC 2748, IETF, January 2000.

[Fuk 06] Fukumoto, N., Yamada, H., and Kawai, H., "Evaluation Result of Transmission Control Mechanism for Multimedia Streams based on the Multi-RTCP Scheme over Multiple IP-Based Networks", *CCNC 2006*, pp. 308–313, January 2006.

[Han 98] Handley, M. and Jacobson, V., "SDP: Session Description Protocol", RFC 2327, IETF, 1998.

[Hei 99] Heinanen, J., Baker, F., Weiss, W., and Wroclawski, J., "Assured Forwarding PHB Group", RFC 2597, IETF, June 1999.

[Kan 05] Kanada, Y., "Multi-Context Voice Communication In A SIP/SIMPLE-Based Shared Virtual Sound Room With Early Reflections", *NOSSDAV 2005*, pp. 45–50, June 2005.

[Kan 08] Kanada, Y., "A Method of DiffServ-based Bandwidth-sharing among Delay-sensitive Traffic and Loss-sensitive Traffic in Backbones", Submitted for *NGI 2008*.

[Kat 97] Katz, D., "IP Router Alert Option", RFC 2113, IETF, February 1997.

[Man 06] Manner, J., ed., Karagiannis, G. and McDonald, A., "NSLP for Quality-of-Service Signaling", draft-ietf-nsis-qos-nslp-11, Internet Draft, IETF, June 2006.

[Moo 99] Moon, S. B., Skelly, P., and Towsley, D., "Estimation and Removal of Clock Skew from Network Delay Measurements", *IEEE Infocom 1999*, pp. 227–234, March 1999.

[Nic 98] Nichols, K., Sblake, S., Baker, F., and Black, D., "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

[Pan 99] Pan, P. and Schulzrinne, H., "YESSIR: A Simple Reservation Mechanism for the Internet", *ACM SIGCOMM Computer Communication Review*, Vol. 29, No. 2, April 1999.

[Ros 02] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and Schooler, E., "SIP: Session Initiation Protocol", RFC 3261, IETF, June 2002.

[Sch 03] Schulzrinne, H., Casner, S., Frederick, R., and Jacobson, V., "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, IETF, July 2003.

[Soh 02] Sohail, S. and Jha, S., "The Survey of Bandwidth Broker", Technical Report UNSQ-CSE-TR-0206, University of New South Wales, Sydney, Australia, May 2002.

[Wes 02] Westberg, L, Csaszar, A., Karagiannis, G., Marquetant, A., Partain, D., Pop, O., Rexhepi, V., Szabo, R., and Takacs, A., "Resource management in DiffSery (RMD): A Functionality and Performance Behavior Overview," *PfHSN 2002*.

[Xu 06] Xu, X., Connor, K., Shaikh, T., Padmanabhan, R., and Umansky, I., "SubRTCP: RTCP Extension for Internal Monitoring of RTP Sessions", draft-xu-avt-subrtcp-00, Internet Draft, IETF, December 2006.