Resource Control and Authentication for a Video Streaming Service in a DiffServ/IP Multicast Network

Roland Balmer and Torsten Braun

Institute of Computer Science and Applied Mathematics (IAM), University of Bern, Neubrückstrasse 10, CH-3012 Bern, Switzerland

With IP multicast, a sender can transmit identical data streams to a group of receivers along a multicast tree. Resource reservations generate extra cost and should therefore be limited to that part of a multicast tree that contains authorised (paying) users only. This paper presents a concept that allows Internet Service Providers (ISPs) to authenticate anonymous users of a video distribution over IP multicast. The ISPs can adjust the resource reservations based on the authentication results. The proposed approach uses signalling based on the Resource Reservation Protocol (RSVP).

Keywords: authentication, multicast, video streaming, resource control, access control

1 Introduction

Delay or loss sensitive services like video conferences are becoming more popular in the Internet. Good service quality requires privileged transport of the corresponding data packets. Techniques to guarantee Quality of Service (QoS) exist in the form of Integrated and Differentiated Services. QoS support for unicast end-to-end traffic is well understood, in particular when the two communicating endpoints are known. In this case the path from the source to the destination is known and all nodes on this path can be configured appropriately.

While for a small receiver group data transmission could still rely on unicast, for larger groups this is not feasible due to bandwidth limitations. In that case, the use of IP multicast is preferable. The principle of IP multicast is simple: Each user desiring to receive the multicast data, just has to join the corresponding multicast group and then it receives all data that are sent to this multicast address.

For a commercial video distribution service data should be transported over the network with high quality. Also in multicast scenarios, the same techniques as used for unicast can be deployed, e.g. Differentiated Services (DiffServ). However, the combination of IP multicast and DiffServ can cause problems: When all members of the multicast group are treated in the same way, all branches of the multicast tree receive the same QoS. Unfortunately, each Internet user can join a multicast group and would then automatically benefit from the privileged transmission and the corresponding resource reservation. Fig. 1 shows an example of a multicast tree with QoS support. To provide the service to paying users only the solid lined part is needed. The dashed part of the multicast tree increases the QoS resource consumption of the service without augmenting the profits. However, the goal would be that only users who have been authenticated in advance and who are paying for the delivered service should really benefit from QoS.

The idea is to limit multicast QoS delivery to authorised users only by reducing the multicast tree to the branches needed for the service. This could be done, if the locations of all paying users were known. This information could be available from the service provider, but the used scenario presented in section 3 allows anonymous users.

Our solution tries to analyse the multicast tree and to deactivate QoS support along all branches without authorised users. An appropriate concept is presented in section 4. Section 5 presents a prototype implementation to validate the concept and section 6 will conclude the paper.

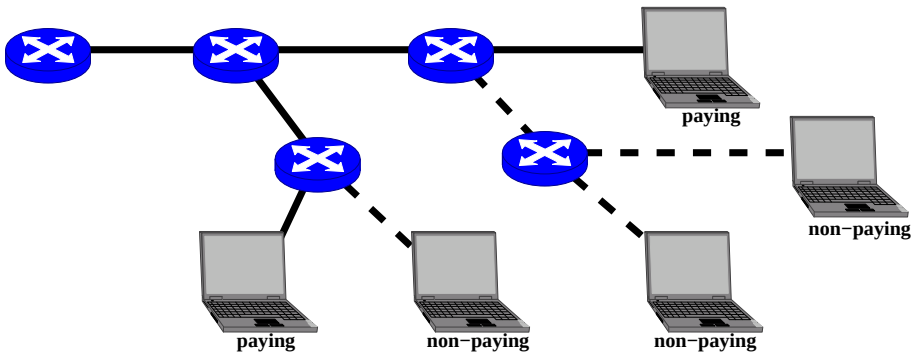

Fig. 1: Example of a multicast tree. Dashed links are used by non-paying users.

2 Related work

Because of the popularity of IP multicast some work has already been done in this research area. To restrict and secure the use of multicast two different directions exist.

The first direction is to guarantee that only authorised (paying) users can benefit from QoS during the multicast transmission. One approach according to the Secure Multicast Workgroup of the IETF (SMuG) [IETb] is to authenticate the video server against all members of the multicast group. An extension is to encrypt the transported data as proposed in [CQNW02]. This solves the problem that non-authorised users can watch the video data, but this does not solve the problem that non-authorised users could join the multicast group and consume QoS resources. Our target scenario (cf. section 3) that the admission control was designed for uses encryption for the transported data by default.

The second direction is to control access to the multicast group. The IETF Multicast and Anycast Group Membership Workgroup (magma) [IETa] is active in access control. The main focus is to inquire whether an authorised user exists in one of the branches of a multicast tree. This requires to modify IGMP, which is not required in our approach. A similar proposal is presented in [IYT01]. Again in this approach a receiver authentication is performed by using IGMP. This creates the problem that the backward compatibility is affected. Furthermore, this approach allows the initiator of the multicast group to collect information that can be used to detect the clients. This might cause problems if clients do not wish that ISPs collect client information.

Other related work is presented in [BW03]. This work addresses the problem from the QoS point of view. The problem is that QoS support for a multicast group can be broken, if the tree is extended by a new branch. QoS support is not adjusted before it can be guaranteed that the new branch supports the required QoS. This check should be done by a central admission control entity. The criteria for the decision is only the availability of the resources and no other information are considered. This solves the basic admission control problem from a resource point of view similar to [SB03], but security/authentication problems remain open.

3 Video broadcast scenario

The idea of a video broadcast service is to transmit video streams over the Internet using IP multicast. The scenario has to fulfil three requirements. First, a user should pay for the service and the payment transactions should be secure. Second, the video data should be transported with QoS support to prevent data loss, large jitter and high delay. And third, he copyright of the transmitted video should be protected.

In an initial scenario we assumed that all data are sent with IP multicast and receive the desired QoS. In this case, QoS is provided in all branches, i.e. also in branches without paying users. This paper introduces a new concept to support more efficiently QoS based on dedicated authentication mechanisms. The description of the scenario is divided into two parts: video broadcasting components and video data protection.

3.1 Video broadcasting components

Fig. 2 shows the different components and their corresponding relations. The ticket server and the financial institute are the two components responsible for the payment. These two are independent from the content provider or the Internet Service Provider (ISP). The financial institute is responsible for all financial transactions. It receives the money from the client and generates tickets (a kind of virtual money) that can be used

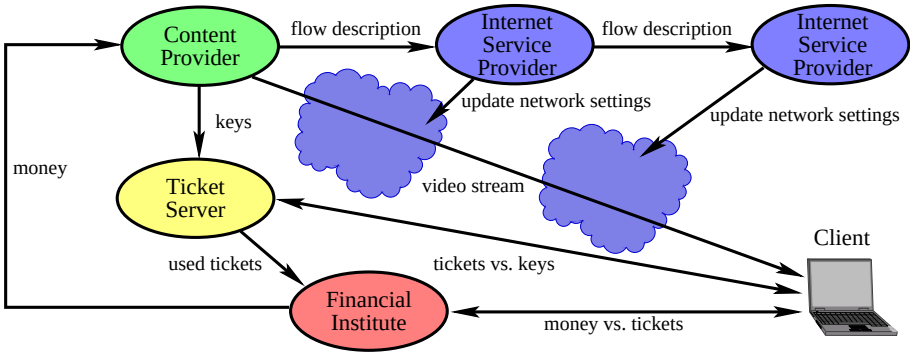

Fig. 2: All components and their relations in the video streaming scenario.

for the service. With these tickets the client can request the keys (needed to decrypt the video) from the ticket server. These keys are generated by the content provider and sent directly to the ticket server over a unicast link. Note that the clients are anonymous to the ticket server. From time to time the ticket server sends all used tickets to the financial institute, which transfers the money to the content provider. The ticket server and the financial institute do not have any impact to our concept. For data privacy and user anonymity reasons the ticket server should never give this data to the ISP. For more information about the financial transactions see [BB01].The content provider receives the money covering the complete costs for transmitting the video to the client. The transmission path might span over several ISPs. For this case, we propose the model developed in the CATI project, where the content provider pays some fees to the first ISP and the first ISP pays from that money some fees for the second ISP [DGBS00].

The other components and their roles are more relevant for user authentication. The responsibility of the ISP is the correct transport of the data over the network.The ISP receives all needed information for flow description directly from the content provider [BGB01]. The subsequent ISPs should signal to each other which flows should experience QoS support. This exchanged information describes the video streams needing QoS support and contains information about source address, destination address, bandwidth, maximum allowed delay and the starting time [GB99]. The information is processed by a central QoS broker that keeps control over the entire network. With this information each QoS broker can configure the QoS support inside its own domain using DiffServ techniques. Since there is not more specific information about users available (in particular whether users are authorised to view a video stream or not), the resources are first reserved for the entire multicast tree. Unused resources could be released at a later moment as described in section 4.

Another important role is performed by the content provider that broadcasts the video stream. The content provider has two interests. First, it wants to protect the copyright and second, it wants to assure that only paying (authorised) clients are able to view the transmitted data. The first goal is achieved by watermarking the video. The second goal is reached by encrypting the video data and by restricting QoS support to branches with authorised clients only. These two techniques influence user authentication.

3.2 Video data protection

In our video broadcast scenario, watermarking is used to protect the video. The purpose of watermarking is not only to identify the sender of the stream, but also to determine the creator of eventually appearing copies. Fig. 3 explains where the watermark is included into the video pictures. The watermark is included into each 8x8 pixel block resulting from the DCT transformation. To prevent users from removing the watermark and to avoid reducing the quality of the video stream, the watermark is included only into a small part of the DCT block. The 8x8 pixel block is divided into three different regions. First, the major called region represents the most important values of the block. A watermark within this region results in a visible quality reduction. Second, the bulk called region represents the most irrelevant values of the block. In most cases these values are near zero. A watermark in this region is easy to detect and can be removed without visible quality reduction, e.g. by setting all values to zero. The importance of these values is so small that this so called bulk part is not encrypted. Third, the watermarked region represents the values that are important enough, so that they can not be removed, but small changes will not reduce the quality too much. Furthermore, the region is large

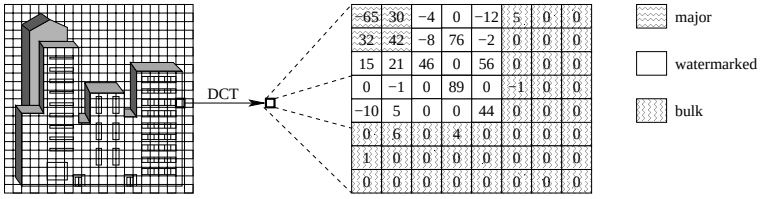

Fig. 3: The three different regions in a DCT transformed 8x8 pixel block: major or most important, used for watermarking and bulk.

enough, so that the watermark can not easily be detected and that the number of different possible watermarks is sufficiently large.

For transport the video picture is divided into different parts. The bulk part is not encrypted, but transported with QoS support. With only the bulk part representing about 90% of the video stream, the resulting video will not be visible. The rest of the data is treated in a different way. Fig. 4 shows the four steps that are applied to the important part (major and watermarked) of the video stream.

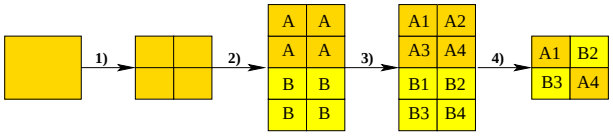

Fig. 4: The four steps to transform a picture into a unique and watermarked picture.

First, each picture is split into different segments. In the example four segments are created. Second, each of these segments is duplicated. In the example two copies of the four segments are created. At this moment the corresponding segments from pictures A and B are identical. Third, each segment receives a unique watermark. Now the segments A1 and B1 are distinguishable. And finally, e user receives a unique set of segments. In the example the user receives the set (A1, B2, B3, A4). From combinations of two different segments, $2^4 = 16$ different sets are possible. To augment the number of unique sets of segments, the picture can be divided into more segments. E.g. using 32 segments, 2^{32} unique sets are possible.

For transport each segment (e.g. A1) is encrypted and transmitted using multicast and QoS support. All segments (e.g. A1) with the same watermark are sent using individual streams. In the example eight ($2 * 4$) streams are created. To reduce the load on all links, a single multicast group is used for each stream. To decrypt the streams the client uses the keys received from the ticket server. For further information about the video streaming refer to [KT00][TK01].

4 Authentication mechanism for QoS-Based Multicast Video Transmission

4.1 Authentication Procedure

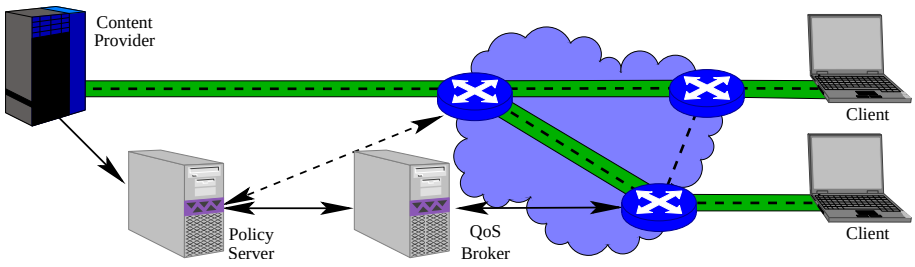

Fig. 5: The authentication scenario to detect users.

As described in the previous Section the QoS broker of each ISP along the multicast distribution tree receives the description of the video stream including information about the source and the destination IP (multicast) addresses. In addition to that information, we need to discover which users are really authorised to view the video. Policy servers can be used at the various ISPs to support this task and to support the decision whether a user is authorised to view videos. Clearly, such decisions are rather complex and should be separated from components such as QoS brokers or even routers. In addition, an appropriate signalling protocol is required to exchange authentication data in a multicast environment. Since the Resource Reservation Protocol (RSVP) [BZB+97] supports both multicast and policy objects in a flexible way [Her00][YYP+01], we decided to use RSVP as a signalling protocol for authentication in our multicast video distribution scenario. Similarly, RSVP has also been proposed to be used for other security related signalling purposes such as firewall configuration [RGKS01]. Please note that other signalling protocols could be used as well. Another candidate might be SIP including extensions for media authorisation [Mar03]. Using RSVP PATH messages ((1) in Fig. 6) will be generated by either the video source (video server) or any other router that is as close as possible to the video source, e.g. an ingress router of the ISP to which the video server is attached to. To simplify the task of the ISP, the video server can send the correct data to the ISP (inform messages (2) and (3)). The PATH messages travel along the multicast tree and are received by the clients. The clients should respond with RESV messages and insert authentication data that allows the ISPs between video server and clients to determine whether authorised clients need to be served by the branches of the multicast tree. RESV messages can be created by the video viewer application or by a separate tool running at the client computer.

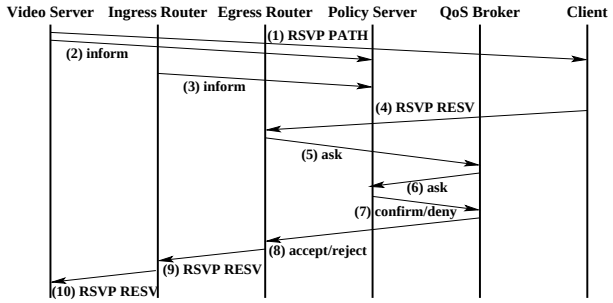

Fig. 6: All exchanged messages for the authentication of a user.

Inserting the policy server results in the extended scenario shown in Fig. 5. Fig. 6 shows the interaction between the components of the scenario. A router might receive authentication data within a RESV message (4) from a receiver and forwards an authentication request via the QoS broker (5) to the policy server (6). The policy server checks the authentication and returns the answer to the QoS broker. The QoS broker receives the answer (7) from the policy server and decides QoS management actions such as disabling or enabling QoS support (e.g., DiffServ expedited forwarding) of the flow on a specific branch of the multicast tree. For example, QoS can be configured on branches with authorised users only, while best-effort might be configured on branches without authorised users. The authentication result is then forwarded to the router originally triggering (8) the authentication request. If the authentication was successful, the RESV message can be forwarded upstream (9 and 10). Another nice feature of RSVP is the merging capability. A router does not need to forward each single request to the policy server and to upstream nodes, but can collect several RESV messages from downstream nodes. It needs only to check periodically whether one of the received RESV messages contains valid authentication data. Another advantage of RSVP is that soft states are used to store the information. The entries are deleted when not refreshed periodically. RSVP can handle an individual refresh period for each entry and the entry is removed if this period has passed three times. The refresh period depends on the interval of the encryption key exchange for the authentication and should be between a third and the half of this exchange period, e.g. the refresh period can be several minutes for a normal video stream or less than 30 seconds (default for RSVP) for live streaming of a 100m final of the Olympic Games (including the presentation of the sprinters).

4.2 Authentication Data

A key question is which authentication data can be used. We propose two different solutions. The first one uses the transported video stream, while the second one is independent from the data stream.

1. **Generated video stream:** In this approach, the authentication data from a client shall prove that the client decrypted the video data correctly and that he used a valid key for this operation. The authentication data may be hashed to a hash value at the client and this hash value can be inserted into a RESV message. The policy server itself also receives and decrypts the video data, stores the most recent hash values calculated from it and compares them with the authentication data to be verified. Obviously, it does not make much sense to use the (unencrypted) bulk part of the video, because every (non-paying) user receives it.

   The other two parts (watermarked and major, see Fig. 3) can also be used and it depends on the desired security level in order to decide which one to use. The strongest authentication can be achieved by using the watermarked part. In this case each image is individual and each user can authenticate only for the streams he can really decrypt. This strong authentication can be used in order to check whether each user has really decrypted the copy of the video stream he paid for. Usually, it would rather make sense to use the non-watermarked part (major). This latter one also needs to be decrypted by a user, but it would be possible for other users to copy that information and to authenticate.

   In any case, authentication is done by comparing the returned authentication data (e.g. the calculated hash code over the video stream) with the authentication data available at the policy server. If the watermarked part is used for authentication, however, the policy server must receive and process all individual streams of the video. This can create severe scalability problems, because the policy server must process all different sets of streams. In case of a large video service, it might therefore be better to perform the authentication with just a few fragments, i.e. not using the entire picture.

   The PATH message can be used for describing which parts of an image shall be used to calculate the authentication data. A special case of using the generated video stream for authentication is, when packets are encrypted independently of each other. In this case, authentication data can be calculated after decrypting single packets. Another issue are scalability concerns in case there are many video services to support. A single policy server might be come overloaded, but additional policy servers can solve this problem.
2. **Handshake method:** This approach is based on a request/reply mechanism without any video data packets. All relevant data are sent within the RSVP messages. For example, the same keys that are used for the video data encryption can be used for authentication and the messages challenging the client to generate authentication data are created by the content provider. While this solution is independent from the transported data, it it is difficult to support anonymous users.

4.3 QoS management

After a QoS broker has detected that there are authorised and / or non-authorised users on a branch, it has to trigger appropriate QoS management actions. We assume that QoS inside the network is provided by DiffServ [BBC+98]. Please note that RSVP is only used as an authentication signalling protocol but not for resource reservation. The QoS broker as part of an ISP's network controls and manages the routers inside the network. The QoS broker establishes QoS for each flow individually, but tries to aggregate different flows into the same DiffServ Expedited Forwarding class. This means that re-markers need to be installed at strategic points in the networks such as split points or egress routers in order to prevent that prioritised traffic is flowing along branches without authorised receivers. Note that any scheduling mechanisms are still working on a coarse level according to DiffServ. More information about the QoS broker and the DiffServ implementation is given in [SB03].

Fig. 7 shows the adaptation of a multicast tree. When the initial reservation for the service is established (a), the resource reservation is prepared on all links inside the network. A certain fraction of bandwidth might be reserved on each link for multicast data. As soon as the multicast tree has been established and the service has started (b), prioritised multicast data of the particular group flows only via the links marked by solid lines. At this point the entire multicast tree receives the same QoS. This state can be achieved without authentication mechanisms. Then, the user authentication can be initiated (c) to detect the branches with authorised / paying

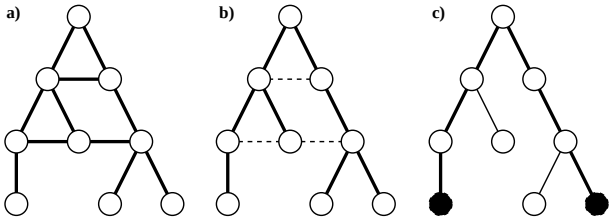

Fig. 7: The three steps from the initial request to the final resource reservation.

users (black). Only the bold branches leading to authorised users will carry prioritised multicast data traffic, while QoS support along the other branches might be deactivated by using appropriate marking mechanisms such as from EF to Best-Effort.

4.4 Discussion

The concept has been designed with special emphasis on scalability for large receiver sets. Although there are several entities that might have central roles for a particular network domain or even for the complete scenario, scalability should not be limited. The key server for example is mainly used prior to the video transmission service. The QoS broker will only be active in case of significant changes required on a branch of the multicast tree. The most critical component, the policy server, can easily be extended by additional processing elements to speedup the calculations needed for authentication. It is also important to note that it is not required to control joins and leaves of single receivers, but it is sufficient to know whether at least one authorised receiver is connected via a particular branch. Furthermore, the developed concept is in particular appropriate to support multi-domain scenarios. Using the video data based authentication, no keys need to be exchanged among service and network providers in order to authenticate a user. The only requirement is that an ISP needs to be able to decrypt the video transmission correctly.

5 Prototype implementation

5.1 Test scenarios

For prototype implementation and proof of concept the two test scenarios shown in Fig. 8 have been used. These scenarios are nearly identical and the differences are explained below. For these prototype scenarios the

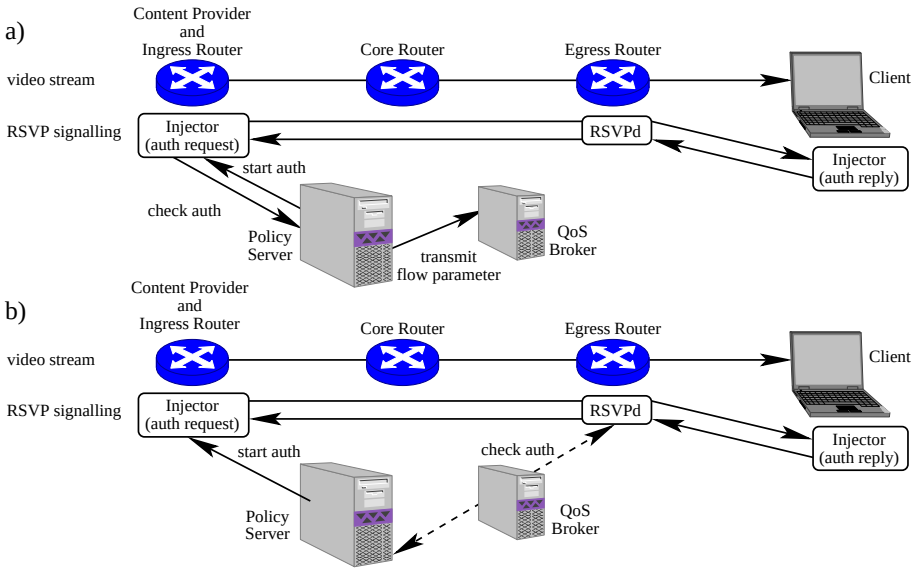

Fig. 8: The different tools that are used to make the authentication.

video stream is simplified in the manner that the video is transported in a single stream only. Authentication

is performed with information from the entire image. The responsible entity for controlling the authentication process is the policy server. To run the authentication a simple RSVP tool, called injector, has been used at the initiator and at the receiving client. The task of the injector tool is to generate authentication request and reply messages as part of RSVP PATH and RESV messages. Between the two ends the RSVP messages are processed by an extended RSVP daemon, running on the egress router. The difference between the two scenarios is, that in scenario a) the authentication request is sent from the first injector (initiator of the RSVP request) to the policy server and in scenario b) the authentication request is sent from the RSVP daemon at the egress router via the QoS broker to the policy server. The answer is returned along the reverse path to the RSVP daemon.

Scenario a) is useful when only a single ISP network is used. Otherwise the scenario b) should be used, because the control can be done at the border of each ISP network.

For example using the scenario depicted in Fig. 8.a) the authentication is done in the following order. First, the policy server asks the injector to initialise the authentication (start auth). The injector creates the request (PATH) and sends it to all clients. At the receiving client the response (RESV) is done automatically and forwarded back to the first injector. When the response arrives, the injector forwards it to the policy server (check auth). The policy server checks the authentication and if the client is allowed to use reserved resources, the policy server sends the flow parameters to the QoS broker. The QoS broker reconfigures the routers along the path if required.

To explain the prototype more exactly, the implementation can be divided into different parts. First, the modifications to RSVP are described, then the tool to generate the RSVP messages is presented. Finally, the connection between the RSVP protocol and the policy server is shown.

5.2 RSVP protocol extensions

First, we look at the adaptations that are needed inside the RSVP protocol. The advantage of RSVP is that it supports policy objects in the initial definition [BZB+97]. The detailed definition of policy data was done later [Her00]. The definition of policy information is flexible and allows the definition of new objects. The valid policy objects are divided into three ranges. The first and the second range are standardised. The third range is open for new and proprietary extensions. For the prototype only new policy objects have been added for transporting the picture numbers, packet numbers and authentication information. The size of the authentication information is chosen flexibly to permit the usage of different authentication data, like a part of the picture or only a short MD5 checksum of the picture.

5.3 Generating RSVP messages

The injector tool is used for initiating the RSVP messages and works directly on the RSVP layer. The tool has two working modes. The first mode called *ask* is used at the ingress router to generate authentication requests (PATH messages) including information required for authentication. To do this the tool receives from time to time the desired image number from the policy server. The second mode called *reply* mode is used at the receiving client. In this mode the injector automatically creates the corresponding authentication reply (RESV message) to the received PATH message. The injector uses a MD5 checksum of the picture to create the authentication information. Because it is not guaranteed that the request for the authentication arrives before the used image, the injector stores a list with the corresponding checksums of the last image. The list has 1024 elements, representing the last 40 seconds of the video stream. This list is updated after each new image.

5.4 Connection from RSVP daemon to policy server

The third entity in the RSVP message exchange is the RSVP daemon. The daemon is usually running on the egress router of an ISP only. If a RESV messages arrives, the daemon forwards the message via the QoS broker to the policy server. The policy server checks the received authentication information. If the authentication succeeds, the response is sent via the QoS broker back to the RSVP daemon. The QoS broker can decide based on the authentication result to activate or deactivate QoS support on a particular branch of multicast tree. Then, it forwards the result to the RSVP daemon, which might forward the RESV message to the next upstream RSVP node. If the authentication fails, the response is sent back to the RSVP daemon, which will discard the RESV message. The policy server only checks whether the authentication information is valid or not. The result indicates only whether the user is able to authenticate, but does not contain any information about the user himself. To store the information for the authentication, the policy server uses a list similar to the one of

the injector. To reduce the load of the policy server, collecting of the image areas can be done outside of the policy server. In this way the policy server uses the resources only for authentication.

5.5 Results of the prototype implementation

The prototype implementation demonstrates that the proposed authentication concept is working from a functional point of view and it has been used for performance measurements. To reduce the resource consumption overhead tools like the injector have been implemented from scratch. For the more complex tools like the video streaming application, the video viewer and for the RSVP daemon existing applications and implementations have been used. The RSVP daemon is based on the ISI implementation [ISI] and a RSVP over DiffServ gateway [BBBG00]. For the video distribution we use video streams and video applications developed by ourselves. The prototype was tested on a test network based on Linux-PCs with 533MHz AMD K6-2 CPUs. The injector was running on the same machine as the video streamer and the video viewer. It should be mentioned that the video viewer application, running on the receiver system and the policy server, uses about 50% of the CPU resources.

To test the performance of the prototype implementation, the injector and the policy server measure the time an action needs. For the policy server this means the time between the arrival of the request and the transmission of the response. For the injector the measurement depends on the mode. For the reply mode the time to receive the request and to generate the reply, and for the ask mode the time between emitting the request and the arrival of the response has been measured.

The prototype implementation has been tested with two different scenarios. First, the policy control is only done at the two ends of the connection (Fig. 8.a), i.e. at the sender or ingress router and the receiver. In this case admission control is done only when the reply returns to the sender. This scenario is used to find out via which egress router the paying receiver is connected and to measure the round trip time between the video streamer and viewer. The results for this scenario are

- that the injector tools do not use too much CPU time and so will not disturb the video streamer or viewer.
- that it takes 1.52 ms between a request and a corresponding response, measured on the sender. From this time, 0.60 ms are used to create the reply on the video viewer side. The rest is used to transmit the data (0.33 ms round trip time) and data processing on the sender that is similar to processing at the receiver. For processing the reply and for the authentication 0.60 ms are needed.

The second scenario (Fig. 8.b) uses all components of the prototype implementation. With this scenario it is possible to show whether the concept works and to identify performance problems. The results of this scenario are

- that the policy server needs only 0.42 ms to check if the authentication information is correct. The check is fast, because there is only a binary comparison between the policy server's information and the authentication information. If we extrapolate this value, we can assume that approximately 2000 requests can be handled per second. Furthermore, 60% of this time is used to refresh the database with authentication values.
- that the time used by the RSVP daemon is huge. Our measurements show that the round trip time between the video streamer and viewer increases to 50 ms. The RSVP daemon itself causes a delay of approximately 48 ms.

Another result of the test with this prototype is, that our own tools do not use too much resources and do not affect the video streaming service. But one problem still exists for the policy server: collecting the authentication information from the video stream must be done outside the policy server. Otherwise, if a viewer would be used for each video stream the policy server would be under heavy load.

6 Summary and outlook

In this paper a working admission control concept for a video streaming service in a DiffServ/IP multicast network has been presented. The concept is flexible and can also be used for other scenarios. The tests with the prototype implementation showed that there are also some details that can be improved. One example is to reduce the delay of the RSVP daemon. A solution might be to eliminate all the code which is not used

for our particular signalling purposes. Furthermore, this concept is not limited to admission control but can be extended to transport other information required by the QoS broker and/or the policy server. The authentication information is only a small part for which policy objects can be used.

7 Acknowledgements

This work has been performed in the framework of the project "QoS support for the Internet based on Intelligent Network Elements" (QuINE, Project no. 2000-066624.01/1), which is funded by the Swiss National Science Foundation (SNF).

References

[BB01] Levente Buttyan and Naouel Ben Salem. A Payment Scheme for Broadcast Multimedia Streams. In *Proceedings ot the 6th IEEE Symposium on Computers and Communications, Hammamet, Tunisia*, pages 668–673, July 2001. ISBN 0-7695-1177-5.

[BBBG00] Roland Balmer, Florian Baumgartner, Torsten Braun, and Manuel Günter. A Concept for RSVP over DiffServ. In *Proceedings of the 9th International Conference on Computer Communication and Network, Las Vegas, USA*, pages 412–417, October 2000. ISBN 0-7803-6494-5.

[BBC+98] S. Blake, D. Blake, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services, December 1998. Internet RFC 2475.

[BGB01] Roland Balmer, Manuel Günter, and Torsten Braun. Video Streaming in a DiffServ/IP Multicast Network. In *Proceedings of the Workshop of Advanced Internet Charging and QoS Technology at Informatik 2001, Vienna, Austria*, pages 159–165, September 2001. ISBN 3-85403-157-2.

[BW03] Roland Bless and Klaus Wehrle. IP Multicast in Differentiated Service Networks, August 2003. draft-bless-diffserv-multicast-07.txt.

[BZB+97] Bob. Braden, Lixia Zhang, Steve Berson, Shai Herzog, and Sugih Jamin. Resource Reservation Protocol, September 1997. Internet RFC 2205.

[CQNW02] Hao Chu, Lintian Qiao, Klara Nahrstedt, and Hua Wang. A Secure Multicast Protocol with Copyright Protection. *ACM Computer Communication Review Journal (ACM CCR)*, 32:42–60, April 2002.

[DGBS00] Gabriel Dermler, Manuel Günter, Torsten Braun, and Burkhard Stiller. Towards a Scalable System for Per-flow Charging in the Internet. In *Proceedings of the Applied Telecommunication Symposium, Washington D.C., U.S.A*, page 25, April 2000.

[GB99] Manuel Guenter and Torsten Braun. Evaluation of Bandwidth Broker Signaling. In *Proceedings of the International Conference on Network Protocols ICNP'99, IEEE Computer Society*, pages 145–152, November 1999. ISBN 0-7695-0412-4.

[Her00] Shai Herzog. RSVP Extension for Policy Control, January 2000. Internet RFC 2750.

[IETa] IETF. Multicast and Anycast Group Membership magma. `http://www.ietf.org/html.charters/magma-charter.html`.

[IETb] IETF. The Secure Multicast Research Group (SMuG). `http://www.securemulticast.org/smug-index.htm`.

[ISI] ISI. ISI RSVP implementation release 4.2a4-1. `ftp://ftp.isi.edu/rsvp/release/rsvpd.rel4.2a4-1.tar.gz`.

[IYT01] Norihiro Ishikawa, Nagatsugu Yamanouchi, and Osamu Takahashi. An Architecture for User Authentication of IP Multicast and its Implementation. IPSJ Journal Vol. 40, No. 10, May 2001.

[KT00] Dimitri Konstantas and Dimitris Thanos. Commercial Dissemination of Video over Open Networks: Issues and Approaches. Internet Objects, Centre Universitaire d'Informatique, University of Geneva, September 2000.

[Mar03] Bill Marshall. Private Session Initiation Protocol (SIP) Extensions for Media Authorization, January 2003. Internet RFC 3313.

[RGKS01] Utz Roedig, Manuel Görtz, Martin Karsten, and Ralf Steinmetz. RSVP as Firewall Signaling Protocol. In *Proceedings of the Sixth IEEE Symposium on Computers and Communications (ISCC'01)*, pages 57–64, July 2001.

[SB03] Günther Stattenberger and Torsten Braun. Performance of a Bandwidth Broker for DiffServ Networks, March 2003. Kommunikation in verteilten Systemen (KiVS03), Leipzig, Germany.

[TK01] Dimitris Thanos and Dimitri Konstantas. COiN-Video : A Model for the Commercial Dissemination of Video over Open Networks. In *Proceedings of the IBC 2001 Conference, RAI Amsterdam*, September 2001.

[YYP+01] Satyendra Yadav, Raj Yavatkar, Ramesh Pabbati, Peter Ford, Tim Moore, Shai Herzog, and Rodney Hess. Identity Representation for RSVP, October 2001. Internet RFC 3182.