Quality-of-Service Support for IP Flows over ATM

Torsten Braun and Stefano Giorcelli
IBM European Networking Center
Vangerowstr. 18
D-69115 Heidelberg
Phone: +49 6221 59-4352
Fax: +49 6221 59-3300
Email: braun@heidelbg.ibm.com

Abstract. This paper describes an extension of a classical IP over ATM implementation which allows to support quality-of-service for IP flows. While standard classical IP over ATM implementations use shared VCs for several IP flows between two end systems, the classical IP over ATM implementation extensions allow to establish separate VCs with individual quality-of-service parameters for single IP flows. The paper describes the modifications of an AIX 4.2 kernel implementation. Mbone video applications using RSVP for resource reservation have been implemented on top of the classical IP over ATM implementation.

1 Introduction

An IP flow is a sequence of packets exchanged between two end points identified by the pair of IP address and port number. A flow is identified by the so-called flow identifier. In IPv4, this is the combination of the two end point identifiers, while in IPv6 the flow identifier may consist of the flow label and the source IP address.

ATM connection establishment following the classical IP over ATM concept [15] means that a single, shared best-effort virtual connection (VC) is established for several flows between any pair of ATM attached IP systems. The single VC is established with quality-of-service (QoS) parameters supporting best-effort service according to [17]. In some situations it is desirable to establish and use a dedicated VC for a flow, in particular for data flows with high quality-of-service requirements. Another application of using dedicated VCs for IP flows is a RSVP based environment. The Resource ReSerVation Protocol (RSVP) [7] is being used in the Internet world to reserve resources in order to provide quality-of-service. ATM VCs can be considered as a very important resource for quality-of-service guarantees. Signalling parameters for ATM VC establishment can be derived from quality-of-service parameters of IP flows exchanged between RSVP systems.

The paper describes an implementation architecture to provide ATM QoS support for IP flows. The architecture is based on a classical IP over ATM implementation, which has been extended to support QoS on application level. System calls have been implemented, which allow to specify traffic descriptor parameters to be used in SETUP messages for VC establishment. If a dedicated VC exists for a flow, the packets of the flow are sent over this VC, otherwise a best-effort VC is used, which must be shared with other flows. The advantage of this approach is that applications need only slightly to be modified to take advantage of the QoS extensions of the classical IP over ATM implementation. Internet addresses can still be used, only additional function calls specifying the desired quality-of-service are required. On transport level, TCP and UDP are currently supported. The architecture is open to other protocols over IP.

After an overview about IP over ATM in Section 2, the implementation architecture of the QoS extensions of the classical IP over ATM implementation is illustrated in Section 3. Section 4 describes applications which can use the QoS extensions. Section 5 gives an outlook to future work.

2 IP over ATM

2.1 Classical IP over ATM

The original Internet architecture (classical IP model) assumes an interconnection of networks. Each network can be divided into many subnetworks (IP subnets). According to the classical IP model, hosts within the same subnet can communicate directly, while hosts on different subnets must exchange packets via routers. When a host wishes to communicate with another host, the destination address is analyzed whether the destination is local (host on the same subnet) or remote (host on a different subnet). In the first case, the packet can be sent directly to the destination, while in the second case it has to be sent to a router, which is has to forward it to the next hop.

These concepts are applied with slight modifications also in the case of ATM subnets. In particular, the notion of Logical IP Subnet (LIS) has been introduced. A LIS consists of a group of hosts and routers which are connected to the same ATM network and belong to the same IP subnet. Only hosts connected to the same LIS communicate directly with each other, while routers are used to reach destinations outside the LIS. Two protocols are used to map IP addresses to ATM addresses within a LIS. The ATM Address Resolution Protocol (ATM ARP) resolves IP addresses to ATM addresses. The Inverse ATM Address Resolution Protocol resolves ATM addresses to IP addresses.

The ATM ARP is based on a centralized approach. Each LIS has an ATM ARP server, which maintains a table of IP and ATM address pairs. The ATM ARP server address has to be statically configured in each host within the LIS. When a host first connects to a LIS, it establishes a connection to the ATM ARP server using the configured address. In response to this connection establishment, the ATM ARP server sends an Inverse ARP request to the host, in order to determine its IP address. The host replies with an Inverse ARP reply, and the resulting address binding is then stored in the server's ATM ARP table.

If a host A wants to send a packet to a host B, it first checks whether the destination IP address is local or not. IF host B resides on the same LIS, a connection can be directly established to it. Host A looks for the IP address of host B in its local ATM ARP table, which stores mappings between IP and ATM addresses. If it is the first packet for host B, no entry can be found in the local ATM ARP table. Therefore, host A must request the ATM address of host B from the ATM ARP server. Host A sends an ATMARP_request to the ATM ARP server and receives from it an ATMARP_reply message containing the ATM address of host B (Figure 1). Host A stores the address mapping in its cache and establishes an ATM VC to host B by an SETUP message. The SETUP message contains quality-of-service parameters for a best-effort service according to [17]. The establishment of the VC requires the reception of a CONNECT message in response to the SETUP message. The following packets from host A to host B are sent via the best-effort VC as soon as it is established.

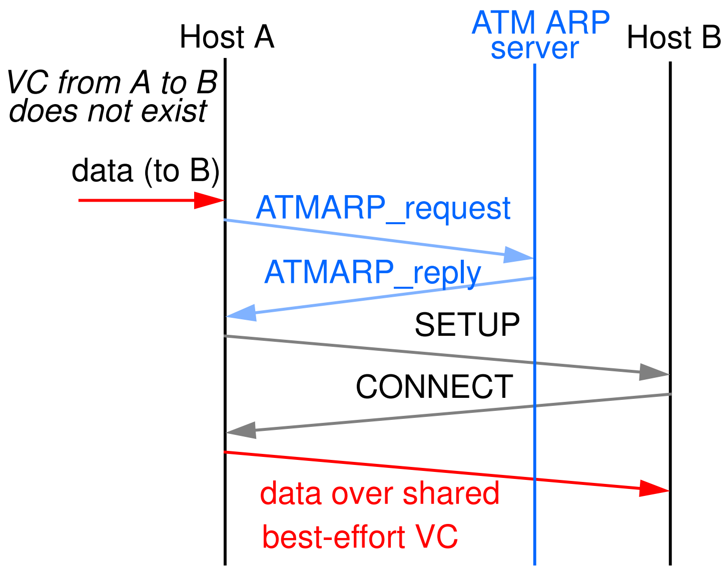

Figure 1: Classical IP over ATM and ATM ARP

The classical IP over ATM model suffers from several limitations. One limitation is that direct VCs between hosts are only established between hosts of a common logical IP subnet. If two hosts do not belong to a common IP subnet, IP traffic must be forwarded via routers although the ATM network infrastructure would allow a direct ATM VC between the two hosts. This short-cut is not allowed by the classical model, so that resulting routes are not optimal, and routers may become performance bottlenecks. The short-cut functionality is enabled by the Next Hop Resolution Protocol (NHRP) [14], developed by the IETF routing over large clouds working group. NHRP allows hosts or routers to find the ATM address of the next hop, given the IP address of the destination. It minimizes the number of hops and allows hosts residing on the same ATM network to establish a direct VC although the hosts are not members of a common IP subnetwork.

2.2 Quality-of-Service support

Another problem of the classical IP over ATM approach is that it does not support quality-of-service on application or IP flow level. This problem is addressed by the Application REquested IP over ATM (AREQUIPA) proposal [1]. It allows ATM attached hosts to set up single ATM connections with QoS requested for dedicated IP flows. Applications can directly benefit from ATM ability to guarantee quality of service. The approach introduces three new API functions in the TCP/IP protocol suite `Arequipa_expect`, `Arequipa_preset` and `Arequipa_close`.

If host A wants to establish a connection to host B, the hosts first exchange their ATM addresses and the port numbers. Host B then invokes `Arequipa_expect` to accept an incoming ATM connection from host A. This function requires as parameters the socket descriptor, the IP address of host A, the port number and the ATM address. Host A invokes

`Arequipa_preset` to open an ATM VC to host B. This function requires a socket descriptor, host B's IP address, port number, ATM address, and QoS parameters. AREQUIPA functions cannot be performed on a router because they do not include complete information about the flow identifier of transmitted data. Only the remote address is passed to the functions, while a router would have to set also the source addresses. The AREQUIPA approach only works in contexts where ATM connectivity is provided end-to-end.

A different approach are interfaces providing the application direct access to ATM (so-called native ATM interfaces). Applications are running directly over AAL5 but not over TCP/IP protocols. Several solutions use existing socket function calls [12][13], where ATM addresses instead of the Internet addresses are used. The socket calls have to be modified and applications have to get and set desired quality-of-service parameters of the ATM connection to be established. Native ATM interfaces are limited to a homogeneous ATM network, and the application also needs to know the ATM address of the receiver host. This can be avoided by adding ATM addresses into the domain name system data base. Another advantage of native ATM sockets is reduced processing overhead due to missing IP and UDP/TCP processing compared to UDP/TCP sockets. However, in [20] it is shown that there are only minor performance differences between a UDP/IP protocol stack and native ATM sockets. A drawback of native ATM interfaces is the lack of reliable services such as required for WWW applications.

To allow applications to specify their QoS requirements the resource reservation protocol (RSVP) has been developed by the IETF [7]. It exchanges QoS requests between end systems and routers. RSVP runs on top of IP or UDP. Similar to ICMP, IGMP or routing protocols, it does not carry user data. RSVP has been designed for simplex data flows. Resources are reserved in only one direction. RSVP is no routing protocol, it obtains routes from a routing protocol and uses this information to forward QoS requests along paths. Within routers, incoming packets are classified by a packet classifier, which looks for the routes and the QoS class for each packet. The packet scheduler of the outgoing interface passes the packets to the underlying link layer providing the appropriate QoS. RSVP allows dynamic QoS reservations. Receivers may request or change resource reservations at any time. The architecture described hereafter can be used to run RSVP over ATM [5]. ATM resources can be reserved for IP flows based on RSVP message exchange. This results in the establishment of QoS-based ATM VCs.

3 Implementation of IP over ATM with quality-of-service-support

3.1 Implementation architecture

The architecture of the classical IP over ATM implementation with QoS support is shown in Figure 2. A1, A2 and A3 are three applications having access to a shared, best-effort VC established by the classical IP over ATM implementation. At a given time, the applications might require a QoS-based ATM VC, either through a RSVP reservation, or directly through an I/O control (ioctl) call to the kernel. After the private QoS-based VC has been established, it is exclusively used for the requesting application. The best-effort VC is kept up for other IP flows after the dedicated VCs are established. The extensions of the classical IP over ATM implementation extension can be divided into four parts:

- the administration of an additional ATM ARP table together with the definition of new data structures to store QoS and flow identifier information (ATM ARP QoS table),

- the introduction of new ioctl commands to set, get, and delete entries from this ATM ARP QoS table,
- the modification of the sender part of the classical IP over ATM implementation taking into account the possible presence of a dedicated VC, and
- the modification of the receiver part of the classical IP over ATM implementation to avoid sending Inverse ARP requests for unidirectional VCs.

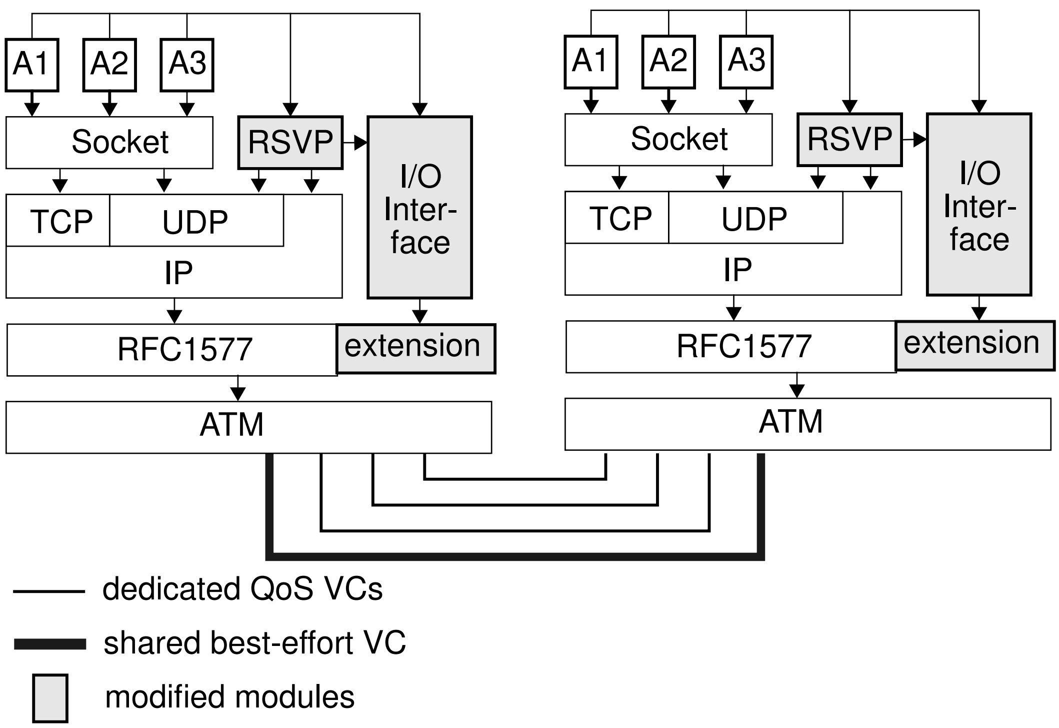

Figure 2: QoS extensions of classical IP over implementation

3.1.1 ATM ARP QoS table

The normal classical IP over ATM implementation builds an ATM ARP table to store the ATM addresses of end systems, together with their IP address and a VC identifier identifying the VC to be used for sending IP datagrams to the specified destination. Several internal functions of the classical IP over ATM implementation access and modify the contents of this table. In order to allow applications or reservation protocols such as RSVP to request a special quality-of-service to be used for a certain IP flow, a second ATM ARP table has been added in addition to the standard ATM ARP table. This ATM ARP QoS table stores information such as the flow identifiers and the corresponding QoS parameters. The functions to access and modify this table and the actions resulting from the modifications are described in the next subsection.

The standard ATM ARP table consists of an array of pointers to double-linked lists indexed through a hash function. To allow the same handling procedures on the ATM ARP QoS entries, the additional ATM ARP QoS table has been designed with a similar format as the standard one (see Figure 3). The two different tables are necessary because the default, best-effort VCs need to be kept up even after the establishment of private QoS connections to allow other flows and best-effort messages to be exchanged.

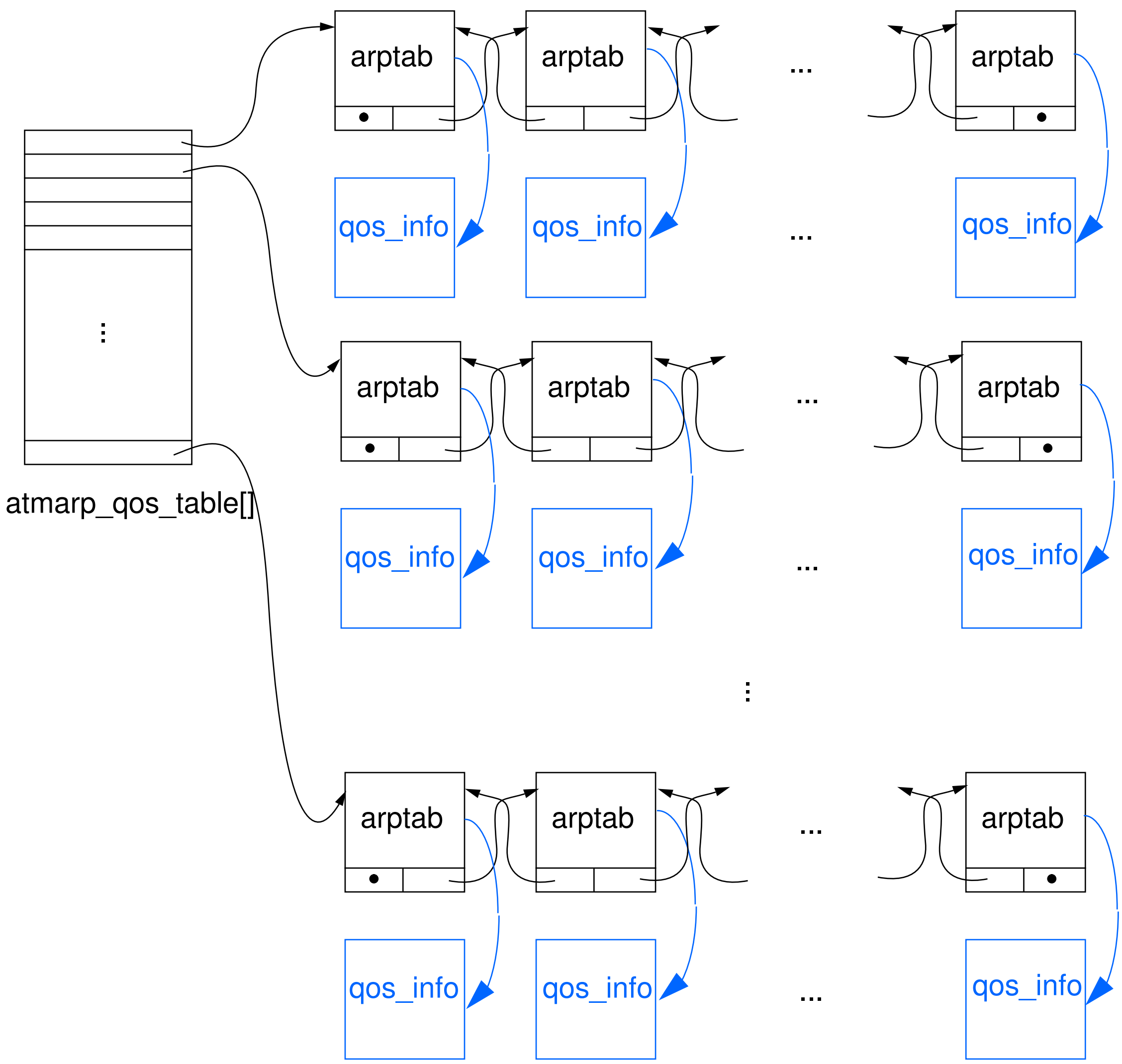

Figure 3: QoS-ATMARP table

To store the information regarding the flow identifier and QoS parameters in each of these ARP table entries, a subfield within this structure is used as a pointer to a new structure, called `qos_info`. The definition of this structure is given below:

```
struct qos_info {
 flowid_t flowid;
 c_par_t conn_par;
 cause_t cause;
};

typedef struct flowid {
 struct in_addr srcaddr;
 int srcport;
 struct in_addr dstaddr;
 int dstport;
} flowid_t;
```

```
typedef struct conn_par {
 int fwd_peakrate_hp;
 int fwd_peakrate_lp;
 int fwd_sus_rate_hp;
 int fwd_sus_rate_lp;
 int fwd_bur_size_hp;
 int fwd_bur_size_lp;
 int best_effort;
} c_par_t;
```

The `c_par_t` type includes a subset of the standard ATM QoS parameters to set up a certain ATM connection with the QoS parameters requested by the application. It could easily be enhanced by additional parameters. The `flow_id_t` type has been defined for the use in IPv4 and contains the IP address and the port number for the source and the destination. The `cause_t` type of the `qos_info` structure contains in the case of a connection setup failure information about the reason for which it was not possible to establish the connection.

3.2 Application programming interface

New I/O control commands allow to create, modify, read, and delete entries in the additional ATM ARP QoS table maintained in the kernel. The user interface to access the table consists of additional commands for the ioctl function.

```
int ioctl (FileDescriptor, Command, ifr)
int FileDescriptor, Command;
struct ifreq *ifr;
```

An application should set an entry for the requesting IP flow in this ATM ARP QoS table through an `ioctl` call to the kernel using the **`ATM_SQOS`** command. A RSVP implementation sets this entry after receiving the flow specification parameters from the next hop by a reservation message. An entry with the same flow identifier must not be already present in the table. However, an entry with the same destination address is required in the normal ATM ARP table. In a RSVP scenario this assumption is always true. When the RSVP reservation message is received, RSVP path messages have been sent by the source over the best-effort VC before and the destination ATM address is already stored in the ATM ARP table. This entry can be used in order to complete the ATM ARP QoS table entry with the destination ATM address. The `ATM_SQOS` command requires the parameters next hop address, flow identifier and ATM connection parameters.

The **`ATM_GQOS`** command gets (reads) an entry from the ATM ARP QoS table corresponding to a specific flow identifier given as a parameter. For example, when RSVP reservation messages arrive at a sending RSVP node, one has to check using the `ATM_GQOS` command, whether there is already an ATM ARP QoS table entry for the flow specified in the RSVP reservation message.

The **`ATM_DQOS`** command deletes an entry from the ATM ARP QoS table. Deleting the entry for a certain IP flow in the ATM ARP QoS table results in the termination of the corresponding VC. Packets of the deleted flow will again be sent over the best-effort VC. The `ATM_GQOS` and the `ATM_DQOS` commands need the next hop address and the flow identifier as parameters.

The **ATM_DUMPQOS** command dumps the contents of the ATM ARP QoS table. It is mainly used for debugging and monitoring purposes.

3.3 Sender modifications

When a data packet for a certain flow has to be sent, the extended classical IP over ATM implementation looks if there is an entry for the flow in the ATM ARP QoS table. If an entry is found and a VC does not exist, it tries to establish the VC by sending a `SETUP` message. In the meantime it sends the packets of the flow for which the connection is being established over the best-effort VC to the destination. As soon as the dedicated VC is established, packets belonging to the flow will be sent over the dedicated VC. For a packet to be sent, the following operations are performed (Figure 5):

- If no ATM ARP QoS table entry is found for the destination address, the normal ATM ARP operations are performed.
- If at least one entry is found, the complete flow identifier is extracted from the packet headers. The source and destination addresses are extracted from the IP header, the source and destination ports from the TCP/UDP header. Only TCP and UDP protocols are currently supported. Another lookup is performed on the ATM ARP QoS table for the complete flow identifier.
  - If no ATM ARP QoS table entry is found, the packet is sent over the best-effort VC.
  - If the ATM ARP QoS table entry is found, there are three cases:
    - If the VC is `UP` the packet is sent over the dedicated QoS VC.
    - If the VC is `CONNECTING` the packet is sent over the best-effort VC.
    - If the VC is `DOWN` a function call is performed in order to get the VC connected, and the packet is sent over the best-effort VC.

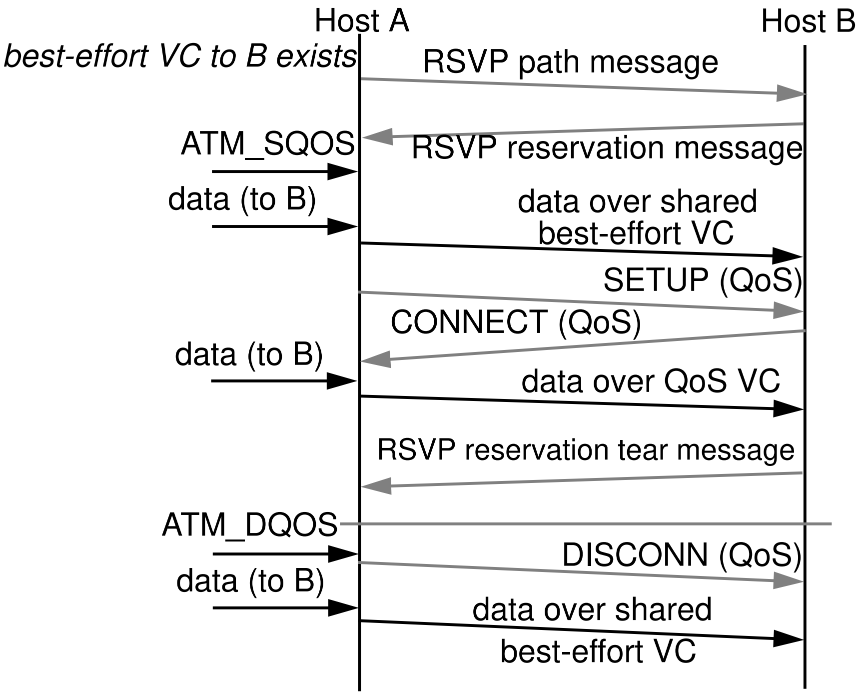

Figure 4: VC setup for IP flows with QoS requirements

The number of additional operations of the sending routine have been kept very low compared to the original classical IP over ATM implementation in order to minimize the performance overhead.

Figure 4 shows the behavior of the sender part of the classical IP over ATM implementation. Data are sent over the QoS VC as soon as it is established. VC establishment is triggered by an `ATM_SQOS` command, e.g. after an RSVP message exchange. VCs are deleted by the `ATM_-DQOS` command, e.g. after tearing the RSVP reservation. After deleting the VC data are sent over the best-effort VC again. The QoS parameters of the ATM ARP QoS entries, which have been configured by an `ATM_SQOS` command, are used for ATM connection setup. The connection parameters, which are included in the `qos_info` structure, are copied into the corresponding fields of the SETUP message.

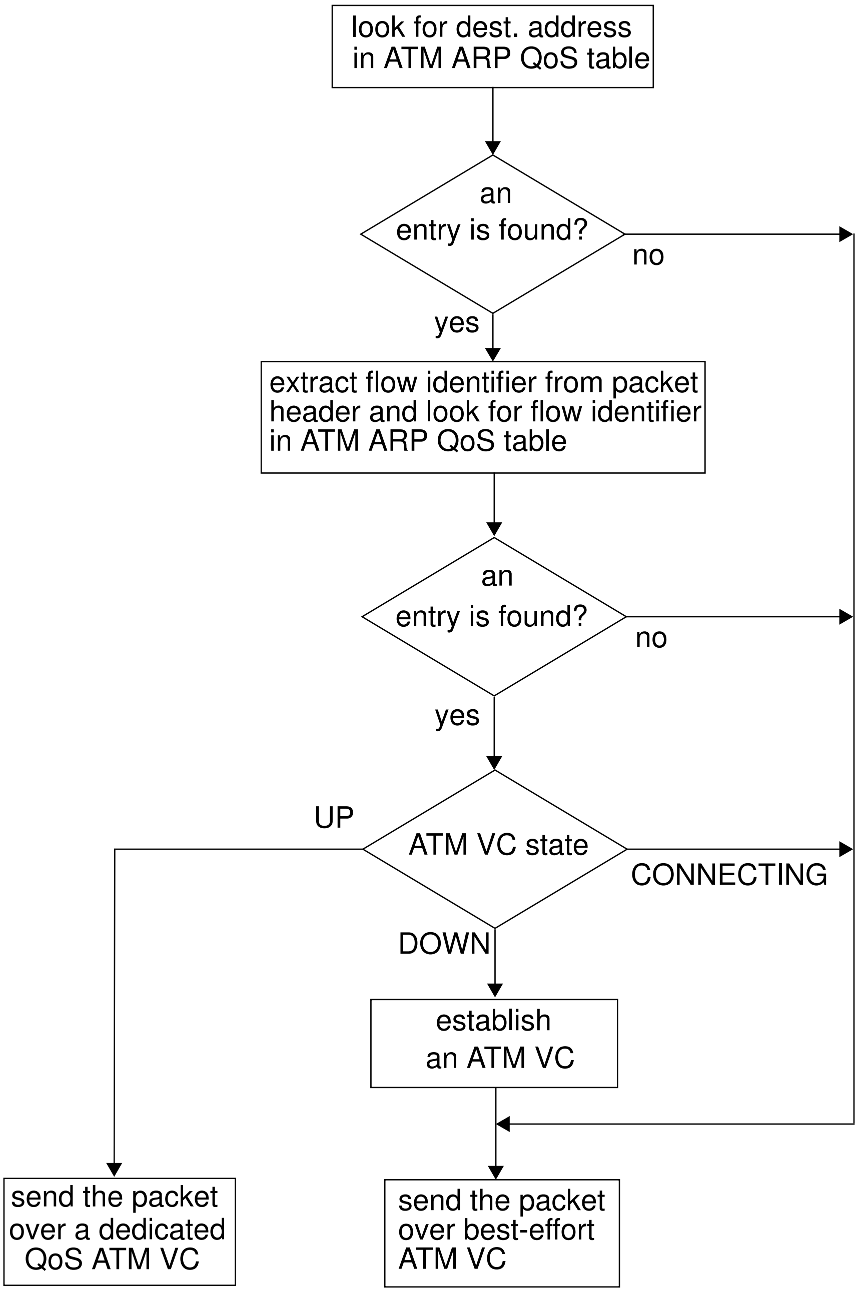

Figure 5: QoS extensions of the sender part of the classical IP over ATM implementation

3.4 Receiver modifications

Handling of incoming calls has also been modified, i.e. the classical IP over ATM implementation had to be adapted. In a normal classical IP over ATM implementation an Inverse ARP message is sent back to the calling system in order to find out its IP address when a connection establishment request is received. Since the additional VCs with QoS support are unidirectional (as RSVP reservations are unidirectional), this Inverse ARP message will not be sent for QoS VCs. QoS VCs are identified by having the broadband Bearer Capability class set to C (for best-effort VCs the used class is X), and the backward sustainable cell rate set to 0 (unidirectional VC).

4 Applications

The extensions to the classical IP over ATM implementation allow user-level programs to establish dedicated VCs for IP flows with specific QoS. Thus, it becomes possible for a resource reservation protocol or for a user-level application to benefit from QoS guarantees across an IP over ATM network. In this section we describe how to use the classical IP over ATM extensions for both of the cases. First, we describe Mbone applications using RSVP over ATM. Second, we explain how to design applications in order to use directly the quality-of-service interface.

4.1 Mbone Applications and RSVP over ATM

The network video (NV) [10] tool and the video conferencing tool (vic) [16] are one of numerous video conferencing tools developed for the use over the IETF multicast backbone (Mbone). Both tools perform resource reservations calling the RSVP daemon [21] using the RSVP API (RAPI) [8]. While NV had already been prepared for the use over RSVP, we adapted vic for its use in an RSVP environment [9]. Currently, only unicast communication is supported. The vic menu has been modified allowing the receiver to specify the desired bandwidth. The actual transmission rate is advertised in the path message according to [18] and displayed at the receiver. The receiving user is then able to select the reserved bandwidth that determines the settings of the QoS parameters in the reservation messages.

The RSVP daemon runs in user space and is designed to communicate with a traffic control kernel in order to perform the resource reservations. In order to make use of the features provided by the classical IP over ATM extensions, the RSVP daemon implementation had to be modified. The RSVP daemon, which is designed for routers and end systems, has three major interfaces:

- Applications communicate with the RSVP daemon via the RSVP Application Programming Interface (RAPI) and an UNIX pipe. The RAPI library routines linked to the application allow an application to create a RAPI session, to make reservations, or to delete them. E.g., a receiver application uses `rapi_reserve` calls to establish or modify reservations for the session. The RSVP daemon generates, sends, receives, and processes RSVP messages.
- A general framework known as RSRR (Routing Support for Resource Reservation) is used as interface to routing protocols. It includes an interface to the multicast routing code and an unicast routing interface to the kernel routing table.

- The RSVP daemon communicates with a traffic control kernel through a traffic control (TC) adaptation module which is a part of the RSVP daemon. The RSVP daemon can be used to establish reservation states in hosts and routers. The daemon may pass this state to the traffic control kernel via the kernel-specific TC adaptation module. The packet classifier, packet scheduler and the admission control module can be considered as parts of a traffic control kernel.

The original release contained a TC adaptation module for the MIT ISPS kernel. In order to perform the additionally defined ioctl calls of the kernel extensions and to reserve ATM resources, a new TC adaptation module for ATM has been implemented. The RSVP daemon cooperates with the TC adaptation module through several function calls. Their interfaces have been nearly kept unchanged compared to the original release.

Within the resulting implementation the Mbone application calls the RSVP daemon by RAPI calls. The RSVP daemon in turn performs function calls to the TC adaptation module. The TC adaptation module then establishes dedicated QoS ATM VCs through the classical IP over ATM implementation extensions. This scenario is shown in Figure 6. The TC adaptation module is the interface between the RSVP daemon and the classical IP over ATM QoS extensions. The TC adaptation module provides several interface functions.

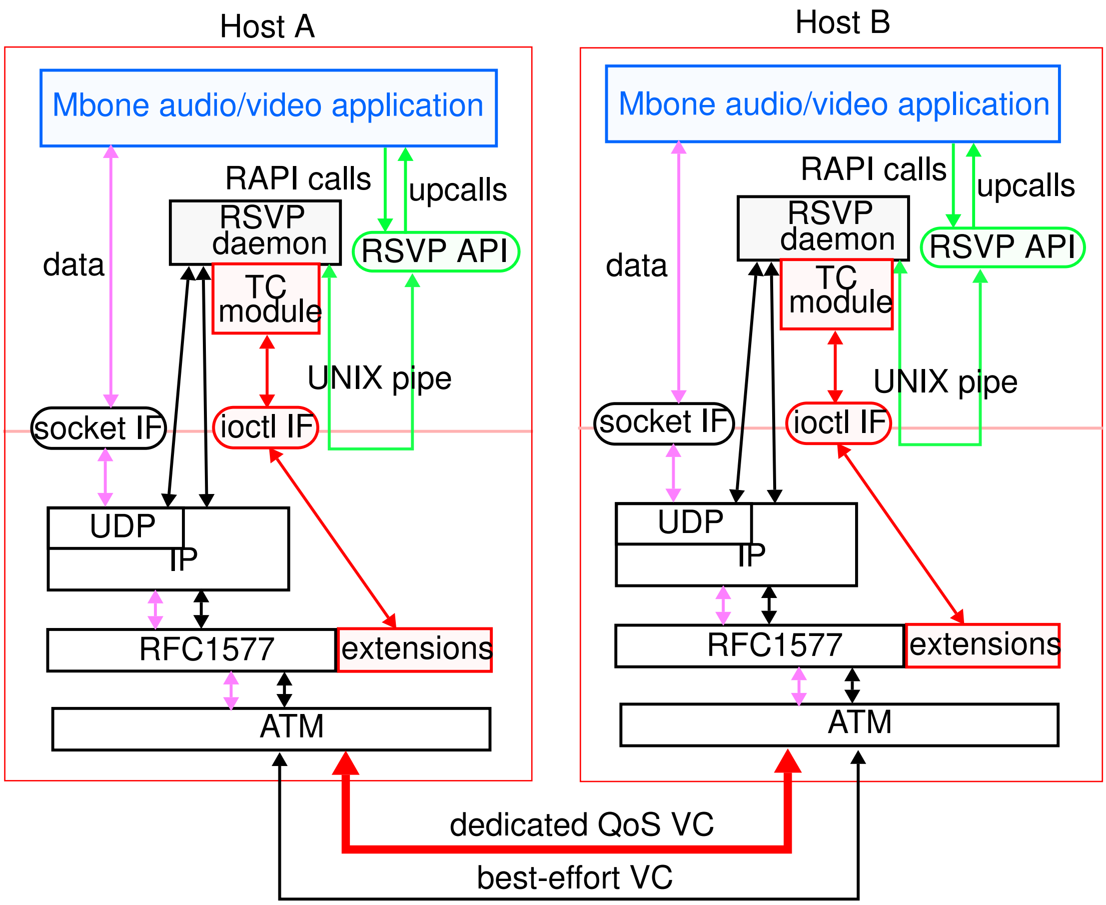

Figure 6: Mbone multimedia application over RSVP and ATM

A RAPI call at the receiver (host B) to `rapi_reserve` triggers the sending of a reservation message to the sender (host A). After receiving a reservation message the RSVP daemon at host A calls the traffic control function **`TC_AddFlowspec`** using the parameters desired flow specification and sender traffic specification. The function translates the QoS parameters into ATM connection parameters, and returns a pointer to the created structure as a handle to the RSVP daemon. QoS mapping between IETF service classes and ATM connection parameters is described in [6].

The RSVP daemon then calls the **`TC_SetFilter`** function within the TC module, in order to associate the entry created by `TC_AddFlowspec` to a specific IP flow, and to add the complete entry to the ATM ARP QoS table. The `TC_SetFilter` function performs an `ATM_GQOS` ioctl call, in order to check whether a private VC already exists for this flow. If no entry is found, an `ATM_SQOS` can be performed, and the VC is prepared to be established. As soon as a data packet has to be sent by host A, the setup procedure for the private VC is triggered, while the packet is sent over the best-effort VC. When the VC is fully established, packets belonging to this specific flow are sent over the dedicated QoS VC.

If host B now changes its reservation (e.g., asking for a higher bandwidth), a new RSVP reservation message is sent to host A. At host A, the **`TC_ModFlowspec`** function is called by the RSVP daemon, with the handle and the modified flow specification as parameters. This function first checks the existence of the entry passed as a parameter within the ATM ARP QoS table (through a `ATM_GQOS` call). If the entry is found, it deletes it through an `ATM_DQOS` call, thus tearing down the private VC. It then translates the new QoS parameters into ATM connection parameters. Finally, it calls `ATM_SQOS` to set a new entry in the ATM ARP QoS table with the new parameters. Again, when a packet needs to be sent by host A, a new VC with the modified ATM parameters is established.

If host B finally wants to delete a reservation, a RSVP reservation tear message is sent to host A. This implies that the RSVP daemon calls the **`TC_DelFlowspec`** function within the TC module, giving the handle as a parameter. The `TC_DelFlowspec` function deletes the entry from the ATM ARP QoS table through an ioctl call to `ATM_DQOS`. This also implies that the corresponding VC is torn down.

4.2 Application-requested QoS-based VCs

Besides the utilization through RSVP, the kernel extensions might be used directly by applications requesting a dedicated pipe across the ATM network. The new ioctl commands provided by the classical IP over ATM extensions can be accessed by any user-level program. Guidelines for a generic application wishing to open and use a QoS-based ATM VC are given in the following.

Assume that the sender and the receiver both reside on the same LIS and that they use classical IP over ATM to exchange IP packets through an ATM network. At a given time, the sender may want to guarantee that the data to be sent will benefit from a certain QoS. This requires to open a dedicated ATM VC. This can be set up by means of the ioctl options provided by the classical IP over ATM extensions. The sender has to perform the following procedure.

- First, a socket must be opened to allow the ioctl function to be called.
- A data structure containing several parameters for the `ATM_GQOS` call must be filled with the destination IP address, the complete flow identifier (source and destination IP addresses and port numbers), and the QoS parameters for the ATM connection such as peak rate, sustainable rate, and maximum burst size.

- An ATM_GQOS ioctl call must be performed in order to make sure that no other dedicated VC already exists for the specified flow.
- An ATM_SQOS ioctl must be performed to set an entry in the ATM ARP QoS table and to prepare the kernel to establish the private VC.
- When the first data packet for the specified flow has to be sent, the kernel extensions will try to set up the VC, while sending the first packet over the best-effort VC. Once the connection is established, packets will start to flow over the private VC established with dedicated QoS parameters.
- If the sender wants to close the private VC, it must call a ATM_DQOS ioctl command. The ATM connection is released and the corresponding entry is removed from the ATM ARP QoS table.

No modifications of the procedure to send data are required. The socket interface is still being used with no additional parameters. The ioctl calls can be performed at any time during a socket connection, and they influence only packets sent after an ATM_SQOS call. No modifications are required to the application on the receiver side. It will keep on receiving data through the same socket without any modification.

5 Future work

The classical IP over ATM implementation does currently not support IP multicast. IP multicast over ATM is still a matter of specification [2]. The implementation architecture is prepared to support multicast and only minor modifications are needed to adapt it. In particular, the ATM ARP QoS table entries should be modified in order to store a list of ATM addresses, instead of a single one to be able to establish point-to-multipoint VCs. The shared explicit reservation style is not implemented in the used RSVP implementation. The kernel extensions do not support it as well. In order to support this style, slight modifications would be needed in the data structures, i.e. the table entry should include a list of senders, instead of a single sender [11].

An important issue which is currently under study at the IETF is the QoS parameters mapping between IETF service classes and ATM service classes [6]. The implementation does not focus specifically on this topic, which is left for future improvements. However, functions calls to translate the QoS are provided within the TC adaptation module of the RSVP daemon, such that only the body of these functions must be modified to change the QoS mapping. The QoS mapping is done in the user space. This simplifies to evaluate different schemes to map IETF integrated services classes and parameters [4][19][22] to ATM parameters [3].

The architecture presented in this paper is a good basis for experiments. Several functional extensions are under discussion. It may be desirable to establish a VC immediately after calling an ATM_SQOS command and not together with the first data packet. This allows that data of the supported flow can be exchanged earlier over the QoS VC. However, if there are no data to exchange directly after the ATM_SQOS command, the early VC setup would waste ATM resources. Another option would be a command to modify the QoS parameters of a flow. In that case, the old VC is torn down after an additional VC with the modified QoS parameters has been established. This avoids that data are exchanged over the best-effort VC between releasing the old VC and the availability of the new VC. However, the probability that the new VC can not be set up because of missing resources increases. Given that a switch can support X Mbps of a single port, modifying the QoS requirements from Y Mbps to Z Mbps is not possible using a modify command as proposed above, if Y + Z > X.

6 Conclusions

The implemented classical IP over ATM implementation extensions allow any application to establish and use dedicated VCs across an ATM network through a well-defined interface, while the standard TCP-UDP/IP socket calls remain unchanged. The QoS VC establishment can be made at any time during the data transmission, and the existing sockets need not to be closed and re-opened. Implementation in routers is possible, and heterogeneous IP over ATM networks are supported through utilization of RSVP. The QoS-based VC management is based on well-defined interfaces, such as new ioctl commands or the RSVP API. The only obvious disadvantage compared to native ATM interfaces consists in the lack of direct VCs between end systems on different LISs. Integrating NHRP into the protocol architecture would eliminate this drawback. Several advantages over native ATM interfaces make the proposed approach attractive to allow QoS support for existing Internet applications over ATM.

7 References

[1] W. Almesberger, J.Y. LeBoudec, P. Oechslin: Application REQuested IP over ATM (AREQUIPA), Internet Draft, June 1996

[2] G. Armitage: Support for Multicast over UNI 3.0/3.1 based ATM Networks, February 1996

[3] ATM Forum Technical Committee: User-Network Interface Specification Version 3.1, ATM Forum, September 1994

[4] F. Baker, R. Guerin, D. Kandlur: Specification of Committed Rate Quality-of-Service, Internet Draft, June 1996

[5] S. Berson: IP Integrated Services Support in ATM, Internet Draft, September 1996

[6] M. Borden: Interoperation of Controlled-Load and Guaranteed Service with ATM, Internet Draft, September 1996

[7] R. Braden, L. Zhang, S. Berson, S. Herzog, S. Jamin: Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification, Internet Draft, May 1996

[8] R. Braden, D. Hoffman: RSVP Application Programming Interface (RAPI) for SunOS/BSD, Internet Draft, August 1995

[9] S. Bretz: Integration of a RSVP Resource Reservation Module into a MBone Multimedia Application, Diploma Thesis, BA Mannheim, September 1996

[10] R. Frederick: Experiences with real-time software video compression, ftp://parcftp.xerox.com/pub/net-research/nv-paper.ps, 1994

[11] S. Giorcelli: RSVP over Implementation over ATM, Professional Thesis, Eurecom Sophia-Antipolis, July 1996

[12] W. J. Hymas, H. Stüttgen, S. Sharma, S. Wise: Socket Extensions for Native ATM, IEEE ATM ‘96 Workshop, San Francisco, August 26-28, 1996

[13] D. Kandlur, D. Saha, M. Willebeek-LeMair: Protocol Architecture for Multimedia Applications over ATM Networks, ACM Computer Communication Review, Vol. 25, No. 3, July 1995, pp. 33-43

[14] D. Katz, D. Piscitello, J. Luciani: NBMA Next Hop Resolution Protocol (NHRP), Internet Draft, September 1996

[15] M. Laubach: Classical IP and ARP over ATM, RFC 1577, January 1994

[16] S. McCanne, V. Jacobson: vic: A Flexible Framework Framework for Packet Video, ACM Multimedia ’95

[17] M. Perez, F. Liaw, A. Mankin, E. Hoffman, D. Grossman, A. Malis: ATM Signaling Support for IP over ATM, RFC 1755, February 1995

[18] S. Shenker, L. Breslau: Two Issues in Reservation Establishment, ACM SIGCOMM ‘95, pp. 14-26

[19] S. Shenker, C. Partridge, R. Guerin: Specification of Guaranteed Quality of Service, Internet Draft, August 1996

[20] R.K. Singh, S.G. Tell, and S.J. Bharat: Comparison of Raw and Internet Protocols in a HIPPI/ATM/SONET Based Gigabit Network, ACM Computer Communication Review, Vol. 26, No. 1, January 1996, pp. 18-28

[21] URL: ftp://ftp.isi.edu/rsvp/release

[22] J. Wroclawski: Specification of the Controlled-Load Network Element Service, Internet Draft, June 1996