Quality of Service and Active Networking on Virtual Router Topologies

Florian Baumgartner and Torsten Braun

Institute of Computer Science and Applied Mathematics,
University of Berne, Neubrückstrasse 10, CH-3012 Berne, Switzerland
[baumgart, braun]@iam.unibe.ch

Abstract. Currently two main approaches exist to realize Quality of Services in the Internet. While Integrated Services are based end to end sessions, Differentiated Services focus on traffic aggregates. Additionally, architectures on higher levels like Bandwidth Brokers are developed. All these concepts have their pros and cons and a seamless interaction is desirable but complex because of the heterogeneousness and dynamics of the Internet. To cope with these problems an approach based on Active Networks is proposed. This paper presents an approach of using Active Networking technology to combine the two main resource reservation techniques Integrated and Differentiated Services and describes a prototype implementation for evaluation purposes.

1 Introduction

There is an ongoing discussion about realising Quality of Services in the Internet. One approach to achieve this was the development of Integrated Services based on the Resource Reservation Setup Protocol. This protocol is based on the idea of resource reservation for single TCP or UDP flow, causing every RSVP capable router to store information about this flow, allocating resources, initiating traffic control components or queueing systems. Even if this works fine in small and medium sized networks, it cannot scale in Internet backbones. On the other hand, RSVP is really able to guarantee bandwidth and delay on a per flow basis, fitting the needs of modern real time applications.
The alternative concept for a Quality of Service supporting Internet are so called Differentiated Services [Nic98], [BBC$^+$98]. The basic idea is the implementation of different traffic classes in the Internet. The differentiation among these classes is done by the Differentiated Service Code Point (DSCP) in the ToS byte of IP packets. According to the DSCP a packet will be put to queues with different priority or dropping algorithms (e.g. see [HBWW99]) causing different packet forwarding. Every DS capable host or network may apply – according to his Service Level Agreement (SLA)– certain types of service to the packet leaving his domain. It is obvious that the performance of Differentiated Services depends crucially on a good network provisioning, that can be provided in the backbone.
Beneath these two concepts Internet Service Providers (ISPs) may use own methods for traffic engineering. To cope with the dynamics arising of the necessary

reservation mappings the benefits of distributed systems are obvious. In the following section we will present concepts for the mapping and interaction of different reservation techniques, while we focus on a flexible approach for the evaluation of such or similar mobile agent systems in large networks.

2 Active Networking for QoS Support in the Internet

The use of mobile agents seems a promising approach for the management of reservation mapping in large networks. In this section we will present the problems arising by the use of different reservation methods and propose an agent based architecture to cope with these.

2.1 Reservation Domains and ISP Service Mapping

Obviously the points of interest for mapping of resource reservations are the borders of regions supporting or favouring different reservation protocols. This is not necessarily the entire network of an ISP. Even within an ISP it might make sense to provide several areas with different supported reservation methods. Because of this we will not refer to an ISP's network but to a so called Reservation Domain (RD), representing a certain topology regarding reservation protocols or administration. Unfortunately, there are some general problems with the mapping of resource reservations to other types or reservation concepts.

- The borders of the reservation domains may change. Most probably they will be located at the border routers of an ISP, but of course no ISP is forced to provide an homogeneous network working with one type of service all over his topology. So an ISP may include several reservation domains (RD) each providing a different resource reservation method. He may offer his customers RSVP in the access networks, but preferring to use Differentiated Service in the backbone because of scalability or allowing only certain users to use specific reservation methods.
- The path of the packets may depend on the desired Quality of Service. But also this routing may not be stable, but requiring a dynamic allocation of paths at the transition points between two RDs depending on current load and available bandwidth. Also the costs of data transport may play an important role. A provider may also establish such transition points within his network to achieve better load sharing.
- Because of the desired traffic aggregation it is desired to aggregate multiple flow to one large reservation. These aggregate reservations should have a lifetime as long as possible to minimize management overhead.
- Information about the amount and type of requested resources might get lost during mapping, because of incompatibilities in traffic specifications. So as few mappings as possible should be applied to a flow or aggregate during it's transport over several RDs. RSVP uses for example a detailed description of a single reservation, while Differentiated Services only acts on aggregates.

So information about a specific flow may get lost, when mapping RSVP to Differentiated Services.

Figure 1 shows two ISPs with their interior routers (black) and the border routers (white). Typically the used reservation method changes at the border router of an ISP, so a reservation for a flow forwarded from ISP A to ISP B might have to be mapped at the border routers (white). As already mentioned one problem at this point is, that there are no really established standards for the description of a reservation, being compatible between the different concepts of resource reservation. RSVP uses for example a quite detailed flow specification $flow_{spec}$ [Wro97]

$$flow_{spec} = \{r, b, p, m, M\}$$

with the token bucket Rate r, the token bucket size b, the peak data rate p, the minimum policed unit m and the maximum packet size M.

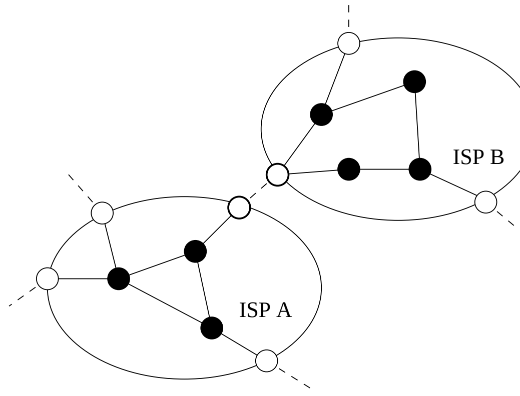

Fig. 1. Two ISPs as RDs with their border routers (white)

Unfortunately the Differentiated Services framework does not describe services as detailed as RSVP. Services are defined by a so called per hop behaviour (PHB). A PHB describes how a packet has to be treated during forwarding, e.g. which kinds of queueing strategies have to be applied. The actual proposed standards include mainly PHBs for a minimum delay guaranteed bandwidth for the realisation of a leased line like services called Expedited Forwarding (EF) ([JNP99] and several classes for flows with multiple drop precedences called Assured Forwarding (AF) [HBWW99]. Furthermore, these proposals do not have to be realised by each ISP, even the DSCP values for the same service may change from ISP to ISP. So at the border of an Reservation Domain (RD) several mappings may be necessary. Obviously the RSVP to DiffServ mapping is one of the most popular cases [BBBG00]. Because of scalability issues an ISP will be interested in mapping RSVP reservation to Differentiated Services decreasing the load of his backbone routers. Because the realisation of Differentiated Services may differ between two ISPs also a mapping of the DSCPs might be necessary.

2.2 Mobile Agents and Service Mapping

The translation of a resource reservation type at the border of a Reservation Domain (RD) faces some problems.

- A RD may depend on the type of an incoming reservation. So a RSVP to DS conversion might be desirable at an ISP's border router to prevent his backbone routers of the RSVP processing load, while a mapping to Differentiated Services is necessary when entering the ISP's backbone using some specific reservation scheme (see also [TH98]).
- Certain reservation mapping methods require a specific setup of a router pair. So, a cooperation of two routers is necessary to setup a tunnel through an ISPs network (e.g. aggregated RSVP [GBH97]). Especially in large networks it might not be reasonable to manage these setups centrally, but leave it to the endpoints to cooperate.
- A new mapping may require a configuration of intermediate routers as a setup of certain queueing disciplines. A central approach would have to determine the path causing a lot of management overhead, while an ingress router may simply launch a capsule to the egress point configuring the devices on it's way through the network, reducing management overhead.
- The mapping components have to access policing information for specific flows. It would be desirable to store the information as close as possible to the agent.

These reasons make the benefits of distributed solutions such as provided by Active Networking obvious. In the following we will describe an architecture for the mapping of resource reservation over several Reservation Domains.
One of the big advantages mobile code can provide is it's self organizing capability. This feature allows to create programs being transmitted through the network, settling at places of interest and performing specific actions there.
Once injected into the network capsules establish agents by searching the network for significant differences in reservation handling between two neighbouring nodes or administrative regions, respectively.
Once an agent is established it listens to traffic being transmitted to his RD, to be able to react on incoming reservation requests. (see figure 2). The network inside such a domain can be assumed to be somewhat homogeneous concerning the available resource reservation methods.

DiffServ to DiffServ Mapping After the injection of capsules, agents occupy the entry points of the homogeneous reservation domain. Each router is configured to announce every DS packet to the agent. The agent will determine an appropriate mapping for the packet, reconfigures the local router to do the proper mapping and forwards a capsule along the packet's path through the

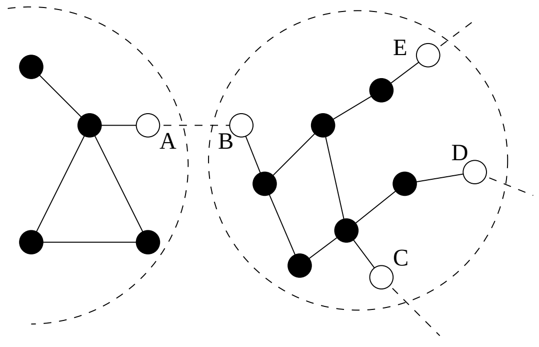

Fig. 2. Reservation Domains with Agents located at the RD borders

reservation domain in order to configure appropriate scheduling mechanisms in each router (see [BB00]). At the egress point of the RD another agent will check, whether another DSCP translation is necessary to meet the needs of the next RD (see section 2.3). There are several methods the agent can decide how to map DSCPs.

encoding: The proper mapping scheme is encoded in the agent itself, requiring an update of agents, when DSCPs are changed.

policy server: The agent might ask a central instance (e.g. a policy server) to determine a proper mapping. The agents then have to cache information about DSCP mappings to minimize the communication overhead.

negotiation: The probably most favourable way would be a negotiation between neighboured agents about the most appropriate mapping scheme. To achieve this the agents need some knowledge about the PHBs behind the DSCPs and methods to determine corresponding PHBs.

Once a mapping and the schedulers are installed, each succeeding packet with the same DSCP can be handled properly without additional overload. The setup of the DSCP mapping and the queueing may have some lifetime, so after a certain idle period the queueing system and translation units may be removed automatically. Whether it makes sense to use mobile code for the realisation of special queuing components has to be evaluated. If only preconfigured service classes will be used the agents only have to reconfigure the DSCP mapping. As it will be described in the next subsection it might also make sense to setup tunnels between the ingress and egress points of an RD. Even when the mapping of certain DSCPs between ingress and egress points does not have advantages regarding scalability, it simplifies the setup of queuing components significantly.

RSVP to DiffServ Mapping A more complex task than the support of different DSCPs is the RSVP-based QoS support. RSVP is based on an end to end scheme, so the RSVP messages used for the setup of an reservation have to

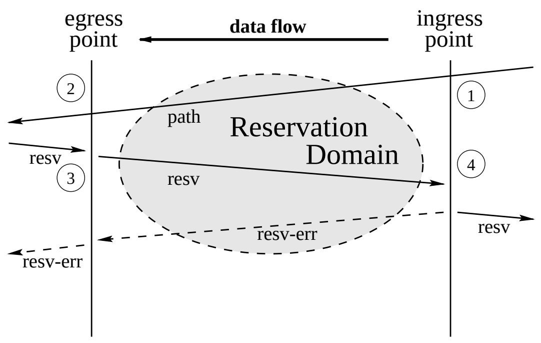

Fig. 3. RSVP Message Flow

be transported through the RD. Figure 3 shows the messages exchanged during a RSVP setup. The two main steps performed by RSVP are:

a) transmission of the *path*-message to notify each router on the path from sender to receiver. This message is sent using the Router Alert Mechanism [Kat97]).
b) transmission of *resv*-message from router to router on the former detected path. At this step the appropriate resources are reserved in every router.

The basic idea is now to setup reservations between ingress and egress points. Several RSVP reservations should be aggregated to a bigger reservation. This traffic aggregate is transported through an IP over IP tunnel to the according egress point (see figure 4) prohibiting interaction of RSVP messages with devices in the RD and simplifying the resource management crucially. Inside the RD, service allocation only has to be done for flows between tunnel endpoints [GBH97]. All packets transported from (B) to (E) are encapsulated, having the same source- and destination address, which simplifies the setup of appropriate resources within the RD.
The setup of the tunnel is proposed to be done by mobile code. When a tunnel has to be setup or resized, the ingress point sends a capsule to the egress point allocating resources for this tunnel within each intermediate router and also initiates the decapsulation at the RD's egress router. A reliable transmission of the capsule to the destination can be achieved by forwarding the capsule with low dropping precedence and the implementation of some kind of simple transport protocol either by the capsule itself or the capsule interpreter.
Acting only on traffic aggregates and using some heuristics, changes to the tunnels will occur quite infrequently, minimizing the load for processing reconfiguration capsules (see [DGB00]). An RSVP reservation has only to be processed at the ingress and the egress point of the RD decreasing the delay to setup the reservation.
As can be seen in figure 3 there are two points at the RSVP message handling process we can use for the RD signalling. We can use the *path*-message (1)

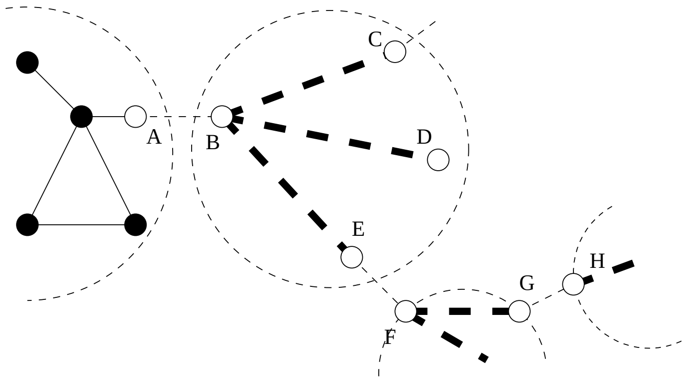

Fig. 4. Tunnels setup by Agents at Ingress points using capsules to configure QoS tunnels through an RD

entering the RD to establish the connection, or use the *resv-message* (3) to setup the resources. The *path*-message is sent earlier but contains only premature information about the requested resources, so a user might alter these data or reserve anything. Only after the *resv*-message has been received (3) we can be sure that the user has accepted the reservation.

2.3 Agent Interaction

As it can be seen on figure 2 ideally on both sides of an RD border agents have settled to perform the reservation translation according to their own and to their neighbours needs. Until now we neglected the use of communication of agents at each side of such a border. Agents can exchange information about the used techniques of resource reservation used in the neighbouring RDs.
Based on this knowledge an agent can decide about the appropriate method of resource reservation or cooperate with other agents to setup for example quality of service tunnels over multiple RDs.
For a central instance it might be hard to supervise all border routers, collecting information about the reservation methods being available in neighbouring domains re-negotiating and configuring border routers and tunnels to an optimal traffic transport.
As it can be seen on figure 4 an agent pair within an RD can use tunnels to set up resources between two border routers (E,F and G,H). Each of these border routers needs resources for encapsulation and decapsulation packets. Of course this does not make sense, when all the traffic leaving the tunnel at E is split up into single flows, encapsulated again by F and decapsulated by G.
An alternative is the extension of the tunnel over multiple domains. When an agent at B starts to set up a tunnel, it forwards the according capsule as described in section 2.2. When the mobile code reaches the egress point it negotiates with an agent at F about an appropriate mapping scheme. If F considers an optimal endpoint for the tunnel to be located at G it might propose this to E. E

transmits the new tunnel endpoint to B, setting up the proper encapsulation methods. Even when the tunnel is now spread over multiple RDs, the underlying resource reservation methods are tasks of the RD's agents. So a capsule sent from B to E sets up resources with their RD, while a capsule from F to G configures their intermediate routers.
The advantages of such a concept is the very local processing of data and the very limited communication over an RD's borders. Agents established by capsules sent to a border router only interact with their direct neighbours, causing only very local traffic. Additionally this simplifies the supervision of the RD since no foreign capsules have to be executed or even transported though an RD. Especially when an RD is equivalent to an ISP this is important.

3 Concept Evaluation by Virtual Routers

3.1 Virtual Router Architecture

A general problem in research on networking is the demand for setting up test beds of sufficient size and complexity to show the desired results or to prove a new concept. Alternatively network simulators like ns [ns],[BEF$^+$00] or OpNet [opn] can be used to prototype a device or a protocol in the special simulation environment and to run the desired tests. So, the simulation normally precedes the setup of the test scenario in a laboratory. Unfortunately, problems occur when combining different technologies. A simulation approach for combining IP forwarding and shaping components with Active Networking is at least not trivial. To encounter these problems we developed an approach to emulate complete (active-)routers including the appropriate IP forwarding. This allows the combination of real hardware and routers with large emulated topologies using several so called Virtual Routers running on the same host.

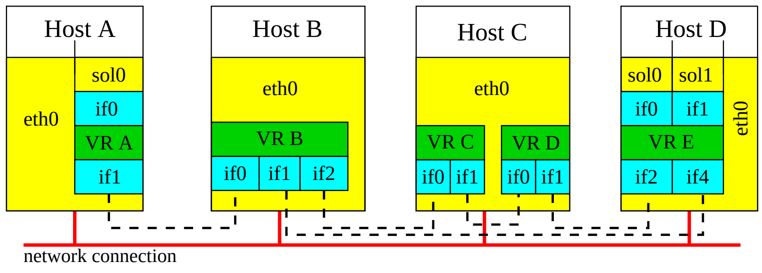

Fig. 5. The components of a VR

This basic idea of combining real hardware with emulated topology is shown on figure 5. The core mechanism, the Virtual Router (VR), is emulating a real IP packet forwarder. Each VR has a couple of interfaces attached. These interfaces can be connected to other interfaces (dashed lines) of other VRs on the same or on a remote host. Alternatively, a VR's interfaces can be connected via Softlink devices (sol$*$) to the network system of the local host.

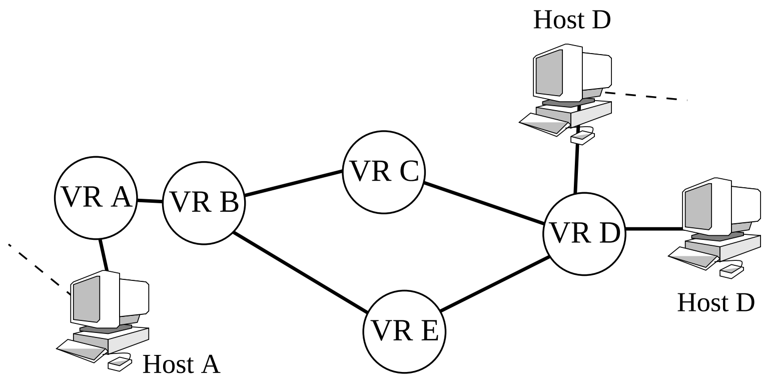

Fig. 6. The components of a VR

Such a Softlink device acts as an interface between the operating systems network layer and a VR. For the OS kernel/user space it looks like a normal Ethernet device, transporting data to user space and vice versa. The network layer of the host system can not detect any differences between the real network and the emulated topology. So, it is possible to define an emulated topology consisting over multiple VRs distributed to several machines. Figure 6 shows the resulting topology. The Softlink device is implemented as a a Linux Kernel module for kernels $\geq$ 2.2.12. The VR itself has been developed for Linux and Solaris.
Virtual Routers are used to realize the network topology to be emulated. Figure 7 shows the principal architecture of the program. Following the primary task the architecture is focusing on IP routing. The VR is completely implemented in plain C++, making the source extensible and easy to port.
The forwarding mechanism acts on standard routing rules, but was extended to allow routing decisions by source addresses, port numbers, protocol fields and ToS values.
As an interface to programs running on this virtual host, the VR has to provide usual IP stacks with TCP, UDP and ICMP. Actually ICMP, UDP and a minimal Capsule Interpreter (CIP) (see section 3.3) were implemented on top of IP.
The main work regarding IP processing is assigned to the interface components underlying the routing mechanism. Figure 7 shows two of them. Each interface can be connected to a Softlink device, acting as a transition point to the real network or to another VR-interface. For the connection to other VR interfaces we use UDP.
Received data is first processed by an IP address translation unit (NAT). This allows to force a source to route also traffic to the source itself over the VR topology, by using dummy destination addresses, which are mapped to the address

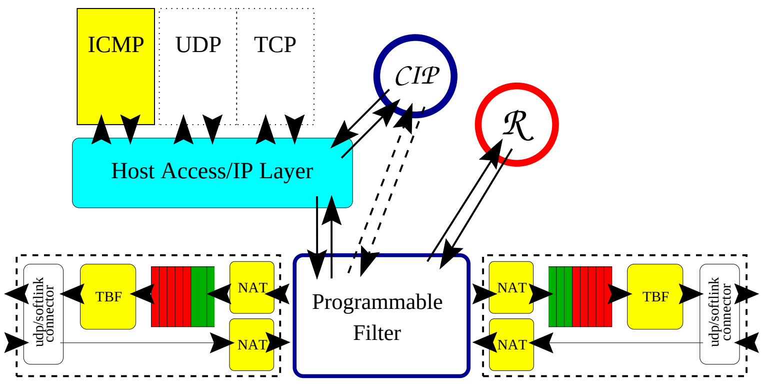

Fig. 7. The components of a VR

of the source in the first VR. This allows a further reduction of necessary computers and simplifies the setup of large topologies. After that step packets are delivered to the filter (PF). This unit is programmable and rules the forwarding of certain packets to higher layers as it will be explained in section 3.2.
Acting as sender, data is also transported through host filter and NAT to be put to the queueing system before transmitted by the Softlink device or sent via UDP. A token bucket filter preceding the connector is used to limit the maximum bandwidth of the interface.
Because of it's flexibility the queueing system is the most complex part of the interface component. It consists of a pool of subcomponents like queues, filters, shapers, schedulers. The current implementation offers the following components: a generic classifier, a Token Bucket Filter, a drop tail queue, a Random Early Detection queue (RED) [FJ93], a Weighted Fair Queueing (WFQ) scheduler, a simple Round Robin (RR) scheduler and a Priority Round Robin (PRR) scheduler. Additional components are a RED queue with three drop precedences (TRIO), a special marker for differentiated services and a Priority Weighted Fair Queueing (PWFQ) scheduler for the implementation of Expedited [JNP99] and Assured Forwarding [HBWW99].
The configuration of the queuing system can be completely done at runtime via API or command line interface (CLI). The object oriented implementation of the queueing system and it's components makes it easy to add or modify single functionalities.
For the configuration of the VR a command line interface and an API have been implemented. The API allows programs (e.g. a capsule interpreter) running on the virtual router to alter interface setting, routing rules and so on. The command line interface is accessible via console or external[1] telnet.

[1] This telnet connection is not provided by the VR, but by the host running the VR making it independent from any changes made to the VR. This simplifies the setup of multiple machines crucially, because no changes to interfaces setup or queueing mechanisms can harm the TCP connection used for the configuration.

3.2 Programmable Filter (PF)

To allow the seamless processing of capsules and to control the access of running agents on specific flows all data transported by an interface is processed by the programmable filter (PF). This filter is programmable via an API and allows daemons running on the virtual router to gain access to flows. An application can determine, which flows it gets by setting up a filter. This can be done by submitting a set of parameters to the PF. The set of supported parameters contains most of the IP headers and options. Of course more than one filter might be setup.

$$PF_{spec} = \{(Des, Des_{Mask}), (Src, Src_{Mask}), (Opt, Opt_{val}, Protocol, ToS)\}$$

This concept has the advantage, that an application only has to provide the filter pattern once rather than processing all incoming data. It simplifies the setup of applications as well as speeds up processing of normal traffic, being forwarded by the VR.
There are two main filter modes: copy and move. During copy mode a packet is forwarded normally and a copy of it is passed to the upper layer. In the move mode the packet is not duplicated, but only passed to the HAL/IP layer (see figure 7). So, an application can process a packet and re-inject it afterwards or completely discard it. An application can override queueing systems and forwarding mechanisms of the VR. For example, a whole queuing system may be applied for a certain type of packets, without affecting the normal processing of standard IP packets.
In addition to forward packets to applications according to the set up of filters, the PF can be configured to preprocess the packets. It is obvious, that an application may not be interested in the entire packet, but only in certain information. The PF provides the functionality to truncate the packet body and provide only the header to the application. It may collect statistical data about a certain packet type like the average, the minimum and maximum bandwidths, the number of matching packets and so on.
It has to be mentioned, that the PF does not provide any security. The PF acts on the filter set up and does not control whether the application is authorized to access these data. If the PF is used to support a Capsule Interpreter (CIP), the decision which capsule is allowed to setup which filter is left to the CIP.

3.3 Capsule Interpreter

To support the exchange and execution of mobile code in a network of VRs, i.e to get a Virtual Active Router (VAR), a Capsule Interpreter (CIP) was implemented. These CIP uses the PF to receive IP packets with the Router Alert Option [Kat97] set. A packet that is forwarded through the VAR with these options is also copied to the CIP, executing the capsule's code.
For some first testing of the capsule forwarding mechanisms we implemented a very simple TCL based CIP. The approach behind was not so much influenced by

architectural or security issues, but by the quick implementation of a mechanism to distribute scripts over a network of Virtual Active Routers. In general it was mainly used to evaluate the concepts of programmable filters and VRs. The interfaces to the Host Access/IP layer are simple and it should be easy to port a more advanced CIP to the VR in future. Especially we evaluated methods for the setup and the configuration of an VR's queueing system to support specific flows.
The capsule format is comparable to the Active Network Encapsulation Protocol (ANEP) [ABG+97]. TCL scripts including a ANEP like header are included in IP packets using the Router Alert Option [Kat97] to be executed by each router they pass. The CIP itself requests these packets by an appropriate configuration of the PF. A capsule may set up additional filters at the PF to process specific flows or to act on the arrival of certain packets. The CIP has to control which data may be accessed by the capsule.

4 Summary and Outlook

The use of Active Networking technology offers great possibilities for the management of large networks. Of course there is a lot of research to do especially in the areas of security and performance. The combined approach of using IP and AN as assumed in this paper could offer the desired performance as well as the enormous flexibility that only mobile code can provide.
The other important aspect is security. In our scenario capsules are only launched by an ISP's own hosts and their activities are limited to the border regions of their RD. Any interaction between foreign agents only takes place either within an ISP's RDs or between neighbouring ISPs, so security solutions should be feasible by applying common security mechanisms like encryption and authentification via asymmetric algorithms. The VAR platform will be optimal for large scale performance evaluation of the proposed service mapping concepts.
A future issue will be the information exchange between single agents to implement a self organizing distributed system optimizing routing and tunnel setup regarding QoS and costs.

5 Acknowledgements

The work described in this paper is part of the work done in the project 'Quality of Service support for the Internet Based on Intelligent Network Elements' funded by the Swiss National Science Foundation (project no 2100-055789.98/1). Parts of the implementation platform were funded by the SNF R' Equip project no. 2160-53299.98/1.

References

[ABG+97] D. Scott Alexander, Bob Braden, Carl A. Gunter, Alden W. Jackson, Angelos D. Keromytis, Gary J. Minden, and David Wetherall. Active network encapsulation protocol. Internet draft, July 1997.

[BB00] Florian Baumgartner and Torsten Braun. Virtual routers: A novel approach for qos performance evaluation. *Quality of future Internet services, QofIS'2000*, September 2000.

[BBBG00] R. Balmer, F. Baumgartner, T. Braun, and M. Günter. A concept for rsvp over diffserv. *IEEE, ICCCN'2000*, October 2000.

[BBC$^+$98] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weis. An architecture for differentiated services. Request for Comments 2475, December 1998.

[BEF$^+$00] Les Breslau, Deborah Estrin, Kevin Fall, Sally Floyd, John Heidemann, Helmy Ahmed, Polly Huang, McCanne Steven, Ya Xu, and Haobo Yu. Advances in network simulation. *IEEE Computer*, pages 59–67, May 2000.

[CDMK$^+$99] Andrew T. Campbell, Herman G. De Meer, Michael E. Kounavis, Kazuho Miki, John B. Vicente, and Daniel Villela. A survey of programmable networks. In *Computer Communications Review*, volume 29/2. ACM SIGCOMM, April 1999.

[DGB00] Gabriel Dermler, Manuel Günter, and T. Braun. Towards a scalable system for per-flow charging in the internet. ATS 2000, 2000.

[FJ93] Sally Floyd and Van Jacobson. Random early detection gateways for congestion avoidance. *IEEE/ACM Transactions on Networking*, August 1993.

[FLK$^+$99] Ted Faber, Bob Lindell, Jeff Kann, Graham Phillips, and Alberto Cerpa. Arp: Active reservation protocol, April 1999. Presentation.

[GBH97] R. Guerin, S. Blake, and S. Herzog. Aggregating rsvp-based qos requests. Internet Draft `draft-guerin-aggreg-rsvp-00.txt`, November 1997.

[HBWW99] Juha Heinanen, F. Baker, W. Weiss, and J. Wroclawski. Assured forwarding phb group. Request for Comments 2597, June 1999.

[ISI81] University of Southern California Information Sciences Institute. Internet protocol. Request for Comments 791, September 1981.

[JNP99] Van Jacobson, K. Nichols, and K. Poduri. An expedited forwarding phb. Request for Comments 2598, June 1999.

[Kat97] D. Katz. Ip router alert option. Request for Comments 2113, February 1997.

[Nic98] S. Nichols, K. Blake. Differentiated services operational model and definitions. Internet Draft `draft-nichols-dsopdef-00.txt`, February 1998. work in progress.

[ns] Ucb/lbnl/vint network simulator - ns (version 2). URL: http://www-mash.CS.Berkeley.EDU/ns/.

[opn] Opnet modeler. URL: http://www.mil3.com.

[TH98] J. Touch and S. Hotz. The x-bone. Third Global Internet Mini-Conference in conjunction with Globecom'98, Sydney, Australia, November 1998.

[Wro97] J. Wroclawski. The use of rsvp with ietf integrated services. Request for Comments 2210, September 1997.