Adaptive End-to-End Quality-of-Service Guarantees in IP Networks using an Active Networking Approach

Roman Pletka
IBM Research, Zurich Research Laboratory, 8803 Rüschlikon, Switzerland
rap@zurich.ibm.com
Burkhard Stiller
University of Federal Armed Forces Munich, 85577 Neubiberg, Germany and
Computer Engineering and Networks Laboratory (TIK), ETH Zürich, 8092 Zürich, Switzerland
stiller@tik.ee.ethz.ch

Abstract

This paper proposes a framework based on an active networking approach to efficiently link Quality-of-Service (QoS) descriptions from an application point of view with an underlying heterogeneous IP networking infrastructure. The main goal is to provide building blocks that cooperate to sense the availability of and deploy distinct QoS capabilities in order to accomplish adaptive end-to-end service guarantees. The building blocks needed in a heterogeneous IP network will be introduced and discussed with respect to safety from abuse of total networking bandwidth, CPU, and memory usage. In conjunction with a new safety hierarchy and a sandbox environment for active-code execution, security risks can be bounded to the level of traditional IP forwarding, control, and management. In particular, the problem of QoS-parameter translation to provide end-to-end service guarantees is addressed, and an example using Diffserv, RSVP, and GPRS in a heterogeneous network is given.

1 Introduction

Limited Quality-of-Service (QoS) support in IP networks is usually achieved by massive over-provisioning nowadays, because over-provisioning is still cheaper than operating a QoS-enabled network. However, this may change in future. Yet the introduction of simple service differentiation such as Diffserv [1] can significantly increase the networking utility without installing additional networking bandwidth. It can be expected that QoS capabilites will be gradually introduced in the Internet. Regardless of whether end users are willing to pay for QoS the number of real-time and multimedia applications, and therewith the amount of networking traffic that would benefit from QoS, is still increasing. It has been shown that the needs of these applications cannot be satisfied in a high-loaded best-effort-based network [2]. At the same time, there is a huge variety in existing QoS architectures, most of them standardized by the IETF (Intserv [3], Diffserv [1], and ST2+ [4]), or proposed by the research community such as SRP [5].

Furthermore, QoS support is rarely used in heterogenous IP networks because there is no end-to-end support of service guarantees and because the increasing variety in QoS-provisioning mechanisms (e.g., policers, schedulers and active queue management) in network nodes complicates their integration into QoS frameworks.

This paper introduces a framework for adaptive end-to-end QoS guarantees using active networks and focusses on its safety requirements to reduce security risks to the level of traditional IP forwarding. Active packets are used for QoS provisioning in conjunction with and as a complement to existing frameworks in order to optimize the usage of existing QoS capabilities. Our framework allows the efficient and dynamic translation of QoS parameters from an unsupported scheme into one that is supported by the networking infrastructure considered. This can be done for a particular router or a heterogeneous domain, and in horizontal (e.g., between networking nodes) as well as vertical (e.g., inside a given networking node) direction. Safety is guaranteed by two different means: First, by restrictions in the active byte-code itself, and second, through the definition of a safety hierarchy. The safety hierarchy allows code to be placed in active routers only under certain restricted conditions.

The framework allows service providers to dynamically define and install QoS translation services, which are necessary owing to the heterogeneity of the Internet. Furthermore, it allows applications and networking nodes to describe their service requirements by placing active code into packets. This code is then used within the network to facilitate the translation and adaptation process in order to find an appropriate service behavior.

The paper is organized as follows. Based on a brief survey of major related work on end-to-end QoS support and active network frameworks in Section 2, we first point out where the discrepancy in QoS provisioning seen from the application's view point and the underlying heterogeneous network, originates (Section 3). Then we propose an abstract node model that integrates these deficiencies, and discuss the minimum set of functionalities such a node model has to fulfill. The requirements and implementation

guidelines for the active networking approach are given in Section 4. In an example (Section 5), active packets are used for QoS provisioning in conjunction with and as a complement to existing frameworks as Intserv/RSVP [3] and Diffserv [1] to illustrate the deployment of QoS guarantees in heterogeneous IP networks. Finally, Section 6 presents some concluding remarks.

2 Related Work

Several capsule-based active networking approaches have been introduced. The ANTS framework [6] based on mobile code and caching techniques has been introduced to enable dynamic and automatic deployment of new protocols in routers and end systems but lacks in providing essential security and safety properties. PLAN [7] and Sprocket [8] from the Smart packet approach provide stronger security guarantees and resource management. However, there exists programs in PLAN that execute in time exponential to the packet size, and Sprocket does not allow the dynamic extension of services as this would require modifying the virtual machine itself. SNAP [9] balances the tradeoffs between flexibility, efficiency, and safety by introducing limitations in the byte-code language. For instance, program loops not allowed in the language, can only be achieved by sending the active packet back to the last active node, resulting in additional computational overhead for packet processing and unnecessarily increasing networking bandwidth usage.

Other frameworks introduce extensible router architectures that allow customization of router functionalities at run-time. Scout [10], a communication-oriented operating system, can be used to build customized forwarding paths in extensible routers. Initially, packets are classified and then processed by a given path from the source to the destination device. Forwarding paths can be customized for individual packet flows. In Click [11], packet-processing paths are build using simple elements with input and output ports that are linked together in a modular way. Router plugins [12] introduces extensibility that can be placed at predefined points called gates in the IP forwarding code. Router plugins are dynamically retrieved from a code server and installed in the router's kernel. Their functionalities range from routing, packet scheduling, and security processing to scaling of video streams [13].

3 QoS Provisioning in a Heterogeneous Environment

3.1 The Discrepancy in QoS Provisioning of Heterogeneous Networks

A growing set of applications such as Voice-over-IP (VoIP) or real-time audio and video streaming combined with application sharing require QoS guarantees. These guarantees must be provided across sub-networks and multiple domains with heterogeneous networking infrastructures,

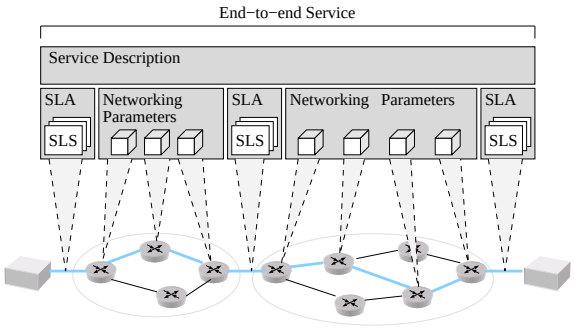


Figure 1: *End-to-end services in a heterogeneous environment. Each cube is an instance of the node model presented in Section 3.2.*

hence the need for functionalities that enable a repeated mapping from a QoS request to the networking parameters of each successive underlying QoS scheme in an end-to-end fashion.

The discrepancy in QoS provisioning seen from the application's view point and the underlying heterogeneous network is illustrated in Figure 1. The service description, known in general by the application but not directly by end users, contains flow-specific traffic description for rate (e.g., CBR, sustainable and peak rate), delay, and maximum loss tolerance. Service Level Agreements (SLA) between a customer and a service provider include a Service Level Specification (SLS) for each service specified in the SLA. Generally, this information is not on a per-connection or application basis and does not include detailed information on the source-destination relationship. Moreover, no implementation details or indications on the networking resources available beyond the boundary concerned are given. In a heterogeneous Internetwork, hardware support can differ for each router. Networking parameters for buffer management, scheduling, and active queue management are neither reflected in SLSs and SLAs nor in the description of the service, as SLAs and SLSs describe the boundary at the edge of the network rather than the behavior of nodes in the network.

The difficulty of maintaining the QoS parameter sequence for end-to-end services stems from the uncertainty how traffic is treated beyond the peering networking domain for which the SLA is valid. This uncertainty is aggravated by the heterogeneity of QoS provisioning capabilities within domains. End-to-end per-connection SLA-negotiation using inter-domain bandwidth brokers would solve the problem, but in turn provokes scalability issues.

3.2 An Abstract Node Model for QoS Provisioning

QoS provisioning in networking nodes can be modeled as shown on Figure 2. Three different operating planes are distinguished. The application plane describes and requests specific QoS guarantees from the network, which will be provided to an application. QoS signaling can be implicit or explicit, and the QoS description absolute, rel-

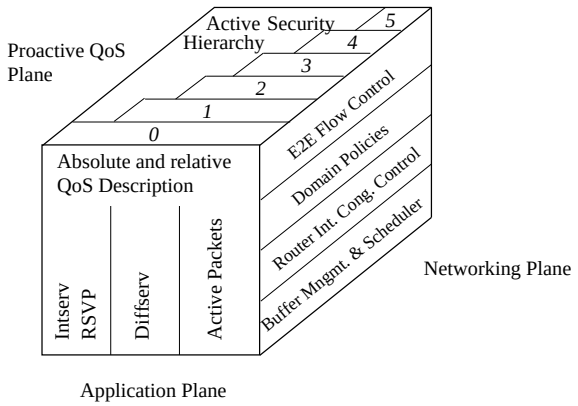


Figure 2: *Node model for QoS provisioning in a proactive environment composed of the application, networking, and proactive plane.*

ative, or within statistical bounds. The networking plane consists of per-hop-based networking parameters, domain policies, and end-to-end flow-control mechanisms. This plane reflects the mapping of QoS descriptions into hardware functions placed at the disposal of the corresponding node. Finally the proactive QoS plane performs the translation of QoS parameters from the application plane to the networking plane, and acts according to these parameters on a per-hop basis. The functionality of the translation process has to be limited by a security hierarchy: Functionalities provided by low-numbered levels can be used in the data-plane packet-forwarding process, whereas higher levels include control-plane functionalities such as adding new policies or router services for QoS translation.

Given the heterogeneity of networks in terms of underlying hardware as well as domain-specific behaviors, the proactive QoS plane cannot be described by simple static means; rather it is appropriate to consider each individual node on the path of a connection as an independent instance of our node model. Eventually, an end-to-end service requiring a certain QoS guarantee can be achieved by chaining these node models as shown in Figure 1.

From this approach the following implicit functionalities of a packet can be derived: Packets are no longer static data containers but can also carry active program code. Data and program code can reside in packets at the same time. These functionalities enable dynamic QoS parameter translation between different QoS frameworks (e.g., Diffserv and Intserv) that are not translatable by simple one-to-one mapping functions.

3.3 Functional Description

Before we show how active packets are used to program the proactive environment, we introduce the functional modules necessary in an active router. They consist of four different building blocks: the discovery process, the QoS translation phase, resource management, and feedback mechanisms.

Discovery process: The discovery process leads to initial behavior bounds [1] that specify upper bounds for available resources. This is done to obtain some a-priori knowledge of QoS availability prior to connection setup. The information gathered on instantaneous resources is then updated periodically. A possible source of information is the traffic-engineering opaque LSA messages in OSPF [14]. Unlike the QoS Broker [15], the discovery process is used within the network and does not deal with QoS discovery from the application or end user's view point. The end-to-end QoS behavior is therefore subject to an ongoing and adaptive process throughout the lifetime of a connection, and is a result of the network discovery process present in the background.

Translation phase: The necessity of active code stems from the fact that QoS translation in general is not a bijective operation, and therefore increases complexity. The following guidelines can be used: Surjective code translation is obtained by projection onto the new QoS space, whereas injective code translation needs additional information based on default mappings and/or educated-guess methods. Bijective translation is primarily achieved with one-to-one table-based mapping. The translation process is done using active code provided by either the network administrator or in certain cases by the application itself as long as safety is not compromised. Classification of the packet allows the appropriate safety level and, if needed, an adequate translation code to be chosen.

Resource management: Resource management comprises the task of maintaining information on the actual status of resource availability. A certain share of the resources can be initially assigned to the active networking element. The resources that are administered consist of QoS-related resources (e.g., maximum bandwidth per traffic class), policies, resources related to the neighborhood, and router services. From these resources the behavior bounds are derived.

Feedback mechanisms: Instantaneous traffic characteristics can deviate from the corresponding QoS reservation and are influenced by numerous factors in the network (e.g., traffic shapers, actions of active queue management (AQM) schemes, the granularity of schedulers, amount of cross-traffic), and in end systems (e.g., round-trip time in TCP). All these factors are variable in time, and affect the end-to-end service. Some of them are controlled by specific feedback mechanisms. Adaptive end-to-end service guarantees are feasible when the interaction between feedback mechanisms is taken into account i.e., their interactive behavior is predictable. The proactive QoS environment uses feedback mechanisms that act in an active node as well as between neighbors, and uses different time scales to update behavior bounds.

[1] The behavior bound consists of a classifier describing to whom the service will be offered, a traffic specification (e.g., sender Tspec), and a resource bound vector that characterizes the maximum resource usage of the router service.

Table 1: Safety hierarchy in active networks.

Safety level	Allowed network functionalities	Packet and router requirements
5	Dynamic router services (active code): registering new router services	Authentication of active packets needed using a public key infrastructure.
4	Complex policy insertion and manipulation	Admission control at the edge of the network, trusted within a domain.
3	Simple policy modifi cation and manipulation	Running in a sandbox environment, limited by predefi ned rules and installed router services.
2	Creation of new packets and resource-intensive router services (lookups etc.)	Sandbox environment based on the knowledge of the instruction performance.
1	Simple packet byte-code	Safety issues solved by restrictions in the language defi nition and the use of a sandbox.
0	No active code present in packets	Corresponds to traditional packet forwarding process.

4 Active Networking Framework

4.1 Security Risks in Active Networks

The use of an active networking approach is justified by the following reasons: First, the deployment of protocols is always limited by a well-defined function space that does not allow interoperability between diverse protocols in heterogeneous Internetworks. Programmability provides the necessary flexibility and acts as glue between protocols that finally enables the use of translation mechanisms between existing QoS frameworks. Second, active networks have the advantage that new functionalities can be deployed dynamically. Unfortunately, active networks entail a considerable number of security issues that have to be solved. Any potential approach has to comply with these technical demands:

- Use of a byte-code language that not only achieves architectural neutrality but also possesses intrinsic safety properties given by the definition of the language itself.
- The definition of a resource bound divides networking resources into a two-dimensional vector consisting of a local part, which is consumed on a router while executing byte-code instructions, and a network part which limits the spread of a packet in the network.
- An appropriate safety hierarchy that monitors control-plane activities.
- Execution of any active byte-code in a secure environment called a Active Networking Sandbox (ANSB).
- Router services dynamically enhance router functionalities to overcome limitations of the byte-code instructions.

In the following section we will discuss the safety hierarchy of the above-mentioned requirements. A more detailed description of the byte-code language and the resource bound can be found in [16, 17].

4.2 The Hierarchical Safety Levels

In this section an adequate solution for scalable and safe active networking, the safety levels, are introduced and discussed. The goal is to protect networking resources from malicious users and to distribute excess resources fairly at the same time. It is not wise to overwhelm the network with strong cryptographic measures with regard to the data plane in which each packet has to be authenticated and encrypted. The network simply has not enough resources to handle these packets in the data path. Therefore, an adequate and sufficient safety hierarchy is a mandatory building block.

Packet classification (a policy-based procedure) determines the safety level of packets. Higher safety levels are likely to be coupled with more restrictive policies. Thus, the safety levels of the proactive QoS plane have a pyramidal shape, which restricts the execution of powerful commands according to policies, and therefore achieves the required safety.

Table 1 shows the proposed safety hierarchy that addresses the problem. Levels 0 and 1 of the hierarchy address the data path. Level 0 corresponds to traditional packet-forwarding process without execution of active code. Packets containing simple active byte-code are allowed on level 1. Safety is solved by restrictions in the language definition and the use of a sandbox environment. SNAP [9] is an example of such a byte-code language. A simple packet byte-code enables immediate QoS provisioning and minimizes security efforts because no verification of plugins has to be done. In traditional routers, QoS decisions are taken using local information and information from packet headers. The main advantage of the new approach is that packets are able to take additional information from preceding hops into account, hence acting in a flexible and distributed manner. A router will execute the active byte-code in a special sandbox environment that ensures safety.

Level 2 provides certain router-service primitives in addition to the byte-code language. As router services in general are more costly in terms of CPU cycles than simple

byte-code instructions, and can differ significantly in the cost as well, the definition of a resource bound as given above is mandatory. [16] identifies and quantifies expensive router services (e.g., router services that allow packets to obtain information on congestion status and policies installed at the given hop) that belong to this safety level.

The next higher level, i.e., level 3 allows modification and manipulation of policies that are installed using router services. As an example the RSVP soft-state mechanism is mentioned here in which state information is updated by RSVP reservation request (Resv) messages to maintain reservation state and RSVP Path messages that store path state in each node along the way. Unlike RSVP this level does not allow the installation of new states in routers.

The insertion of policies and complex rule manipulation is handled by level 4. In the RSVP example (cf. Section 5), this reflects in the creation of new state information in a router.

Level 5 finally allows the installation of dynamic router services that can provide and maintain information for active packets in lower levels. For security reasons, no level 4 packets from outside a service provider's network are allowed to enter the domain. Thus only certain management hosts within the ISP domain are allowed to inject such packets. Nevertheless when a certain well-known service should be installed, agents that reside at the network edge can, for example, translate the requested service into a predefined level 4 active code that will then install the service in the ISP's network.

5 Example Applications

Figure 3 sketches a heterogeneous Internetwork environment to illustrate an example of the deployment of end-to-end QoS guarantees in heterogeneous IP networks. The sender is directly attached to an Intserv domain, and therefore utilizes RSVP Tspec messages to describe the desired QoS, while the receiver is a mobile client attached to a General Packet Radio Service (GPRS) [18] network.

The Intserv/RSVP Domain

The first domain supports full RSVP on all routers in the domain and might correspond to a small ISP placed at the edge of the network (e.g., in the metropolitan area) and therefore is capable of handling per-flow RSVP signaling. This domain does not provide any active routers at all and active packets are forwarded as regular IP packets.

The Diffserv Network with Active Nodes

The second domain, a core ISP, full RSVP support cannot be provided for scalability reasons. Therefore it makes sense to install dynamic router services in which the behavior bound directly derives from the SLA established with the Intserv domain. Simplified RSVP support is then provided using dynamic router services. Active packets that enter this domain and have a safety level higher than 1 are simply preempted.[2] Only authenticated and authorized packets are allowed to install dynamic router services on the active routers in the Diffserve domain. These dynamic router services then allow RSVP messages to be handled in order to dynamically configure hardware classifiers and policers for data packet processing.

Figure 4.a) shows the sequence executed when adding a new router service. The active packet arriving at the node asks to register a new router service given as payload. As the active packet is not preempted, the byte-code interpreter executes the register request by passing the information to the active-code translator. The active-code translator installs new policies, reserves resources, and, if successful, registers the new service in the service table. At the same time classifiers for the RSVP messages are set up.

In active routers at the edge, RSVP Resv messages are accepted using policies from the policy database. In the core, no further verification of the RSVP Resv messages has to be done, and the content of the message is used to install appropriate filters and flow parameters according to the behavior bounds given in the policy and resource databases (Figure 4.b).

A pure Active Networking Domain

In a pure active networking domain we can imagine the following scenario. Within the domain, no native RSVP support is given, but active packets with security level up to 1 are executed (Figure 4.c). Here, QoS adaptation takes place directly in the data path. Small active-code sections can use the DSCP and local congestion status information to influence the forwarding behavior of a given packet within the limits provided by the ANSB. This ISP provides a slightly larger degree of freedom to applications, but in principle is not willing to guarantee anything more than that. Because of charging of active packets at the edge and the safety properties of the byte-code, this network can be exploited with a minium of administration effort. The service offered is comparable to the well-known postal priority service.

Although this domain does not support strict QoS guarantees, the end-to-end service is improved, and can in the best case even hide the lack of full RSVP support in all routers on the data path.

Mobile Network using a GPRS Backbone

The GPRS architecture [18] supports interworking of GPRS and IP networks. A Gateway GPRS Support Node (GGSN) is acting as the interface between the GPRS backbone and the IP network. Packets are then forwarded to the corresponding Serving GPRS Support Node (SGSN) which is responsible for the delivery of data packets from

[2] Preemption can be done by using a preemption flag in the active header, stackable active headers, classification based on the active code, or domain tags derived from an Autonomous System (AS). In terms of operating expenses and complexity, only the use of a preemption flag is reasonable, although a possible asymmetric behavior has to be accepted. Thus active routers at the ingress of a domain are responsible for setting this flag correctly.

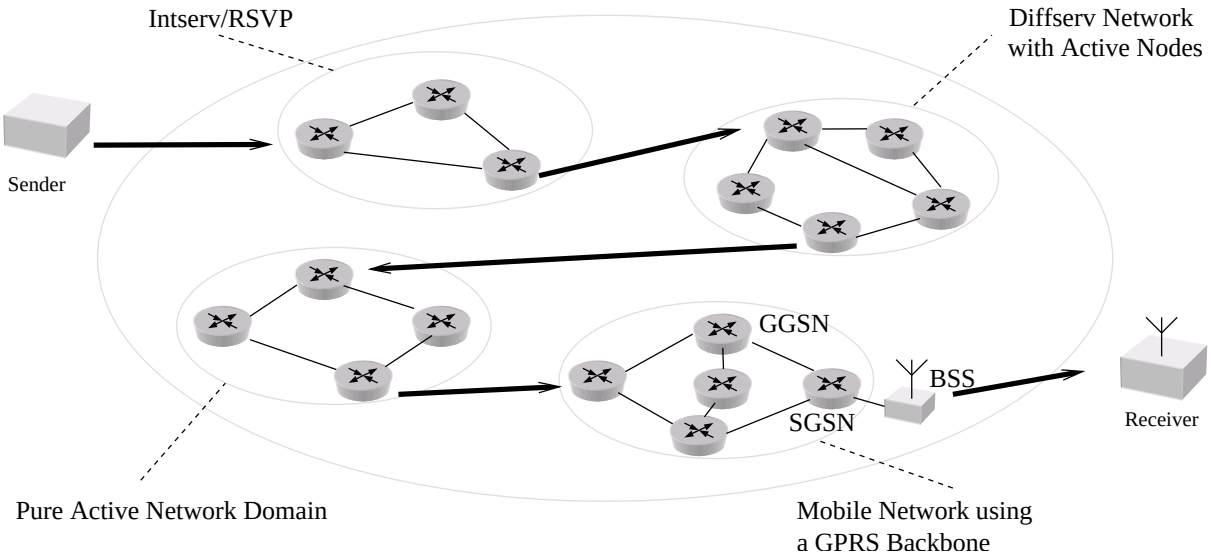
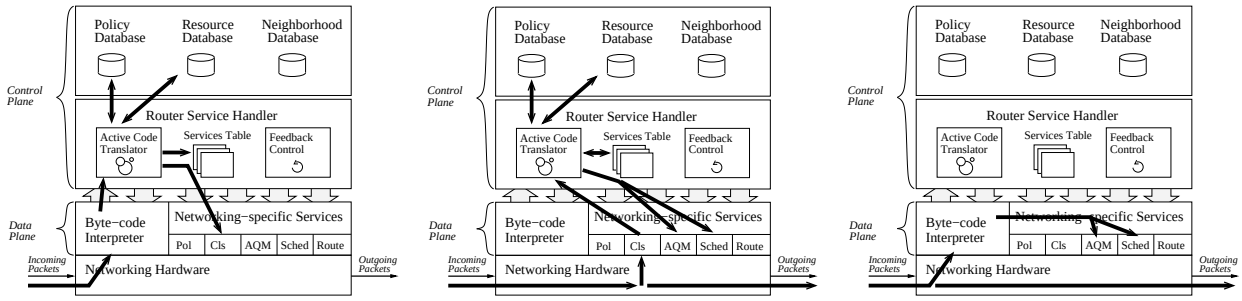


Figure 3: *End-to-end service in a heterogeneous environment crossing three different domains.*



(a) Service registration for simplified RSVP functionalities in the Diffserv Network using active packets with safety level 5.

(b) RSVP Resv message handled by the ANSB in the core of the Diffserv Network.

(c) Active packets with safety level 1 can profit from limited QoS functionalities in the pure active networking domain.

Figure 4: *Sequence of procedures in the ANSB for different packets in different domains.*

and towards the mobile stations through the Base Station Subsystems (BSS) in its area. GPRS allows the definition of QoS profiles in terms of service priority, reliability, delay and throughput. The appropriate translation process between the Diffserv network and the GPRS backbone can be achieved by router services installed on the GGSN similar to the Diffserv network with active nodes presented above.

6 Summary and Conclusion

This paper shows how the end-to-end quality of service can be improved with active networks. Based on existing QoS capabilities, the active networking approach proposed here provides the necessary tools for dynamic translation between different QoS schemes and, therewith, enables efficient linkage of QoS parameters to build end-to-end services.

The safety hierarchy introduced consists of six safety levels, and provides the necessary safety guarantees and enables dynamic router services for QoS translation in the control plane on the one hand. On the other hand, it also allows the execution of simple active code, even in the data path.

In general, information on the network infrastructure and network topologies is not publicly available because ISPs consider this information as sensitive. The use of active networks as proposed in this paper does not expose this information. First, only a part of the available resources can be placed at the disposal of the ANSBs, and second, the safety hierarchy ensures that only information required for a given task will be made available, e.g., active code in the data path restricted to safety level 1 cannot use dynamic router services.

It is clear that this paper only showed a small portion of possible solutions. Nevertheless, given the flexible approach using Active Networks, the applicability to other scenarios is ensured.

We successfully implemented the lower safety levels on a IBM PowerNP 4GS3 [19] network processor. In the fu-

ture, we intend to focus on higher safety levels that would allow certain forwarding and control functionalities to be off-loaded directly onto a network processor in a highly dynamic way. We believe that QoS provisioning can significantly benefit from network processors using Active Networks for the dynamic off-loading of functions from the control point.

Acknowledgments

The authors would like to thank Robert Haas and Marcel Waldvogel for the valuable comments and and helpful discussions.

References

[1] S. Blake, D. Blake, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for Differentiated Services. RFC 2475, Internet Engineering Task Force, December 1998.

[2] Ulrich Fiedler, Polly Huang, and Bernhard Plattner. Towards provisioning Diffserv intra-nets. In *Proceedings of the 9th International Workshop on Quality of Service (IWQoS 01)*, June 2001.

[3] R. Braden, D. Clark, and S. Shenker. Integrated Services in the Internet architecture: an overview. RFC 1633, Internet Engineering Task Force, June 1994.

[4] K. McCloghrie and M. Rose. Internet Stream Protocol version 2 (ST2) protocol specification - version ST2+. RFC 1819, Internet Engineering Task Force, August 1995.

[5] W.Almesberger, T.Ferrari, and J.-Y. Le Boudec. SRP: a scalable resource reservation protocol for the Internet. In *IEEE Workshop on Quality of Service IWQoS*, May 1998.

[6] David J. Wetherall, John Guttag, and David L. Tennenhouse. ANTS: A toolkit for building and dynamically deploying network protocols. In *Proceedings of IEEE OPENARCH '98*, April 1998.

[7] Michael W. Hicks, Pankaj Kakkar, Jonathan T. Moore, Carl A. Gunter, and Scott Nettles. PLAN: A packet language for active networks. In *International Conference on Functional Programming*, pages 86–93, 1998.

[8] B. Schwartz, W. Zhou, A. Jackson, W. Strayer, D. Rockwell, and C. Partridge. Smart packets for active networks. *ACM Computer Commun. Rev.*, January 1998.

[9] J. T. Moore, M. Hicks, and S. Nettles. Practical programmable packets. In *Proceedings of the 20th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM '01)*, April 2001.

[10] D. Mosberger and L. L. Peterson. Making paths exlicit in the scout operating system. In *Proceedings of the Second USENIX Symposium on Operating System Design and Implementation (OSDI)*, pages 153–167, October 1996.

[11] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The Click modular router. *ACM Trans. on Computer Systems*, 18(3):263–297, August 2000.

[12] Dan Decasper, Zubin Dittia, Guru Parulkar, and Bernhard Plattner. Router plugins: a software architecture for next-generation routers. *IEEE Trans. on Networking*, 8(1):2–15, February 2000.

[13] R. Keller, S. Choi, M. Dasen, D. Decasper, G. Fankhauser, and B. Plattner. An active router architcture for multicast video distribution. In *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2000)*, March 2000.

[14] Dave Katz, Derek Yeung, and Kireeti Kompella. Traffic engineering extensions to OSPF. draft-katz-yeung-ospf-traffic-07.txt, August 2002. Work in progress.

[15] Klara Nahrstedt and Jonathan M. Smith. The QoS broker. *IEEE Multimedia*, 2(1):53–67, 1995.

[16] Andreas Kind, Roman Pletka, and Burkhard Stiller. The potential of just-in-time compilation in active networks based on network processors. In *Proceedings of IEEE OPENARCH '02*, June 2002.

[17] Roman Pletka and Burkhard Stiller. Adaptive end-to-end Quality-of-Service guarantees in IP networks using an active networking approach. TIK-Report Nr. 138, Computer Engineering and Networks Laboratory (TIK), ETH Zürich, June 2002.

[18] Christian Bettstetter, Hans-Jörg Vögel, and Jörg Eberspächer. GSM phase 2+ general packet radio service GPRS: Architecture, protocols, and air interface. *IEEE Communication Surveys*, 2(3), 1999.

[19] IBM PowerNP NP4GS3 network processor datasheet. http://www.ibm.com/chips/techlib/techlib.nsf/products/IBM_PowerNP_NP4GS3, February 2002.