https://doi.org/10.26552/com.C.2001.4.31-35

Octavian Pop – Tamás Máhr – Tímea Dreilinger – Róbert Szabó *

ARCHITEKTÚRA DOJEDNÁVANIA ŠÍRKY PÁSMA V SIEŤACH DIFFSERV

BANDWIDTH BROKER ARCHITECTURE FOR DIFFSERV NETWORKS

Bandwith Broker (BB) je automatizovaný manažér zdrojov v architektúre diferencovaných služieb. Riadi alokáciu zdrojov podľa požiadaviek na kvalitu služby v rámci jednej alebo následných domén podľa dostupných zdrojov a kontraktu o kvalite služby, ktorý bol dohodnutý medzi zákazníkom a poskytovateľom služby. Ak je použité multidoménové riadenie zdrojov, systémy dojednávania šírky pásma vyjednávajú navzájom v mene iniciátora služby. Navyše sa tieto systémy tiež zúčastňujú v komunikácii tranzitných domén koordinovaním dohôd o kvalite služby poskytovanej na rozhraní medzi doménami. Boli navrhnuté rôzne architektúry BB a niektoré z nich boli dokonca aj implementované. Všetky sú ale len v predbežnom štádiu alebo neriešia podstatné problémy, ako napr. obojsmerná alokácia zdrojov. V tomto článku navrhujeme BB architektúru, ktorá má funkcie riadenia pre autorizáciu, podporuje kvantitatívne a kvalitatívne služby, dojednáva obojsmernú alokáciu zdrojov a podporuje riadenie zdrojov, ktoré je nezávislé od výrobcu, t. j. nevyžaduje žiadne úpravy softvéru alebo hardvéru v chrbticových smerovačoch.

A Bandwidth Broker (BB) is an automated resource manager in the Differentiated Services architecture. It manages Quality of Service resource allocation requests within a single or successive DiffServ domains based on the available resources and on the Service Level Agreements formerly negotiated between the customer and its service provider. If used for multi-domain resource management, BBs negotiate among each other on behalf of the service initiator. Additionally, BBs also participate in transit domain communication by coordinating SLAs across domain boundaries. Various BB architectures have been proposed in the recent years, some of them even have been implemented, but all of them are in a preliminary stage or do not address important issues such as bi-directional resource allocation. In this paper we propose a BB architecture that includes policy manager functions for authorization, supports quantitative and qualitative services, handles bi-directional resource allocation and features a vendor-independent resource allocation approach, i.e., neither software nor hardware modification of core routers are required.

1. Introduction

Internet quality of service insurance has attracted substantial interest in recent years, because the best-effort service quality currently offered by the Internet does not meet requirements of real-time applications appearing on the market. Therefore, service providers are finding it necessary to offer their customers various levels of service. Continuing research has yielded two approaches for providing *Quality of Service (QoS)*. The *Integrated Services (IntServ)* architecture [2] differentiates services on a per-flow basis. The *Differentiated Services (DiffServ)* [1] [3] architecture, on the other hand, divides the network into domains with their own resources and aggregates traffic flows into classes to avoid the scalability problem of IntServ caused by the per-flow differentiation.

In this architecture QoS enforcement is basically realized by two entities [1]: *core routers* that are located internally in a DS domain without any interface to the outside world, and *edge routers* that connect one DS domain to a node in another DS domain or in a domain that is not DS capable. According to the former separation core routers only handle IP packets according to their pre-assigned service classes attempting to provide the required quality. On the other hand, edge routers realize the storing of flow related information and most of the complicated functions e.g. policy control and flow classification.

In the DS architecture two service models are distinguished, where the *service* is the overall treatment of a defined subset of a customer's traffic within a DS domain or end-to-end. In the *absolute service*, the user is assured of the requested performance level at the expense of strict admission control functions. Accordingly, requests are rejected if there is not enough available resource to accommodate the new service without violating the quality of already existing services. With *relative services*, users are only assured of relative differentiation; which can be provided without explicit admission control function by some active queue management algorithms or packet schedulers.

* **Octavian Pop, Tamás Máhr, Tímea Dreilinger, Róbert Szabó**
High Speed Networks Laboratory, Department of Telecommunications and Telematics, Budapest University of Technology and Economics,
H-1117, Pázmány Péter sétány 1/D., Budapest, Hungary, Tel.: +36-1-463 2187 Fax: +36-1-463 1763,
E-mail: {pop, mahr, dreiling, szabor}@ttt-atm.ttt.bme.hu

The *Internet Engineering Task Force* (IETF) has also developed RFCs that not only describe the above mentioned DiffServ functional architecture, but standardize some externally observable forwarding attributes called *Per-Hop-Behavior* (PHBs) of a DiffServ-compliant node.

Efficient use of absolute services implies signaling: customers should be able to signal their demands to the service provider. There is an obvious need for on-demand call admission control mechanisms to admit connection requests to the network based on available resources in the network. Following a positive acknowledgement edge routers must be configured accordingly. A policy manager should further authenticate and check the authorizations of each service requester based on the SLAs. *SLA (Service Level Agreement)* [1] is a service contract between a customer and a service provider that specifies the service customer should receive, where the customer may be a user organization or another DS domain. Flows traversing multiple DiffServ domains can only be served by enabling communication among successive *Internet Service Providers* (ISPs). All of these functions can be achieved by entrusting the resources of each DiffServ domain to a centralized manager agent called a *Bandwidth Broker* (BB) [6].

Figure 1 shows a sample network scenario [4]. Roughly, if user A wants to reserve resources for sending data to user B, he will send a request to the BB1. BB1 checks the SLA of user A *(policy server function)* and if authentication and permissions are adequate BB must check resource availability in its domain *(call admission control).* Since the reservation is initiated for multiple domains in our case, there is a need for communication between BB1 and BB2 *(inter-domain communication).* Note that BB2 also has to make the call admission control for its own domain to establish end-to-end reservation. Finally, the BB1 configures the edge routers to classify the user A data into the appropriate class.

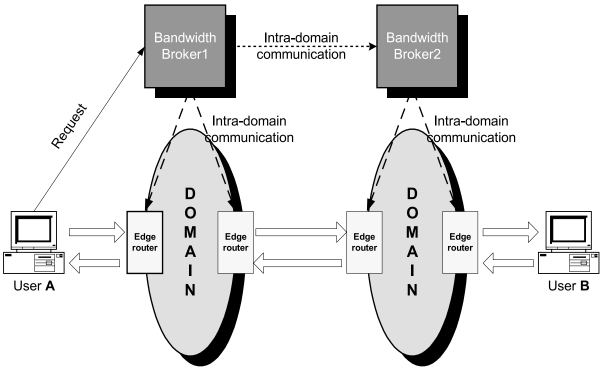

Fig. 1. Bandwidth Broker in a DiffServ domain

Our goal was to design and implement a Bandwidth Broker architecture for a Linux-based DiffServ test bed. We designed and added to our BB a resource reservation scheme that requires neither software nor hardware upgrades in core routers. There already exists a similar idea independently developed and presented in [7], however the method to determine the reservation path for flows was not sufficiently elaborated or presented. Therefore, we created and implemented a lightweight route discovery technique that is independent from the underlying routing protocol and the domain topology. Our proposal also handles path changing in a network. Another important issue is the problem of reservations for *destination requested flows (backward reservation)*, i.e., where the service requesting entity acts as a sink for a QoS flow while the sender is unaware of QoS signaling. Note that most candidate QoS applications require either bi-directional or such a destination requested flow, e.g. Internet telephony or video-on-demand. To the best knowledge of the authors neither the problems of reservation for destination requested flows nor their possible solutions are discussed in the literature. However, in our proposal we address this problem too.

In what follows, section 3 outlines our BB architecture. It covers the supported services and the necessary QoS forwarding mechanisms as well as our proposed resource allocation technique. We present our resource reservation signaling focusing on the proposed path discovery scheme for call admission control and the backward resource reservation method. Section 4 sets forth some implementation issues and finally we conclude the work in section 5.

2. The proposed BB Architecture

2.1. Supported Services

Our objective was to concentrate on two very typical services, one in the absolute, and one in the relative service group. Our first supported service is a *Virtual Leased Line* (VLL), the second, a relative one, is called *Better than Best Effort* (BBE), which is based on *assured forwarding* per-hop-behavior (AF-PHB) [12]. The VLL service, based on *expedited forwarding* per-hop-behavior (EF-PHB) [13] offers guaranteed bandwidth, delay and jitter characteristic for real-time applications. In contrast, BBE offers only three packet dropping levels that assure a higher-level class always has a lower dropping probability than a lower-level class. Since network resource usage cannot grow unbounded beyond the actual capacity, for VLL service it is necessary to support call admission control. In contrast, BBE requires no additional mechanisms, but the forwarding function, which itself assures the necessary relative service differentiation. BBE is only effective with applications using TCP, because it contains flow and congestion control mechanisms allowing active queue management algorithms to take back transmission rates by dropping packets. This way congestion results an unfair competition for resources between TCP and UDP flows [14]. Applications expected to potentially use VLL service in the future will require resource allocation from and/or to the service requester.

In order to use any of the VLL and BBE services users have to reach an agreement (SLA) with the service provider.

2.2. Call Admission Control

Connection admission control refers to the process performed by BB of admitting connection to the network based on available resources in the network. As we mentioned before, VLL requires a strict admission control scheme.

2.2.1 Distributed vs. centralized Call Admission Control

Some schemes use *hop-by-hop (distributed) methods*, where the overall decision is the sum of successive decisions performed locally at each node along the path that the flow traverses. Notable example protocols are RSVP (Resource reSerVation Protocol) [8] and Boomerang [9]. This scheme has two major drawbacks. First, this is vendor-dependent and each router has to support the same function to be effectively deployed. Second, timing considerations also need to be addressed: routers knowing of free resources in other routers on a timely basis could result in greater network utilization.

Alternatively, one may *centralize the admission control decision* for an administrative domain. However, this requires the BB to be aware of the flow's path and resource availability along this path. We solved this problem as follows: the BB determines the path in question via a lightweight route discovery method. Since this uses an IP's record route feature (a mandatory part of the protocol), this does not break compatibility with other routing protocols, and does not require modification of hardware/software in routers. This brings up another problem: change in the routing tables of domain routers is harder to handle. One may use an SNMP trap, but this requires the routers and topology maintenance of the domain. We propose to check the routes of active connections periodically instead. This also does not require modification of core router software/hardware.

2.2.2. Measurement vs. reservation based admission control

The decision may be based on *measurements*, this is mainly adequate for a centralized scheme. Edge routers may measure some performance metrics in a DiffServ domain and the results may directly determine the acceptance or rejection of new connections. The measurements can also be made on an end-to-end basis by checking the resource availability with probe traffic [10] [11]. The other alternative is to use a *reservation based admission control*, i.e. the used/available resources are tracked and allocation requests are rejected accordingly. We have used this kind of call admission as the VLL service must provide hard guarantees for real-time applications thus one cannot risk the overbooking here.

2.3. Resource Reservation Signaling

Resource reservation signaling consists of two sub-parts: route discovery and resource reservation for self-destined flows.

2.3.1. Route discovery signaling

We named our proposed route discovery scheme used along the call admission control to *Discover Forwarding Path* (DFP) mechanism and works as follows. Customers demand services by passing *Resource Allocation Requests* (RAR) messages to their BBs. The RAR contains the required service type, traffic descriptor (for VLL), destination address, ingress edge router, flow direction and time duration of service reservation. The customer may pass the RAR to the IP address of the BB. In our method, customers send the RAR to their ingress router, which in turn forwards it to the BB of the domain. The RAR of the user can thus omit the ingress edge router address; the edge router itself will add it to the RAR packet (Figure 2, messages #1 and #2). Upon the reception of a new RAR message in the BB, it asks the ingress edge router (message #3) to send a special empty IP packet with the IP *record route* option preset toward the flow's destination. This DFP packet follows the path that the flow will take. All core routers add automatically their used outgoing interface address to the DFP packet (Figure 2 messages #4, #5 and #6). Egress edge routers must capture these DFP packets before leaving the domain to turn it back to the BB (Figure 2. message #7). The BB now knows which router interfaces' utilization have to be checked and accepts/rejects the flow accordingly.

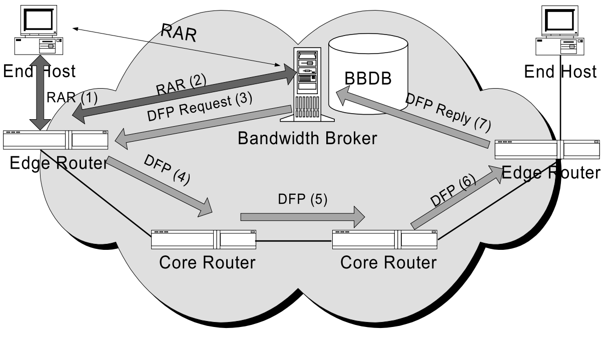

Fig. 2. Route discovery method

2.4. Backward resource reservation

We designed resource reservation signaling with resource reservation going from a sender to the service requester (backward direction) in mind. This is not a trivial venture, as initially nothing prevents destination-originated flows (service requester) from having different paths from source-originated flows. Technically, any resource reservation must be started from the source's domain and not from the service requester's. The RAR is then sent to the destination domain's BB, and therein lies our problem: the destination domain's BB does not have global network topology knowledge. Our ICMP-based solution is presented in Figure 3.

The numbers in figure 3 depict the following sequence of events:

1. User sends the RAR to its BB (through ingress edge router) for a flow having the destination the user itself.
2. BB belonging to the user's domain sends this request in a special marked ICMP-ECHO *request* packet to the flow source.
3. The source end-host automatically replies with an ICMP-ECHO *reply* packet

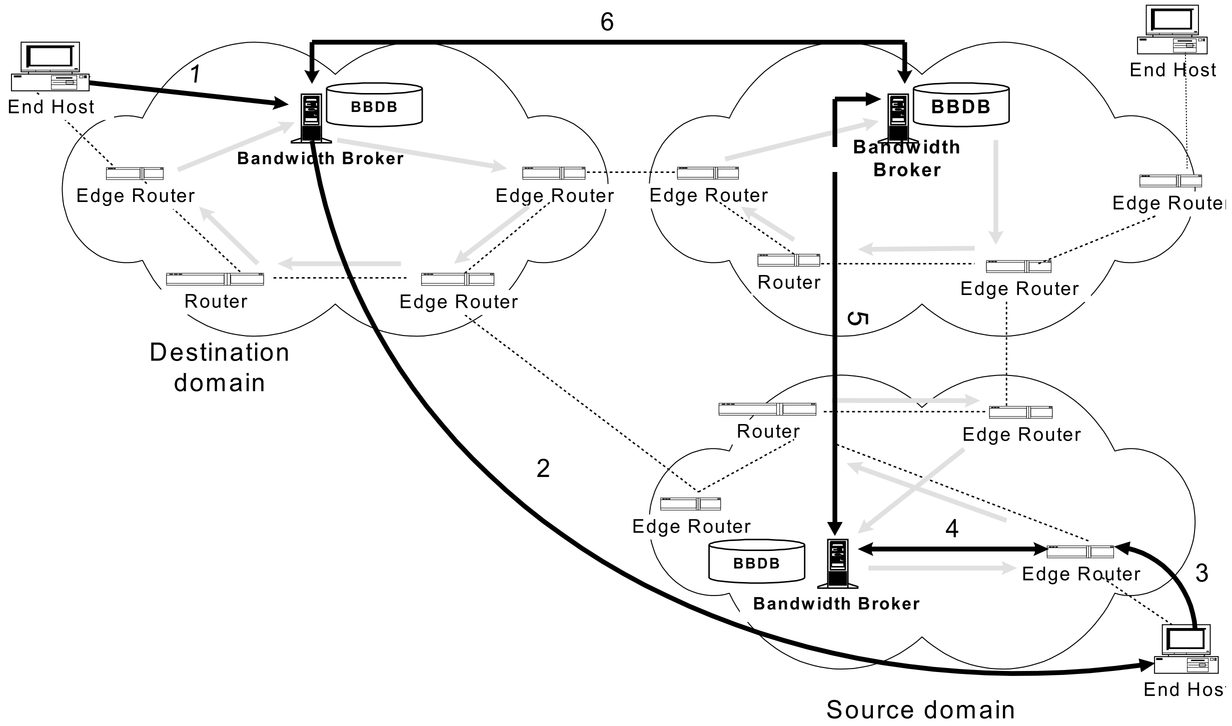

Fig. 3. Backward resource reservation

4. The source's edge router catches this packet, as it is a special marked ICMP-ECHO *reply*. Then, the withdrawn request is sent to the BB of the source domain.
5. This BB performs local Call Admission Control (CAC), and sends the request to the neighboring domain.
6. The BB of the neighboring domain also performs local CAC, and the request gets back to the destination domain. The BB of the destination domain – depending on the collected information and its resources – may accept or reject the request.

3. Implementation issues

Both our BB functions and the edge router control plane functionality are implemented in C programming language, while the policy database is stored in PostgreSQL.

Our BB implementation exhibits a modular architecture. The two pluggable modules are the *SQL module* and the *DFP module* that are connected to the core by predefined interfaces and can be exchanged. This is useful, if we for instance implemented a new

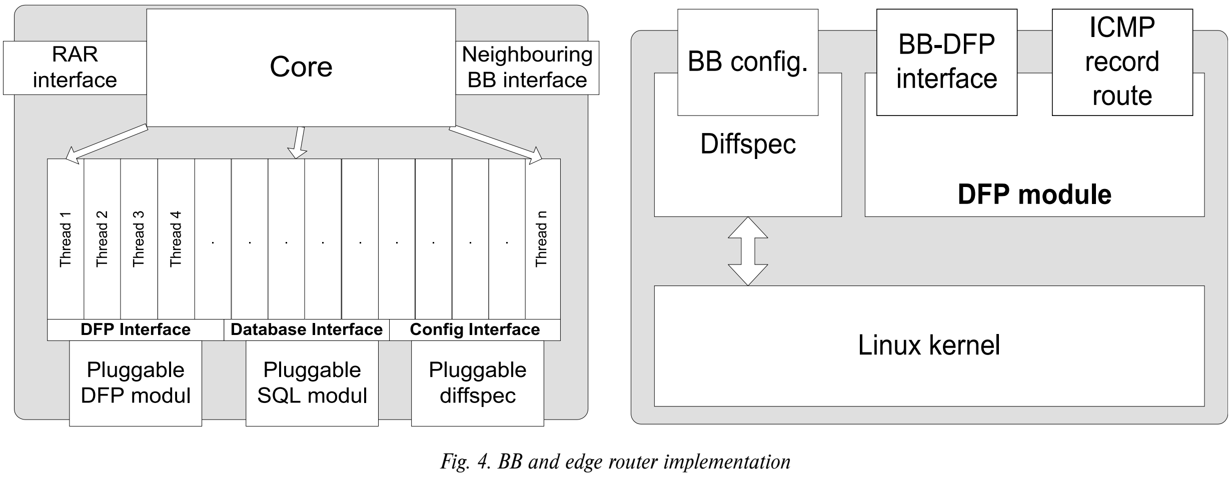

Fig. 4. BB and edge router implementation

DFP method or if one wants to replace the database server. The SQL module gives abstract access to a database and different implementations of the interface functions allow a given database to fit in the architecture seamlessly.

The BB core has two socket interfaces, which is supported to reach its services. One interface we defined for users to communicate with through the edge routers and one for neighboring BBs. Connection requests arrive to these sockets and the core runs threads for each request. The inter-domain request handling thread is not yet designed because there areconsiderations that interoperability issues should also be considered toward different vendor implementations. The user request handling thread accomplishes the CAC decision task after receiving the RAR from the user.

The edge router software consists of two parts. One of them is the *diffspec* daemon for configuring QoS mechanisms developed at University of Kansas [5] that handles the diffserv configuring requests of the BB (configures filters, meters, etc.). The other is the DFP handler that consists of two more processes. One to receive the DFP requests from the BB (ingress edge router) and send the IP addresses to the BB (egress edge router), while the other one launches the DFP packet (ingress) and catches them up (egress).

4. Conclusions

In this paper we have presented a Bandwidth Broker (BB) architecture. In the BB approach to DiffServ networks a centralized automated resource manager performs admission control, resource provisioning and other policy decisions.

Our proposed architecture manages QoS resource allocation requests arriving at a DiffServ domain based on available resources and SLAs in effect among users and ISPs. Hence our BB provides policy and resource-based call admission control, as opposed to other proposed solutions.

Since we chose the centralized call admission model, we designed and implemented a lightweight route discovery technique. This is independent from the underlying routing protocol and domain topology, solves the path-changing problem by periodic queries and it does not require any modification of core router software.

We proposed a possible implementation of resource reservation for destination-requested flows, i.e., a backward resource reservation, based on ICMP-ECHO messages. At the best information of the authors neither the former problem nor its solutions are discussed in the literature.

5. References

[1] BLAKE, S., BLACK, D., CARLSON, M., DAVIES, E., WANG, Z., and WEISS, W.: *An Architecture for Differentiated Services*, IETF RFC 2475, December, 1998.

[2] SHENKE, S., WROCLAWSKI, J.: *General Characterization Parameters for Integrated Services Network Elements*, IETF RFC 2215, 1997.

[3] BERNET, Y., BINDER, J., BLAKE, S., CARLSON, M., CARPENTER, B. E., KESHAV, S., OHLMAN, B., VERMA, D., WANG, Z., WEISS, W.: *A Framework for Differentiated Services*, IETF-draft, http://www.ietf.org, 1999.

[4] NEILSON, R., WHEELER, J., REICHMEYER, F., HARES, S.: *A Discussion of Bandwidth Broker Requirements for Internet2 Qbone Deployment*, August, 1999.

[5] http://www.ittc.ukans.edu/˜kdrao/845/

[6] NICHOLS, K., JACOBSON, V., ZHANG, L.: *A Two-bit Differentiated Services Architecture for the Internet*, Internet-draft, 1999.

[7] ZHI-LI ZHANG, ZHENHAI DUAN, LIXIN GAO, and YIWEI THOMAS HOU: *Decoupling QoS Control from Core Routers: A Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services*, Sigcomm 2000

[8] BRADEN, R., ZHANG, L., BERSON, S., HERZOG, S., JAMIN, S.: *Resource ReSerVation Protocol (RSVP) - Version 1 Functional Specification*, IETF RFC 2205

[9] FEHÉR, G., NÉMETH, K., MALIOSZ, M., CSELÉNYI, I., BERGKVIST, J., AHLARD, D., ENGBORG, T.: *Boomerang - A Simple Protocol for Resource Reservation in IP Networks*, IEEE WS on QoS Support for Real-Time Internet Applications, Vancouver, Canada, June 1999

[10] VIKTÓRIA ELEK, GUNNAR KARLSSON and ROBERT RÖNNGREN: *Admission Control Based on End-to-End Measurements*, IEEE INFOCOM 2000

[11] BIANCHI, G., CAPONE, A., PETRIOLI, C.: *Throughput Analysis of End-to-End Measurement-Based Admission Control in IP*, IEEE INFOCOM 2000

[12] HEINANEN, J., BAKER, F., WEISS, W., WROCLAWSKI, J.: *Assured Forwarding PHB Group*, IETF RFC-2597, 1999.

[13] JACOBSON, V., NICHOLS, K., PODURI, K.: *An Expedited Forwarding PHB*, IETF RFC-2598, 1999.

[14] SALLY FLOYD, KEVIN FALL: *Promoting the Use of End-to-End Congestion Control in the Internet*, IEEE/ACM Transactions on Networking, August 1999.