**Proceedings of the Third IASTED International Conference**
**Communications, Internet, and Information Technology**
**November 22-24, 2004, St. Thomas, US Virgin Islands**

# SCALABLE ARCHITECTURE FOR PROVIDING PER-FLOW BANDWIDTH GUARANTEES

Vasil Hnatyshin
Department of Computer Science
Rowan University
Glassboro, NJ 08028
USA
hnatyshin@rowan.edu

Adarshpal S. Sethi
Department of Computer and Information Sciences,
University of Delaware,
Newark, DE 19716
USA
sethi@cis.udel.edu

**ABSTRACT**
Despite numerous efforts, the problem of providing per-flow Quality of Service in a scalable manner still remains an active area of research. This paper introduces a scalable architecture for support of per-flow bandwidth guarantees, called the Bandwidth Distribution Scheme (BDS). The BDS maintains aggregate flow information in the network core and distributes this information among boundary nodes as needed. Based on the feedback from the network core the boundary nodes dynamically adjust resource allocations of individual flows. The BDS architecture consists of three main components: the admission control, the resource distribution mechanism, and the protocol for distributing the aggregate flow requirements in the network. This paper describes components of the BDS architecture and illustrates how they operate together to achieve scalable per-flow QoS.

**KEY WORDS**
Quality of service, bandwidth distribution, network feedback, resource allocation

## 1. Introduction

To solve the problem of providing scalable per-flow Quality of Service, a number of service differentiation models have been proposed. The Integrated and Differentiated Service (DiffServ) models are among the most prominent approaches to providing Quality of Service in the Internet. The Integrated Services model [1] requires each router in the network to reserve and manage resources for the flows that travel through it. In large networks, millions of flows may simultaneously travel through the same core routers. In such cases, managing resource reservations on a per-flow basis may cause enormous processing and storage overheads in the core routers. As a result, the Integrated Services model is considered to be not scalable to large networks and thus is not widely deployed in the Internet. The DiffServ model [2] attempts to solve the scalability problem of the Integrated Services approach by combining flows that have similar quality of service requirements into traffic aggregates or classes. The DS core routers process incoming traffic based on the class the packets belong to and thus maintain and manage resource reservations only on a per-class/per-aggregate basis. The DiffServ approach provides a scalable solution to the QoS problem but it supports only coarse per-aggregate guarantees which in certain cases may not be adequate.

This paper examines the architecture of an alternative approach, called the Bandwidth Distribution Scheme (BDS). The BDS core routers do not maintain per-flow information (e.g. bandwidth requirements of individual flows); instead core routers keep aggregate flow requirements. The amount of information kept in the network core is proportional not to the number of flows but to the number of edge routers, which we believe does not raise scalability concerns. The edge nodes maintain per-flow information and fairly allocate network resources (e.g. bandwidth) among individual flows according to the flow requirements and resource availability. The dynamic resource allocation at the edge routers is enabled by the network feedback which consists of periodic path probing and explicit congestion notifications. Overall, the BDS architecture consists of: the admission control mechanism, which determines if a new flow can be admitted into the network, the resource allocation mechanism, which fairly distributes available bandwidth among individual flows, and the protocol for distribution of the aggregate flow requirements, which provides feedback to the network routers about the changes of network characteristics.

The BDS approach relies on the basic idea of performing per-flow management at the network edges and processing traffic aggregates in the network core. This idea is not new and has been examined before, for instance in [2, 3, 4, 5]. However, the primary contribution of this work is a novel approach to aggregating flow information in the network core, dynamically distributing it among edge nodes, and then using the aggregate flow requirements for fair distribution of available bandwidth among individual flows.

The rest of the paper is organized as follows. Section 2 presents an overview of the BDS architecture. Section 3 introduces specifications and definitions used in the BDS approach. The resource management mechanism is presented in Section 4, while Section 5 discusses the BDS protocol for dynamic distribution of aggregate flow requirements. Section 6 discusses implementation issues of the Bandwidth Distribution Scheme. Finally, discussion

and conclusions are presented in Sections 7 and 8 respectively.

## 2. The overview of the BDS architecture

The X-BDS architecture provides a scalable solution to the problems of fair per-flow bandwidth distribution and congestion control. This architecture consists of three components: a set of specifications and definitions, the resource management mechanism, and the Requested Bandwidth Range (RBR) Distribution and Feedback (RDF) protocol. The BDS specifications and definitions consist of the network architecture which defines the working environment of the BDS and makes the proposed solutions scalable to large networks, definition of the flow requirements which outlines the format for user expectations for traffic treatment, and definitions of fairness which specify what it means for the resource distribution to be fair. The BDS resource management mechanism consists of per-flow admission control that denies access into the network for those flows that violate existing per-flow guarantees and per-flow resource allocation that dynamically allocates available bandwidth to individual flows. The Requested Bandwidth Range (RBR) Distribution and Feedback protocol (RDF) is the glue that holds the BDS approach together. The RDF protocol is a distributed explicit message exchange protocol that provides feedback about the changes of network characteristics. More specifically, the RDF protocol distributes aggregate flow requirements among the routers in the network and generates explicit congestion notifications when needed. Figure 1 illustrates the BDS architecture.

| Specifications and Definitions | Resource Management | RDF Protocol |
|---|---|---|
| *Network Architecture* | *Admission Control* | |
| *Flow Requirements* | *Resource Allocation* | |
| *Definitions of Fairness* | | |

**Figure 1. The BDS Architecture**

## 3. The BDS Specifications and Definitions

### 3.1 Network Architecture

The Internet consists of a large number of routers that are traditionally grouped into independent network domains as shown in Figure 2. A cluster of interconnected routers that are governed by the same administrator are called a *network domain*. Each network domain contains two types of nodes: the *edge* or *boundary* routers and the *core* routers. Traffic enters a network domain through the edge nodes called *ingress* routers. It further travels through the core routers to reach the network boundary and exits the domain through the edge nodes called *egress* routers.

The BDS core routers do not perform per-flow management and treat arriving traffic on per-aggregate basis, in a way similar to that of the Differentiated Services nodes [2]. The BDS core routers provide feedback to the boundary nodes about the changes of network conditions. The edge nodes maintain per-flow information and manage activation and termination of the flows. Based on the provided feedback the edge nodes compute the fair shares of bandwidth for their flows and then allocate available resources accordingly.

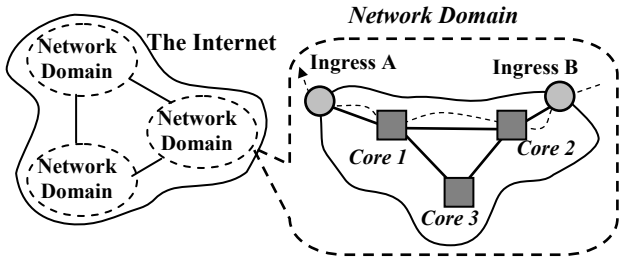


**Figure 2. The BDS Network Architecture**

It is reasonable to assume that the number of active flows that enter and exit the network domain through a particular edge router is fairly small. Thus, managing per-flow information at the network boundaries allows this network architecture to scale well to large networks [2]. Furthermore, this architecture does not require the BDS approach to be set-up everywhere in the Internet at once. Instead, each network domain can choose to support the Bandwidth Distribution Scheme at its own discretion, which facilitates incremental deployment of the BDS architecture in the Internet. If a network domain decides to support the BDS, then a certain amount of resources are allocated for the BDS traffic. These resources are fairly distributed among the BDS flows only, thus isolating the BDS traffic from the rest of the non-BDS flows whithin this domain. This paper examines the architecture of the Bandwidth Distribution Scheme within the confines of a single network domain. We plan to address the issue of inter-domain traffic and deployment of the BDS approach in the Internet in future work.

This paper defines a "flow" to be a sequence of packets that travel from a given source host to a given destination host. We only consider the flows that receive the BDS treatment and which are therefore subject to the BDS resource allocation. Similarly, the terms “resources”, “capacity”, “load,” or “bandwidth” mean the resources, bandwidth, etc. explicitly allocated by the network administrator for the BDS traffic. This definition of a flow, while different from the more conventional definition as a sequence of packets between individual source-destination applications (e.g., TCP or UDP streams), was chosen to simplify the presentation of the BDS scheme. The BDS architecture, as presented here, can be easily extended to apply to the conventional definition of a flow.

### 3.2 Flow Requirements and Definitions of Fairness

In this paper we assume that both the minimum and the maximum transmission rates of a flow are known ahead of time. Thus, the flow requirements are defined in the form of a range which is called the Requested Bandwidth Range (RBR). The RBR of flow $f$, $RBR^f$, consists of two

values: a minimum rate, $b^f$, below which the flow cannot operate normally, and the maximum rate, $B^f$, that the flow can utilize.

$$RBR^f = [b^f, B^f] \quad (1)$$

Consider a core router's interface $k$ and a set of flows, $F^k$, that travel through it. The set $F^k$ can be divided into two disjoint subsets: the subset, $F_B^k$, of flows that have link $k$ as their bottleneck and the subset, $F_{NB}^k$, that contain all the other flows. These subsets are called *bottleneck flows* and *non-bottleneck flows*, respectively.

$$F^k = F_B^k \bigcup F_{NB}^k \quad (2)$$

The *aggregate bottleneck RBR* and the *aggregate RBR* on interface $k$ are defined as follows:

$$b_B^k = \sum_{f \in F_B^k} b^f \qquad B_B^k = \sum_{f \in F_B^k} B^f \quad (3)$$

$$b^k = \sum_{f \in F^k} b^f \qquad B^k = \sum_{f \in F^k} B^f \quad (4)$$

The aggregate bottleneck RBR is the sum of the RBRs of the bottleneck flows on link $k$, while the aggregate RBR is the sum of the RBRs of all the flows that travel through link $k$. The total allocated rate of the non-bottleneck flows is called the *non-bottleneck rate* and is denoted as $R_{NB}^k$. The amount of bandwidth left for distribution among the bottleneck flows is the difference between the capacity of link $k$ and the non-bottleneck rate. This value is called the *bottleneck capacity,* $C_B^k$.

$$C_B^k = C^k - \sum_{f \in F_{NB}^k} R^f = C^k - R_{NB}^k \quad (5)$$

When a link is not fully utilized, its bottleneck capacity could be larger then the sum of the allocated rates of the bottleneck flows. We introduce two definitions of fairness. First, the proportional fair share, $FS_f^k$, of the flow $f$ on link $k$ is defined as follows:

$$FS_f^k = b^f + (C_B^k - b_B^k)\frac{b^f}{b_B^k} = C_B^k \frac{b^f}{b_b^k} \quad (6)$$

Using definition (6), each flow is allocated its minimum requested rate plus a share of leftover bandwidth. We call this definition of fairness *proportional fairness* because each flow receives an amount of bandwidth proportional to its minimum requested rate. This definition of fairness should not be confused with Kelly's proportional fairness [6, 7], which deals with different issues. Throughout the rest of this paper, the term "proportional fairness" refers to the definition of fairness specified by equation (6).

A second definition of fairness uses a similar idea, except that the excess bandwidth is distributed proportionally to the difference between the flow's maximum and minimum requested rates. The idea is to allocate resources proportionally to the amount of bandwidth a flow needs to be completely utilized. We assume that a flow is completely utilized when it sends traffic at its maximum requested rate, $B^f$. That is why this definition of fairness is called *maximizing utility fairness*. The maximizing utility fair share, $FS_f^k$, of flow $f$ on link $k$ is computed as follows:

$$FS_f^k = b^f + \left(C_B^k - b_B^k\right)\frac{B^f - b^f}{B_B^k - b_B^k} \quad (7)$$

# 4. The Resource Management Mechanism

## 4.1 The Admission Control

The BDS network guarantees that each flow would receive at least its minimum requested rate, $b^f$, while the leftover resources in the network are fairly distributed among participating flows. To achieve these guarantees, the network allocates to each flow an amount of bandwidth not smaller than the flow's minimum requested rate, and denies network access to those flows whose minimum rate guarantees cannot be met.

The purpose of admission control is to determine if a new flow can be admitted into the network at its minimum rate without violating existing QoS guarantees of other flows. The problem of admission control was extensively examined in the literature [6, 8, 9]. Traditionally, there are two types of admission control: *parameter-based* and *measurement-based*. In parameter-based admission control, the decision to admit a new flow is derived from the parameters of the flow specification. Usually, this type of admission control relies on worst-case bounds and results in low network utilization, although it does guarantee supported quality of service. Measurement-based admission control relies on measurements of the existing traffic characteristics to make the control decision. Measurement-based admission control supports higher network utilization. However, measurement-based admission control may occasionally cause the quality of service levels to drop below user expectations because of its inability to accurately predict future traffic behavior.

Since the network guarantees that each flow will receive at least its minimum requested rate, the edge nodes should check the current resource allocation on a path before granting a new flow request. Thus, to admit a new flow into the network, the edge routers verify that the sum of the minimum requested rates of all the flows that follow a particular path, including a new flow, is smaller than the capacity of the bottleneck link on that path. Link $k$ is a *bottleneck link* for flow $f$ traveling on path $P$ if $k$ limits the transmission rate of $f$ on $P$.

We formally define the BDS admission control as follows. Consider a network consisting of a set of $L$ unidirectional links, where link $k \in L$ has capacity $C^k$. The network is shared by the set of flows, $F$, where flow $f \in F$ has the RBR of $[b^f, B^f]$. At any time, the flow transmits packets at a rate $R^f$, called the *allocated rate*, which lies between $b^f$ and $B^f$. Let $L_f \subseteq L$ denote the set of links traversed by flow $f$ on its way to the destination. Also let $F^k \subseteq F$ denote the set of flows that traverse link $k$. Then a new flow $\phi$ with the RBR of $[b^\phi, B^\phi]$ is accepted in the network if and only if:

$$b^{\phi} + \sum_{f \in F^k} b^f \le C^k \quad \forall k \in L_{\phi} \tag{8}$$

Thus, a new flow, $\phi$, is accepted into the network only if the sum of the minimum requested rates of the active flows, including the new flow, is not larger than the capacity of each link on the path of flow $\phi$ to the destination. Equation (8) is often called the *admission control test*.

### 4.2 The Resource Allocation

To distribute bandwidth according to equations (6) – (7), the resource management mechanism requires the knowledge of such link characteristics as the aggregate bottleneck RBR and the bottleneck capacity. However, these characteristics are not readily available in the network. Instead the core routers keep track of the capacity, the arrival rate, and the aggregate RBR for each outgoing link and distribute this information among the edge nodes. The edge nodes use the aggregate RBR and link capacity instead of the aggregate bottleneck RBR and the bottleneck capacity to compute fair shares of individual flows. The edge nodes compute the fair share of flow $f$ on its bottleneck link $k$ using the proportional and maximizing utility definitions of fairness as shown below. However, the flows that do not have link $k$ as their bottleneck would not adjust their allocated rates.

$$FS_f^k = b^f + (C^k - b^k)\frac{b^f}{b^k} = C^k \frac{b^f}{b^k} \tag{9}$$

$$FS_f^k = b^f + \left(C^k - b^k\right)\frac{B^f - b^f}{B^k - b^k} \tag{10}$$

Clearly, such resource distribution may leave link $k$ underutilized because the non-bottleneck flows will transmit data at rates below their fair shares on link $k$. Thus, the edge nodes require additional means for utilizing the leftover resources. A "water-filling" technique employed for implementation of the max-min fairness [10, 11] allows the edge nodes to completely distribute leftover capacity. The idea of the "water-filling" is to increase allocated rates of individual flows as long as the bottleneck link is not fully utilized.

Periodic path probing, that delivers information about availability of resources on the path, enables the edge routers to implement the "water-filling" technique. Thus, in the presence of excess bandwidth the edge routers increase allocated rates of individual flows until available bandwidth on the path is consumed completely. It was shown in [12] that by distributing leftover bandwidth proportionally to the individual flow requirements the resource management mechanism achieves an optimal resource distribution defined by equations (6) – (7).

The resource management mechanism enforces resource allocation through the token bucket. Thus, if a flow transmits data above its allocated rate then the token bucket discards all excess traffic of that flow. As a result, the flow injects the amount of data into the network that corresponds to its share of the allocated resources.

## 5. The RDF Protocol

The feedback protocol that governs the information sharing between the nodes in the BDS network is called the RBR Distribution and Feedback (RDF) protocol. The RDF protocol is one of the most important components of the BDS. The RDF protocol operates as the "glue" that holds the BDS architecture together by supplying information to the admission control and the resource management mechanism. The RBR Distribution and Feedback protocol consists of two major components that determine its name: distribution of the aggregate RBR and periodic feedback from the network.

The RBR Distribution and Feedback protocol consists of three distinct phases: the path probing phase, the RBR update phase, and the notification phase. Each phase is classified based on the information flow as either edge-to-core or core-to-edge. During the edge-to-core phases, the information travels from the edge nodes into the network core to update the aggregate flow requirements stored in the *Interfaces* Tables. At the same time, during the core-to-edge phases, information about the status of the network core is being distributed among the edge routers to refresh network information stored in the *Path* and the *Link* Tables.

The path probing phase discovers characteristics of a particular path and works as follows. The ingress node periodically generates a probe message on a path while the egress node sends the probe message with collected path characteristics back to the ingress node. The ingress node uses received information to update its Path and Link Tables. In addition, the core routers can discover the edge nodes that send traffic through their interfaces using the path probing phase. For example, upon the probe message arrival, the core routers retrieve the identity of the edge node that generated this probe and update corresponding edge node entry in the Interfaces Table, which contains the identity of the edge router and a countdown timer. If the entry in the Interfaces Table for this edge router already exists, then the core node resets the corresponding countdown timer. Otherwise, the core router creates a new entry for this edge router. The core router discards the edge node's information whenever the countdown timer expires.

The purpose of the RBR update phase is to notify the core routers about the changes to the aggregate RBR information upon flow activation or termination. The edge routers initiate the RBR update phase by generating the RBR update message on a particular path. Each core router renews its Interfaces Table based on the information received from the RBR update message. The egress node terminates progress of the RBR update message.

Only in the event of congestion do the core routers initiate the notification phase. In this case, the core routers generate congestion notification messages to the edge routers asking them to adjust allocated rates of their flows. The edge routers update their Path and Link Tables and re-compute the fair shares of the corresponding flows based on the information received from the congestion notification messages.

## 6. Implementation Issues

This section discusses implementation details of the BDS architecture. In particular, it describes a set of data structures maintained in the routers of the BDS domain that allow answering the following questions: How do the core routers identify the edge nodes that should be notified about the network changes? How do the edge routers identify the flows that should adjust their allocated rates in the event of the network change?

### 6.1 The BDS Traffic Processing at the Edge Routers

The edge routers maintain the Service Level Agreement (SLA) Table that keeps track of the per-flow requirements negotiated between the end-user and the network domain. An entry in the SLA Table usually contains the following information: the source and destination addresses, the requested bandwidth range, the current allocated rate of a flow, and possibly flow status (e.g., active, idle, etc.). Since the flow SLA is usually negotiated ahead of time and is not frequently updated, the SLA Table is sorted and indexed based on the source-destination pair. This sorting allows efficient retrieval and update of flow information.

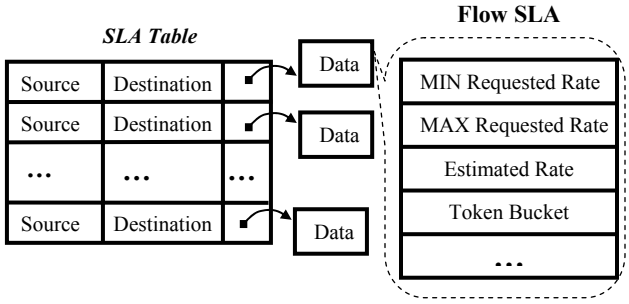


**Figure 3. The SLA Table**

The source-destination pair was chosen as the primary index into the SLA Table because we define a flow as a sequence of packets that travel from the source node to the destination. This definition of a flow simplifies the overall traffic classification at the edge routers. Identifying flows based only on the source-destination pair may limit flexibility of the flow differentiation (e.g., such flow identification does not allow distinguishing different flows that originate from the same source and travel to the same destination). However, such flow identification has little or no effect on the overall BDS architecture, since the flow definition could be easily extended to support a more complex differentiation without any modification to the main BDS components (e.g., admission control, resource management, or the feedback protocol). For example, the end user and the network domain can use the Type of Service (ToS) field of the IP header as an additional parameter for flow discrimination, which would require a slight modification of the classification element and the SLA Table only.

Let us examine how an arriving packet is processed at the edge router. Upon a new packet arrival, an edge node uses the header of the arriving packet to determine the identity of the flow the packet belongs to. Then, the edge node retrieves the status of the flow, which specifies if a flows is active or not, along with the flow's characteristics from the SLA Table.

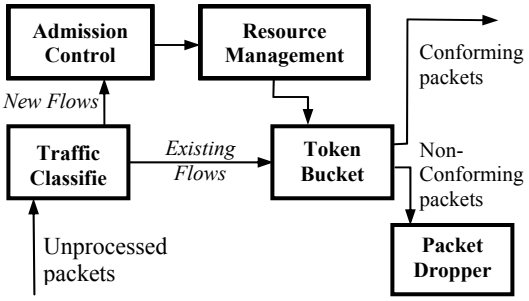


**Figure 4. Packet processing at the edge nodes**

If a packet belongs to a new flow, then the edge node initiates the admission control procedures. If the flow can be accepted into the network, then the edge node allocates a share of available resources to the new flow, which may entail adjusting allocated rates of the other flows, and forwards the packet further. Otherwise, the packet is dropped, and the source is notified that its request cannot be granted.

If the packet belongs to an already active flow, then the edge node verifies if the packet conforms to the rate requirements by passing the packet through that active flow's token bucket. The token bucket of each flow has its token generation rate set to the flow's allocated rate as determined by equations (10) and (11). If the packet conforms to the token bucket specification, then the packet is forwarded further. Otherwise, the packet is discarded. Figure 4 shows the packet processing at the edge router.

As Figure 4 shows, a packet from a new flow passes through the admission control and the resource management units before being forwarded. However, to compute the allocated rate of a flow, or to determine if the flow can be admitted into the network, an edge node needs to know the path that the flow will traverse and identity and characteristics of the bottleneck link. To provide the necessary information for the admission control and the resource management units, edge nodes maintain additional data structures that keep track of the active paths. We define a *path* or a *route* as a sequence of interfaces or links from the ingress router to the corresponding egress node. An *active path* is a route that is currently being used by at least one active flow.

The table that keeps track of all active paths and their characteristics is called a *Path Table* and is indexed by the identity (e.g., IP address) of the egress node. An entry in the Path Table contains a complete path to the corresponding egress node and a list of flows that traverse it. To minimize the amount of redundant information kept in the Path Table, the edge routers also maintain a *Link Table*. The Link Table keeps track of the individual link characteristics and is indexed by the IP address of the link (or more precisely, the IP address of the outgoing interface for the link). The list of flows and the complete path maintained in each Path Table entry are implemented via linked lists of pointers as shown in Figure 5.

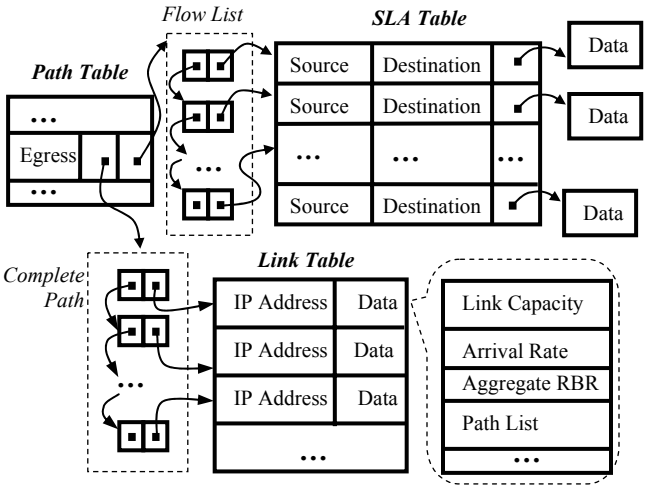


**Figure 5. Data structures maintained in the edge node**

The SLA Table is primarily used for packet classification, while the Path and the Link Tables are primarily used for admission control and resource management. For example, to determine if a new flow could be admitted into the network, the edge router identifies the path that the flow will follow, retrieves information about the bottleneck link on that path, and then performs the admission control test. In addition, the Path and the Link Tables allow the edge nodes to identify the flows influenced by the network changes.

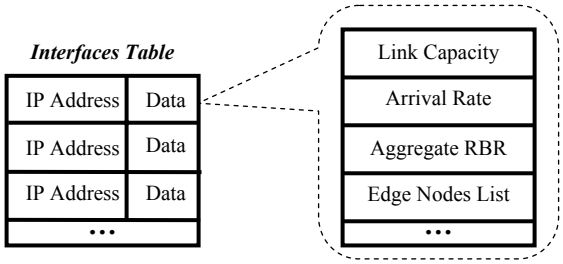


**Figure 6. The Interfaces Table**

Let us assume that characteristics of a particular link $k$ in the network were changed causing the flows that travel through $k$ to adjust their allocated rates. Once the edge router learns about this event it identifies the flows that should adjust their allocated rates as follows. First, the edge router identifies the entry in the Link Table of link $k$. Then, the edge router retrieves the list of paths that $k$ belongs to. Next, for each entry in the path list, the edge node retrieves a set of flows that follow the corresponding path. The union of the resulting flow sets is a list of all the flows that travel through link $k$ and thus should adjust their allocated rates.

### 6.2 The BDS Traffic Processing at the Core Routers

Now let us examine the data structure that helps the core routers to identify a set of edge nodes that should be notified in the event of network changes (e.g., the change of the characteristics of the outgoing link). Each core router maintains a single Interfaces Table (Figure 6) that keeps track of the core router's interface characteristics. Each entry in the Interfaces Table maintains an identity (e.g., IP address) of the outgoing link and its characteristics, including the list of edge nodes that send traffic though this interface.

When characteristics of the link change (e.g., the link became congested), the core router consults the Interfaces Table to retrieve identities of the edge routers send traffic through that link. The core router generates and sends explicit notification messages to each router in the retrieved list. Upon such message arrival, the edge routers adjust allocated rates of the corresponding flows as described in Section 6.1.

## 7. Discussion and Related Work

Most of the current architectures that support QoS in the Internet have emerged from various proposals by the Internet Engineering Task Force (IETF). In 1994, the IETF introduced Integrated Services [1] architecture, followed by the Differentiated Services [2] model in 1998. Although both approaches address the same problem of supporting quality of service in the Internet, they are different in terms of implementation and provided services. Integrated Services supports end-to-end guarantees on a per-flow basis, while DiffServ attempts to provide end-to-end guarantees based on per-hop assurances for a small set of predefined traffic classes. At the implementation level, Integrated Services requires per-flow management in the network core, while the Differentiated Services model employs a network architecture that pushes per-flow management to the network edges.

The architecture presented in this paper attempts to combine advantages of the Differentiated and Integrated Services models and provides support for building per-flow QoS services in a scalable manner. The network architecture enables the Bandwidth Distribution Scheme to become scalable, while per-flow bandwidth distribution at the network edges provides framework for deployment of the QoS services on a per-flow basis.

The RDF protocol relies on periodic path probing and explicit network feedback for dynamic bandwidth allocation and congestion control. This idea is not new and it has been examined before. In particular, the Explicit Congestion Notification (ECN) extension to IP [13] uses binary feedback to notify ECN-capable transports about congestion occurrences. Unlike the ECN extension, the network feedback in the BDS model not only notifies the edge routers about congestion but also carries additional information such as the arrival rate and aggregate flow requirements on the congested link. A similar idea is used in ATM networks for Available Bit Rate (ABR) congestion control [14], where the feedback carried by the resource management cells also includes rate information. The Explicit Control Protocol (XCP) [15] generalized the ECN proposal by sending additional information about congestion. XCP also does not require per-flow information in the network core. However, unlike BDS, XCP is not a rate-based but a window-based protocol that separates utility control from the fairness control.

The BDS approach is designed primarily for support of per-flow bandwidth guarantees. A similar feedback-based idea of providing dynamic per-flow bandwidth allocation for elastic traffic sources called simple rate control algorithm was introduced in [3]. A traffic source is called elastic if it does not require a fixed rate and can adjust its transmission rate as needed. Unlike BDS, the boundary nodes in the simple rate control algorithm employ knowledge of the level of network congestion and the user utility functions to determine a fair resource distribution among elastic sources. The end users obtain the level of congestion through the explicit acknowledgements (ACK) that carry the number of congested links on a particular path.

The Stateless-Core approach [4] provides an interesting solution for supporting per-flow QoS without keeping per-flow information in the network core. The main idea of this scheme relies on the Dynamic Packet State (DPS), where control information is carried in the IP header of the data packets [5]. The routers use the DPS information to provide per-flow guarantees without maintaining per-flow state in the network core. However, because of such features as required route pining without which the SCORE/DPS model will fail, use of existing IP header fields to encode control information, additional per-packet processing in the network core, inability to distribute excess bandwidth, and possible network underutilization due to use of the upper bound of the aggregate reservation for admission control, the Stateless-Core architecture may not be deployed soon in today's Internet [4].

Evaluation of the BDS architecture [12, 16, 17] showed that the Bandwidth Distribution Scheme is capable of fair distribution of available bandwidth among individual flows under all network conditions and thus, can support deployment of per-flow QoS services. Furthermore, the BDS eliminates congestion when it arises, keeps network utilization high by fair distribution of leftover bandwidth among corresponding flows, causes small overhead, and dynamically adjusts resource allocation in respect to network changes. Finally, evaluation of the BDS approach suggests that the Bandwidth Distribution Scheme is scalable to large networks, although further investigation of this feature under more realistic conditions is still needed.

## 8. Summary and Conclusions

In this paper we presented a scalable architecture for deployment of per-flow QoS services in computer networks. The BDS consists of three major components: the admission control, the resource management mechanism, and the RDF protocol. These components together with the network architecture and a set of definitions and specifications form the architecture for the Bandwidth Distribution Scheme. This paper in detail describes the BDS components, the data structures maintained in the BDS routers, and how the Bandwidth Distribution Scheme operates to achieve fair distribution of bandwidth among individual flows.

**REFERENCES:**


[1] R. Braden, D. Clark, S. Shenker, "Integrated Services in the Internet Architecture: an Overview", June 1994, IETF RFC 1633.

[2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss," An Architecture for Differentiated Services," December 1998. IETF RFC 2475.

[3] K. Kar, S. Sarkar, L. Tassiulas, " A Simple Rate Control Algorithm for Maximizing Total User Utility," *Proc. of INFOCOM'01 2001*, April 2001.

[4] I. Stoica, "Stateless Core: A Scalable Approach for Quality of Service in the Internet," Ph.D. Thesis, Carnegie-Mellon Univ., 2000.

[5] I. Stoica, H. Zhang, "Providing Guaranteed Services without Per-Flow Management," *Proc. ACM SIGCOMM'99*, September 1999.

[6] F. Kelly, P.B. Key, and S. Zachary, "Distributed Admission Control," *IEEE Journal on Selected Areas in Communications*, *18* (2000), pp.2617-2628.

[7] F. Kelly, A. Maulloo, and D. Tan, "Rate Control for Communication Networks: Shadow Prices, Proportional Fairness and Stability," *Journal of the Operational Research Society*, *49*(3), 1998, 237-252.

[8] L. Breslau, S. Jamin, and S. Shenker, "Comments on the Performance of Measurement-Based Admission Control Algorithms," *Proc. of IEEE INFOCOM'00*, March 2000.

[9] R. Gibbens and E. Kelly, "Measurement-based connection admission control", *Proc. of 15th International Tel-traffic Congress*, Amsterdam, Netherlands, June 1997.

[10] P. Marbach, "Priority Service and Max-Min Fairness," In *Proc. of IEEE INFOCOM'02*, June 2002.

[11] H. Tzeng and K. Sui, "On Max-Min Fair Congestion Control for Multicast ABR Service in ATM," *IEEE Journal on Selected Areas in Communications*, 15(3), 1997, 545-556.

[12] V. Hnatyshin, "Dynamic Bandwidth Distribution Techniques For Scalable Per-Flow QoS," Ph.D. Thesis, University of Delaware, 2003.

[13] K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", September 2001, IETF RFC 3168.

[14] S. Kalyanaraman, R. Jain, S. Fahmy, R. Goyal, and B. Vandalore, "The ERICA Switch Algorithm for ABR Traffic Management in ATM Networks," *IEEE/ACM Transactions on Networking*, *8*(1), 2000, 87-98.

[15] D. Katabi, M. Handley, and C. Rohrs, "Internet congestion control for future high bandwidth-delay product environments," *Proc. ACM SIGCOMM'02*, August 2002.

[16] V. Hnatyshin and A. S. Sethi, "Reducing load distribution overhead with message aggregation," *Proc. of the 22nd IEEE IPCCC'03 Conference*, Las Vegas, NV, 2003, 227-234.

[17] V. Hnatyshin and A.S. Sethi, "Fair and Scalable Load Distribution in the Internet," *Proc. of the International Conference on Internet Computing*, pp. 201-209, June 2002.