Constructing End-to-End Traffic Flows for Managing Differentiated Services Networks

Jae-Young Kim[1], James Won-Ki Hong[1], Sook-Hyun Ryu[1], and Tae-Sang Choi [2]

[1] Department of Computer Science and Engineering
Pohang University of Science and Technology
{jay,jwkhong,shryu}@postech.ac.kr
[2] Internet Architecture Team
Internet Technology Department
Electronics and Telecommunications Research Institute
choits@etri.re.kr

Abstract. Differentiated Services (DiffServ), presently being standardized by IETF, is considered to be a promising solution for supporting different service characteristics to different classes of network users on the Internet. The IETF DiffServ working group has defined a general architecture of DiffServ and is elaborating more detailed features. A simple but powerful management mechanism is needed to operate, provision, monitor and control DiffServ networks. Managing end-to-end traffic flows is one of the key components for managing DiffServ networks. Various high-level management functions can be built by using the flow information. In this paper, we present our work on designing a system architecture for managing DiffServ networks using the SNMP framework. DiffServ routers with SNMP agents have been developed, and a management system constructing end-to-end traffic flows has been designed.

1 Introduction

Over the past decade, the number of devices and the number of Internet users have increased at an exponential rate and the network traffics caused by data transfers will continue to rise. While previous network bandwidths were sufficient to carry text-based application data, current network bandwidths are no longer sufficient to handle multimedia, real-time network traffic flows.

Because the increase rate of network bandwidth is much slower than the increase rate of network usage, bottleneck points, where bandwidth is insufficient for network users, are commonly observed. In such situations, every packet competes for access to the bandwidth and the result is packet loss, unexpected delays, and jitter. However, both Transmission Control Protocol (TCP) and Internet Protocol (IP), two network protocols for delivering packets in the Internet, were originally designed in the best-effort service model.

But users' requirements have been changing. Users want to get different service qualities for different types of services they obtain. Integrated Services (IS) [1] with Resource reSerVation Protocol (RSVP) [2] signaling is the first approach to provide such a service on the Internet. RSVP attempts to provide per-flow QoS support assurances with dynamic resource reservation. A flow is defined by the 5-tuple, consisting of source and destination IP address, transport protocol, and source and destination port. However, since RSVP/IS relies on per-flow states and per-flow processing in every network node, it is difficult to deploy RSVP/IS in large carrier networks like the Internet.

Differentiated Services (DiffServ) is an alternative approach to provide differentiated service qualities to different classes of users. DiffServ uses aggregation of traffics in each routing decision point. Type of Service (ToS) field is used for distinguishing these traffic aggregates. Since the ToS is much simpler than the 5-tuple information, it is easier to implement DiffServ than RSVP/IS [3, 4, 5].

DiffServ applies administrative domain concepts. Within one domain, core routers forward traffics according to the ToS field of traffic aggregates. Between two different domains, there are edge routers which perform classification of flows based on 5-tuple information like RSVP/IS. Since the edge routers mark the ToS field of incoming traffics, core routers do not need to handle complex information.

Although the IETF DiffServ working group has defined several standards for DiffServ, the management aspect of DiffServ is not yet fully standardized. Current standards have defined only the operational aspects of DiffServ. When deploying DiffServ in network nodes, various management functions are needed for remote control of a large number of DiffServ nodes. Possible management considerations are how to configure each DiffServ router, how to change its configuration, and how to monitor or meter traffic each router handles.

Constructing end-to-end traffic flows in DiffServ networks is a key component of managing DiffServ networks. An end-to-end traffic flow consists of a routing path from a source edge router to a destination edge router and performance parameters of packet streams with a given ToS field over the routing path. Various high-level management functions such as bottleneck detection, topology mapping, Service Level Agreement (SLA) monitoring, etc., can be built by using the flow information. Since current management efforts are only focusing on element management of each DiffServ router, the end-to-end traffic flows have to be constructed by using the current element management functions.

In this paper, we propose a method for managing end-to-end traffic flows in DiffServ networks using the SNMP management framework. The IETF DiffServ working group has defined DiffServ MIB for managing DiffServ-enabled network devices. Based on this MIB, we have developed an SNMP agent system that operates in Linux-based DiffServ routers. A central DiffServ manager handles management functions on DiffServ routers with SNMP. The manager constructs end-to-end traffic flows for supporting various high-level management functions. Furthermore, a Web-based DiffServ management console that provides easy-to-use interfaces running in a Web browser is designed.

The rest of this paper is organized as follows. Section 2 explains the architecture of differentiated services proposed by IETF. Section 3 considers the management issues for DiffServ networks. Section 4 describes the detailed constructing methods and applications of end-to-end DiffServ flows and Section 5 shows how to develop a DiffServ management system. Finally, Section 6 summarizes our work and discusses directions for future research.

2 Architecture of DiffServ

DiffServ proposes a basic method to differentiate a set of traffic among network nodes. The method is based on a simple model where traffic entering a network is classified and possibly conditioned at the boundaries of the network, and assigned to different behavior aggregates. Each behavior is identified by a single Differentiated Services Code Point (DSCP).

DSCP is the most-significant 6 bits from the IPv4 Type-Of-Service (ToS) octet or IPv6 traffic class octet. This 6-bit field indicates how each router should treat the packet. This treatment is called a Per-Hop Behavior (PHB). PHB defines how an individual router will treat an individual packet when sending it over the next hop through the network. Being 6 bits long, the DSCP can have one of 64 different binary values.

Four types of PHBs have been defined as standard thus far [4, 6, 7, 8]. They are default, class-selector, Assured Forwarding (AF), and Expedited Forwarding (EF). Table 1 summarizes the standard PHBs and DSCP values accordingly.

Table 1. Standard PHBs

PHB Name	DSCP	Description
Default	000000	best-effort (RFC 1821)
Class-selector	xxx000	7 classes (RFC 2474)
AFxy	xxxyy0	4 classes with 3 drop probabilities (RFC 2597)
EF	101110	no drop (RFC 2598)

A DiffServ-enabled network node has several components for handling DiffServ. Fig. 1 explains five components of DiffServ architecture; classifier, meter, marker, shaper, and dropper [9, 10] in a traffic conditioning block (TCB).

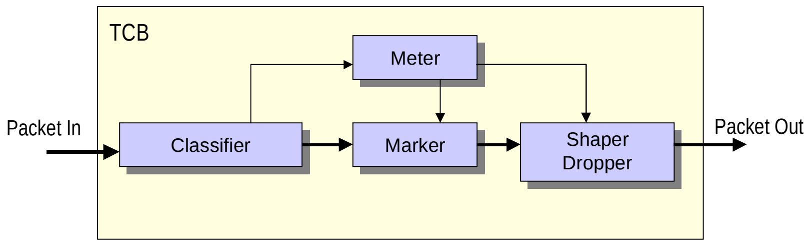

Fig. 1. Basic Traffic Conditioning Block of DiffServ

A classifier selects network packets in a traffic stream based on the content of some portion of the packet header. There are two types of classifiers, the Behavior Aggregate (BA) classifier based on the DiffServ values, and the Multi-Field (MF) classifier based on the value of a combination of 5-tuple information. A meter measures the temporal properties of the stream of packets selected by a classifier. It passes state information to other conditioning actions to trigger a particular action for each packet. A marker sets the DSCP of a packet and a shaper delays some or all of the packets in a traffic stream in order to bring the stream into compliance with a traffic profile. A dropper discards some or all of the packets in a traffic stream in order to bring the stream into compliance with a traffic profile.

DiffServ router is a fundamental DiffServ-enabled network node. The conceptual model and requirements of the DiffServ routers are discussed in IETF [11, 12]. The DiffServ router is considered to have routing component, set of TCBs, queuing component, and configuration and monitoring module that are organized as in Fig. 2.

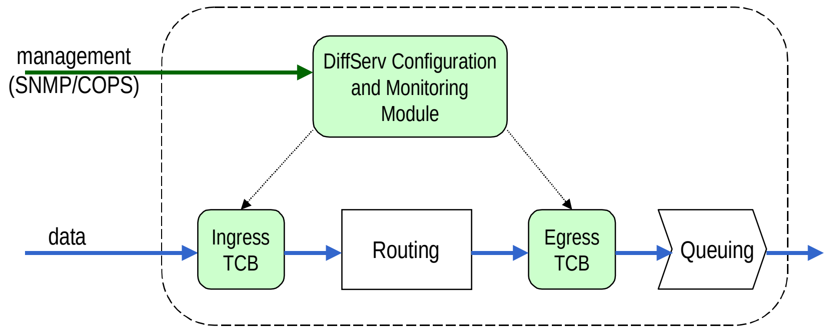

Fig. 2. Conceptual Model of a DiffServ Router

DiffServ-related components are separated from the routing component to simplify the addition of DiffServ capability to the existing router. There is a set of TCBs cascaded both at the ingress point and the egress point. Traffic conditioning can be performed either at the ingress point or at the egress point, or both. Queuing component is a set of underlying packet queues which keep packets before the routers send them out. The management module for DiffServ router can be operated in several ways such as SNMP or COPS [13, 14]. The management module configures TCB parameters and monitors the performance of each TCB. The detailed approach for managing DiffServ networks are explained in the next section.

3 Management of Differentiated Services

Managing DiffServ networks includes a set of various management functions. Current IETF approach for managing DiffServ networks is based on the SNMP framework. The SNMP framework is simple and a de-facto standard for managing Internet-related network devices. We investigate the structure of DiffServ MIB defined by the IETF and provide an overview of on-going efforts to define the MIB.

The IETF DiffServ working group currently suggests an SNMP Management Information Base (MIB) for the DiffServ architecture [15]. The MIB is designed according to the DiffServ implementation conceptual model [12] for managing DiffServ routers in the SNMP framework. The initial draft was proposed on July 1999, with the detailed definitions currently being elaborated and extended in the working group. Table 2 summarizes the primary object tables defined in the DiffServ MIB.

Table 2. DiffServ MIB Structure

Element	Table Name	Description
Classifier	Classifier	list of classifiers
	SixTupleClfr	5-tuple classifier + DSCP value
Meter	Meter	metering parameters
Action	Action	mark / count / absolute drop
Queue	AlgDrop	algorithmic dropper
	Queue	queuing parameters
	Scheduler	shaping parameters

The DiffServ table entries are linked each other with the RowPointer textual convention. RowPointer object is used for pointing an entry in the same or different table [16]. The DiffServ MIB represents a TCB as a series of table entries linked together by RowPointers. With this scheme many different TCBs can be represented in the object tables efficiently. Each table contains several MIB objects to configure, monitor, and modify DiffServ characteristics in a network node. By getting and setting these object values via SNMP, the SNMP manager can control DiffServ-enabled network nodes from a remote location.

However, the current DiffServ MIB is only for managing the characteristics of one DiffServ router. It does not provide a complete network picture of a set of DiffServ routers in one administrative domain. In order to provide such high-level management functions, the current management framework should be extended.

4 Constructing End-to-End DiffServ Flows

We defined a DiffServ flow as a sequence of network packets with the same DSCP value in a DiffServ domain. Every network service provided from a DiffServ network can be represented as a DiffServ flow from a set of source nodes to a set of destination nodes. Possessing information on such DiffServ flows can help understand the current service status. Information on the DiffServ flow consists of two parts: topology and performance. Topology information represents router-to-router connectivity. A path from a set of source edge routers to a set of destination edge routers must be provided. Performance information represents a number of performance parameters of a given DiffServ path. The performance information can be obtained by combining performance parameters of each router in a DiffServ path.

In this section, we suggest a method to create end-to-end DiffServ flows by combining routing information from MIB II and DiffServ performance parameters from DiffServ MIB. The end-to-end DiffServ flow information can be used as a basic component for providing sophisticated high-level management functions.

4.1 Method

A DiffServ flow consists of topology and performance information. Topology information is constructed from routing tables and performance information is constructed from DiffServ MIB values. Constructing end-to-end DiffServ flows thus consists of two phases, as in Fig. 3. First, the topology generator produces the topology information as a linked list of routers and the performance analyzer aggregates performance parameters of each router in the routing path by using the topology information. MIB II and DiffServ MIB are used to construct the information.

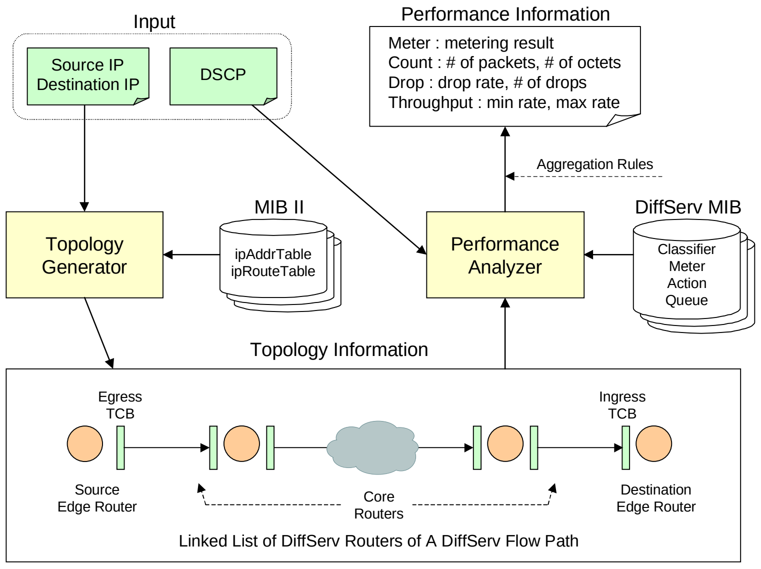

Fig. 3. Construction Process of DiffServ Flow Information

Since each DiffServ router supports routing protocols, the router keeps a routing table containing a list of next hop routers for a given destination IP address. The MIB II has the routing table and a central SNMP manager can retrieve the routing table information to construct a whole routing connectivity map in a DiffServ domain. Two MIB tables, ipAddrTable and ipRouteTable are used to create topology information. The ipAddrTable contains IP addresses of all network interfaces in a router and the ipRouteTable contains the IP routing table that has the next hop host and network interface for a set of destination IP addresses. By combining them we can obtain every source-to-destination routing path. Given a source-destination pair, the topology generator outputs a linked list of DiffServ routers composing a DiffServ flow path.

DiffServ flow performance information is obtained from the DiffServ MIB. Each DiffServ router has performance parameters observed locally. The parameters include metering parameters, counter values, numbers of dropped packets, minimum and maximum rates of packet transmission, and so on. These parameters are calculated and maintained for each DSCP value; that is, the DiffServ MIB of a DiffServ router contains all the performance parameters of DiffServ flows it processes. When a linked list of routers composing a DiffServ flow path is given, the performance analyzer aggregates values of the parameters from each DiffServ router one by one and produces end-to-end performance information of a DiffServ flow.

One important consideration in calculating end-to-end performance information is that the performance parameters contained in the DiffServ MIB in each router do not distinguish packets with different IP source/destination pair. Defined by the DiffServ concept, every core router forwarding packets between the source node to the destination node, only looks up the DSCP value in the header of each packet. Thus performance parameters from DiffServ MIB are for aggregated traffic with a given DSCP value, not for specific traffic flow from a given source to a given destination, which we want to analyze. The traffic flow that we want to distinguish is mixed with other flows with the same DSCP value but with different source/destination pairs.

From this observation, we make rules to follow when aggregating performance parameters. First, absolute values, such as counter values, should be translated to relative values. For example, number of dropped packets should be changed to rate of dropped packets so that the drop rate of a specific end-to-end DiffServ flow can be calculated by accumulating each drop rate in the router list. If there are three routers with 10% drop rates for a specific DSCP flow in the end-to-end routing path, the overall drop rates for the end-to-end DiffServ flow is calculated as 30%. Second, some parameters, such as throughput rates, should be calculated by finding out minimum or maximum values. For example, minimum throughput of an end-to-end DiffServ flow is calculated by finding out the minimum throughput in every router because the end-to-end throughput is bounded by the router with the least throughput.

4.2 Management of DiffServ Flows

By following the proposed method, we can obtain information of a set of end-to-end DiffServ flows in a DiffServ domain. Given a source/destination pair and a DSCP value, topology and performance information of a DiffServ flow from the source to the destination is constructed. Since the flow information gives a network view of DiffServ flows in a DiffServ domain to network administrators, various network management functions can be performed.

- Network topology management

Network topology can be created with the DiffServ flow information. Network connectivity and performance data should be kept in a management system in a certain format. The topology is not static. Numbers of DiffServ flows appear and disappear constantly. Managing the topology should follow such dynamic changes and show the current status.

- Bottleneck detection and rerouting

By analyzing the DiffServ flow information we can find out the location of the traffic bottleneck point. At the bottleneck point, the DiffServ flow cannot satisfy the required throughput. Drop rates go up and the metering result fails. The management system should resolve such occurrences. Rerouting of forwarding paths can be one solution. Routing tables can be modified for high-priority traffic to avoid the bottleneck points.

- Service Level Agreement (SLA) monitoring and reporting

Customers of the DiffServ network always want to know that the quality of service they utilize meets the SLA. Further, service providers want to monitor the service quality they provide to the customers. The service quality measurement turns out to be easy when we have DiffServ flow information. Performance parameters of DiffServ flows from a certain customers' network, which can be monitored and summarized to report the SLA satisfaction.

- Accounting and billing

When the DiffServ is deployed commercially in the Internet backbone, it is necessary for the Internet service providers to keep the usage record of their customers and request fees from them for the amount and quality of the Internet usage. DiffServ flow concepts can be applied to calculate the usage pattern and appropriate amount of fees.

These high-level management issues are under research currently. In the next section, we design a DiffServ management system as an initial framework for supporting the above functions.

5 Developing a DiffServ Management System

In this section, we present a detailed design and on-going implementation processes of a DiffServ management system based on the SNMP framework. The system is currently under development in Linux platforms.

5.1 Design Architecture

The architecture consists of three distinct layers, as depicted in Fig. 4. The three-tier architecture includes a network management system (NMS) client running in a Web browser, an NMS server containing a Web server and DiffServ manager, and network elements performing DiffServ routing and SNMP management.

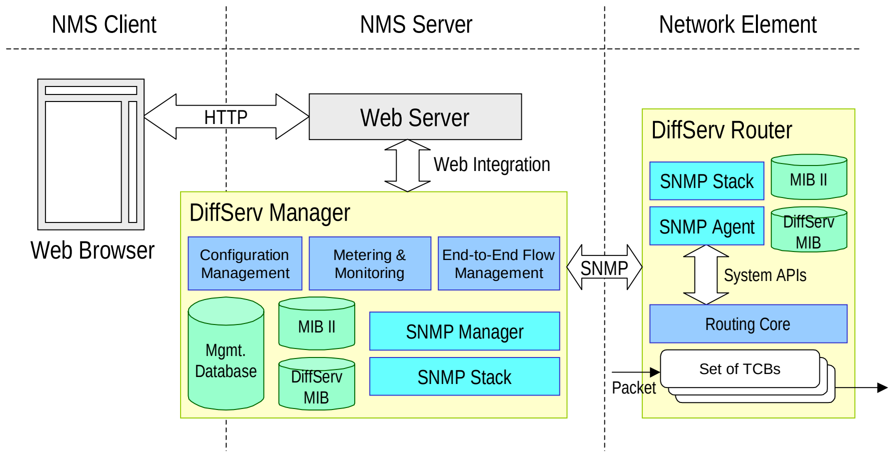

Fig. 4. Design Architecture of the DiffServ Management System

The NMS server is a central server for managing a set of DiffServ routers and providing management interfaces to a set of Web browsers. The Web server located in the NMS server layer has a role to provide a Web-based management interface in Web browsers. The integration of the Web server and the DiffServ manager can be accomplished in various ways such as a basic HTML file access method, a Common Gateway Interface (CGI) method, and a Java applet/servlet method.

The DiffServ manager performs three high-level DiffServ management functions, which are configuration management, metering and monitoring, and end-to-end flow management. The management database is used for storing and retrieving the combined and analyzed data from the MIB II and DiffServ MIB. At the bottom of the DiffServ manager, an SNMP manager communicates with a set of SNMP agents running in different DiffServ routers within a DS domain.

Three high-level DiffServ management functions perform sophisticated and extended management functions. Configuration management function performs remote configuration provisioning. Every DiffServ parameter is determined and enforced via the configuration management function. Metering and monitoring function periodically observes the status of DiffServ routers and compares the results with predefined desirable performance metrics. Such conformance test results are necessary for modifying behaviors of a DiffServ router. Flow management function summarizes all the DiffServ flows in a DS domain and provides the end-to-end DiffServ flow characteristics. The function collects routing tables and DiffServ flow information and constructs overall end-to-end parameters of each DiffServ flow.

DiffServ routers are managed network elements in the design architecture. A DiffServ router contains a routing core module to control a set of TCBs that execute packet forwarding according to various DSCP values, and an SNMP agent module to handle SNMP manager requests for the DiffServ MIB. System-dependent APIs are used to connect the SNMP agent module and the routing core module. The values of DiffServ MIB variables are determined by specific system-dependent system calls. The methods of retrieving and setting DiffServ parameters in the routing core module need not be the same among different implementation architectures.

Within a DiffServ domain, numerous DiffServ routers and DiffServ management clients interwork with each other. The three-tier architecture offers distinct advantages in such environments. One centralized DiffServ manager controls a set of DiffServ routers while providing management interfaces to a set of management clients at the same time. However, by separating the management user interfaces from the manager itself, the DiffServ manager is able to concentrate on management functions and thus the performance of the DiffServ manager can be improved.

5.2 Implementation

Linux, a shareware operating system, supports QoS features in its networking kernel from the kernel version 2.1.90 [17]. The QoS support offers a wide variety of traffic control functions, which can be combined in a modular way. Based on this Linux traffic control framework, W. Almesberger et al. have designed and implemented basic DiffServ classification and manipulation functions required by DiffServ network nodes [18]. The extended DiffServ features are freely available in the form of a kernel patch package [19]. By installing the DiffServ package, a Linux system is able to perform DiffServ router functions.

However, the current Linux DiffServ implementation does not show sufficient management functionality. There is no management architecture and every script setup must be manually configured and modified in local machines. Further, metering and monitoring functions of DiffServ are not fully supported. Our work focuses on this lack of management functionality.

A DiffServ agent is an SNMP agent with MIB II and DiffServ MIB running on the Linux DiffServ router. Basically the agent extracts DiffServ parameters from the Linux traffic control kernel and modifies the appropriate MIB values on the request from a DiffServ manager. The agent also receives management operations from a DiffServ manager and performs the appropriate parameter changes in the Linux traffic control kernel.

The organization of our Linux DiffServ router implementation is explained in Fig. 5. There are two process spaces in the Linux operating system, the user space and the kernel space. Extending from Linux traffic control framework, the Linux DiffServ implementation resides in the kernel space. In the user space, the DiffServ SNMP agent is implemented. Communication between the DiffServ agent and the Linux traffic control kernel is effected via NetLink sockets [20]. The NetLink socket is a socket-type bidirectional communication link located between kernel space and user space. It transfers information between them.

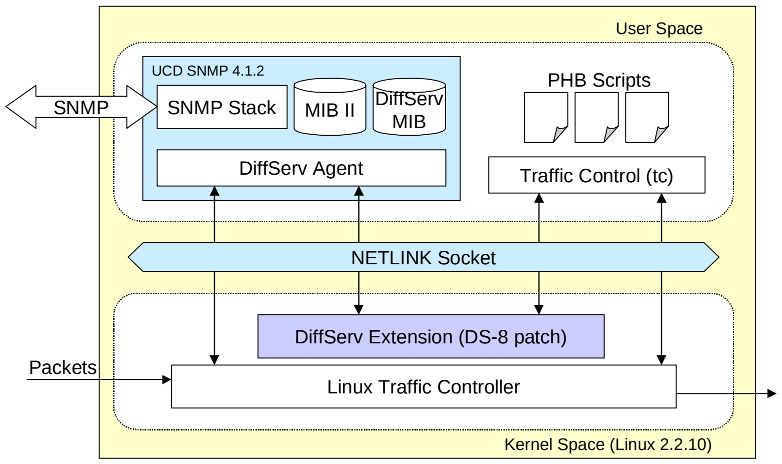

Fig. 5. Organization of Linux DiffServ Router Implementation

The agent has been implemented by using UCD SNMP agent extension package [21]. UCD SNMP 4.1.2 provides the agent development environment. The DiffServ agent uses the traffic control program (tc) or NetLink socket directly for accessing DiffServ parameters in kernel space and manipulates the values of MIB II and DiffServ MIB.

A Web-based DiffServ management system is currently under development in our work. Java programming language is chosen as our development environment because Java applets can be executed in Web browsers very conveniently.

The central DiffServ manager integrated with a Web server is also being developed in a Linux system. It can configure, monitor, and report the characteristics of DiffServ routers and DiffServ networks. A set of DiffServ flow information is constructed by following the method in Section 4 and stored in a PostgreSQL database of version 7.0.2 [22]. For human managers responsible for a DiffServ network, network topology management function and bottleneck detection and rerouting function are in a prototyping stage.

6 Conclusion and Future Work

Differentiated Services (DiffServ) is gaining acceptance as a promising solution for providing QoS support in the Internet. This paper has proposed a method to manage DiffServ using the SNMP framework. Since current research efforts from the IETF DiffServ working group focus mainly on the operational and functional descriptions of DiffServ, a detailed management framework for DiffServ is urgently needed. We have overviewed management concepts for DiffServ by categorizing management operations in the layered architecture and then presented on-going work to define MIB for managing DiffServ-enabled network nodes in the IETF working group.

To overcome current management functional limits and extend the management capability to sophisticated high-level functions, we have suggested a method to construct and maintain end-to-end DiffServ flows by combining MIB II and DiffServ MIB, and showed the applicability of DiffServ flow information. And then we have proposed a DiffServ management system with a flexible three-tier architecture using the SNMP framework. Further, we have developed a DiffServ agent system working in a Linux platform and a Web-based manager system. Management interfaces running in a Web browser enable users to control DiffServ routers conveniently.

In order to improve the proposed DiffServ management system, we are currently working on the following topics.

A systematic method for representing the proposed DiffServ flow information is needed. The proposed construction process must be extended to produce a formal and graphical description of the DiffServ flows. Standardized data formats and graphical representations such as a directed graph with different shapes of vertex are currently being developed.

Scalability of the proposed system should be improved. Current management framework needs constant polling to every router in the management domain. This might not be appropriate, especially in large ISP backbones. To address the scalability problem, the three-tier architecture can be extended to support distributed management functionality with multiple DiffServ managers located in the middle layer. Also instead of polling the routing table, the agent can initiate sending routing change notification to managers by using the SNMP trap method.

Integration with a policy framework is highly recommended. To simplify the system, we have excluded policy management features in this paper, but such a policy framework needs to be integrated with the current SNMP framework for flexible and intelligent configuration and adaptation of DiffServ routers. Future work includes studying the meta-information model for policy representation and designing policy operational modules.

Finally, performance evaluation of the management system we are developing is considered. Because general DiffServ routers handle a huge amount of high-speed traffic, the DiffServ agent must not affect the routing performance of the DiffServ routers. A DiffServ management system needs to be implemented in such a way as to minimize performance degradation.

References

1. R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," IETF RFC 1633, June 1994.
2. R. Braden et al., "ReSerVation Protocol (RSVP) Version 1 Functional Specification," IETF RFC 2205, September 1997.
3. R. Rajan et al., "A Policy Framework for Integrated and Differentiated Services in the Internet," IEEE Network, September/October 1999, pp.36-41.
4. J. Heinanen, "Use of IPv4 TOS Octet to Support Differential Services," IETF Internet-Draft, draft-heinanen-diff-tos-octet-01.txt, November 1997.
5. B. Carpenter and D. Kandlur, "Diversifying Internet Delivery," IEEE Spectrum, Vol. 36, No. 11, November 1999, pp.57-61.
6. K. Nichols et al., "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," IETF RFC 2474, December 1998.
7. J. Heinanen et al., "Assured Forwarding PHB Group," IETF RFC 2597, June 1999.
8. V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB," IETF RFC 2598, June 1999.
9. S. Blake et al., "An Architecture for Differentiated Services," IETF RFC 2475, December 1998.
10. Y. Bernet et al., "A Framework for Differentiated Services," IETF Internet-Draft, draft-ietf-diffserv-framework-02.txt, February 1999.
11. Y. Bernet et al., "Requirements of Diff-serv Boundary Routers," IETF Internet-Draft, draft-bernet-diffedge-01.txt, November 1998.
12. Y. Bernet, A. Smith, S. Blake, and D. Grossman, "A Conceptual Model for Diffserv Routers," IETF Internet-Draft, draft-ietf-diffserv-model-03.txt, May 2000.
13. J. Boyle et al., "The COPS (Common Open Policy Service) Protocol," IETF Internet-Draft, draft-ietf-cops-07.txt, August 1999.
14. R. Yavatkar et al., "COPS Usage for Differentiated Services," IETF Internet-Draft, draft-ietf-rap-cops-pr-00.txt, December 1998.
15. F. Baker, K. H. Chan, and A. Smith, "Management Information Base for Differentiated Services Architecture," IETF Internet-Draft, draft-ietf-diffserv-mib-03.txt, May 2000.
16. W. Stalling, SNMP, SNMPv2, SNMPv3, and RMON 1, 2, 3rd Edition, Addison-Wesley, 1999.
17. S. Radhakrishnan, "Linux – Advanced Networking Overview – Version 1," a technical paper of Department of Electrical Engineering and Computer Science, University of Kansas, August 22, 1999.
18. W. Almesberger, J. H. Salim, and A. Kuznetsov, "Differentiated Services on Linux," IETF Internet-Draft, draft-almesberger-wajhak-diffserv-linux-01.txt, June 1999.
19. W. Almesberger, Differentiated Services on Linux, Internet Web site, http://lrcwww.epfl.ch/linux-diffserv/.
20. ITU-T Recommendation M.3010, "Principles for a Telecommunications Management Network," 1996.
21. UCD-SNMP homepage, http://ucd-snmp.ucdavis.edu/.
22. PostgreSQL homepage, http://www.postgresql.org/.