ICACT ’2000 Paper Submission
Contact Information

Title :

Management of Differentiated Services Using the SNMP Framework

Name :

Jae-Young Kim

Mailing Address :

Distributed Processing & Network Management Lab.
Dept. of Computer Science and Engineering
Pohang University of Science and Technology
San 31, Hyoja-dong, Nam-gu
Pohang, Korea
790-784

E-mail Address :

jay@postech.ac.kr

Telephone :

+82-562-279-5659

Fax :

+82-562-279-5699

Choice of Topics :

Network Management, Operation and Maintenance

Management of Differentiated Services Using the SNMP Framework

Jae-Young Kim, Myung-Sup Kim, Won-Ki Hong
Department of Computer Science Engineering
Pohang University of Science and Technology
{jay, mount, jwkhong}@postech.ac.kr

Tae-Sang Choi, Yoon-Hee Jung, and Sung-Won Sohn
Electronics Telecommunications Research Institute
{choits, yhjung, swsohn}@etri.re.kr

Abstract — **QoS support for the Internet traffic becomes an important future requirement of various network devices such as routers and switches. Differentiated Services (DiffServ) is considered as an efficient feasible solution for providing different service characteristics to different classes of network users. IETF DiffServ working group has already defined a general architecture of DiffServ and is extending its detail features. Network vendors also start implementing DiffServ-enabled network devices. However, the management of DiffServ is not fully standardized yet. In this paper, we present a management architecture of DiffServ within the SNMP framework. IETF DiffServ MIB has been analyzed and we have designed a DiffServ management system. In order to realize and validate our design, DiffServ routers have been constructed in Linux systems and we have implemented a DiffServ agent on each DiffServ routers. Management functions are operated in a Web-based manager that provides user-friendly, easy-to-use management interfaces.**

Keywords — **Differentiated Services, SNMP Agent, Internet Management, Web-based Management**

1. Introduction

Within the past decade, network environments become explosively widespread all over the world. As the Internet and the WWW expands its boundaries, the network traffic caused by data transfers over the Internet has also increased. The previous bandwidth, which was enough to carry text-based application data, is not sufficient anymore for the current multimedia and real-time network traffic flows.

As the increase rate of network bandwidth is quite slower than the increase rate of network usage, there are always bottleneck points where bandwidth is not enough for network users. In such situations every packet compete with each other to get the bandwidth and this results in packet loss, unexpected delay and jitter. However, Transmission Control Protocol (TCP) and Internet Protocol (IP) were originally designed in best-effort service model. They handle all packets with the same priority, first-come and first-served.

But the users' requirement is changing. They want to get different service qualities for different types of services they obtain. Integrated Services (IS) with Resource reServation Protocol (RSVP) signaling is the first approaches to realize providing such services in the Internet [22, 26]. RSVP/IS 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 is relied on per-flow state and per-flow processing in every network nodes, it is very 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 points. Type of Service (ToS) field is used for distinguishing these traffic aggregates. Since the ToS is much simpler than 5-tuple information, DiffServ implementation is easier than RSVP/IS and also more feasible solution [3, 9, 28]. DiffServ use administrative domain concepts. Within one domain, core routers forward traffics according to the ToS field of the 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 on the incoming traffics, core routers do not need to handle complex information on traffics. The basic architecture of DiffServ is further explained in Section 2.

Though IETF DiffServ working group has defined several standards in DiffServ, management of DiffServ is not fully standardized yet. Current standards have defined only the operational aspects of DiffServ. When deploying DiffServ in network nodes various management functions are needed to control a large number of DiffServ nodes remotely. From the management point of view, network element management, network management, and service management need to be defined.

In this paper, we have constructed an SNMP management framework for managing DiffServ. 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. Also we have developed a Web-based DiffServ manager that provides easy-to-use interfaces running in any Web browser.

The rest of this paper is organized as follows. Section 2 explains the architecture of differentiated services proposed

by IETF. Section 3 presents the design of a DiffServ management system. Section 4 describes the detailed implementation on Linux systems and Section 5 summarizes our work and discusses possible future work.

2. Architecture of DiffServ

IETF DiffServ working group has defined the basic architecture of DiffServ. In this section, we summarize current standards and investigate several open issues.

2.1. General Concepts of DiffServ

DiffServ proposes a basic method to differentiate set of traffics 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 overall types of PHBs have been defined as standards so far [9, 10, 13, 14]. They are default, class-selector, Assured Forwarding (AF), and Expedited Forwarding (EF). Table 1 summarizes the standard PHBs and DSCP values.

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)

DiffServ-enabled network nodes handle classes of network packets differently by using the DSCP values in the packet header within an administrative domain. DS domain is a contiguous set of DiffServ-enabled nodes which operate with a common service provisioning policy and set of PHB groups implemented in each node. There are two kinds of DiffServ nodes in a DS domain. Edge routers are located at the boundary of the DS domain and classify and possibly condition ingress traffic to ensure that packets which transit the domain are appropriately marked to select a PHB from one of the PHB groups supported within the domain. Internal routers, called core routers, only check the DSCP values of forwarded packets and perform predefined PHB actions on the packets. It requires no per-flow state in backbone and trunk routers. This internal behavior saves times and resources for providing different service qualities to different classes of users.

2.2. Components

A DiffServ-enabled network node needs to have several components for handling DiffServ [11, 12]. Figure 1 explains five components of DiffServ architecture; classifier, meter, marker, shaper, and dropper.

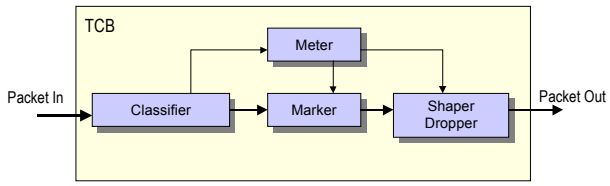

Figure 1 : Basic Traffic Control Block of DiffServ

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, Behavior Aggregate (BA) classifier based on the DiffServ values and 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.

2.3. Current Open Issues

There still remain a lot of things to be done in DiffServ architecture [3, 28]. The following issues are currently researched.

- Network management

The current DiffServ architecture concerns only how to operate with the DSCP and PHB in each network node. There are not enough so-called FCAPS management issues in standards. Recently IETF DiffServ working group proposes an SNMP MIB definitions for DiffServ network nodes.

- End-to-end flow management

DiffServ network nodes handles traffic without any interaction with neighbor nodes. Though this simplifies functions of a network node, no interaction sometimes makes it difficult to understand the end-to-end flow behaviors. In order to control the end-to-end traffic characteristics in one domain, a kind of overall control function is needed.

- Integration with the policy framework

Common Open Policy Service (COPS) protocol in the IETF policy framework has been proposed for flexible and configurable control of network nodes [23, 24]. Integrating the policy framework in Integrated Services and DiffServ is being researched recently.

In this paper we focus on adding network management functionality in the current DiffServ architecture. We have designed a DiffServ management system in SNMP management framework.

3. Design of a DiffServ Management System

In this section, we present the detailed design processes of a DiffServ management system in the SNMP framework. First the IETF DiffServ MIB is analyzed from the operational viewpoint. And then the requirements of a DiffServ management system are listed, and finally, the design architecture of our system is described.

3.1. DiffServ MIB

IETF DiffServ working group currently suggests an SNMP Management Information Base (MIB) for the DiffServ architecture [5]. The MIB is designed following the DiffServ implementation conceptual model [20]. Table 2 summarizes six object tables defined in the DiffServ MIB.

Table 2 : DiffServ MIB Structure

Table Name	# of Objects	Description
Aggregate	1	DSCP value
MFClassifier	13	multi-field classification value
Classifier	8	classifier list
TBMeter	6	token-bucket meter value
Action	15	mark / count / drop
Queue	7	priority queue

Each table contains several MIB objects that configure, monitor, and modify DiffServ characteristics in a network node. By controlling these object values via SNMP, SNMP managers can manage DiffServ-enabled network nodes.

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 [2]. A TCB consists of a number of different classifiers, meters, actions, and queues. The DiffServ MIB represents a TCB as a series of table entries linked by RowPointers. With this scheme a lot of different TCBs can be represented in the six tables efficiently. For example, several TCBs have the same BA and MF classifiers, the same actions or meters on the same queues. Thus the same parameters need not to be defined separately. The current MIB design provides parameter sharing for different TCBs. If the DiffServ MIB defines a TCB table contains entries for all the TCBs, the size of table would be much more than the sum of the current six table entries. Figure 2 shows the linked relationship of six tables in the DiffServ MIB.

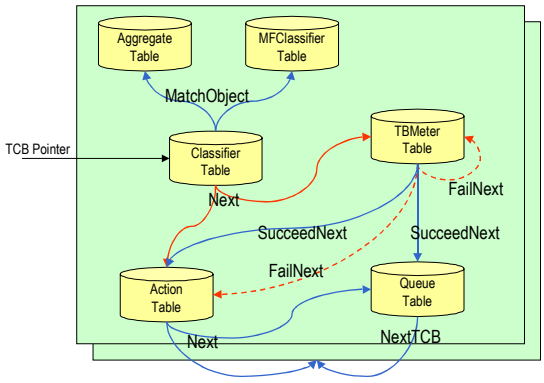

Figure 2 : MIB Table Relationship

All the TCB pointers in a DiffServ network node are contained in the classifier table. From this table entry the TCB configuration linked list starts. Each entry in the classifier table has two RowPointer objects, MatchObject and Next. MatchObject points either an Aggregate table entry or an MFClassifier table entry and Next points either a TBMeter table entry or an Action table entry. TBMeter table entry is used for determining the conformance of packet flows and the next TCB component is chosen on the fact whether the current packet flow succeeds in or fails for the meter parameters. When succeeded the packet flow is sent to the appropriate Action or Queue table entry, and when failed the packet flow is sent to another Meter table entry or the appropriate Action table entry. Action and Queue table entry has a pointer to the next TCB classifier and this closes a TCB configuration.

3.2. Requirements of a DiffServ Management System

For developing a DiffServ management system, we have analyzed requirements of a DiffServ management system. The followings are selected requirements.

- Management in the SNMP framework

SNMP is considered as a standard management framework for Internet management. In order to simplify the system, we have chosen the SNMP framework for our management system. The IETF DiffServ MIB is the key management information in the SNMP framework.

- Web-based management interfaces

We have adopted Web-based management interfaces for our management system. Since Web browsers are ubiquitous, easy-to-use, and platform-independent, the management interfaces running in Web browsers provide cost-effective, portable, and convenient interfaces to human managers.

- Remote configuration provisioning

When DiffServ is deployed in an organization, there might be lots of DiffServ-enabled network nodes working together in the organization's backbone network. A DiffServ management system is able to configure numbers of DiffServ nodes remotely.

- Metering and status monitoring

Network traffic changes its characteristics dynamically. In order to make DiffServ nodes to be cope with the dynamic traffic variations, a DiffServ management system needs to have realtime metering and status monitoring functionality. Periodic SNMP polling and history keeping should be performed in a DiffServ manager. Further, graphical representation of status changes is also needed. Based on the results, a DiffServ management system decides to modify configurations of a set of DiffServ nodes.

- End-to-end flow management

A DiffServ flow is defined as a traffic flow with the same DSCP value in a DS domain. There is a set of different DiffServ flows coming in and out in a DS domain. However, as descibed in the previous section, DiffServ nodes do not interact with each other to keep end-to-end statistics of each DiffServ flow. From the management point of view, the end-to-end characteristics of DiffServ flows are important because the network service quality is

determined by the end-to-end performance parameters, not by the individual performance paramters in each DiffServ node. Therefore, a DiffServ management system is able to summarize performance parameters of each DiffServ node and report the end-to-end DiffServ flow characteristics.

3.3. Design Architecture

We have designed a DiffServ management system satisfying the previous requirements. The architecture consists of three distinct layers as depicted in Figure 5. The three-tier architecture includes a network management system client running in a Web browser, a network management system server containing a Web server and DiffServ manager, and a network element performing DiffServ routing and SNMP management.

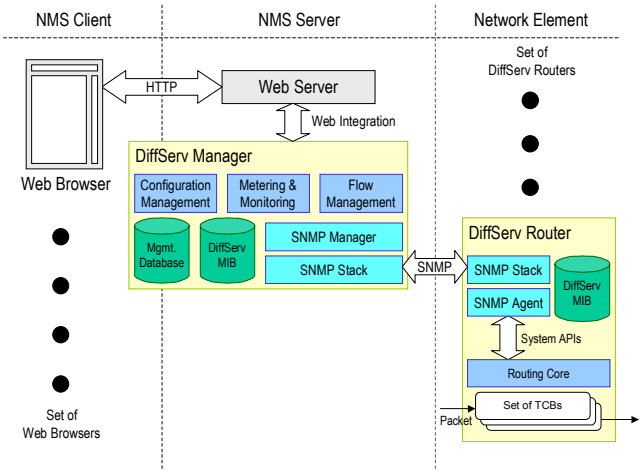

Figure 3 : Design Architecture of the DiffServ Management System

The NMS server is the 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 interfaces in Web browsers. The integration with Web server and DiffServ manager can be achieved in various ways such as a basic HTML file access method, a Common Gateway Interface (CGI) method, and a Java application method.

The DiffServ manager performs three high-level DiffServ management functions – configuration management, metering and monitoring, and flow management. The management database for storing and retrieving additional information other than the DiffServ MIB is needed in order to provide high-level DiffServ management functions. At the bottom of the DiffServ manager, an SNMP communicates with a set of SNMP agents running in different DiffServ routers within a DS domain.

Three high-level DiffServ management functions performs sophisticated and extended management functions for providing value-added management operations to users. Configuration management function performes remote configuration provisioning. Every DiffServ paramteter 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 peformance metrics. Such conformance test results are necessary for modifying DiffServ router behaviors in the configuration management function. The flow management function summarizes all the DiffServ flows in a DS domain and provides the end-to-end DiffServ flow characteristics. The function collects DiffServ flow information from every DiffServ router and constructs an overall end-to-end parameters of each DiffServ flow. The parameters are useful for understanding quality of network services.

The DiffServ routers are managed network elements in the design architecture. A DiffServ router contains a routing core module for controlling a set of TCBs that execute packet forwarding according to various DSCP values and an SNMP agent module for handling SNMP manager requests for the DiffServ MIB. System-dependent APIs are used for connecting 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 DS domain there might be lots of both DiffServ routers and DiffServ management clients interworking with each other. The three-tier architecture has advantages in such network management 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 the management functions and thus the performance of the DiffServ manager can be improved. To address the scalability problem, the three-tier architecture can be easily extended to support distributed management functionality with multiple DiffServ managers located in the middle layer. Obviously, there must be a concrete way of combining the multiple DiffServ managers in this case.

4. Implementation

Based on the previous design architecture, we have implemented a DiffServ management system. DiffServ routers have been constructed in Linux systems and we have added an SNMP agent for the DiffServ routers. A web-based DiffServ manager has been also developed for delivering various management services to human users very conveniently.

A DiffServ agent is running on a Linux DiffServ router. The agent has DiffServ MIB values that are extracted from the Linux traffic controller via NetLink sockets. A DiffServ manager is integrated with Web technologies. The manager interacts with several agents via SNMP and interacts with human managers via HTTP. Human managers can operate every DiffServ routers in their control within easy-to-use Web browsers. Implementation details of the manager and the agent are described in the following subsections.

4.1. DiffServ in Linux Systems

Linux, a shareware operating system, supports QoS

features in its networking kernel from the kernel version 2.1.90. The QoS support offers a wide variety of traffic control functions, which can be combined in a modular way. The traffic control code in the Linux kernel consists of four conceptual components (queuing discipline, class, filter, and policing) and configured logically as in Figure 4 [7, 8].

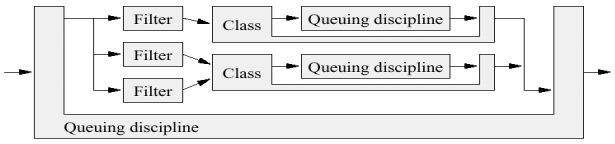

Figure 4 : Linux Traffic Control Blocks

Each network device has a queuing discipline associated with it, which controls how packets enqueued on that device are treated. Queuing disciplines may use filters to distinguish among different classes of packets and process each class in a specific way. Policing, resides within a filter, performs the conformance test on classes of packets and appropriate actions on each class according to the test results.

Based on this Linux traffic control framework, W. Almesberger et al. have designed and implemented basic DiffServ-field classification and manipulation functions required by DiffServ network nodes [8]. The extended DiffServ features are freely available in the form of a kernel patch. The current DiffServ package (DS-6) available from DiffServ on Linux web site [28] contains the kernel patch, a control program (tc), and several PHB scripts constructed with the control program. By installing the DiffServ package, a Linux system is able to perform DiffServ router functions.

However, the current Linux DiffServ implementation does not have enough management functionality. There is no manager-agent architecture and every script setup should be manually configured and modified in local machines. Also metering and monitoring functions of DiffServ are not fully supported. Our work has been done on this lack of management functionality.

4.2. Linux DiffServ Agent

The DiffServ agent is an SNMP agent with 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 the Linux DiffServ router implementation is explained in Figure 5. There are two process spaces in the Linux operating system, user space and kernel space. The Linux DiffServ implementation extended from the Linux traffic control framework resides in kernel space. In user space, the DiffServ SNMP agent is implemented combining with the Linux traffic control program. Communications between the DiffServ agent and the Linux traffic control kernel are transferred via NetLink sockets. The NetLink socket is a socket-type bidirectional communication link located between kernel space and user space. It transfers information between them.

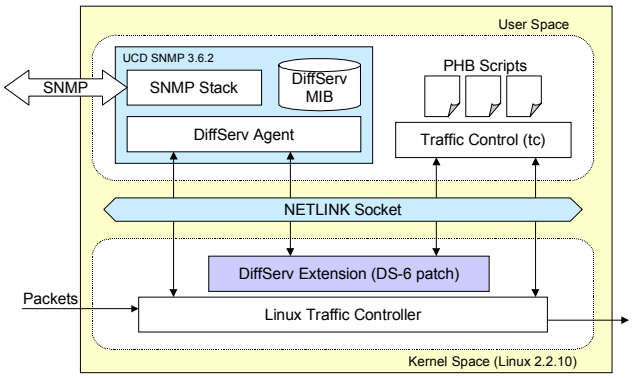

Figure 5 : Linux DiffServ Router Implementation

The agent has been implemented by using UCD SNMP agent extension package [6]. UCD SNMP 3.6.2 provided 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 DiffServ MIB.

4.3. Web-Based DiffServ Manager

We have developed a DiffServ manager as an NMS server and a Web-based GUI application as an NMS client. The integration of Web technology within network management systems has benefits of ubiquitous, easy-to-use, and user-friendly interfaces. A Web server has been integrated in the NMS server as in Figure 5. The manager functions consist of the following subfunctions.

- Agent list management
- Agent summary reporting
- MIB browser
- Flow browser
- Realtime statistics reporting

Agent list management keeps the record of DiffServ agents managed by a DiffServ manager. The manager is able to add, modify, or delete an entry for a DiffServ agent. Agent summary reporting provides an overall agent status to users. The reporting contains an IP address, a system name, and installed network interfaces of a DiffServ router. Also the summary report links to MIB and flow browsers. The MIB browser is a basic MIB value retriever. When a manger wants to look up specific MIB object values, this MIB browser can be used efficiently.

Flow browsers enable a user to easily understand the flows a DiffServ router handles. Since DiffServ MIB tables are linked together as in Figure 2, it is difficult to see a specific traffic flow in the MIB tables. The flow browser makes it easy by extracting MIB values according to flow concepts. Every different flow is listed in the flow browser and users can also see the real-time statistics of flows.

Realtime statistics reporting generates realtime graphs of statistical MIB variables. For every DiffServ flow in every DiffServ router, the Action MIB table contains the number of conforming packets, the number of conforming octets, and the number of dropped packets. The DiffServ manager periodically retrieves the values from every DiffServ router and stores them as history records. When the DiffServ

clients request to view the data, the DiffServ manager generates a statistical graph for each MIB variable.

The Web-based user interfaces providing the DiffServ manager functions are showed in from Figure 6 and Figure 7.

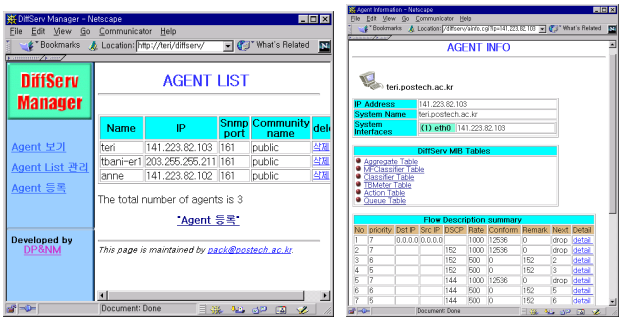

Figure 6 : Agent List and Agent Summary Report Window

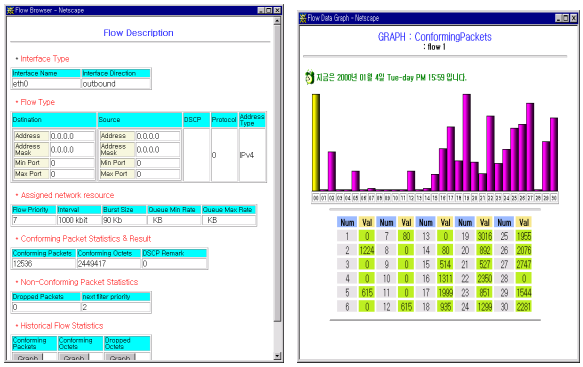

Figure 7 : Flow Browser and Realtime Statistic Graph

5. Conclusion and Future Work

Differentiated Services (DiffServ) is a promising solution for providing QoS support in the Internet. Although the operational architecture has been standardized from IETF, the management architecture is not yet fully standardized and needs to be refined and extended. In this paper we have proposed a way of managing DiffServ in the SNMP framework. We have designed and developed a DiffServ agent system working in Linux platform and a Web-based manager system which manages DiffServ agent systems. Management interfaces running in any Web browser enable users to control DiffServ routers conveniently.

In order to improve the functionality of our system, we are working on the following topics [4].

Performance evaluation of our management system needs to be considered. Because general DiffServ routers handle a huge amount of high-speed traffics, the DiffServ agent must not affect the routing performance of the DiffServ routers. A DiffServ management system needs to be implemented to minimize performance degradation factors.

Integrating with the policy framework is highly recommended. To simplify the system, we have excluded the policy management features in this paper, however, the 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 meta-information model for policy representation and designing policy operational modules.

And finally, there are already several proprietary implementations of DiffServ routers from hardware vendors. We have plans to integrate our system with commercial products so that our system will be a feasible solution for managing DiffServ in the next Internet structure.

References

[1] D. Perkins and E. McGinnis, Understanding SNMP MIBs, Prentice-Hall, 1997.

[2] W. Stalling, SNMP, SNMPv2, SNMPv3, and RMON 1 and 2, 3rd Edition, Addison-Wesley, 1999.

[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] T. Choi, Y. Jung, and S. Sohn, "Design and Implementation of Seamless End-to-End Internet QoS Management System," paper submitted to NOMS'2000, May 2000.

[5] F. Baker, K. H. Chan, and A. Smith, "Management Information Base for Differentiated Services Architecture," IETF Internet-Draft, draft-ietf-diffserv-mib-01.txt, October 1999.

[6] UCD-SNMP Homepage, http://ucd-snmp.ucdavis.edu/.

[7] W. Almesberger, "Linux Network Traffic Control – Implementation Overview," ftp://lrcftp.epfl.ch/pub/people/almesber/pub/tcio-current.ps, April 1999.

[8] W. Almesberger, J. H. Salim, and A. Kuznetsov, "Differentiated Services on Linux," IETF Internet-Draft, draft-almesberger-wajhak-diffserv-linux-01.txt, June 1999.

[9] J. Heinanen, "Use of IPv4 TOS Octet to Support Differential Services," IETF Internet-Draft, draft-heinanen-diff-tos-octet-01.txt, November 1997.

[10] K. Nochols et al., "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers," IETF RFC 2474, December 1998.

[11] S. Blake et al., "An Architecture for Differentiated Services," IETF RFC 2475, December 1998.

[12] Y. Bernet et al., "A Framework for Differentiated Services," IETF Internet-Draft, draft-ietf-diffserv-framework-02.txt, February 1999.

[13] J. Heinanen et al., "Assured Forwarding PHB Group," IETF RFC 2597, June 1999.

[14] V. Jacobson, K. Nichols, and K. Poduri, "An Expedited Forwarding PHB," IETF RFC 2598, June 1999.

[15] K. Nichols, V. Jacobson, and L. Zhang, "A Two-bit Differentiated Services Architecture for the Internet," IETF RFC 2638, July 1999.

[16] Y. Bernet et al., "Requirements of Diff-serv Boundary Routers," IETF Internet-Draft, draft-bernet-diffedge-01.txt, November 1998.

[17] M. Daniele et al., "Internet Endpoint MIB," IETF Internet-Draft, draft-ops-endpoint-mib-02.txt, November 1999.

[18] S. Brim, B. Carpenter, and F. Le Faucheur, "Per Hop Behavior Identification Codes," IETF Internet-Draft, draft-ietf-diffserv-phbid-00.txt, October 1999.

[19] IETF DiffServ Working Group Homepage, http://www.ietf.org/html.charters/diffserv-charter.html.

[20] Y. Bernet, A. Smith, and S. Blake, "A Conceptual Model for Diffserv Routers," IETF Internet-Draft, draft-ietf-diffserv-model-01.txt, October 1999.

[21] D. Grossman, "New Terminology for Diffserv," IETF Internet-Draft, draft-ietf-diffserv-new-terms-00.txt, October 1999.

[22] R. Braden et al., "ReSerVation Protocol (RSVP) Version 1 Functional Specification," IETF RFC 2205, September 1997.

[23] J. Boyle et al., "The COPS (Common Open Policy Service) Protocol," IETF Internet-Draft, draft-ietf-cops-07.txt, August 1999.

[24] R. Yavatkar et al., "COPS Usage for Differentiated Services," IETF Internet-Draft, draft-ietf-rap-cops-pr-00.txt, December 1998.

[25] R. Rajan et al., "Schema for Differentiated Services and Integrated Services in Networks," IETF Internet-Draft, draft-rajan-policy-qosschema-01.txt, April 1999.

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

[27] W. Almesberger, Differentiated Services on Linux, Internet Web site, http://lrcwww.epfl.ch/linux-diffserv/.

[28] B. Carpenter and D. Kandlur, "Diversifying Internet Delivery," IEEE Spectrum, Vol. 36, No. 11, November 1999, pp.57-61.