Policy based SLA Management in Enterprise Networks

Dinesh Verma, Mandis Beigi and Raymond Jennings

IBM Thomas J Watson Research Center
PO Box 704
Yorktown Heights, NY 10598, USA
{dverma,raymondj,mandis}@us.ibm.com

Abstract. The Differentiated Services Architecture defines the mechanisms that are needed to offer multiple classes of services in an IP network. While this model offers significant scaling advantages over the signaling-oriented approach of Integrated Services, the management of a differentiated services network remains a complex problem. Since the operation of a differentiated network involves numerous access routers, core routers and servers, a consistent operation is difficult to achieve by independently configuring each device. In this paper, we explore a scheme to enable a network administrator to manage and configure DiffServ networks from a central location and also abstract away the specific details of device configuration, and allow him/her to express the management of the network in terms of application-oriented performance metrics. This leads to a simplification of network management task, which can be exploited to support business needs of an enterprise network, such as honoring Service Level Agreements provided to its customers.

1. Introduction

The IP networks of today are transitioning from a best-effort service model to one that can provide different service levels in the network. The interest and activities of the technical community in the area has led to the development of two standards for Quality of Service (QoS) -- the RSVP signaling (IntServ/RSVP) approach [1], which was followed by the differentiated services (DiffServ) approach [2]. Integrated services with RSVP signaling approach attempts to provide per-flow QoS assurances with dynamic resource reservation, where a flow corresponds to a transport session (such as a TCP connection or UDP stream to a multicast address group). In order to avoid the scaling issues associated with the per-flow state needed in the RSVP approach, the DiffServ approach attempts to provide different service levels to traffic aggregates, and the service level of each packet is identified by means of a 6-bit field in the IP header. Each distinct value of the field identifies the processing to be done for a specific level, which are called Per Hop Behavior (PHB) in DiffServ terminology.

The DiffServ standards can be deployed by an Internet Service Provider (ISP) to offer different service levels to its customers, or by an enterprise network operator to offer different classes to the various applications running over an enterprise intranet. An ISP has to resolve the complex issues associated with pricing of different service levels. In an enterprise environment, the pricing issue is relatively less important, and classes of services are more likely to be defined in terms of the perceived importance a traffic to the business needs of the enterprise. In both types of deployment, the tricky issue arises in the management of the differentiated services network.

A sample deployment of the differentiated services in an enterprise network is illustrated in Figure 1. We assume that the wide area network (WAN) is bandwidth-constrained, and consists of routers that support DiffServ. The WAN is connected to two LANs, one hosting several clients, and the other hosting some servers. The access routers examine the 5-tuples (source and destination IP address, source and destination port numbers and the protocol) in a packet headed towards the WAN and mark then with a specific value of the DS field. The core routers look at the different markings and process the packets accordingly. Marking can also be done by the servers in the network. In this case, the access router at the server site needs to examine only the DS field of the packets entering the WAN and optionally changing it.

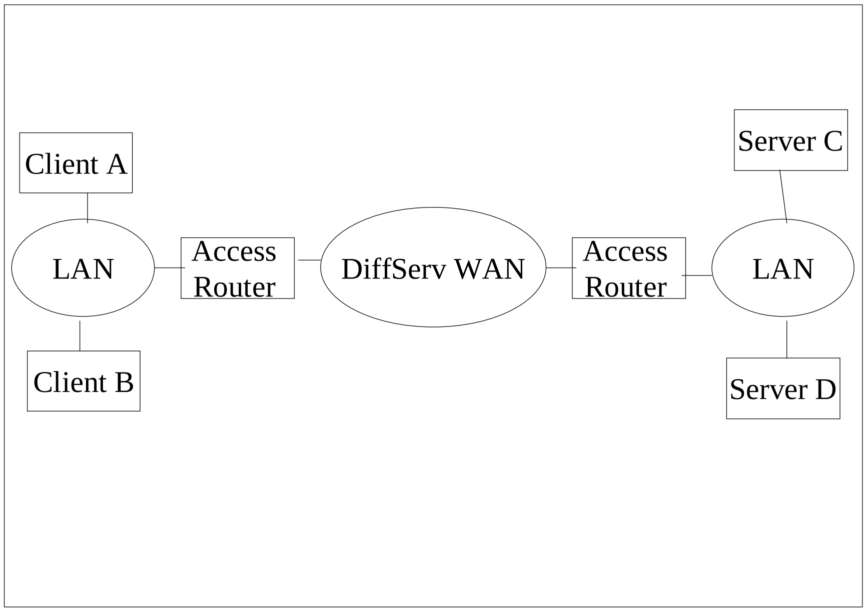


Fig. 1. A Sample DiffServ Capable Enterprise Network

In a typical enterprise, the number of servers and routers runs into hundreds and thousands. Independent configuration of all these devices in order to map one application (e.g. clearing credit card transactions) to a higher service level than another application (e.g. email) is tedious and can easily lead to missed devices and inconsistent configuration. Furthermore, the amount of resources to be reserved at a specific core router for any specific PHB (or service level) is dependent on routing topology and the performance associated with the service levels. The determination of the appropriate resources needed at specific routers needs to be done by means of an automated tool.

As a result, the network administrator in an enterprise needs tools that allow configuration of networks at a higher level of abstraction than that offered by individual configuration of the network devices. In this paper, we present a management tool for Differentiated Services, which we have built to simplify managing DiffServ in an enterprise environment. With relatively minor modifications, the same tool can be used in an ISP environment.

Section 1 (this section) introduced the problem and provides the background. In Section 2, we present the architecture of the QoS management tool that we have developed. Section 3 describes the view of QoS management that a network operator sees, as well as the view of QoS management that devices in the network experience. Section 4 provides details on how the network operator's views are transformed into device configuration and Section 5 provides details on the other components that comprise the management tool. Section 6 gives an example of managing the tools in an enterprise, including some screen views associated with the tool. Finally, we identify open issues and areas of future work.

2. The QoS Management Tool

The QoS Management tool is software used by a network administrator to configure and administer the DiffServ components in an enterprise network. The components of the management tool are as shown in Figure 2.

The QoS management tool consists of five major components:

- The *Graphical User Interface* is the means by which an administrator can input the objectives for deploying QoS in the network. The objectives constitute the business Service Level Agreements (SLAs) that the network is required to meet to satisfy the performance needs to its customers. The SLAs are specified in terms of the performance that is expected from each of the applications in the network.
- The *Resource Discovery* component is responsible for determining the topology of the network and the users and applications that are operational in the network. The component is also responsible for identifying the capabilities of each device in the network, e.g. the set of PHBs that are supported at the core routers, or the capabilities of a server to retrieve configuration information stored in the network. Some servers (e.g. IBM System 390) and routers (e.g. Nways 2216) are capable of obtaining configuration information stored in a network directory and configure themselves.
- The *view transformation logic* component is responsible for ensuring that the business SLAs that are specified by the network administrator are mutually consistent, correct and feasible with the existing capacity and topology of the

network. It also translates the business SLAs into device configuration information[1] that can be distributed to the different components.

- The *Configuration Distributor* is responsible for ensuring the device configuration is distributed to the various devices in the network. The distribution of the configuration can be done in a variety of ways. One way would be by storing them into a repository from where different devices can retrieve it[2]; making a configuration file and copying it over to the device. Another way is to have a program, which can log into the device console remotely and issue commands to configure it appropriately.
- The *performance monitor* keeps track of the performance of the network, and compares the performance obtained by different traffic flows in the network. It determines whether the business SLAs specified by the network administrator are being satisfied or not. Our current prototype only validates SLA compliance. We are exploring adaptive models that exploit feedback from the performance monitor.

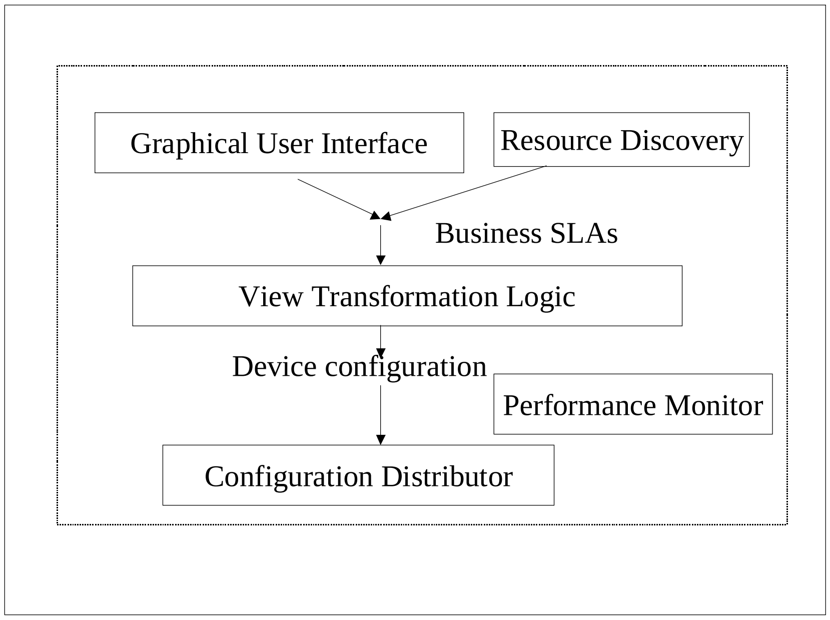


Fig. 2. Components of the QoS Management Tool

1 The device configuration in terms of PHBs is abstract in the sense that it is not mapped to how a router actually implements a PHB (and can be called an abstract policy). In this paper, we would use configuration due to its clarity.

2 The technique for storing configuration information in a repository follows the preferred architecture of policy framework working group within IETF [8].

Both the business SLAs and device configuration are specified in XML using a specialized DTD. The configuration distributor is responsible for the translation of the XML tags into the specific notation used in the configuration files, or for the schema used within the LDAP directory.

There are several advantages of using XML for the representation of SLAs and configuration information. XML provides a human-readable format for representing information, which is useful in the debugging and deployment of the tool. Due to the wide availability of parsers and validators for XML documents, the effort needed in the development of the QoS management tool is reduced. Furthermore, due to the flexibility of information representation in XML, adding new types of information is relatively easy.

3. QoS Views

As described earlier, there are two views of QoS management, one corresponding to business SLAs, which is manipulated by the network administrator, and one corresponding to the individual device configuration. Both the business level SLAs and device configuration information consists of a number of tables. The tables of the SLA information can be populated by the network administrator or by means of the resource discovery module, while the tables of device configuration information are generated by the logic component. Business SLAs are manipulated by the network administrator, while the device configuration view is used by the configuration distributor. The business SLA view would be easier for the network administrator to specify while the device configuration view would be the easier one for devices in the network to interpret.

The business SLA information manipulated by the network administrator consists of the elements shown in Figure 3. Each element in Figure 3 corresponds to a table, which is populated with details specific to the enterprise network. Arrows indicate tables that contain links to other tables, and some significant attributes in each table are outlined. Similarly, the device configuration view is shown in Figure 4.

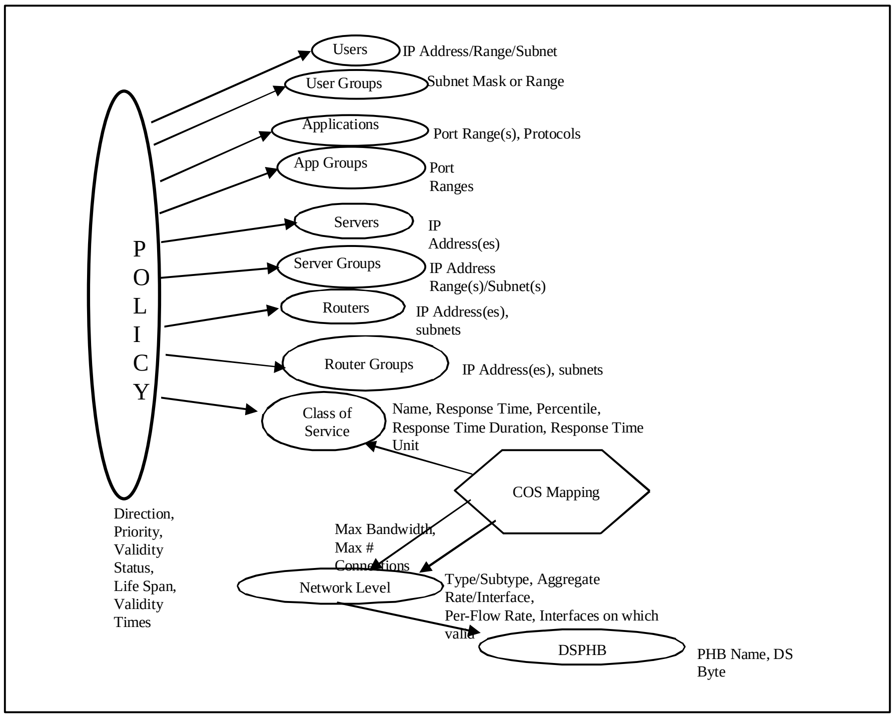


Fig. 3. High Level View for QoS Management

The network administrator sees two types of active entities in the network: users and applications. Each of these can be grouped into user groups or application groups. A user is characterized by the machine name or an IP address. If a user does not have a static IP address, the machine name is specified rather than the IP address. A user group is essentially an alias for a group of IP addresses. A user group may be defined as a collection of different users, or as a range of IP addresses. Thus, a user group may be defined for a department such as accounting which uses IP addresses from the subnet with addresses from the class C network address of 9.2.75.0. Individual machines from accounting may use DHCP to assign addresses dynamically within this network.

An application is characterized by a class of TCP or UDP port numbers that it uses. This model works well for applications that run on well known ports, or can be configured to use port numbers that are dynamic but constrained to take port numbers between a specific range. This characterization does not work for applications that use dynamic port ranges without any constraints.

The topology of the network enumerates the servers and routers that are present in the network. The goal of the topology generation is to determine the devices that need to be configured in the network, and to generate the appropriate configuration information for the listed devices. With some distribution mechanisms, the need to determine network topology can be avoided. However, the availability of network topology results in a better configuration information for the devices.

The administrator can define several classes of service in the network. The class of service measures the responsiveness of traffic flows belonging to that class. The responsiveness of a service is characterized by the round-trip delay experienced by packets in the network. This metric is readily available for TCP-based applications where the round trip time estimates can be obtained from the server's stack measurements. For UDP-based applications or the ISP environments, this metric can be obtained by statistical probing techniques (see [3] for an example of such a probing technique).

A policy is defined for a user (or user group) accessing an application (or application group) running on a server (or group of servers). The policy maps this traffic aggregate into a class of service. A policy also has a life-span, which defines the period when the policy is active. A life-span has a beginning time and an ending time. In addition to the life-span, the policy may be further restricted to occur only on specific days or dates by specifying a validity-time field.

The last step of managing application level performance is defining the class of service (COS) in terms of the PHBs that are supported at the different network devices. A list of the PHBs that are supported at the different devices is obtained by means of resource discovery. A table of COS mappings defines how each class of service is to be implemented in the network. COS mappings can be defined differently at different servers or routers in the network. The COS mapping table defines limits on the amount of resources that each class of service should enforce, e.g. limits on the network bandwidth, or bounds on the total number of connections belonging to that class should be permitted at a server.

Like the business level view, the device configuration view consists of a set of tables. These tables are divided into several groups, each group consisting of three tables. Within each group, the first table enumerates the set of devices, which constitute the group. The second table enumerates the set of PHBs, which are supported in all the devices in the group. The third table specifies the device configuration rules that map specific traffic aggregates to specific network PHBs. All devices in the group are configured identically and support the same set of PHBs. The device configuration rules consist of the fields in the protocol headers that define a traffic aggregate. Such a traffic aggregate can be defined by means of a 6-tuple, which consists of a subset of the source and destination IP addresses (or subnets), protocol, source and destination port address ranges, and the incoming DS field. Instead of the source and destination IP addresses, domain names may be specified to accommodate hosts that use DHCP and do not have a static IP address. This eliminates the need to recomputed device configuration when DHCP leases expire or change.

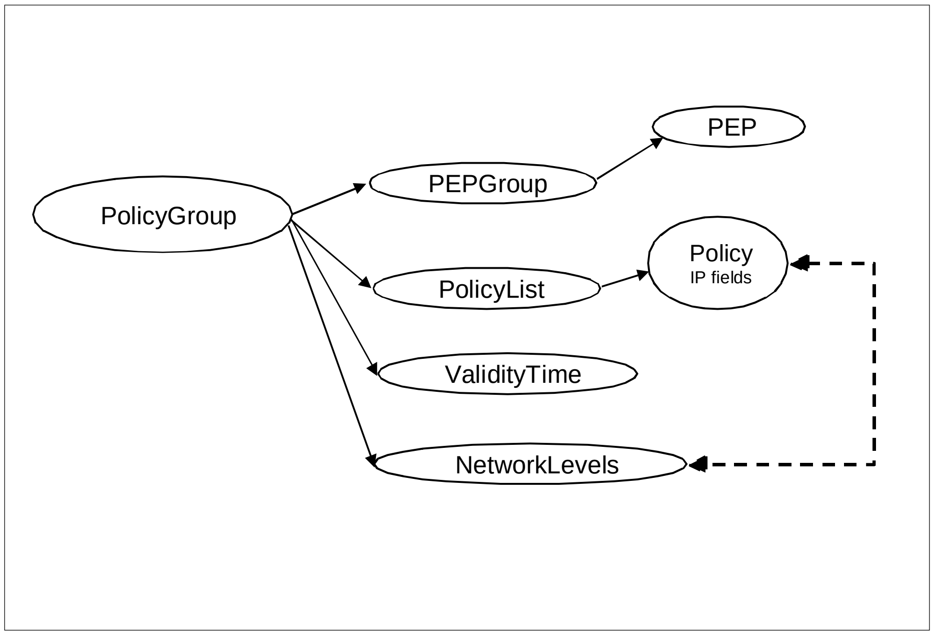


Fig. 4. Low Level View for QoS Management

4. View Transformation Logic

The view transformation logic validates the information provided in the business SLAs, and transforms them into the configuration of devices in the network. The logic furthermore ensures that the policies specified are mutually consistent and cover all aspects of interest to the network administrator.

The view transformation logic takes a parsed XML tree [4] representing the business level view and validates its semantic correctness. The validation of the syntax of the policy specification is done automatically by XML parsers/generators given the document type definition (DTD) which defines the syntax of the business level specification. The semantic validation consists of the following three types of checks:

- *Bounds Checks*: This check validates that the values taken by any parameter in the policy specification in within the specific constraints determined by the network administrator. As an example, a network administrator should be able to specify that all response times in any defined class of service be less than 1000 ms.

- *Constraint Checks*: This check validates that the values taken by any two parameters in the policy specification are within constraints that make sense. For example, two attributes of a class of service are response time and the duration over which the response time has to be measured. The latter must be larger than the former, and the network administrator should be able to specify how large the response time ought to be.
- *Consistency Checks*: These checks ensure that each traffic flow is mapped onto exactly one service class, and that each service class is properly defined at all the interfaces. These checks are applied in the manner described below.

The three types of checks can be looked upon as validating an entry in a table, a row in a table, and resolving conflicts between two rows in a table.

The approach taken for consistency checking is to map each row of a table into a region in a hyper-space defined by the columns of the table. The scheme is best illustrated by means of simplified policy table, which consists of only three columns, a user group, an application group and a service class. Two of these columns, the user group and the application group form the two axes of a 2-dimensional space. Any row, which specifies the user groups and application groups forms a region in the 2-dimensional space. This region is marked with the service class described in the row. If there is any region, which is marked with two service classes, there is a potential conflict.

With the representation of policies used in the business SLAs, the space being compared consists of six dimensions comprising of: the clients (user or user groups), the application (or application group), the server (or server groups), the life span, the validity time and the conflict resolution priority. The class of service is the column used for marking the regions in hyper-space. If two policies are found to be conflicting, the user is asked to change the conflict resolution priority to eliminate the conflict.

To determine if classes of service are mapped properly to network PHBs, the space of comparison consists of two dimensions: the class of service being mapped and the set of interfaces for which the mapping is valid. The network PHB, which the class of service is being mapped to, is used for marking the hyper-space. If the same class of service is being mapped to different PHBs at the same interface, it is an error, which must be corrected by the network administrator.

After validating that SLAs are consistent and well-formed, the view transformation logic translates the SLAs into device configuration information. The step essentially maps applications to corresponding port number ranges, users/user groups to the corresponding IP addresses and the class of service to corresponding PHBs at each of the interfaces in the network. After the device configuration rules are obtained, they are grouped to reduce the number of irrelevant rules that may be applied to a device.

In order to group the device configuration rules, each rule is considered to be either relevant or irrelevant to a server or router. To a server, a rule is relevant if the source or destination IP address range contained in the device configuration rule contains the IP address of the server. For a router, a device configuration rule is considered relevant only if the shortest-hop[3] route taken between the source and destination IP addresses passes through an access router. In most of the configuration, this rule applies only when the source or the destination IP address range is behind that access routers.

After the generation of rules associated, the devices in the network are divided into several groups. Two devices are placed in the same group if and only if they support the same set of network PHBs and have the same set of relevant policies. The device configuration rules are then distributed to the individual devices.

One function that is needed by the policy tool is the determination of the bandwidth needed for each PHB in the core routers. This requires linking up the bandwidth requirement of each class of service with the current routing topology in the network. Unlike the case of determining the relevance of a policy, we need to determine whether a router lies on the current route taken by a traffic flow, as opposed to whether a router could potentially be on one of the possible shortest hop paths. Our implementation is not currently linked to a routing daemon in the network.

5. Component Description

Although the view transformation logic is the most crucial portion of policy management, other components, namely configuration distributor, resource discovery, performance monitoring and the user interface are essential for its operation. These are described in the following sections.

5.1 Configuration Distributor

The distribution of device configuration rules to the individual devices may be done in a variety of ways. Among the possible methods that can be used are:

– *Populating a Repository*: The management tool can write out the device configuration rules into a configured repository in the network. Individual servers and routers pull the configuration information from the repository and configure themselves. The repository supported in our implementation is an LDAP directory server [5].

  This approach provides the easiest option for the QoS management tool. However, it does require that the devices in the network be capable of pulling QoS configuration information out from the *LDAP directory*. All devices may not be

[3] An implicit assumption is that OSPF or RIP routing protocols and the shortest hop path is used for routing IP packets in the network.

capable of such a function. Furthermore, since many common LDAP server implementations do not support asynchronous notification to a client when an entry changes, there is a lag between the time when a policy is entered at the management tool and when it becomes effective at a device.

An alternative repository would be a *web-server* with servlets or cgi-bin *scripts* that implement functions such as populating an LDAP directory, pushing configuration files to the devices, or invoking command line scripts. This approach can address most of the issues associated with managing policies using an LDAP directory.

– A provisioning protocol such as COPS can be used to distribute the configuration in an standard manner. Another alternative is to use SNMP for configuration. These approaches are being explored by the various working groups within the IETF [9] [10].

– *Distributing Configuration Files*: The QoS management tool can generate the configuration files that would be needed at each device and copy them remotely over to the appropriate router and server. The approach works for all types of devices, and does not require any specific software to be running at the device. However, the QoS management tool has to understand each type of device and the format of configuration file that can be used with it. Most servers support configuration of differentiated services by means of configuration files.

– *Command Line Interfaces*: Most routers permit the ability for remote administration by means of a telnet session and specific command lines. The QoS management tool can use automated scripts to specific routers and control the configuration of the router using commands specific to the router. As in the case of configuration files, the QoS management tool has to understand the scripts that can be used for different types of routers.

Regardless of the choice of the distribution mechanism, care must be exercised when configuring devices that unstable network behavior is not introduced while transiting from one network configuration to another.

5.2 Resource Discovery Mechanisms

Resource discovery identifies the set of active routers, applications and users in the network. It is included as part of the QoS manager to simplify the task of the network administrator.

The most basic form of resource discovery is by means of manual configuration by the operator. This requires that all users, applications, user groups, application groups and the network topology be input by the user. In a large network, such a manual input is rather tedious.

Fortunately, automation can assist in discovering the various entities in the network. The network topology (servers and routers) can be determined automatically using an SNMP manager. The exploration mechanism would work provided an SNMP agent is active at all routers and servers. This is usually true in an enterprise environment. Workstations and desktops usually do not have SNMP enabled, but their discovery is not crucial for the purpose of policy management. Some other topology discovery mechanisms that can be used are described in [6]. While the resource discovery by SNMP can yield information about the servers and routers in the network, the grouping of the servers into administrative domains needs be done manually.

In order to locate the users and user groups in an organization, one can look up the corporate directory. While most corporate directories list the users and their grouping into organizations, the information related to the machines or the set of addresses being used by a specific user is not readily available. Finally, the set of applications needs to be discovered. This information can be obtained automatically by looking at the network configuration of the servers that are running at different machines, which provides information about the applications that may be running at the server, and the ports they are configured to use. We are currently investigating tools that can be used to automate the discovery of users, their machines, and applications.

5.3 Performance Monitoring and GUI

The SLAs offered in the enterprise QoS are provided in terms of the TCP round-trip delays, which are present in the network. The conformance to business SLAs can be determined by monitoring the performance recorded by the stack of the servers. On the S/390 and AIX servers, the stack is augmented with a performance monitoring agent as per the architecture described in [7]. Among the attributes measured by the performance monitoring agent, the round trip delay observed by packets belonging to a specific policy class is maintained. This information can be obtained by an SNMP manager which is included as part of the QoS manager. Our current implementation does not include an SNMP manager, and does not support performance monitoring.

The GUI which is included as part of the QoS manager allows the user to manipulate the different tables that are included as part of the business SLA view. It also allows a user to view the device configuration that is generated as a result of the view transformation process, and to look at the entries in the LDAP repository which is used for configuration distribution.

6. Sample QoS Configuration

In this section, we present a simple example of a network in which QoS using DiffServ can be implemented. The network is as shown in Figure 5. It consists of a core network (WAN) and three client sites labeled as Research, Accounting, and Engineering. The three client sites access a server site with two main servers, labeled as eComm and CICS.

The active applications in the network consist of a web-server running on port 80, a mail server using ports 11-13, a video server configured to use port ranges between 4000-6000, and applications that run legacy transactions on the server. These applications use SNA encapsulated in IP using either the DLSW protocol which uses TCP and port 265, or the IBM Enterprise Extender protocol which uses port 300 and is UDP-based. DLSW and Enterprise Extender based applications are considered business-critical, while the mail server is considered an office support application. The video server and the web-server are defined as staff utility applications.

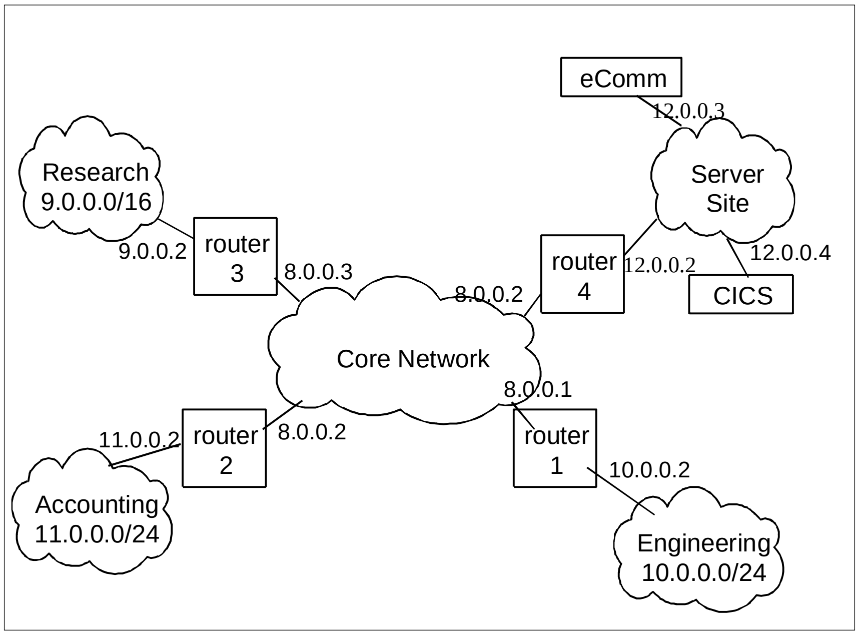


Fig. 5. Sample Enterprise Network for QoS Example

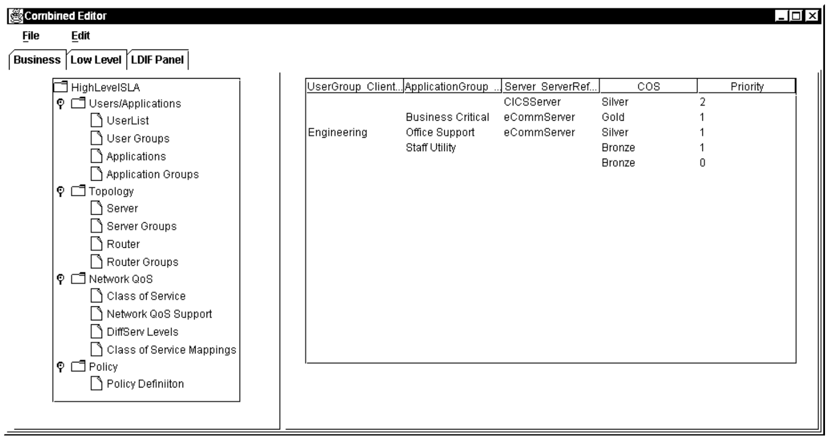
As part of the business policies, the network administrator has defined three classes of services in the network, the bronze service is the default one and has a round trip response time of less than 1 second. The silver service has a response time of less than 0.5 seconds while the Gold service has a response time of less than 100 ms. The network administrator wants to enforce the following policies in the network:

- Any access to CICS server gets the silver service.
- Business critical applications on eComm Server get the gold service.
- Engineering access to office support applications on eComm Server gets the silver Service.

- Staff utility applications get the bronze service.
- The default service is the bronze service.

The five policies expressed above are input by means of the GUI panel shown in Figure 6. The blank entries have an implied semantics of any application group or user group.

The last column in the panel is the conflict resolution priority for policies that may conflict with each other. While the above set of policies may appear to be just fine, there is a conflict between the first and fourth policies as both of them can be applied to the case of an access to a staff utility application running on the CICS Server. The tool is capable of determining these conflicts. One can correct the conflict in a variety of ways, e.g. by deleting the fourth policy (which is redundant due to the default service class) or by changing the conflict resolution priority of one of the rules to a higher value (which is the approach shown).



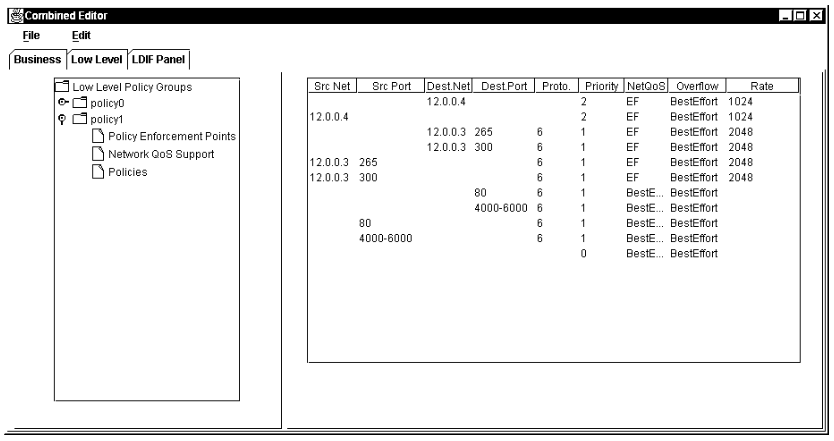
UserGroup Client...	ApplicationGroup ...	Server ServerRef...	COS	Priority
		CICSServer	Silver	2
	Business Critical	eCommServer	Gold	1
Engineering	Office Support	eCommServer	Silver	1
	Staff Utility		Bronze	1
			Bronze	0

Figure 6. High Level Policy Panel

Fig. 6. GUI Panel for Entering High Level Policies

These business policies are translated into the device configuration view. With this particular setting, the device configuration policies fall into two groups, one that applies to the eComm Server, and the other that applies to the CICS Server and all the four routers in the network. The device configuration view is as shown in Figure 7. This reflects the classic 5-tuple classifier found in the description of Differentiated Services.

The final view (which we do not show due to the length constraints of the paper), consists of the entries that will be stored in the LDAP repository. Even with the simple set of policies shown in the network above, there are close to 50 entries stored in the directory for the set of 5 policies described above. For a real enterprise, the number of entries that need to be stored in the directory would be much bigger. Thus, tools offering a higher level view of QoS management would be of great help in simplifying the management of QoS in networks.



Src Net	Src Port	Dest.Net	Dest.Port	Proto.	Priority	NetQoS	Overflow	Rate
		12.0.0.4			2	EF	BestEffort	1024
12.0.0.4					2	EF	BestEffort	1024
		12.0.0.3	265	6	1	EF	BestEffort	2048
		12.0.0.3	300	6	1	EF	BestEffort	2048
12.0.0.3	265			6	1	EF	BestEffort	2048
12.0.0.3	300			6	1	EF	BestEffort	2048
			80	6	1	BestE...	BestEffort	
			4000-6000	6	1	BestE...	BestEffort	
	80			6	1	BestE...	BestEffort	
	4000-6000			6	1	BestE...	BestEffort	
					0	BestE...	BestEffort	

Fig. 7. GUI Panel for Low Level Policies

7. Conclusions and Future Work

In this paper, we have described a scheme that can simplify the task of managing Quality of Service in Differentiated Services networks. This scheme provides an abstraction of the network that deals with applications, customers and classes of service rather than the specifics of the PHBs that are required of individual routers. Such a high-level view can offer significant simplification of the task of QoS administration. In addition to the support for differentiated services, a similar mechanism can be used for configuring RSVP support in network.

We are currently working to extend a similar high-level management for security in enterprise networks. Some areas of the tool remain to be finished, including the integration of a performance monitoring component and the investigation of effective resource discovery mechanisms for applications and servers. Similarly, our support for dynamic port numbers and dynamic host addresses needs further work. Despite these limitations, we believe such a tool can offer significant advantages to a network administrator, and assist in keeping track of business SLAs and network performance.

8. Acknowledgements

This work has evolved due to several discussions with many colleagues, including Jay Aiken, Roy Brabson, Arvind Krishna, Dilip Kandlur and Edward Ellesson. Vishal Bhotika was instrumental in coding significant portions of the management tool in a very short amount of time.

References

1. R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, *Resource ReSerVation Protocol (RSVP) Version 1 Functional Specification.* RFC2205, Sept. 1997.
2. S. Blake et. al. , *An Architecture for Differentiated Services,* Internet RFC2475, December 1998.
3. M. Beigi et. al, *Low Overhead Continuous Monitoring of IP Network Performance*, Proceedings of Symposium on Performance Evaluation of Computer and Telecommunication Systems, July 1999.
4. W3C XML Specifications, available at URL http://www.w3.org/TR/REC-xml.
5. M. Wahl, T. Howes and S. Kille, *Lightweight Directory Access Protocol V3,* RFC 1997, March 1995.
6. R. Siamwalla, R. Sharma, and S. Keshav, *Discovering Internet Topology*, July 1998. http://www.cs.cornell.edu/skeshav/papers/discovery.pdf.
7. A. Mehra et al., Policy-Based Differentiated Services on AIX, Internet Computing Magazine, October 2000.
8. Ineternet Engineering Task Force Policy Framework Working Group, Charter available at URL http://www.ietf.org/html.charters/policy-charter.html.
9. **The IETF Resource Allocation Protocol Working Group, charter available at URL** http://www.ietf.org/html.charters/rap-charter.html**.**
10. **The IETF SNMP Configuration Working Group, charter available at URL** http://www.ietf.org/html.charters/snmpconf-charter.html.