Tsaruk Yaroslav

INFORMATION BASE FOR REPRESENTATION OF MANAGEMENT INFORMATION IN DIFFSERV

Master's Thesis work

Mobile Computing

9/6/2005

University of Jyväskylä
Department of Mathematical Information Technologies

Author: Tsaruk Yaroslav

Contact information: E-mail:yatsaruk@cc.jyu.fi

Title: Information base for representation of management information in DiffServ

Work: Master's Thesis

Number of pages:

Study line: Mobile Computing

Department: University of Jyväskylä, Department of Mathematical Information Technologies

Keywords: Common Information Model, Management Information Base, Policy Core Information Model, Simple Network Management Protocol, QoS, Integrated Services, RSVP, Differentiated Services, MPLS, Constraint Based Routing, Policy Information Base, COPS, Policy Enforcement Point.

Abstract: In this work has been done analysis of frameworks, which applied for presenting information in Differentiated Services. Frameworks for analyzes were selected CIM (Common Information Model), SNMP (Simple Network Management Protocol) and COPS (Common Open Policy Service). During analyze of network management frameworks pay attention for presentation information. Frameworks SNMP and CIM share nearly the same data presentation MIB. COPS for presenting information use PIB (Policy Information Base). Also during analyze process have been considered such framework characteristics as: protocols for transmission network management information, security issues and existing software for developing application based on appropriate frameworks. In the introduction part considered overview of Quality of Services and brief overview both of the approaches, which exists for providing appropriate QoS (Integrated Services, Differentiated Services).

Acknowledgements

On the way to complete this work a lot of people make assistance to me. I would like to thank all of them.

Fist of all, I want to show my gratitude and appreciation to my supervisors: Timo Hamalainen and Alexander Sayenko. Based on their experience my supervisors gave me useful and valuable advices and provide necessary consultation regarding my topic.

Also, I would like to thank to teachers in Kharkov University of Radioelectronics and University of Jyväskylä. They give me that knowledge and skills that I have now.

I want to thank my mother and father, who help me and support me in each of my undertaking.

Finally, I want to say “thank you” to Vagan Terziyan and Helen Kaykova. They were the founders of Master’s program between Kharkov University of Radioelectronics and University of Jyväskylä. After coming to Finland they support and still give useful advices to me.

Abbreviations

ABR - Available Bit Rate

ATM - Asynchronous Transfer Mode

BA - Behaviour Aggregate

BGP - Border Gateway Protocol

BOF - Birds of a Feather

CBR - Constant Bit Rate

CDV - Cell Delay Variation

CFI - Canonical Format Indicator

CLP - Cell Loss Priority

CLS - Controlled Load Service

COPS - Common Open Policy Service Protocol

CoS - Class of Service

DA - Destination Address

DQDB - Distributed Queue Dual Bus

DSBM - Designated Subnet Bandwidth Manager

DVMRP - Distance Vector Routing Multicast Protocol

FCS - Frame Check Sequence

FDDI - Fiber Distributed Data Interface

FIFO - First in First out

FTP - File Transfer Protocol

GS - Guaranteed Service

ICMP - Internet Control Message Protocol

IEEE - Institution of Electrical and Electronic Engineers

IETF - Internet Engineering Task Force

IGMP - Internet Group Management Protocol

IP - Internet Protocol

IPv4 - Internet Protocol Version 4

IPv6 - Internet Protocol Version 6

IS - Internal System

IntServ - Integrated Services

LANs - Local Area Networks

LLC - Logical Link Control

LU - Local Use

MAC - Media Access Control

MBONE - Multicast Backbone

MBS - Maximum Burst Size

MF - Multi-field

MPLS - Multiprotocol Label Switching

MTU - Maximum Transmission Unit

NHRP - Next Hop Resolution Protocol

OOPS - Open Outsourcing Policy Service

OSPF - Open Shortest Path First

PASTE - Provider Architecture for Differentiated Services and Traffic Engineering

PCR - Peak Cell Rate

PHB - Per-Hop Behaviour

PIM - Protocol Independent Multicast

PT - Protocol Type

QOSPF - QoS-OSPF

QoS - Quality of Service

RED - Random Early Discard

ResV - Reservation Request

RFC - Request for Comment

RIF - Routing Information Field

RSVP - Resource Reservation Protocol

RSpec - QoS Specification

RTP - Real-time Transport Protocol

SBM - Subnet Bandwidth Manager

SLA – Service Level Agreement

SONET - Synchronous Optical Network

TCP - Transmission Control Protocol

TPID - Tag Protocol ID

TR - Token Ring

TSpec - Traffic Specification

ToS - Type of Service

UBR - Unspecified Bit Rate

UDP - User Datagram Protocol

UNI - User-Network Interface

VBR - Variable Bit Rate

VC - Virtual Circuit

VLAN - Virtual Local Area Network

WAN - Wide Area Network

WFQ - Weighted Fair Queuing

Contents

1 Introduction

1.1 Background

1.1.1 Introduction

Nowadays Internet provides mostly Best Effort Service. Traffic is processed as quickly as possible, but there is no guarantee as to timeliness or actual delivery. With the rapid transformation of the Internet into a commercial infrastructure, demands for service quality have rapidly developed [10].

It is becoming obvious that several service classes will likely be demanded. One service class will provide predictable Internet services for companies that do business on the Web. Such companies will be willing to pay a certain price to make their services reliable and to give their users a fast feel of their Web sites. This service class may contain a single service. Or, it may contain Gold Service, Silver Service and Bronze Service (according to old terminology), with decreasing quality. Another service class will provide low delay and low jitter services to applications such as Internet Telephony and Video Conferencing. Companies will be willing to pay a premium price to run a high quality videoconference to save travel time and cost. Finally, the Best Effort Service will remain for those customers who only need connectivity [42, 12].

Whether mechanisms are even needed to provide QoS is a hotly debated issue. One opinion is that fibers and Wavelength Division Multiplexing (WDM) will make bandwidth so abundant and cheap that QoS will be automatically delivered. The other opinion is that no matter how much bandwidth the networks can provide, new applications will be invented to consume them. Therefore, mechanisms will still be needed to provide QoS. Here we simply note that, even if bandwidth will eventually become abundant and cheap, it is not going to happen soon. For now, some simple mechanisms are definitely needed in order to provide QoS on the Internet. Our view is supported by the fact that all the major router/switch vendors now provide some QoS mechanisms in their high-end products.

The Internet Engineering Task Force (IETF) has proposed many service models and mechanisms to meet the demand for QoS [12]. Notably among them are the Integrated

Services/RSVP model [4,41], the Differentiated Services (DS) model [12,45,3], MPLS [43,38], Traffic Engineering [38] and Constraint Based Routing [39].

The Integrated Services [4] model is characterized by resource reservation. For real-time applications, before data are transmitted, the applications must first set up paths and reserve resources. RSVP is a signalling protocol for setting up paths and reserving resources. In Differentiated Services, packets are marked differently to create several packet classes. Packets in different classes receive different services. MPLS is a forwarding scheme. Packets are assigned labels at the ingress of a MPLS-capable domain. Subsequent classification, forwarding, and services for the packets are based on the labels. Traffic Engineering is the process of arranging how traffic flows through the network. Constraint Based Routing is to find routes that are subject to some constraints such as bandwidth or delay requirement.

There are many papers on each topic: Integrated Services, RSVP, Differentiated Services, MPLS, Traffic Engineering and Constraint Based Routing. Authors present their knowledge in a good and clear way, but they have never been discussed together in one paper. Analysed all papers, I try to briefly describe each of this topic and show for readers relationships among them in this chapter. Also I try to grasp the big picture of the QoS framework.

In this chapter, I give an introduction to Integrated Services, RSVP, Differentiated Services, MPLS, Traffic Engineering and Constraint Based Routing. I describe how they differ from, relate to, and work with each other to deliver QoS on the Internet.

1.1.2 Integrated Services and RSVP

The Integrated Services model [4] proposes two service classes in addition to Best Effort Service. They are:

1) Guaranteed Service [24, 33] for applications requiring fixed delay bound;

2) Controlled Load Service [24, 33] for applications requiring reliable and enhanced best effort service.

The philosophy of this model is that "there is an inescapable requirement for routers to be able to reserve resources in order to provide special QoS for specific user packet streams, or flows. This in turn requires flow-specific state in the routers".

RSVP [5] was invented as a signalling protocol for applications to reserve resources. The signalling process is illustrated in Figure 1.1. The sender sends a PATH Message to the receiver specifying the characteristics of the traffic. Every intermediate router along the path forwards the PATH Message to the next hop determined by the routing protocol. Upon receiving a PATH Message, the receiver responds with a RESV Message to request resources for the flow. Every intermediate router along the path can reject or accept the request of the RESV Message. If the request is rejected, the router will send an error message to the receiver, and the signalling process will terminate. If the request is accepted, link bandwidth and buffer space are allocated for the flow and the related flow state information will be installed in the router.

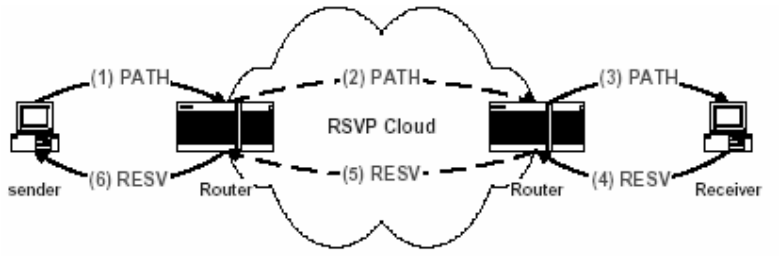

Figure 1.1 - RSVP signalling

Recently, RSVP has been modified and extended in several ways to reserve resources for aggregation of flows, to set up Explicit Routes (ERs) with QoS requirement, and to do some other signalling tasks. This is a hotly debated issue in the IETF and is beyond the scope of this paper.

Integrated Services is implemented by four components: the signalling protocol (e.g. RSVP), the admission control routine, the classifier and the packet scheduler. Applications requiring Guaranteed Service or Controlled-Load Service must set up the paths and reserve resources before transmitting their data. The admission control routines will decide whether a request for resources can be granted. When a router receives a packet, the classifier will perform a Multi-Field (MF) classification and put the packet in a specific

queue based on the classification result. The packet scheduler will then schedule the packet accordingly to meet its QoS requirements.

The Integrated Services/RSVP architecture is influenced by the work of Ferrari et al. [11]. It represents a fundamental change to the current Internet architecture, which is founded on the concept that all flow-related state information should be in the end systems [16].

Base on collected information we could formulate main weak sides related with the Integrated Services architecture. The problems are:

1) amount of state information increases proportionally with the number of flows. This places a huge storage and processing overhead on the routers. Therefore, this architecture does not scale well in the Internet core;

2) requirement on routers is high. All routers must implement RSVP, admission control, MF classification and packet scheduling;

3) ubiquitous deployment is required for Guaranteed Service. Incremental deployment of Controlled-Load Service is possible by deploying Controlled-Load Service and RSVP functionality at the bottleneck nodes of a domain and tunnelling the RSVP messages over other part of the domain.

1.1.3 Differentiated Services

In the previous chapter we define several weak sides of the RSVP and Integrated Services. Because of the difficulty in implementing and deploying Integrated Services and RSVP, Differentiated Services (DS) [31] is introduced.

1.1.3.1 Introduction to Differentiated Services

IPv4 header contains a TOS byte. The meaning of these bytes could be found in [31,9]. Applications can set three bits in the TOS byte to indicate the need for low delay or high throughput or low loss rate service. However, choices are limited. Differentiated Services defines the layout of the TOS byte (termed DS field) and a base set of packet forwarding treatments (termed Per-Hop Behaviours, or PHBs)[3]. Marking the DS fields of packets

differently, and handling packets based on their DS fields can create several differentiated service classes. Therefore, Differentiated Services is essentially a relative-priority scheme.

In order for a customer to receive Differentiated Services from its Internet Service Provider (ISP), it must have a Service Level Agreement (SLA) with its ISP. A SLA basically specifies the service classes supported and the amount of traffic allowed in each class. A SLA can be static or dynamic. Static SLAs are negotiated on a regular, e.g. monthly and yearly, basis. Customers with Dynamic SLAs must use a signalling protocol, e.g. RSVP, to request for services on demand.

Customers can mark DS fields of individual packets to indicate the desired service or have them marked by the leaf router based on MF classification.

At the ingress of the ISP networks, packets are classified, policed and possibly shaped. The classification, policing and shaping rules used at the ingress routers are derived from the SLAs. The amount of buffering space needed for these operations is also derived from the SLAs. When a packet enters one domain from another domain, its DS field may be re-marked, as determined by the SLA between the two domains.

Using the classification, policing, shaping and scheduling mechanisms, many services can be provided, for example, Expedited forwarding [23] for applications requiring low delay and low jitter service; Assured forwarding [19] for applications requiring better reliability than Best Effort Service.

Note, that the Differentiated Services only defines DS fields and PHBs. It is the ISPs' responsibility to decide what services to provide.

Differentiated Services is significantly different from Integrated Services. First, there are only a limited number of service classes indicated by the DS field. Since service is allocated in the granularity of a class, the amount of state information is proportional to the number of classes rather than the number of flows.

Differentiated Services is therefore more scalable. Second, sophisticated classification, marking, policing and shaping operations are only needed at boundary of the networks. ISP

core routers need only to implement Behaviour Aggregate (BA) classification. Therefore, it is easier to implement and deploy Differentiated Services.

There is another reason why the second feature is desirable for ISPs. ISP networks usually consist of boundary routers connected to customers and core routers/switches interconnecting the boundary routers. Core routers must forward packets very fast and therefore must be simple. Boundary routers have no need to forward packets very fast, because customer links are relatively slow. Therefore, they can spend more time on sophisticated classification, policing and shaping. Boundary routers at the Network Access Points (NAPs) are exceptions. They must forward packets very fast and do sophisticated classification, policing and shaping. Therefore, they must be well equipped.

In the Differentiated Services model, incremental deployment is possible for Assured Service. Incapable routers simply ignore the DS fields of the packets and give the Assured Service packets Best Effort Service. Since Assured Service packets are less likely to be dropped by DS-capable routers, the overall performance of Assured Service traffic will be better than the Best Effort traffic.

1.1.3.2 An End-to-End Service Architecture

In this section, service architecture for Differentiated Services is presented. This architecture provides Assured Forwarding Service, Expedited Forwarding Service (in older terminology it has the name Premium Service [15]) in addition to Best Effort Service.

Assured Service is intended for customers that need reliable services from their service providers, even in time of network congestion. Customers will have SLAs with their ISPs. The SLAs will specify the amount of bandwidth allocated for the customers. Customers are responsible for deciding how their applications share that amount of bandwidth.

SLAs for Assured Service are usually static, meaning that the customers can start data transmission whenever they want without signalling their ISPs.

Assured Service can be implemented as follows. First, classification and policing are done at the ingress routers of the ISP networks. If the Assured Service traffic does not exceed

the bit-rate specified by the SLA, they are considered as in profile. Otherwise, the excess packets are considered as out of profile. Second, all packets, in and out, are put into an Assured Queue (AQ) to avoid out of order delivery. Third, the queue is managed by a queue management scheme called RED with In and Out, or RIO.

RED (Random Early Detection) is a queue management scheme that drops packets randomly. [14]

Because in packets have low loss rate even in the cases of congestion, the customers will perceive a predictable service from the network if they keep traffic conformant. When there is no congestion, out packets will also be delivered. The networks are thus better utilized.

Best Effort traffic can be treated differently from Assured Service out traffic or they can be treated identically.

Expedited Forwarding provides low-delay and low-jitter service for customers that generate fixed peak bit-rate traffic. Each customer will have a SLA with its ISP. The SLA specifies a desired peak bit-rate for a specific flow or an aggregation of flows. The customer is responsible for not exceeding the peak rate. Otherwise, excess traffic will be dropped. The ISP guarantees that the contracted bandwidth will be available when traffic is sent. Expedited Forwarding is suitable for Internet Telephony, Video Conferencing, or for creating virtual lease lines for Virtual Private Networks (VPNs).

Because Expedited Forwarding is more expensive than Assured Service, it is desirable for ISPs to support both static SLAs and dynamic SLAs. Dynamic SLAs allow customers to request for Expedited Forwarding on demand without subscribing to it. Admission control is needed for dynamic SLAs.

First, by admission control, the amount of premium traffic can be limited to a small percentage, say 10%, of the bandwidth of input links. Second, excess packets are dropped at the ingress routers of the networks. Non-conformant flows cannot impact the performance of conformant flows. Third, premium packets are forwarded before packets of

other classes; they can potentially use 100% of the bandwidth of the output links. Since most links are full duplex, the bandwidth of the input links equals to the bandwidth of the output links. Therefore, if expedited traffic is distributed evenly among the links, these three factors should guarantee that the service rate of the EFQ is much higher than the arrival rate [17, 22]. Therefore, arriving expedited packets should find the EFQ empty or very short most of the time [34, 18]. The delay or jitter experienced by expedited packets should be very low.

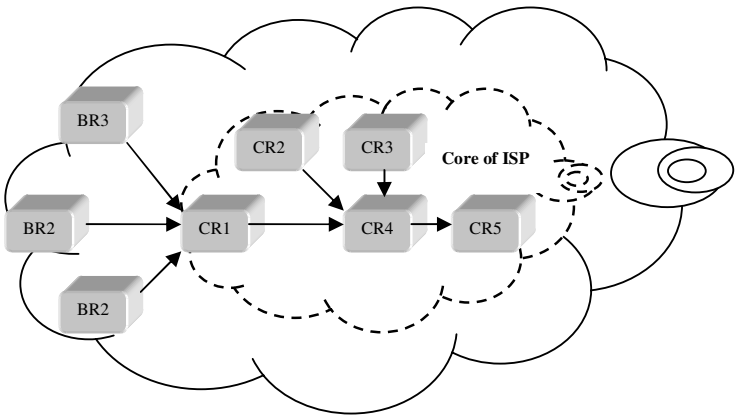

Figure 1.2 - The distribution of premium traffic in an ISP.

However, unbalanced distribution of premium traffic may cause a problem for Expedited Forwarding. In ISP networks, aggregation of traffic from the boundary routers to a core router, e.g. CR1 in Figure 1.2, is inevitable. But this is not a problem, because the output link is much faster than the input links. However, aggregation of expedited traffic in the core at CR4 may invalidate the assumption that the arrival rate of expedited traffic is far below the service rate. Differentiated Services alone can not solve this problem. Traffic Engineering/Constraint Based Routing must be used to avoid such congestion caused by unbalanced traffic distribution.

By limiting the total amount of bandwidth requested by expedited traffic, the network administrators can guarantee that expedited traffic will not starve the Assured and Best Effort traffic.

1.1.4 MPLS

MPLS is a forwarding scheme [38]. It evolved from Cisco's Tag Switching. In the OSI seven-layer model, it is between Layer 2 (L2, link layer) and Layer 3 (L3, network layer).

Each MPLS packet has a header. The header contains a 20-bit label, a 3-bit Class of Service (COS) field, an 1-bit label stack indicator and an 8-bit TTL field. The MPLS header is encapsulated between the link layer header and the network layer header. A MPLS capable router, termed Label Switched Router (LSR), examines only the label in forwarding the packet. The network protocol can be IP or others. This is why it is called Multi-Protocol Label Switching.

When a packet enters an MPLS domain, it is assigned an MPLS label, which specifies the path the packet is to take while inside the MPLS domain. Throughout the interior of the MPLS domain, each MPLS router switches the packet to the outgoing interface based only on its MPLS label. At the same time, the packet gets marked with a new label prior to transmission. The COS field is used to choose the correct service queue in the outgoing interface. At the egress to the MPLS domain, the MPLS header is removed and the packet is sent on its way using normal IP routing.

MPLS is a label-based message forwarding mechanism. By using labels, it can set up explicit routes within an MPLS domain. A packet's forwarding path is completely determined by its MPLS label. If a packet crosses all MPLS domains, an end-to-end explicit path can be established for the packet. Label also serves as a faster and efficient method for packet classification and forwarding.

MPLS also to route multiple network layer protocols within the same network and can be used as an efficient tunneling mechanism to implement traffic engineering.

For example, the switching tables must be pushed down to the MPLS routers from a central controller, similar to a policy server. Configuring these tables can be quite complex, which leads to scalability problems.

MPLS needs a protocol to distribute labels to set up Label Switched Paths (LSPs). Whether a generic Label Distribution Protocol (LDP) should be created or RSVP should be extended [1] for this purpose is another hotly debated issue. MPLS labels can also be piggy-backed by routing protocols. A LSP is similar to an ATM Virtual Circuit (VC) and is uni-directional from the sender to the receiver. MPLS LSRs use the protocol to negotiate the semantics of each label, i.e., how to handle a packet with a particular label from the peer. LSP setup can be control driven, i.e., triggered by control traffic such as routing updates. Or, it can be data driven, i.e., triggered by the request of a flow or a Traffic Trunk. In MPLS, a traffic trunk is an aggregation of flows with the same service class that can be put into a LSP. The LSP between two routers can be the same as the L3 hop-by-hop route, or the sender LSR can specify an Explicit Route (ER) for the LSP. The ability to set up ERs is one of the most useful features of MPLS. A forwarding table indexed by labels is constructed as the result of label distribution. Each forwarding table entry specifies how to process packets carrying the indexing label.

Packets are classified and routed at the ingress LSRs of a MPLS-capable domain. MPLS headers are then inserted. When a LSR receives a labeled packet, it will use the label as the index to look up the forwarding table. This is faster than the process of parsing the routing table in search of the longest match done in IP routing. The packet is processed as specified by the forwarding table entry. The outgoing label replaces the incoming label and the packet is switched to the next LSR. This label-switching process is similar to ATM's VCI/VPI processing. Inside a MPLS domain, packet forwarding, classification and QoS service are determined by the labels and the COS fields. This makes core LSRs simple. Before a packet leaves a MPLS domain, its MPLS label is removed.

MPLS LSPs can be used as tunnels. After LSPs are set up, a packet's path can be completely determined by the label assigned by the ingress LSR. There is no need to enumerate every intermediate router of the tunnel. Compared to other tunneling mechanisms, MPLS is unique in that it can control the complete path of a packet without explicitly specifying the intermediate routers.

In short, MPLS is strategically significant because:

- it provides faster packet classification and forwarding,
- it provides an efficient tunneling mechanism.

These features, particularly the second one, make MPLS useful for Traffic Engineering.

1.1.5 Traffic Engineering and Constraint Based Routing

QoS schemes such as Integrated Services/RSVP and Differentiated Services essentially provide differentiated degradation of performance for different traffic when traffic load is heavy. When the load is light, Integrated Services/RSVP, Differentiated Services and Best Effort Service make little difference. Then, why not avoid congestion at the first place? This is the motivation for Traffic Engineering.

1.1.5.1 Traffic Engineering

Network congestion can be caused by lack of network resources or by unbalanced distribution of traffic. In the first case, all routers and links are overloaded and the only solution is to provide more resources by upgrading the infrastructure. In the second case, some parts of the network are overloaded while other parts are lightly loaded. Unbalanced traffic distribution can be caused by the current Dynamic Routing protocols such as RIP, OSPF and IS-IS, because they always select the shortest paths to forward packets. As a result, routers and links along the shortest path between two nodes may become congested while routers and links along a longer path are idle [28]. The Equal-Cost Multi-Path (ECMP) option of OSPF, and recently of IS-IS, is useful in distributing load to several shortest paths. But, if there is only one shortest path, ECMP does not help. For simple networks, it may be possible for network administrators to manually configure the cost of

the links, so that traffic can be evenly distributed. For complex ISP networks, this is almost impossible.

Traffic Engineering is the process of arranging how traffic flows through the network so that congestion caused by uneven network utilization can be avoided. Constraint Based Routing is an important tool for making the Traffic Engineering process automatic.

Avoiding congestion and providing graceful degradation of performance in the case of congestion are complementary. Traffic Engineering therefore complements Differentiated Services.

1.1.5.2 Constraint Based Routing

In a sentence, Constraint Based Routing is used to compute routes that are subject to multiple constraints.

Constraint Based Routing evolves from QoS Routing. Given the QoS request of a flow or an aggregation of flows, QoS Routing returns the route that is most likely to be able to meet the QoS requirements. Constraint Based Routing extends QoS Routing by considering other constraints of the network such as policy.

The goals of Constraint Based Routing are:

- selecting routes that can meet certain QoS requirement;
- increasing the utilization of the network.

While determining a route, Constraint Based Routing considers not only topology of the network, but also the requirement of the flow, the resource availability of the links, and possibly other policies specified by the network administrators. Therefore, Constraint Based Routing may find a longer and lightly-loaded path better than the heavily-loaded shortest path. Network traffic is thus distributed more evenly.

In order to do Constraint Based Routing, routers need to distribute new link state information and to compute routes based on such information.

Distribution of Link State Information

A router needs topology information and resource availability information in order to compute QoS routes. Here, resource availability information means link available bandwidth. Buffer space is assumed to be sufficient and is not explicitly considered.

One approach to distribute bandwidth information is to extend the link state advertisements of protocols such as OSPF and IS-IS. Because link residual bandwidth is frequently changing, a tradeoff must be made between the need for accurate information and the need to avoid frequent flooding of link state advertisements.

To reduce the frequency of link state advertisements, one possible way is to distribute them only when there are topology changes, or significant bandwidth changes, e.g., more than 50% or more than 10 Mbps. A hold-down timer should always be used to limit the frequency of such advertisements. A recommended timer value is 30 seconds [28].

In the end of this chapter let's define pro and cons of Constraint Based Routing. The Pros of Constraint Based Routing are:

- meeting the need of QoS requirements of flows better;
- the improved network utilization.

The Cons of Constraint Based Routing are:

- increased communication and computation overhead;
- increased routing table size;
- longer paths may consume more resources;
- the potential routing instability.

In Constraint Based Routing, an essential issue is routing granularity. Routing can be destination based, source-destination based, class based, traffic trunk based or flow based. Routing with finer granularity is more flexible, and thus more efficient in terms of resource utilization and more stable. But the computation overhead and storage overhead are also higher.

1.1.5.3 The Position of Constraint Based Routing in the QoS Framework

In this section, we describe the relationships between Constraint Based Routing and other components in the QoS framework.

Relationship between Constraint Based Routing and Differentiated Services. Constraint Based Routing is to select the optimal routes for flows so that their QoS requirements are most likely to be met. It is not to replace Differentiated Services, but to help Differentiated Services to be better delivered. Figure 1.2 shows an example in point.

Relationship between Constraint Based Routing and RSVP. RSVP and Constraint Based Routing are independent but complementary. For a router with Dynamic Routing, when a RSVP PATH Message is received, it will be forwarded to the next hop determined by the Dynamic Routing protocol. The QoS requirements of the flow and the load of the networks are not considered in selecting the next hop. However, with a router running Constraint Based Routing, such information is considered.

The next hop of the RSVP messages determined by Constraint Based Routing therefore may be different. In either case, the actual reservation of resources for the flow is done by RSVP. In short, Constraint Based Routing determines the path for RSVP messages but does not reserve resources. RSVP reserves resources but depends on Constraint Based Routing or Dynamic Routing to determine the path.

Relationship between Constraint Based Routing and MPLS. Given that MPLS is a forwarding scheme and Constraint Based Routing is a routing scheme, MPLS and Constraint Based Routing are, in theory, mutually independent. Constraint Based Routing determines the route between two nodes based on resource information and topology information. It is useful with or without MPLS. Given the routes, MPLS uses its label distribution protocol to set up the LSPs. It does not care whether the routes are determined by Constraint Based Routing or by Dynamic Routing.

However, when MPLS and Constraint Based Routing are used together, they make each other more useful. MPLS makes it possible to do Constraint Based Routing at the traffic trunk granularity without introducing MF classification to the core routers. MPLS's per-

LSP statistics provide Constraint Based Routing with precise information about the amount of traffic between every ingress-egress pair. Given such information, Constraint Based Routing can better compute the routes for setting up LSPs. In combination, MPLS and Constraint Based Routing provide powerful tools for Traffic Engineering.

Note that when Constraint Based Routing is done at the granularity of traffic trunk, multiple LSPs may be set up between an ingress-egress pair. The number of Sink Trees needed for a network of N egress routers may become k*N, where k is a small constant.

1.2 Related work

The QoS becomes one of the important topics in networking. There is exigency to manage every days growing network traffic. Integrated and differentiated services try to solve this problem. As far as show practice, the most suitable solution for Internet is DiffServ. DiffServ provides separation by different classes of traffic and mark them. It is much more useful than take care about each flow. Such organization as DMTF provides their solution for network management field. CIM is going to manage storing and presentation data. CIM technology could be used for structuring and presentation information in different fields. Network management is one field of implementation. One more mechanism provided by DMTF is WBEM. The purpose of WBEM is realizing standardized, access, share and aggregate network management information in heterogeneous environment. There are several implementations of CIM/WBEM technology. Pegasus is one of CIM/WBEM realization presented by Open Group[37].

Another organization, such as IETF, develop COPS framework. At the beginning it was developing as a protocol for transferring management information in RSVP. But then it becomes separate framework for policy management. COPS framework widely supported by Intel Corporation. Intel provides broad specter of hardware and software, which support COPS[21].

Also there are a lot of researchers, who is trying to analyze and design solution, which allows using all of these technologies in one infrastructure.

1.3 Research problem statement

Given the SLAs, ISPs must decide how to configure their boundary routers so that they know how to handle the incoming traffic. This process is called Resource Allocation.

For static SLAs, boundary routers can be manually configured with the classification, policing and shaping rules. Resources are therefore statically allocated for each customer. Other customers can share unused resources.

For a dynamic SLA, resource allocation is closely related to the signaling process. The BB in the customer domain uses RSVP to request for resources from its ISP. The boundary routers or Bandwidth Broker can make the admission control decisions in a distributed manner at the ISP side. If boundary routers are directly involved in the signaling process, they are configured with the corresponding classification, policing and shaping rules when they grant a request. If a BB is involved rather than the boundary routers, then the BB must configure the boundary routers when it grants a request.

So, the main task of this thesis is to analyze all existing nowadays framework, which allow presenting, storing and transmitting network management information between network nodes (e.g. BB, router). The comparison all of this nowadays exists framework should show advantages and disadvantages each of this technologies.

1.4 Structure of the thesis

The rest of the thesis is organized as follows.

The chapter 2 contains more detail analyse of Differentiated Services. In this section presented information about Data path mechanism, mechanism for definition the route of packet. In this chapter also could be found information about scheduling mechanism and

describing different algorithms of scheduling. One more topic discussing in this chapter is control path mechanism.

The frameworks, which realize interchange of management information between network nodes discussed in chapter 3. These frameworks help to represent, store and transfer management information. As a frameworks for working with management information have been selected such as: SNMP, CIM and COPS.

The comparison of framework for representation information reported in the chapter 4. In the beginning of the chapter have been selected the list of parameters and all comparison process based on this factors. In the summary to this chapter is provided conclusion about weak and strong side each of this technology.

2 Differentiated Services architecture

To address some of the problems associated with IntServ, differentiated services (DiffServ) have been proposed by the IETF with scalability as the main goal. DiffServ is a per-aggregate-class based service discrimination framework using packet tagging [3]. Packet tagging uses bits in the packet header to mark a packet for preferential treatment. In IPv4, the type-of-service (TOS) byte is used to mark packets. The TOS byte consists of a 3-bit precedence field, a 4-bit field indicating requests for minimum delay, maximum throughput, maximum reliability and minimum cost, and one unused bit. However, these bits were never widely used. DiffServ redefines this byte as the DS field, of which six bits make up the DSCP (Differentiated Service CodePoint) field, and the remaining two bits are unused. The interpretation of the DSCP field is currently being standardized by the IETF. DiffServ uses DSCP to select the per-hop behavior (PHB) a packet experiences at each node. A PHB is an externally observable packet forwarding treatment, which is usually specified in a relative format, compared to other PHBs, such as relative weight for sharing bandwidth or relative priority for dropping. The mapping of DSCPs to PHBs at each node is not fixed. Before a packet enters a DiffServ domain, according to the service quality the packet is required and entitled to receive the DSCP field is marked by the end-host or the first-hop router. Within the DiffServ domain, each router only needs to look at DSCP to decide the proper treatment for the packet. No complex classification or per-flow state is needed.

DiffServ has two important design principles, namely pushing complexity to the network boundary and the separation of policy and supporting mechanisms. The network boundary refers to application hosts, leaf (or firsthop) routers, and edge routers. Since a network boundary has relative small number of flows, it can perform operations at a fine granularity, such as complex packet classification and traffic conditioning. In contrast, a network core router may have a larger number of flows, and it should perform fast and simple operations. The differentiation of network boundary and core routers is vital for the scalability of Diff-Serv.

The separation of control policy and supporting mechanisms allows these to evolve independently. DiffServ only defines several per-hop packet forwarding behaviors (PHBs) as the basic building blocks for QoS provisioning, and leaves the control policy as an issue for further work. The control policy can be changed as needed, but the supporting PHBs should be kept relatively stable. The separation of these two components is a key to the flexibility of DiffServ. A similar example is Internet routing.

It has very simple and stable forwarding operations, while the construction of routing tables is complex and may be performed by a variety of different protocols. (This often reflects a software-hardware split, where PHBs are implemented in hardware, while the control policy is implemented in software.) Currently, DiffServ provides two service models besides best effort. Expedited Forwarding [23] is a guaranteed peak rate service, which is optimized for very regular traffic patterns and offers small or no queuing delay. This model can provide absolute QoS assurance. One example of using it is to create "virtual leased lines", with the purpose of saving the cost of building and maintaining a separate network. Assured service is based on statistical provisioning [19]. It tags packets as In or Out, according to their service profiles. In packets are unlikely to be dropped, while Out packets are dropped first if needed. This service provides a relative QoS assurance.

2.1 Data Path Mechanisms

Having outlined the frameworks, we will discuss the details of Internet QoS mechanisms along two major axes: data path and control path. Data path mechanisms are the basic building blocks on which Internet QoS is built. They implement the actions that routers need to take on individual packets, in order to enforce different levels of service. Control path mechanisms are concerned with configuration of network nodes with respect to which packets get special treatment what kind of rules are to be applied to the use of resources.

We first discuss the basic data path operations in routers, which presented on the Figure 2.1, including packet classification, marking, metering, policing, and shaping. Then we cover the two basic router mechanisms, queue management and scheduling.

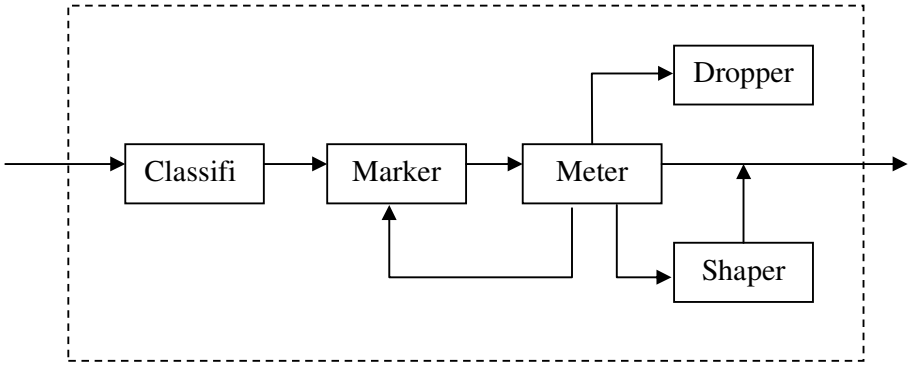

Figure 2.1: Basic data path operations

They are closely related, but they address rather different performance issues. Queue management controls the length of packet queues by dropping or marking packets when necessary or appropriate, while scheduling determines which packet to send next and is used primarily to manage the allocation of bandwidth among flows.

2.1.1 Basic Packet Forwarding Operation

As a packet is received, a packet classifier determines which flow or class it belongs to based on the content of some portion of the packet header according to certain specified rules. There are two types of classification:

General classification performs a transport-level signature-matching based on a tuple in the packet header. It is a processing-intensive operation. This function is needed at any IntServ-capable router. In DiffServ, it is referred as multifield (MF) classification, and it is needed only at network boundary.

Bit-pattern Classification sorts packet based on only one field in the packet header. It is much simpler and faster than general classification. In DiffServ, it is referred as behavior aggregate (BA) classification which is based only on DS field. It is used at network core routers.

After classification, the packet is passed to a logical instance of a traffic conditioner which may contain a meter, marker, shaper, and dropper. A marker marks certain field in the packet, such as DS field, to label the packet type for differential treatment later. A meter is used to measure the temporal properties of the traffic stream against a traffic profile. It decides that the packet is in profile or out of profile, then it passes the state information to other traffic conditioning elements. Out of profile packets may be dropped, remarked for a different service, or held in a shaper temporarily until they become in profile. In profile packets are put in different service queues for further processing. A shaper is to delay some or all of packets in a packet stream in order to bring the stream into compliance with its traffic profile. It usually has a finite buffer, and packets may be discarded if there is insufficient buffer space to hold the delayed packets. A dropper can be implemented as a special case of a shaper by setting shaper buffer size to zero packets. It just drops out-of-profile packet. The function of a dropper is known as traffic policing.

2.1.2 Queue Management

One goal of Internet QoS is to control packet loss. It is achieved mainly through queue management. Packets get lost for two reasons: damaged in transit or dropped when network congested. Loss due to damage is rare, so packet loss is often a signal of network congestion.

To control and avoid network congestion, we need some mechanisms both at network end-points and at intermediate routers. At network end-points, we depend on the TCP protocol, which uses adaptive algorithms such as slow start, additive increase and multiplicative decrease. Inside routers, queue management is used. Our goals are to achieve high throughput and low delay. The effectiveness can be measured by network power, which is the ratio of throughput to delay.

The buffer space in the network is designed to absorb short termdata bursts rather than be continuously occupied. Limiting the queue size can help to reduce the packet delay bound.

Traditionally, packets are dropped only when the queue is full. Either arriving packets are dropped (tail drop), the packets that have been in the queue the longest are dropped (drop

front) or a randomly chosen packet is discarded from the queue. There are two drawbacks with drop-on-full, namely lock-out and full queues. Lock-out describes the problem that a single connection or a few flows monopolize queue space, preventing other connections from getting room in the queue. The "full queue" problem refers to the tendency of drop-on-full policies to keep queues at or near maximum occupancy for long periods. Lock-out causes unfairness of resource usage while steady-state large queues results a longer delay.

To avoid these two problems, we need active queue management, which drops packets before a queue becomes full. It allows routers to control when and how many packets to drop. An important example of such algorithm is Random Early Detection (RED).

RED controls the average queue size using time-based exponential decay, and it marks (or drops) arriving packets probabilistically. The probability of marking increases as the estimated average queue size grows. It uses two thresholds: minth and maxth, shown in Figure 2.2.

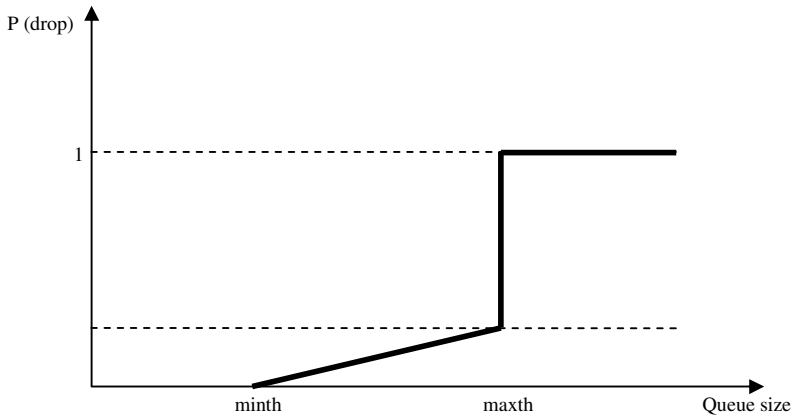

Figure 2.2 - RED queue management algorithm

When the average queue size is less that minth, no packets are marked. This should be the normal mode of operation. When the average queue size is greater that maxth, every arriving packet is marked. This mode should occur only under congestion. When the average queue size is between minth and maxth, each arriving packet is marked with a probability $p \in [0,maxp]$, where p is a function of the average queue size. This is the congestion avoidance phase. Packets get marked in proportion to the flow's link share.

RED has two important properties: It avoids global synchronization of TCP by introducing randomness and it has no bias against bursty traffic.

RIO refines RED with In/Out bits. The idea of RIO is to tag packets as being “In” or “Out” according to their service profiles, and preferentially drop packets that are tagged as being “Out” if congestion occurs. RIO uses two sets of parameters, one for In packets, and one for Out packets. The probability of marking an In packet depends on avg in, the average queue size for In packets, while the probability of marking an Out packet depends on avg total, the average total queue size for all (both In and Out) packets.

One more mechanism that could be used for congestion avoidance is Weighted Random Early Detection (WRED) [35]. Weighted RED (WRED) generally drops packets selectively based on IP precedence (weights). Packets with a higher IP precedence have more chance to become dropped than packets with a lower precedence. Thus, higher priority traffic is delivered with a higher probability than lower priority traffic. However, you can also configure WRED to ignore IP precedence when making drop decisions so that non-weighted RED behavior is achieved.

WRED is useful on any output interface where you expect to have congestion. However, WRED is usually used in the core routers of a network, rather than the edge. Edge routers assign IP precedence to packets as they enter the network. WRED uses this precedence to determine how it treats different types of traffic.

The main difference between WRED and RIO is that WRED uses one average queue length to calculate drop probabilities, while RIO uses two average queue lengths. WRED calculates its average queue length based on all packets present in the queue. RIO does that too but, in addition, it calculates a separate queue length for packets in the queue tagged as in profile [6].

One of the main benefits is that WRED provides separate thresholds and weights for different IP precedence, allowing you to provide different qualities of service for different traffic. Standard traffic may be dropped more frequently than premium traffic during periods of congestion.

2.1.3 Scheduling

Packet delay control is an important goal of Internet QoS. Packet delay has three parts: propagation, transmission, and queuing delay. Propagation delay is given by the distance, the medium and the speed of light, about 5 μs/km. The per-hop transmission delay is given by the packet size divided by the link bandwidth. The queuing delay is the waiting time that a packet spends in a queue before it is transmitted. This delay is determined mainly by the scheduling policy.

Besides delay control, link sharing is another important goal of scheduling. The aggregate bandwidth of a link can be shared among multiple entities, such as different organizations, multiple protocols (TCP, UDP), or multiple services (FTP, telnet, real-time streams). An overloaded link should be shared in a controlled way, while an idle link can be used in any proportion.

Although providing delay guarantee and rate guarantee, are crucial for scheduling. Scheduling needs to be kept simple since it needs to be performed at packet arrival rates. For example, at OC-48 rates, a scheduler only has 100 ns per packet to make a scheduling decision.

Scheduling can be performed on a per-flow basis or a per-traffic-class basis or combination of these two results in a hierarchical scheduling. There are varieties of scheduling algorithms.

- First Come First Serve (FCFS) is the simplest scheduling policy. It has no flow or class differentiation, no delay or rate guarantee.
- Priority scheduling provides a separate queue for each priority class. Basically, it is a multiple-queue FCFS scheduling discipline with the higher priority queue being served first. It has a coarse granularity class differentiation. But is has no delay or rate guarantee for individual flows.
- Weighted Fair Queuing (WFQ) is variation of weighted round robin scheduling, where the weights are coupled with reserved link rates. It can provide end-to-end

delay guarantee on a per-flow basis. But it cannot provide separate delay and rate guarantee. A resulting problem of this is that a low bandwidth flow will experience high delay. There are many variants of WFQ, most of them can be compared with GPS (Generalized Processor Sharing), which is defined for a fluid model of traffic, and serves as a theoretic reference model.

- Earliest Deadline First (EDF) is a form of dynamic priority scheduling. Each packet is assigned a sending deadline, which is the sum of arrival time and delay guarantee. Coupled with traffic shapers, EDF can provide separate delay and rate guarantee.

2.2 Control Path Mechanisms

In this section, we discuss the control path mechanisms including admission control, policy control, and bandwidth brokers.

2.2.1 Admission Control

Admission control [25] implements the decision algorithm that a router or host uses to determine whether a new traffic stream can admitted without impacting QoS assurances granted earlier. As each traffic stream needs certain amount of network resources (link bandwidth and router buffer space) for transferring data from source to destination, admission control is used to control the network resource allocation. The goal is to correctly compute the admission region, since an algorithm that unnecessarily denies access to flows that could have been successfully admitted will underutilize network resource; while an algorithm that incorrectly admits too many flows will induce QoS violations.

There are three basic approaches for admission control: deterministic, statistic, and measurement-based. The first two use a priori estimation, while the later one is based on the current measurement of some criteria parameters. The deterministic approach uses a worst-case calculation, which disallows any QoS violation. It is acceptable for smooth traffic flows, but it is inefficient for flows and leads to lower resource utilization. Both statistical and measurement-based approaches allow a small probability of occasional QoS violation to achieve high resource utilization.

2.2.2 Policy Control

Policy [32] specifies the regulation of access to network resources and services based on administrative criteria. Policies control which users, applications, or hosts should have access to which resources and services and under what conditions. Instead of configuring individual network devices, ISPs and corporate administrators would like to regulate the network through policy infrastructure, which provide supports for allowing administrative intentions to be translated into differential packet treatment of traffic flows.

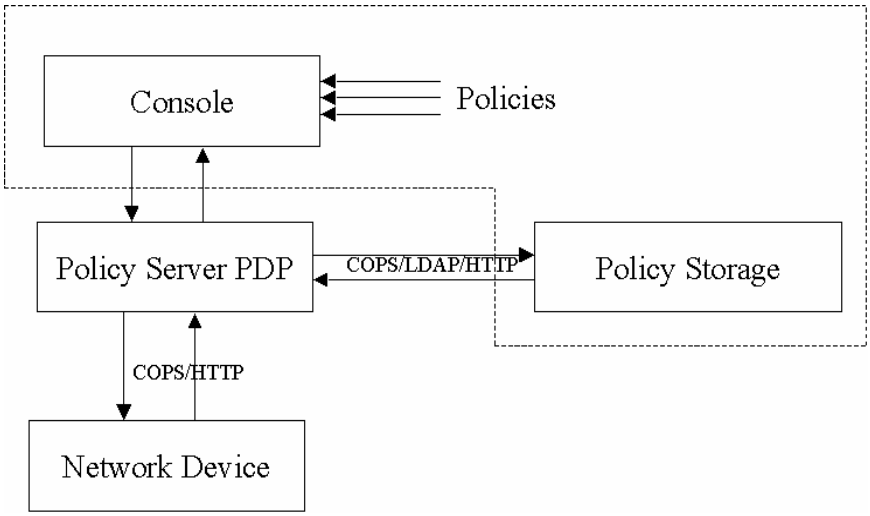

Figure 2.3 - Policy architecture

In Figure 2.3 depicted typical policy architecture. Each domain may contain one or more policy servers whose, function is to make policy and configuration decisions for network elements. The policy server has access to a policy database (possibly through LDAP or SQL) as well as authorization and accounting databases. Each policy entry specifies a rule of "if certain condition happens, it will take certain action". A human network operator working at a management console would use a GUI management application, which interfaces to the policy server through a set of Policy API (PAPI). This allows the operator to update and monitor policy changes in the policy database.

2.2.3 Bandwidth Brokers

A bandwidth broker (BB) [36] is a logical resource management entity that allocates intra-domain resources and arranges inter-domain agreements. A bandwidth broker for each domain can be configured with organizational policies and controls the operations of edge routers. In the view of policy framework, a bandwidth broker includes the function of PDP and policy database, while edge routers serve as PEPs.

In its inter-domain role, a bandwidth broker negotiates with its neighbor domains, sets up bilateral agreement with each of them, and sends the appropriate configuration parameters to the domain's edge routers. The schema of such architecture presented in Figure 2.4.

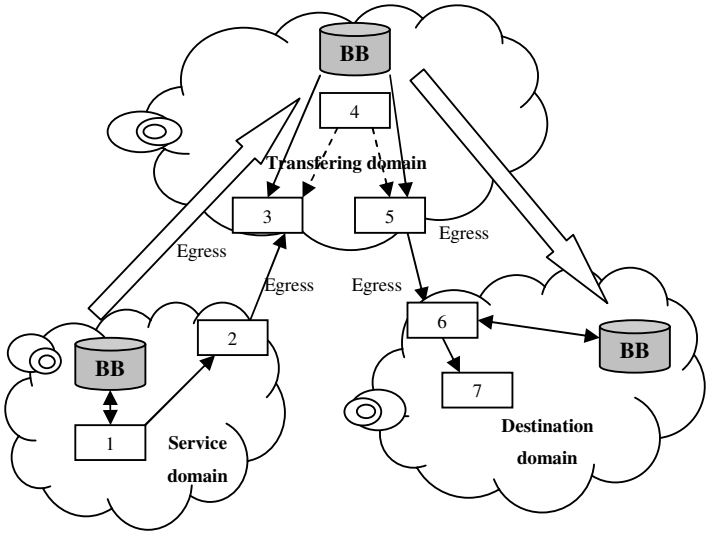

Figure 2.4 - Bandwidth broker

Bilateral agreement means that a bandwidth broke only needs to coordinate with its adjacent domains. End-to-end QoS is provided by the concatenation of these bilateral agreements across domains, together with adequate intra-domain resource allocation.

Within a domain, a bandwidth broker performs resource allocation through admission control. The choice of the intra-domain algorithm is independent of the inter-domain negotiation.

The architecture of a bandwidth broker bears some similarity to current Internet routing, in which BGP4 serves as the standard inter-domain router protocol, many choices are

available for intra-domain routing, and the concatenation of AS-to-AS (Autonomous Systems) forwarding provides end-to-end data delivery.

2.3 Service Allocation in Customer Domains

Given a SLA, a customer domain should decide how its hosts share the services specified by the SLA. This process is called Service Allocation.

There are basically two choices.

- Each host makes its own decision as to which service to use.
- A resource controller called Bandwidth Broker (BB) makes decision for all hosts.

A BB can be a host, a router or a software process on an exit router. It is configured with the organizational policies and it manages the resources of a domain. A domain may also have backup BBs. Since all hosts must cooperate to share a limited amount of resources specified by the SLA, it is technically better to have a BB to allocate resources.

At the initial deployment stage, hosts need no DS mechanism. They simply send their packets unmarked. The exit routers marked them before sending them out to the ISPs. The packets are treated as Best Effort traffic inside the customer domain. In later deployment stages, hosts may have some signaling or marking mechanisms. Before a host starts sending packets, it may decide the service class for the packets by itself or it may consult a BB for a service class. The host may mark the packets by itself or may send the packets unmarked.

If the host sends the packets unmarked, the BB must use some protocols, e.g., RSVP or LDAP (Light-weight Directory Access Protocol) [43,40], to set the classification, marking and shaping rules at the leaf router directly connected to the sender so that the leaf router knows how to mark the sender's packets.

If the SLA between a customer and its ISP is dynamic, the BB in the customer domain must also use some signaling protocol to request resources on demand from its ISP. From now on, we assume that RSVP is used as the signaling protocol.

3 Technology for presentation network management information

3.1 Introduction

Nowadays, there are several network management frameworks, which become standard, such as: SNMP, CIM, COPS. Some of these frameworks are already well known and have been used during long period of time, another become famous not so long time ago. All of these frameworks have advantages and weak sides. In this section, we consider each of these network management frameworks more detail. We discuss general issues and try to assume the key issues of each technology.

3.2 SNMP

SNMP framework was developed in 1988, since that time it becomes a widely used standard. SNMP was the first information-oriented protocol. All previous protocols were command oriented. SNMP operations are implemented using objects called variables that are maintained in managed devices. Rather than issuing commands, a network management station checks the status of a device by reading variables, and controls the operation of the device by changing (setting) variables.

SNMP framework presented as a union of technologies and solutions. SNMP framework provides solutions for presenting and structuring management information and directly SNMP protocol, used for transmitting data between network nodes. The presentation and structuring management information organized via SMI and MIB. The management data variables in a managed device are maintained in a logical collection called a management information base (MIB)[2]. The objects in the MIB are often called MIB objects, and are typically collected into sets called MIB modules.

Network management system contains two primary elements: a manager and agents. The Manager is the console through which the network administrator performs network management functions. Agents are the entities that interface to the actual device being managed. Bridges, Hubs, Routers or network servers are examples of managed devices that

contain managed objects. These managed objects might be hardware, configuration parameters, performance statistics, and so on, that directly relate to the current operation of the device in question. These objects are arranged in what is known as a virtual information database, called a management information base, and also called MIB. SNMP allows managers and agents to communicate for the purpose of accessing these objects. In the Figure 3.1 presented architecture of SNMP framework system. As far as you can see, in addition to the primary element there are two optional. The first one is policy storage, place where contains information about network management policies. The second one needed for manipulation and loading policies to the storage. System based on SNMP framework could use policy management. SNMP framework provides just architecture for controlling different network parameters, presented as variables. For example, could be provided such variables: time of work network device, incoming traffic, outgoing traffic, etc. In conjunction with policy framework presented approach for automatic control and management network devices (elements). The policy is a rule or set of rule, base on which organized control and management networks elements. This approach is much more convenient, because you should not be care of tracking different network parameters. All you need to do is just create a policy and load it to the storage. After this one of the module in SNMP Manager will take care about tracking and alteration necessary variables. Interaction between SNMP manager and SNMP agent organized via SNMP protocol. In spite of this interaction between Policy Storage and Policy client could be realized via various protocols suitable for transferring policy information, such as LDAP, HTTP/HTTPS, etc.

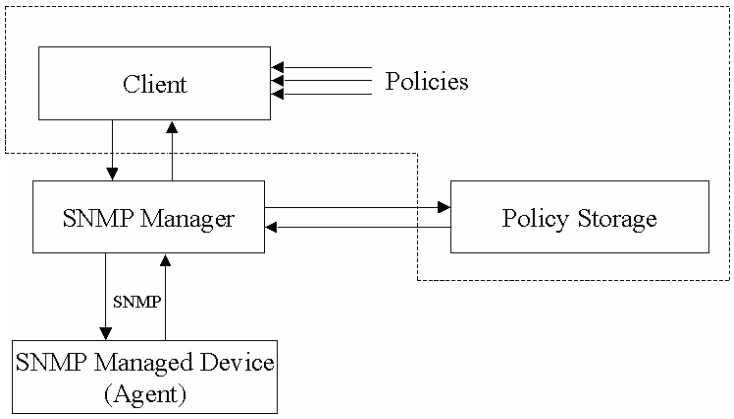

Figure 3.1 - Architecture of SNMP system

One of the important parts of the SNMP framework is transport mechanism (SNMP protocol). The main function of the SNMP Protocol is to allow management information, in the form of Management Information Base (MIB) objects, to be communicated between SNMP-capable devices. The protocol operations of the SNMP Protocol are what describe how this communication is performed. For SNMP to be useful in enabling the management of a network, it must allow a network administrator using a network management station (NMS) to easily check the status of SNMP agents in managed devices. In data communications, there are two general techniques that are used in a situation where one entity needs to be kept informed about activity or occurrences on another:

- Poll-Driven: This term refers to the general technique of having the one who wants the information ask for it; just like someone might conduct a political poll. In SNMP, the NMS would poll SNMP agents for information. A common “real life” example of polling is the model used by the regular mail service; every day you go to check your mailbox to see if you have any mail.
- Interrupt-Driven: This term refers to having a device with information that another needs to know decide to send the information of its own “volition”. In SNMP, this would refer to an SNMP agent sending information to an NMS without being asked. This is the model used by that most famous of “interrupters”, the telephone.

Due to the obvious strengths and weaknesses of these models, The SNMP Protocol is designed to use both. Polling is used for the periodic gathering of routine information, such as checking the usage statistics and general status of a device. Interrupts are used in the

form of traps that a network administrator can set on a managed device. These traps cause an SNMP agent to interrupt an NMS, when an event of import occurs.

Management information is stored in a virtual information store known as Management Information Base (MIB). Objects in MIBs are organized in a way that is described by Structure of Management Information (SMI). The language used to describe MIB objects is a reduced set of ASN.1 constructions. This only allows existence of scalars and two-dimensional arrays in the MIBs. Extension of management information is also possible by creating new MIBs or augmenting existing ones.

3.3 CIM

Common Information Model (CIM) is a framework presented by DMTF group in June 1999. It is a conceptual information model for describing management information that is not bound to a particular implementation. This allows interchanging of management information between management systems and applications. This can be either "agent to manager" or "manager to manager" communications that provides for Distributed System Management. CIM framework consists of two parts: The CIM Specification and the CIM Schema.

The CIM Specification describes the language, naming, Meta Schema and mapping techniques to other management models such as SNMP MIBs, and DMTF MIFs etc. The Meta Schema is a formal definition of the model. It defines the terms used to express the model and their usage and semantics. The elements of the Meta Schema are Classes, Properties, and Methods. The Meta Schema also supports Indications and Associations as types of Classes and References as types of Properties.

The CIM Schema provides the actual model descriptions. The CIM Schema supplies a set of classes with properties and associations that provide a well-understood conceptual framework within which it is possible to organize the available information about the managed environment. The structure of CIM Schema includes three layers:

The Core Schema is an information model that contains information common to all areas of management and includes in each model. This is basic information model.

Common Schemas are second information model, which contains common management information models, but just for this area. For example, Common Schema includes such models: systems, devices, networks, applications, metrics, databases, etc. These models define classes addressing each of the management areas in a vendor-neutral manner.

Extension Schemas represent organizational or vendor-specific extensions of the Common Schema. As example of extension schemas could be schema of adaptation for operating systems (UNIX or Microsoft Windows).

Managed Object File (MOF) is a format for definition of the CIM Schema. MOF is an ASCII or UNICODE file that can be used as input into an MOF editor, parser or compiler for use in application.

CIM, which is based upon an object-oriented model, provides a uniform data model to define and describe all devices in, and aspects of, an enterprise-computing environment. In CIM each type of device a storage array described in a common and consistent way, irrespective of the vendor and the device architecture. In contrast, in a non-CIM environment each type of device would be modeled and interfaced with in a unique way.

In parallel with developing and application CIM, the DMTF launched the Web Based Enterprise Management (WBEM) initiative. The purpose of the WBEM initiative was to develop a standardized, non-proprietary, environment-independent way to access, share and aggregate management information in a heterogeneous computing environment. WBEM can be though of as an umbrella, unifying several pieces of standard technology into a standardized management model and unified management interface. WBEM is currently comprised of three core components: data model; CIM standard; data encoding and language standard; XML encoding of CIM data; data transport mechanism; CIM operations over HTTP. CIM, as previously described, provides a common method for modeling and describing managed objects. XML encoding of CIM data allows CIM data to be presented in the industry standard XML format. CIM operations over HTTP provide an

industry standard protocol and platform independent method of transmitting CIM data. WBEM currently relies on these three components to provide a comprehensive standard management interface, but it is flexible and extensible enough to incorporate future standards and technologies in a seamless fashion.

CIM and WBEM provide management application developers with a common framework and standardized interface to develop against; different management applications can collect data from a variety of storage network devices in a standardized fashion. Figure 3.2 provides an overview of a CIM/WBEM managed environment, and details the major components involved in a CIM/WBEM solution. The major components in a typical CIM/WBEM implementation are the managed object, the CIM provider, the CIM object manager (CIMOM), the CIM server and the CIM client.

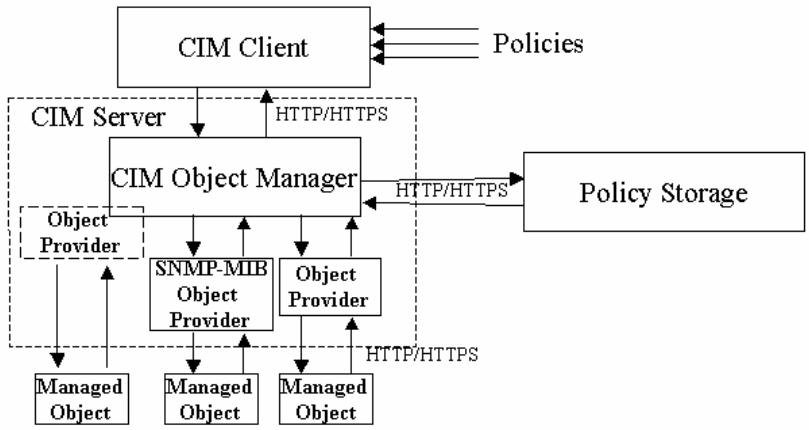

Figure 3.2 - Architecture of SNMP system

Managed objects are any devices or entities that are represented by a CIM object model, and can therefore be managed by CIM/WBEM compliant applications. CIM currently specifies object models for a broad range of physical-servers, storage devices, FC switches and appliances, tape, IP networking devices and logical-applications, operating systems, users, policies, databases-entities. New devices and more detailed and sophisticated models are introduced with each new version of the CIM schema. The DMTF releases a new version of the CIM schema approximately every six months. The power of CIM/WBEM stems from the fact that devices from different vendors, and with different architectures. For example, a modular storage array from one vendor and a monolithic storage array from

a second vendor-utilize the same object model and have an interoperable management interface. Without CIM/WBEM, in contrast, each device would be uniquely modeled and have a proprietary management interface.

The CIM Provider is the software component, which functions as the interface layer between a managed object and the CIMOM. The CIM Provider's primary purpose is to interface with a managed object, via any available management interface, and translate between CIM's standardized data format and a managed object's proprietary data format and management interface. The communication between a CIM Provider, CIMOM and managed object is bi-directional. The Provider can either request data from a managed object, to populate a CIM object with or transfer CIM object data to a managed object. The interface between a CIM Provider and a managed object is specific to the managed object. In many cases, the interface is a vendor specific proprietary API, but it may also be an industry standard interface. A CIM Provider may be embedded in a managed object, or may reside on an external piece of hardware; in the later case, it is referred to as a 'proxy' CIM Provider. Today the interface between a CIMOM and a CIM Provider is generally proprietary, but an industry effort-known as Pegasus seeking to provide a standardized CIMOM/CIM Provider interface is underway.

In the Figure 3.2 shows three examples of CIM Provider implementations. In the first case, there is a proxy CIM Provider, bundled together with the CIMOM, interfacing with a storage array via the storage vendor's proprietary API. With this model, in a heterogeneous environment, each unique type of hardware would need its own specialized CIM Provider. In the second case, there is also a proxy CIM Provider, interfacing with a device via an industry standard FC SNMP-MIB interface. In this model, where the CIM Provider is interfacing with an industry standard interface, one type of CIM Provider can interface with any device supporting the industry standard interface. In the final case, there is a CIM Provider embedded in a server's OS, interfacing with the CIMOM via the proposed Pegasus interface. In this model, which is really a second generation of CIM implementation each managed object would have its own CIM provider, and be "CIM enabled". These three examples illustrate the design and deployment flexibility that the CIM Provider brings to a CIM/WBEM implementation; the CIM Provider is an abstraction

layer, which allows CIM/WBEM to seamlessly interface with a broad range of standard and proprietary management interfaces.

CIM clients are any applications that use CIM/WBEM to communicate with CIM managed objects. Because CIM is so comprehensive, there is the potential for a broad range of CIM clients. Figure 3.2 shows three common applications that would greatly benefit from utilizing CIM/WBEM Storage Resource Management, SAN device management and Backup/Data Replication management. Without CIM/WBEM, developers of these applications must integrate with a wide variety of incompatible management interfaces, and understand the specifics of each device that they interface with. CIM/WBEM changes this to one standardized management interface with well-documented device models. If CIM client applications require specialized data not incorporated into the CIM schema, CIM/WBEM provides a well-defined methodology for adding vendor unique extensions. Moving from today's proprietary management model to CIM/WBEM offers several compelling advantages.

CIM/WBEM benefits for SRM CIM/WBEM based management offers numerous advantages to a wide range of people: application developers, system integrators, OEMs and end-users will all benefit from adopting CIM/WBEM.

Although CIM and WBEM have been around for several years, solutions employing CIM/WBEM technology are just starting to come to market. There are several reasons for the recent acceptance of CIM/WBEM-the advancement of the CIM schema to the point where comprehensive storage network management solutions are possible, the lack of other industry standard management interfaces, the promotional efforts of the DMTF and SNIA. One of the primary reasons for the adoption of CIM/WBEM, however, is that management application developers, integrators, OEMs and end users are all experiencing acute management pain. Solutions based on CIM/WBEM have the ability to efficiently provide substantive management functionality, reduce TCO and alleviate the management pain. All end users who have deployed, or are going to deploy, storage networks should look to CIM/WBEM technology, and vendors who are CIM-enabled, to simplify storage network management.

3.4 COPS

COPS was developed by the IETF RSVP Admission Policy (RAP) Working Group, which is developing a scalable policy control model for RSVP. The group is working with the IETF POLICY Working Group to ensure that COPS supports policy information exchange between PDPs and PEPs. The original purpose of the COPS protocol was to be used with RSVP for outsourcing policy decisions from an RSVP enabled router to some other entity that actually makes these decisions.

So, the COPS (Common Open Policy Service) protocol is a simple query and response protocol that allows policy servers (PDPs) to communicate policy decisions to network devices (PEPs)[7]. In order to be flexible, the COPS protocol has been designed to support multiple types of policy clients. It can be recognized that the PDP/PEP interface can be used to transfer information related to the request of resource by QoS clients and for the allocation of resources by Resource Allocation Servers (e.g. Bandwidth Broker) in a Differentiated Service network.

The IETF Policy Framework (POLICY) Working Group has developed a policy management architecture that is considered the best approach for policy management on the Internet. The typical COPS system architecture is illustrated in Figure 3.4.

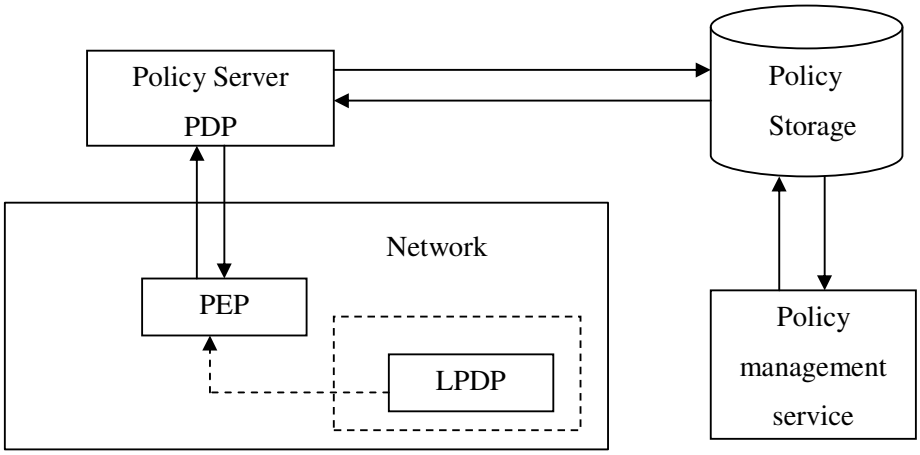

Figure 3.4 - Architecture of COPS system

The architecture of typical COPS system consists:

- Policy management service is a graphical user interface for specifying, editing, and administering policy.
- Dedicated policy repository is a place to store and retrieve policy information, such as an LDAP server or a DEN (Directory Enabled Network) device[43].
- PDP (policy decision point) is a resource manager or policy server that is responsible for handling events and making decisions based on those events (i.e., at time x do y), and updating the PEP configuration appropriately.
- PEP (policy enforcement point) exists in network nodes such as routers, firewalls, and hosts. It enforces the policies based on the "if condition then action" rule sets it has received from the PDP.
- LPDP (local policy decision point) is a scaled-down PDP that exists within a network node and is used in cases when a policy server is not available. Basic policy decisions can be programmed into this component.

Hence, it is sensible to add this resource allocation functionality in the COPS framework. In particular, there are at least two cases where it is sensible to use COPS. The first case is on the interface between an edge node and a resource control node for handling resource allocation in a network provider domain. The second case is on the interface a customer (client of a QoS enabled network) and the network provider: here COPS can be used as a protocol to signal dynamic admission control requests.

Common Open Policy Service protocol realizes exchange policy information between a Policy Decision Point and its clients, called Policy Enforcement Points. The protocol employs a client/server model, where the PDP is the server, and the PEPs are the clients. It uses a persistent TCP connection as its transport, thus there is no need for reliability mechanisms in the protocol itself. COPS supports two models, outsourcing and policy provisioning.

The policy-provisioning model is supported in DiffServ architecture where user contacts the PDP. However, outsourcing model, in which the user approaches the PEP (e.g. a router), which in its turn contacts the PDP is supported by IntServ/RSVP architecture. In Figure 3.5 illustrated the components of COPS outsourcing model and sequence of interaction between them. Then PEP should change the state it sends request to the PDP and get appropriate response.

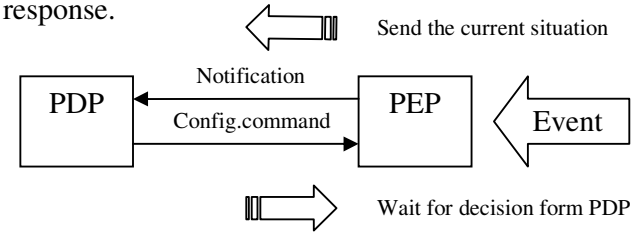

Figure 3.5 - Architecture of COPS outsourcing model

For COPS to support policy provisioning, a new client type has been introduced. This new client type is called COPS for Provisioning (COPS-PR)[8]. Provisioning model is asynchronous. In the Figure 3.6 presents architecture of provisioning model and sequence of interaction between different components of system. The PDP may proactively provision the PEP reacting to external events (such as user input), PEP events, and any combination thereof (N:M correlation).

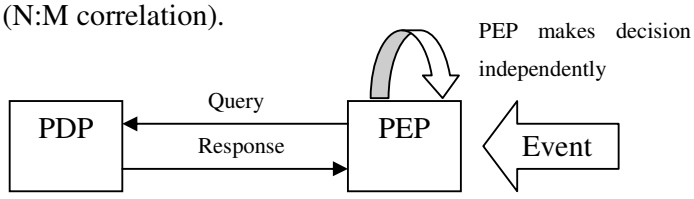

Figure 3.6 - Architecture of COPS provisioning model

Once the PEP has established communication with the PDP, PDP sends its policy capabilities to the PDP. Based on the capabilities the PDP then sends the entire applicable policy configuration to the PEP. COPS-PR supports atomic set operations on complete policy rule instances, so in SNMP MIB terminology, on complete table rows. COPS-PR does not support read or write access to individual parameters of policy rule instances (in MIB terms, to individual columnar objects).

It is independent of the policy type and can carry information about such things as QoS, Virtual Private Networks and security. COPS-PR assumes a data model that is based on the concept of named policy information that is found in Policy Information Bases (PIBs). The PIB name space is common to both the PEP and the PDP. PIB is not part of the COPS-PR protocol.

COPS uses a naming scheme for pieces of information that is very similar to the Object IDentifier (OID) naming scheme used in SNMP. The encoding of information inside COPS protocol messages is very efficient: pieces of repeated naming information are not actually encoded in the messages. As a result, the size of a COPS message for a given amount of information is small.

To provide the high level security, COPS messages can use the HMAC algorithm, IPSEC or other security mechanisms. This will provide authentication and security of the channel between the PEP and the PDP.

4 Comparison of different ways of presentation management information

4.1 Specifying the set of characteristics for comparison

In the pervious chapter we examined each of frameworks in general. In this chapter we are going to discuss it more detail and concentrate on comparison three of these technologies for management network. Among huge amount of parameters for comparison we selected the most valuable. The set of parameters is going to underline deferent solutions and more precisely show the difference between them. Also selected set of parameters allows showing the most suitable field of using each of these technologies. We have to notice that during comparison have been discovered a lot of as commons as differences.

The first parameter for comparison is information presentation. This parameter reflects aspects concerning presentation of management information. As I have mentioned from the previous chapter, each of presented technologies has original structure for presentation management information. These structures, bases, have similarity. The second comparison parameter is transport mechanism. Each of technologies uses different solutions for transmission and receiving information for and from managed device. As a characteristic for comparison transport mechanism could be presented in two ways. The first one is comparison network protocols using for interaction. The second one is comparison way of presentation information for transmission.

The next parameter for comparison is set of operations, which allows executing each framework. Considering such parameter as simplicity we are going to analyse how complicated is to design and develop network management application based on according framework. Such factor as scalability shows us how easy system is going to be increased and updated. Also one of the important factors that we are going to analyse is security aspect of each technology. During analysing security aspects we are going to proceed how much attention has been paid to security in each technology.

The list of parameters that we are going to analyse presented below.

- Information presentation
- Transport mechanism
  - Protocol
  - Presentation for transformation
- Set of operation
- Simplicity
- Scalability
- Security

4.2 Information presentation

4.2.1 SNMP

The Structure of Management Information (SMI) standard is responsible for defining the rules for how MIB objects are structured, described and organized. SMI allows dissimilar devices to communicate by ensuring that they use a universal data representation for all management information. The presentation of information is important issue for Differentiation Services. The main reason is efficiency and productivity of network management system directly related with presentation of information and storing it. The information in efficient system should be presented in such way to make marking and classification stages faster. The information should be sufficient and not excessive.

In SNMP technology all information is presented in MIB (Management Information Base). MIB is a set of objects put in order. So all MIB presented as a tree. The SNMP manager or the management application uses a well-defined naming syntax to specify the variables to the SNMP agent. Object names in this syntax are called Object Identifiers (Object IDs or OIDs). OIDs are series of numbers that uniquely identify an object to an SNMP agent.

OIDs are arranged in a hierarchical, inverted tree structure. The OID tree begins with the root and expands into branches. Each point in the OID tree is called a node and each node will have one or more branches, or will terminate with a leaf node. The format of OID is a sequence of numbers with dots in between. There are two roots for Object Identifiers, namely iso and ccit. iso starts with .1 and ccit starts with .0. Most Object Identifiers start with .1.3.6.1, where 1=iso, 3=org, 6= dod, 1 = internet. The internet subtree branches into mgmt and private.

To understand the concept of relative and absolute Object Identifiers, let us consider the AdventNet Object Identifier .1.3.6.1.4.1.2162. It specifies the path from the root of the tree. The root does not have a name or a number but the initial 1 in this OID is directly below root. This is called an absolute OID. However, a path to the variable may be specified relative to some node in the OID tree. For example, 2.1.1.7 specifies the sysContact object in the system group, relative to the internet (1.3.6.1) node in the OID tree. This is called a relative OID.

The internet subtree branches into mgmt and private. All the standard MIBs are under mgmt, while the private MIBs are under the private.enterprises subtree.

The standard MIBs are those that have been approved by the IAB. Equipment and software vendors define the private MIBs unilaterally. A branch within the private, enterprises subtree is allocated to each vendor who registers for an enterprise Object Identifier. The distinction between the standard and private MIBs is based on how the variables are defined.

The best example of a standard MIB is the [30] (also known as MIB-II). It is a MIB module, which is typically supported by all SNMP agents on TCP/IP-enabled devices or systems. This MIB file contains a description of the object hierarchy on the managed device, as well as the Object ID, syntax, and access privileges for each variable in the MIB.

One key aspect of MIBs is that, only the types of objects on the managed device are specified by the MIB and not the specific objects (or instances). For example, ifInOctets in [30] MIB specifies a type of object, for number of input octets on an interface, but the specific objects or instances of that type are specified as ifInOctets.1, ifInOctets.2, etc.,

depending on the number of interfaces. When specifying an object to the SNMP agent, a proper Object ID, which includes the instance, needs to be used by the manager. When not properly specified, the agent responds with a "No such variable" error.

To obtain values of objects from the agent, you need to specify the instance of the object. Appending an instance index to the object identifier specifies the instance of an object. For example, the last 0 in:

.iso.3.dod.1.mgmt.mib.1.sysUpTime.0

An instance index of "0" (zero) specifies the first instance, "1" specifies the second instance, and so on. Since sysUpTime is a scalar object, it has only one instance. Therefore, an instance index of zero is always specified when retrieving the value of a scalar object. An instance index higher than 0 can only be used in the case of columnar objects (in table), which can have multiple instances.

As a set of rules used to specify the format for defining managed objects used Structure of Management Information (SMI). SMI describes the MIB naming tree that is used to identify managed objects and defines the branch of the MIB tree where SNMP managed objects reside. The SMI does not define a managed object but describes a format for defining a managed object.

The crucial thing for Differentiated Services is consideration in information base mechanism for realization policy-based configuration management mechanism. The policy-base mechanism allows making classification of traffic more efficient and productive.

According to [44] SNMP management system contains: several agents (nodes, which have access to management instrumentation, run a command responder and a notification originator), at least one manager (SNMP entity that contains a command generator and a notification receiver) and management protocol to convey management information between the SNMP entities.

Adding policy-enabled capabilities to this existing architecture seems straightforward. A couple of MIB modules need to be added to the agent. These are the Policy Management

MIB module and a domain specific MIB module. Different queuing mechanisms, classification, metering and marking procedures can be used to implement the desired per-hop-behaviour. To model these structures the DiffServ MIB module contains classifier, marker, meter, queue set and queue tables as well as action (drop action, mark action and count action) tables.

In [2] discussed implementation of policy-enabled management information base according to the SNMP management framework. These MIB models the same functional data path elements, allowing the network manager to assemble them in any fashion that meets the relevant policy.

The DiffServ Policy MIB module is designed to interoperate with the Policy Management MIB and the DiffServ MIB modules for an integrated architecture of both network-wide (policy-based) and device-specific network management.

This module is intended to be used on top of DiffServ MIB module (which operates at a device level) to create an interface towards the PolicyMIB module (which operates on a network-wide scale).

The DiffServ Policy MIB module acts like an interface between high-level "network wide" policy definitions (that affect configuration of the DiffServ subsystem) and instance specific information described in the DiffServ MIB module.

Policies are executed on the managed devices, thus characteristics of objects can easily be inspected and operations described by policies can be performed on them.

4.2.2 CIM

CIM management information base is using object-oriented approach. Using this approach allows using all benefits of object-oriented approach in MIB structure. Such benefits as inheritance and encapsulation allows to structures management information in more easier and more efficient way.

CIM schema uses an object-oriented approach for describing the schema that management applications can use to communicate with each other. CIM uses a state-of-the-art approach to Object Oriented Modeling. UML was used extensively to describe classes, methods, attributes and relationships that are applicable to Management Systems. Using Object Oriented approach allowed for greater flexibility in the design of CIM. This can be seen easily from the way that the model was broken up. There is a small set of classes that define the Core Model, which captures notions applicable to all areas of management, independent of system or implementation. Built around the Core Model is the Common Model. This captures notions that are common to particular management areas, but independent of a particular technology or implementation. From the classes and associations between basic models each vendor defines its own proprietary model to cover any management area.

Partly due to their original specification and partly due to the design followed, CIM is a much broader, cleaner model that can capture management characteristics ranging from device elements to complex applications and allow for a much greater level of control granularity between both the Management Station and the agent or between Management Stations.

CIM is implementation independent when it comes to actually supporting a model like that in an agent. Though for CIM, one of the biggest gains is its representation in an XML format and the various options that exist today in presenting an XML format in different applications (e.g. XSLT). This capability can be exploited by a management application in several situations. Different sets of style sheets can be written for different circumstances. The style sheets vary from those that are general purpose to those that are very specific to a particular application or management scenario. XSL styles sheets could be produced that turn the management information represented by XML into MOF (Management Object Format) syntax. Property page style sheets could be written to display the properties of specified management objects in tabular form. Independent Software Vendors could produce their own XSL style sheets for custom graphical renderings of devices or collections or domain specific collections of management objects.

As far as we notice, CIM provides a suitable information model for specifying users, devices or components to which policy applies. Although there has been work within the IETF/DMTF on defining an information model for representing policies in CIM, but as the result of this work have not been used. I have not found any projects or information about application definition policies for CIM objects. I have found project [27], where researchers using Ponder, as a language for definition policies. In this project evaluates the use of Ponder for specifying both management and security policies, and then refine the language to cater for any shortcomings identified in the evaluation. The DiffServ part of the CIM network sub-model and the DiffServ metrics sub-model have been implemented within the CIM Object Manager (CIMOM) that the WBEM Services project provides. This project assumes that using Ponder, as a policy definition language is more suitable that proposed by IETF/DMTF. One more outcome from this project is confirmation that CIM provide useful solution for specifying network management information, such as users, devices or components to which policy applies.

4.2.3 COPS

COPS technology use PIB for presentation management information. All the information of provisioned policies in COPS-PR is kept by sets of PIB. PIB is based on the model of Structure of Management Information (SMI) and Management Information Bases (MIBs) as used with SMNP. Inside PIB, all the policy data is classified according to type or class of the policies.

A Policy Information Base (PIB) is an SPPI-based definition of the data that can be exchanged between PDP and PEP. PIBs are used in a bulk-provisioning scenario when the PDP must pre-download all the policies relevant to a PEP's interface capabilities and roles. PIBs are used in an outsourcing scenario when the PEP is required to send dynamic parameters to the PDP, so the PDP can send policy decisions to the PEP. PIBs provide a clean way to enable new services without changes to the protocol. The [29] specifies Structure of Policy Provisioning Information (SPPI) that defines numerous constructs, along with their semantics, that can be used while defining PIBs. SPPI also defines how the data definition in PIBs can be reused, thereby avoiding redundancy.

The IETF is in the process of standardizing numerous PIBs. The need for standardizing PIBs arises from the desire to have a PDP support devices from multiple vendors to enable services in a heterogeneous environment. The PIB is a conceptual tree namespace of Provisioning Classes (PRCs) and Provisioning Instances (PRIs). There may be multiple instances (PRIs) of any PRC. Each PRI is identified by a Provisioning Instance Identifier (PRID). A PRID is a unique name in a COPS object. A PRID for a PIB tree is '1.2.3.4.5'. The first four numbers represent the PRC class ('1.2.3.4') and the last number represent the PRI ('5'). Figure 4.1 shows an example PIB tree.

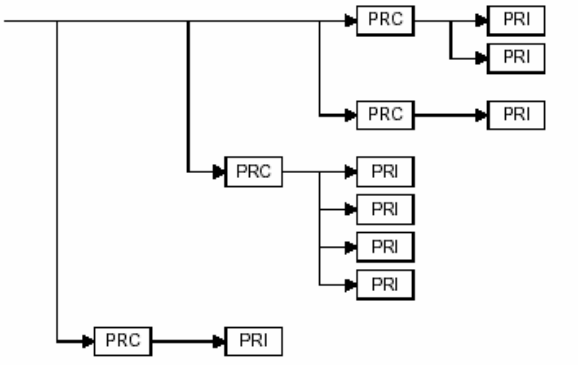

Figure 4.1 - The PIB Tree

Specific clauses in the SPPI allow the PRCs defined in generic PIBs to be reused and specialized for service specific.

PIBs via well-known object oriented concepts such as inheritance. This makes the data model defined in any PIB extensible. The SPPI also defines the notion of "subject-categories" which map to COPS client types. This clause allows a particular PIB written to address policy rules for a specific domain (such as IPSec policy) to be reused for some other application PIB. An example would be a Tunnel Configuration PIB that reuses the IPSec PIB for IPSec based VPNs. This reuse model allows for integrated data models that reduce the time to develop subject matter PIBs. The architecture of reuse COPS framework presented in Figure 4.2.

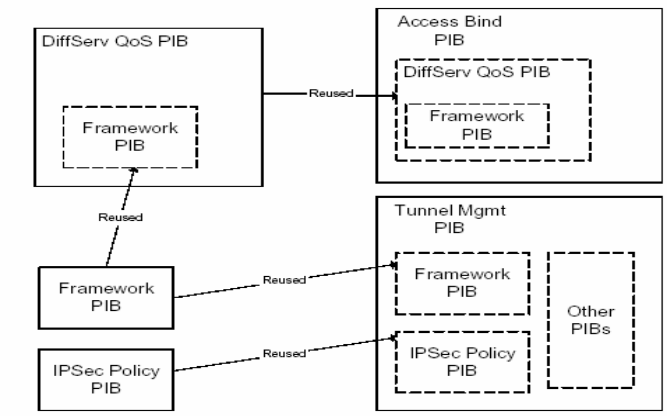

Figure 4.2 - PIB Reuse

The Framework PIB is a standard PIB designed to be used in conjunction with service-specific PIBs and consists of reusable data structures that can be used to:

- Export Scoped capability and implementation limitation information for PEP.
- Export PEP PRC support information.
- Manipulate Abstract Interface usage information via Roles.
- Mixed-Mode Outsourcing and Provisioning.
- Share PRIs via referencing across PIB instances.
- Reuse other frequently used PRCs.

The Framework PIB contains capability and implementation limitation reporting PRCs, which can group capabilities, such as reporting aggregated information for sets of interfaces having similar capabilities. Implementation limitations can also be exported to specify subsets of values for policy parameters, specific algorithms that are supported, and so on.

The Framework PIB contains tables that report PRC support information to the PDP. This PRC specifies the PRCs understood by the PEP and the attributes in the PRC supported by it. This is used to support different versions of service-specific PIBs.

The PDP can use both the capability and implementation limitations and the PRC support information to proactively size the policy before sending the rules to the PEP, instead of reactively changing policies after the PEP reports errors due to implementation limitations or capabilities not supported.

The Framework PIB allows policies to be sent to the PEP in an aggregated manner or at the finer granularity of a single interface. It supports this via assigning abstract, user-defined "roles" to the capability sets exported by the PEP. The PDP can assign roles depending on the usage of the interfaces to which the policies will be applied. These tables allow these abstract roles to be associated with any device-specific entity to which policies can be bound.

The Framework PIB contains the base set of types used to specify typed and untyped references to PRIs. This PIB also defines an indirection PRC used to reference across PIB instances active on the PEP. This allows mixed-mode operation on PEP. Mixed-mode operation gives the PEP the choice to outsource (pull) policy from the PDP when required, or to allow the PDP to dynamically provision (push) policy rules on the PEP when policies change. This flexible operation allows various service models to use this framework as required in their control plane. Mixed-mode operation also facilitates the use of COPS-PR as a converging point for policy control of various signalling protocols that may need outsourcing for policy decisions, but require provisioning to set up session scope to define when to outsource these policy requests. Other than the generic PRCs for this framework, this PIB also defines some highly reusable PRCs, such as the IP, 802 filter, and marker PRCs.

The presentation of role in the PIB is simply a string that is associated with an interface that a PDP can use to target policies. One role could be assigned to multiple interfaces or across multiple devices. The interfaces could have a multiple roles or roles combinations.

The PIB propose for DiffServ mechanism for presenting and structuring network management information [13]. In advance to presenting information about configuration and parameters, there are tables, which contain information describing the capabilities and limitations of the device using a general extensible framework. These tables are provided to the PDP and help the PDP with the configuration of functional elements that can be realized by the device.

This capabilities and limitations exchange allows a single or multiple devices to support many different variations of a functional datapath element.

In DiffServ PIB, the ingress and egress portions of a router are configured independently, but in the same manner. Each of the interfaces realizes such functionality as: classification packets, according to appropriate rules; determination of data stream according to the metering parameters; counting and marking the traffic with a Differentiated Services Code Point (DSCP); applying appropriate policy; enqueue the traffic.

The PIB consists from the following elements: Data Path Table, Classifier Tables, Meter Tables, Action Tables, Algorithmic Dropper Tables, Queue and Scheduler Tables, Capabilities Tables.

The DiffServ PIB module includes the base PRCs for setting DiffServ policy, queues, classifiers, meters, etc., and also PRC that allow a PEP to specify its device characteristics to the PDP.

4.3 Transport mechanism

4.3.1 SNMP

One of the important goals in DiffServ is transportation information between management and managed nodes. Because of the heterogeneous network domain and devices, belong to this domain. The Simple Network Management Protocol (SNMP) is the protocol responsible for allowing network management stations on a TCP/IP internetwork to perform management tasks with managed devices. The core of the protocol consists of a

set of protocol operations that allow management information to be exchanged between SNMP agents and managers. Having previously examined the generalities of SNMP and what MIB objects are, we can now get down to the "nitty gritty" of how management information is actually communicated using SNMP.

Message generation in SNMP is a bit different than the typical TCP/IP client/server model used for most other protocols. There aren't really any formal "clients" and "servers" in SNMP, since management information can be obtained from any device—it is distributed. Most of the message exchanges use a matched pair of request and reply messages. The network management station (NMS) usually acts as the client in these exchanges, sending a particular get or set request to an SNMP agent, which plays the role of server for the information it contains. However, SNMP agents aren't usually considered "servers" in the conventional sense of the term.

SNMP traps deviate from the normal request/reply model of message generation entirely. When a trap is triggered, an SNMP agent sends a trap message to a network management station on its own, not in reaction to receiving a request. Since trap messages are unconfirmed there is no reply. Note, however, that the SNMPv2/v3 InformRequest-PDU message is confirmed, and a response message is thus sent back to the NMS that generates it.

Once a message has been generated, it is sent using the protocols at the levels below the application layer where SNMP resides. The current SNMP standard set separates description of protocol operations and PDUs from the methods used to actually send them. Starting with version 2, SNMP has defined several transport mappings that describe how SNMP PDUs can be sent over a variety of internetworking protocol suites, including TCP/IP, OSI, IPX/SPX (Novell) and Appletalk.

Many of the specific details of SNMP messaging depend on the transport mapping that is used in a particular implementation. SNMP is of course primarily used on TCP/IP internetworks, and TCP/IP is where our main interest lies in this part of the Guide. I will therefore continue this discussion by looking at transport issues when SNMP is used over IP.

The standard IP transport mapping for SNMP calls for it to be carried using UDP. This decision goes back to the initial implementation of SNMPv1 (before there were distinct transport mappings.) UDP was likely chosen because it is more efficient for the simple request/reply messaging scheme SNMP uses; the many TCP features were not considered necessary and add overhead that SNMP's designers wanted to avoid. It is possible that TCP could be used to carry SNMP, defined as a different transport mapping, but I don't believe this is actually done.

In order to achieve robustness of management, SNMP protocol uses the connectionless transport protocol UDP. The use of UDP allows SNMP information communication to be “streamlined”, since there is no need to establish a TCP connection, and since message headers are shorter and processing time slightly reduced. But the use of UDP introduces a couple of issues that SNMP implementations must be concerned with.

The first issue is that of message length. SNMP PDUs can carry many MIB objects, which means they could potentially be rather large. However, UDP is limited in the size of message it can carry (where TCP is not). The standards specify that SNMP entities must accept messages up to at least 484 bytes in size. They also recommend that SNMP implementations be able to accept even larger messages, up to 1472 bytes, which would correspond to the largest size message that can be encapsulated in an Ethernet frame (1,500 bytes, allowing 20 bytes for the IP header and 8 for the UDP header.) This disadvantage doesn’t allow us to transmit a big amount of network management information in DiffServ. The capabilities of SNMP allow us transmit large amount of network management information, but because of the using wrong protocol this capabilities become nothing. This is crucial feature of SNMP protocol which makes inconvenient to use SNMP protocol in DiffServ domain, where there is a need in transmission big amount of configuration information as fast as possible.

The use of the GetBulkRequest-PDU message type in SNMPv2/v3 requires particular care, since it allows a single request to result in many MIB objects being sent back in a response. The Max Repetitions parameter must be chosen conservatively so the SNMP agent doesn't try to send an enormous message that won't fit.

The second issue, which make inconvenient to use SNMP transmission protocol in DiffServ domain is also related with one of the disadvantages UDP protocol – unguarded delivery packets. DiffServ needs to be sure that all configuration network management information has been delivered.

UDP is the price we pay for its efficiency and simplicity: a lack of transport features. UDP doesn't guarantee data delivery or handle retransmissions, which means a request or reply could in theory is lost in transit. Only the device that initially sends a request can know if there was a problem with transport—it sends the request, and if it receives no reply knows either the request or response got lost. This puts the responsibility for retransmission on the part of the device that sends the request message.

NMSes sending requests to SNMP agents generally use a timer to keep track of how much time has elapsed since a request was sent. If the response doesn't arrive within a certain time interval, the request is sent again. Because of how SNMP works, having a request be received more than once accidentally will normally not cause any problems (a property known as idempotence). The NMS does need to employ an algorithm to ensure that it does not generate too many retransmissions and clog the network (especially since congestion might be causing the loss of its messages in the first place.)

Since traps are unconfirmed, there is no way for the intended recipient of a trap PDU to know if did not arrive, nor is there any way for the sender of the trap PDU to know. This is just a weakness in the protocol; the overall reliability of TCP/IP (and the underlying networks) ensures that these messages are not lost very often.

4.3.2 CIM

DiffServ has a strict demand to for transportation of information between management and managed nodes. For transmission information in CIM framework could be used CIM-XML protocol, CIM-XML message encoded in HTTP message. CIM-XML protocol is a protocol regulation representation of CIM Message in XML format. Information in the CIM-XML message organized according to predefined structure. In the Example 4.1 presented CIM-XML message.

```
<CLASS NAME="CIM_LogicalPort" SUPERCLASS="CIM_LogicalDevice">
<QUALIFIER TRANSLATABLE="true" NAME="Description" TYPE="string">
<VALUE>The abstraction of a port or connection point of a Device. This object should be instantiated when
the Port has independent management characteristics from the Device that includes it. Examples are a Fibre
Channel Port and a USB Port. This class would not be instantiated for an Ethernet Port which is not managed
independently of the EthernetAdapter.</VALUE>
</QUALIFIER>
<PROPERTY NAME="Speed" TYPE="uint64">
<QUALIFIER TRANSLATABLE="true" NAME="Description" TYPE="string">
<VALUE>The speed of the Port in Bits per Second.</VALUE>
</QUALIFIER>
<QUALIFIER TRANSLATABLE="true" NAME="Units" TYPE="string">
<VALUE>Bits per Second</VALUE>
</QUALIFIER>
</PROPERTY>
<PROPERTY NAME="MaxSpeed" TYPE="uint64">
<QUALIFIER TRANSLATABLE="true" NAME="Description" TYPE="string">
<VALUE>The max speed of the Port in Bits per Second.</VALUE>
</QUALIFIER>
<QUALIFIER TRANSLATABLE="true" NAME="Units" TYPE="string">
<VALUE>Bits per Second</VALUE>
</QUALIFIER>
</PROPERTY>
</CLASS>
```

Example 4.1 – CIM-XML message

One more way for interchanging information between server and client in CIM is encapsulation CIM-XML message in HTTP message. HTTP message, as a typical message consist of header and body. The header contains not only standard HTTP fields. The addition fields in the header of HTTP using for more accurate specification of transmitted data. Information about the version of the CIM HTTP protocol contains in the Man field. Other fields contain detail information about operation and its type. The body of the HTTP message is the XML file, which specifies parameters of the operation. The example of CIM-XML message presented below in Example 4.2.

```
M-POST /cimom HTTP/1.0
Content-Type: text/xml;charset=UTF-8
Accept: text/xml, application/xml
Man: http://www.dmtf.org/cim/mapping/http/v1.0;ns=48
48-CIMProtocolVersion: 1.0
48-CIMOperation: MethodCall
48-CIMMethod: GetClass
48-CIMObject: root%2Fcimv2
User-Agent: Java1.2.1
Host: edoc5-pc
Content-length: 445
<?xml version="1.0" encoding="UTF-8"?>
```

```
<CIM DTDVERSION="2.0" CIMVERSION="2.0">
<MESSAGE ID="2005:1:24:11:0:44:58:1" PROTOCOLVERSION="1.0">
<SIMPLEREQ>
<IMETHODCALL NAME="GetClass">
<LOCALNAMESPACEPATH>
<NAMESPACE NAME="root" />
<NAMESPACE NAME="cimv2" />
</LOCALNAMESPACEPATH>
<IPARAMVALUE NAME="ClassName">
<CLASSNAME NAME="cim_logicalport" />
</IPARAMVALUE>
</IMETHODCALL>
</SIMPLEREQ>
</MESSAGE>
</CIM>
```

Example 4.2 - CIM-XML message encoded using HTTP

CIM operations over HTTP provide an industry standard protocol and platform independent method of transmitting. CIM architecture uses HTTP protocol as a transport mechanism. In case of using secure connection mechanism using HTTPS. For more efficient use of HTTP protocol to the header of HTTP protocol have been added new fields. These fields contain information about CIM Protocol version, CIM operation, method and object. Thanks for using this additional fields in the header it makes convenient to use CIM transport mechanism in DiffServ. Using additional information allows performing each request more efficient. It allows optimizing the parsing of each message. One more feature, which makes using CIM's transport mechanism is guarantee delivering all configuration information to the network node and make it much more rapidly, because of the using TCP transport protocol. Instead of SNMP, CIM uses as a transport layer protocol TCP. In TCP realized delivery confirmation mechanism. One more feature TCP is establishing persistent connection. Both of these facts guarantee reliable delivery of each send packet.

One of the disadvantages of CIM-XML format is big amount of transferred information. Therefore some network management systems use compression mechanisms for reducing amount of transferred information. One of the examples is OpenWBEM network management system.

4.3.3 COPS

COPS protocol is a mechanism for exchange information between PEP and PDP using PIB (Policy Information base) model. COPS and PIB carries policy configuration information from PDP to PEPs. The COPS framework architecture and interaction between components illustrated on Figure 4.3.

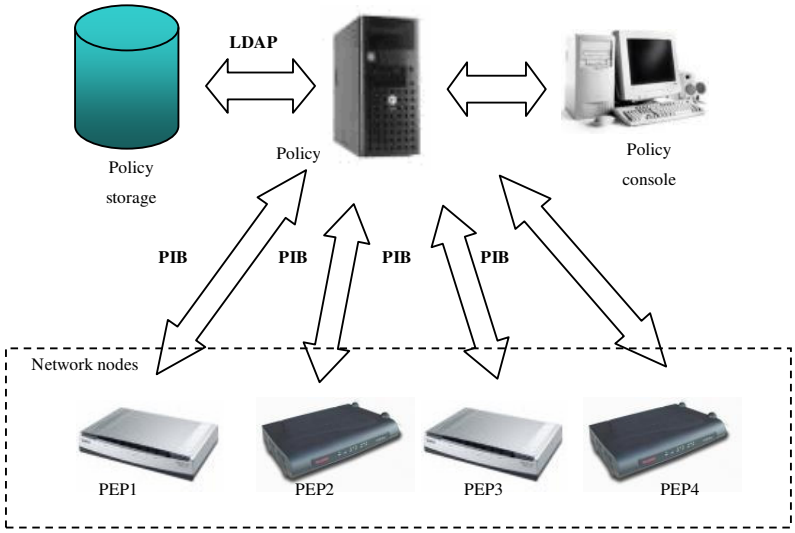

Figure 4.3– Interaction between different elements in COPS architecture

The main purpose of PIB is pushing policy data to the network devices. The capability of COPS protocol allows realizing transferring policy information to the large number of network nodes. This issue is important in DiffServ networks, where there is a need to push policy to the several networks nodes.

The interaction between PEP and PDP is initiated by PEP. As a first step interaction between PDP and PEP is specifying capabilities to the PDP. Base on received data PDP sends policy for execution on PEP. As a response from PEP, PDP receives confirmation about installation of received policy. Also as a response from PEP could be report about error, that occurred in the PEP side. About some time PDP sends update policy. The steps of interaction between PDP and PEP are illustrated on Figure 4.4.

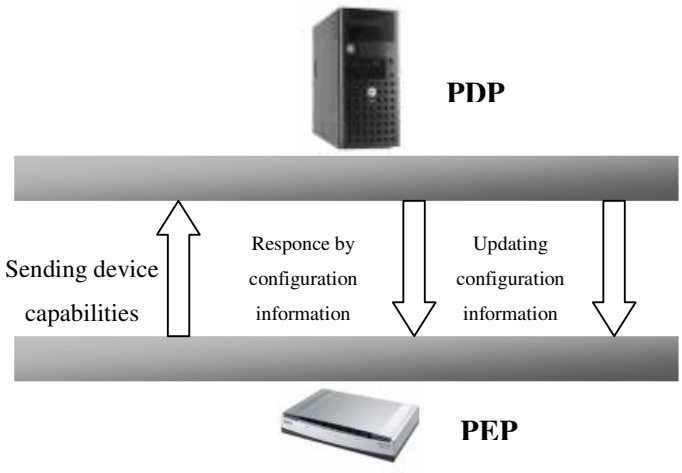

Figure 4.4 - Steps of interaction between PDP and PEP

The PIB was developed with COPS-PR in mind as a transport protocol, but found little bit another application. The PIB could be also used in SNMP framework. The PIB could replace one of the main weaknesses of SNMP – scalability. The PIB is SMI compatible. The experience of SNMP was assumed during developing of PIB.

COPS is a state full protocol. COPS is a reliable protocol because of using as a transport layer protocol TCP. The key feature of TCP protocol is establishing persistent connection and getting confirmation of delivery. COPS is bi-directional and asynchronous.

4.4 Set of operations

4.4.1 SNMP

In this section provided a detailed description of the operations performed by the SNMP Protocol. I begin with a general description of how SNMP operates and the two basic methods devices use to communicate.

The main function of the SNMP Protocol is to allow management information, in the form of Management Information Base (MIB) objects, to be communicated between SNMP-capable devices. The protocol operations of the SNMP Protocol are what describe how this communication is performed. Before looking at these operations individually in detail, it is

instructive to take an overall look at information exchange methods used in SNMP. The list of SNMP operations is presented in the Table 4.1.

Table 4.1 – Set of operations which allow to execute SNMP technology

Name of operation	Description of operation
Get	Allows the NMS to retrieve an object instance from the agent.
GetNext	Allows the NMS to retrieve the next object instance from a table or list within an agent. In SNMPv1, when an NMS wants to retrieve all elements of a table from an agent, it initiates a Get operation, followed by a series of GetNext operations.
GetBulk	New for SNMPv2. The GetBulk operation was added to make it easier to acquire large amounts of related information without initiating repeated get-next operations. GetBulk was designed to virtually eliminate the need for GetNext operations.
Set	Allows the NMS to set values for object instances within an agent.
Trap	Used by the agent to asynchronously Inform the NMS of some event. The SNMPv2 trap message is designed to replace the SNMPv1 trap message.
Inform	New for SNMPv2. The Inform operation was added to allow one NMS to send trap information to another.

Lets discuss more detail some of this operation. In DiffServ network the one of the important issues is controlling the network management parameters. The control parameters is realized throw control of network management information. SNMP framework allows to organize information exchange between nodes in DiffServ network. Observing management information variable allows control network and make appropriate action to get the best network performance. This would be a simple poll operation to read one or more management information variables, used by one SNMP entity (typically an SNMP manager) to request or read information from another entity (normally an SNMP agent on a managed device). SNMP implements this as a simple two-message request/response protocol exchange, similar the request/reply processes found in so many TCP/IP protocols.

This information request process typically begins with the user of an application wanting to check the status of a device or look at information about it. As we have discussed, all this information is stored on the device in the form of MIB objects. The communication, therefore, takes the form of a request for particular MIB objects and a reply from the device containing those object's values. In simplified form, the steps in the process are as follows:

1. SNMP Manager Creates GetRequest-PDU: Based on the information required by the application and user, the SNMP software on the network management station creates a GetRequest-PDU message. It contains the names of the MIB objects whose values the application wants to retrieve.

2. SNMP Manager Sends GetRequest-PDU: The SNMP manager sends the PDU to the device that is being polled.

3. SNMP Agent Receives and Processes GetRequest-PDU: The SNMP agent receives and processes the request. It looks at the list of MIB object names contained in the message and checks to see if they are valid (ones the agent actually implements). It looks up the value of each variable that was correctly specified.

4. SNMP Agent Creates Response-PDU: The agent creates a Response-PDU to send back to the SNMP Manager. This message contains the values of the MIB objects requested and/or error codes to indicate any problems with the request, such as an invalid object name.

5. SNMP Agent Sends Response-PDU: The agent sends the response back to the SNMP Manager.

6. SNMP Manager Processes Response-PDU: The manager processes the information in the Response-PDU received from the agent.

The use of this PDU is fairly obvious; where one of the three Get PDUs specifies a variable whose value is to be retrieved, the SetRequest-PDU message contains a specification for variables whose values are to be modified by the network administrator.

Remember that SNMP does not include specific commands to let a network administrator control a managed device. This is in fact the “control method”, by setting variables that affect the operation of the managed device.

The set process is the complement of the get process; the same basic idea, pretty much, but a reversal in how the object values “travel” and what is done with them. The process follows these steps:

- SNMP Manager Creates SetRequest-PDU: Based on the information changes specified by the user through the SNMP application, the SNMP software on the network management station creates a SetRequest-PDU message. It contains a set of MIB object names and the values to which they are to be set.

- SNMP Manager Sends SetRequest-PDU: The SNMP manager sends the PDU to the device being controlled.

- SNMP Agent Receives and Processes SetRequest-PDU: The SNMP agent receives and processes the set request. It examines each object in the request along with the value to which the object is to be set, and determines if the request should or should not be honored.

- SNMP Agent Makes Changes and Creates Response-PDU: Assuming that the information in the request was correct (and any security provisions have been satisfied), the SNMP agent makes changes to its internal variables. The agent creates a Response-PDU to send back to the SNMP Manager, which either indicates that the request succeeded, or contains error codes to indicate any problems with the request found during processing.

- SNMP Agent Sends Response-PDU: The agent sends the response back to the SNMP Manager.

- SNMP Manager Processes Response-PDU: The manager processes the information in the Response-PDU to see the results of the set.

As far as we saw from this chapter, SMNP framework proposes well suited set of operation. It suits for getting and setting values of appropriate object instances.

According to SNMP framework, interaction between agent and manager organized by specify set of operation. SNMP framework provides operations for set and get values specified object. In case with DiffServ it is not useful. It means that each time when some node in the network wants to get or set value of some object it should send specified operation. The DiffServ network's nodes forced to do a lot of set/get operation. As a result of using SNMP operations could be creation a lot of requests and increasing network traffic.

4.4.2 CIM

DMTF defines the set of operation for CIM framework. The CIM Operations over HTTP Specification defines a set of operations that a WBEM client implements to operate in an open, standardized manner. WBEM Operations are Protocol Independent. The CIM framework's operations could be two types: single or multiple. The single (Individual) operation is operation, which perform just one operation. Multiple (Batched) operations allow perfuming set of operations.

For invocation one or more methods should be generated CIM Operation Message requests. There are two types of methods: extrinsic and intrinsic.

Extrinsic methods defined as a method on a CIM Class in some Schema. Extrinsic methods are invoked on a CIM Class (if static) or Instance. One of the example of single operation could be calling terminate() method on class Win32_Process.

Intrinsic methods model a CIM operation that operates on a schema or namespace. The methods defined via the CIM Operations over HTTP Specification. Intrinsic methods are further characterized by the fact that they are made against a CIM Namespace.

Availability of such type as extrinsic allows to use operation much more efficient in DiffServ networks. Using extrinsic operation allows to reduce the amount of transferred information to the operation side.

The DMTF categories the following operation requests as types of intrinsic methods. CIM operations allow to act with such data type as: Data, Meta Data, Queries. Abilities of CIM framework operation operate with wide variety of data allows to work much more efficient with network management information. Presence of functions, which could operate with query allow to make operations against namespace using just one operation. This solution allows to perform operation much more efficient, specially in DiffServ networks.

In the Table 4.2 presented set of CIM operations sorted by group.

Table 4.2 – Set of operations which allow executing CIM technology

Group	**Dependency**	**Methods**
Basic Read	None	GetClass EnumerateClasses EnumerateClassName GetInstance EnumerateInstances EnumerateInstanceName GetProperty
Basic Write	Basic Read	SetProperty
Instance Manipulation	Basic Write	CreateInstance ModifyInstance DeleteInstance
Schema Manipulation	Instance Manipulation	CreateClass ModifyClass DeleteClass
Association Traversal	Basic Read	Associators AssociatorNames References ReferenceNames
Query Execution	Basic Read	ExecQuery
Qualifier Declaration	Schema Manipulation	GetQualifier SetQualifier DeleteQualifier EnumerateQualifiers

Each of this operation allows doing some operation with network management data. Let's consider more detail each of this operation. The set of operations, presented in Table 4.2 divided in several groups, such as: Basic read, Basic write, Instance Manipulation, Schema Manipulation, Association Traversal, Query Execution, Qualifier Declaration.

Basic Read is a set of operations, which allow to get CIM classes, Enumeration of subclasses of CIM classes in target Namespace. Also to the set of Basic Read group operations belongs operation for reading instance, enumerate of Instances, enumeration the names of instances and extraction the value of property from CM instance

To Basic Write operation group belongs operation of setting appropriate value to the property.

Instance Manipulation operations allow to create, modify and delete instances of CIM class

Schema manipulation presents the set of operation for manipulation with class(create, modify, delete).

Association Traversal is a group, which include operations for enumeration CIM Objects (Classes or Instances) or names of CIM Objects that associated of refer to particular source CIM Object.

Query Execution is a group includes operation for execution query against the target Namespace.

Qualifier Declaration is a group of operations, which takes care about Qualifier. In this group of operation presented set of operation for retrieving, creation, updating, deleting qualifiers and enumeration Qualifier declarations from the target Namespace.

4.4.3 COPS

COPS technology supports only atomic operations on complete table entries. The list of COPS operations presented in Table 4.3. The interaction between PEP and PDP begins with opening connection and security negotiation. In case, if the negotiation was failed the connection will be closed via Client-Close. The next important operation is key

maintenance. This operation allows negotiating the key, which will be using. In the connection was open successfully begins interaction between server and client. The request new policy from PDP, PEP sends Request. At all time the PEP module is expected to abide by the PDP's decisions. After receiving and installation requested information PEP should notify the PDP of any state (Provisioning model). During all time the connection verified via Keep-Alive operation.

Table 4.3 – Set of operations which allow executing COPS technology

Operation name	Description
Client-Open	Open connection
Client-Accept	Accepting connection
Client-Close	Closing connection
Request	Request for policy information
Decision	Response from PDP for PEP request
Report State	Notifying PDP about policy state
Delete Request State	Removing policy information from PEP
Synchronize State Req	Synchronisation request
Keep-Alive	Operation for validation connection
Synchronize Complete	Successful synchronization message

The set of operation, which COPS propose is full and enough for DiffServ network. The operations cover the whole functionality that needs DiffServ network. COPS framework proposes operations for pushing and removing policy from PEP and reporting to PDP about results. There are operation for opening and closing connection between PEP and PDP. Such additional operation as Keep-Alive allows to make a network management information more reliable. The mechanism of policy synchronization between PDP and PEP realize via special operations.

4.5 Simplicity

In the previous chapters we already analyzed a lot of network management framework characteristics. In this chapter we are going to consider simplicity characteristic. The factor, which guaranty of perspective using new technology is application this technology to the real life tasks. There are a lot of examples, which prove that in addition to the wise and intellectual solution each technology should be adapted to the real life tasks. Another way it could not become popular and widely used. The simplicity is factor, which indicate how difficult to create software application using appropriate technology. During research simplicity characteristic we consider already exist solution for fast producing the efficient software based on appropriate technology.

4.5.1 SNMP

SNMP is a longer exist technology. During this time people use it and create utility and other sort of software. This standard became a legend of network management domain. Nowadays, there are a lot of software developer kits for developing SNMP applications. There are a different solutions and libraries for different programming languages and different platforms.

One of the proposed software developer kits is DynamicSNMP SDK from Monfox company. The DynamicSNMP Software Development Kit (SDK) is a Java toolkit for the rapid development of agent and manager applications for SNMPv1, SNMPv2 and SNMPv3. It is specifically tailored to meet the needs of both hardware and software companies in today's competitive marketplace.

This development kit proposes really useful and easy way to create software application using Java programming language.

Another Java SDK for creating network management applications solution provided by jSNMP Enterprises company.

jSNMP Enterprise provides application developers a cross-platform means of communicating with SNMP devices and services. It is written entirely in Java and is completely portable. jSNMP provides complete SNMP v1/v2c/v3 support including trap/inform handling and authentication and privacy password alteration.
A traditional Java SNMP SDK, Java SNMP API, or SNMP Java library requires the user to manually construct SNMP request packets. The jSNMP interface, however, allows the user to communicate with network devices by specifying the Object Identifier (OID) of interest rather than worrying about the intricacies of SNMP, such as the Basic Encoding Rules (BER) or Protocol Data Unit (PDU) format.
jSNMP has been optimized for minimizing network traffic and maximizing efficiency, allowing for a high degree of scalability. In addition, the package has been optimized for use with multiple simultaneous connections, allowing for a 3-tier model.
To support a 3-tier model, jSNMP includes the tools necessary to create distributed network management applications using RMI and CORBA. The core jSNMP engine can be run as an RMI server. Client applications can be written to use either the RMI interface or a remote version of the easy-to-use jSNMP interface. Either way, lightweight clients, running in or out of browsers, can be created using a single back-end jSNMP service.
Running jSNMP as an RMI server provides a number of distinct benefits. All communication with managed devices flows through the service. This reduces the load placed on the devices, and allows the service to perform several optimizations. For instance, requests bound for the same managed device can be packaged together in one PDU, even if the requests came from different clients. Also, values are cached for a user-defined period of time. If a request is received within the defined time window, a cached value is returned. This removes the need for additional network traffic. Better security, easier administration, and improved trap handling are also possible in a 3-tier model.
The use of a robust RMI server, distributed capabilities, and high-level SNMP Java interfaces, combine to create a powerful combination that will save SNMP developers both time and money.

Since SNMP OIDs are difficult to remember, developers may prefer to make a SNMP request with the name associated with an OID instead of the OID's dotted decimal notation

(e.g., ifAdminStatus.1 instead of 1.3.6.1.2.1.2.2.1.7.1). jSNMP Enterprise includes the powerful Java MIB compiler jMIBC, which takes a series of MIB files and produces a dictionary that is used by the jSNMP Enterprise SnmpMIBService and SnmpMIBDictionary classes to translate OIDs to/from common names, to retrieve an OID's status, access, type, abstract type, and description, and to translate OID enumerated values.

4.5.2 CIM

The Solaris WBEM SDK was proposed by Sun Company. This SDK helps to realize network management application on Solaris platform.

The Solaris WBEM SDK is a set of APIs that contain the components necessary to write management applications. These applications communicate with WBEM-enabled management devices using XML and HTTP communication standards.

Solaris WBEM applications request information or services from the Common Information Model (CIM) Object Manager through the WBEM APIs. These APIs represent CIM objects as Java classes. You use the APIs to describe managed objects and to retrieve information about managed objects in a system environment. The advantage of modelling managed resources by using CIM is that those objects can be shared across any system that is CIM-compliant.

One more interesting open source project is Pegasus. Open Group executes it.

The goal of this project is to implement the DMTF CIM and WBEM standards. It is designed to be portable and highly modular. It is coded in C++ so that it effectively translates the object concepts of the CIM objects into a programming model but still retains the speed and efficiency of a compiled language. Pegasus is designed to be inherently portable and builds and runs today on most versions of UNIX, Linux, and Microsoft Windows.

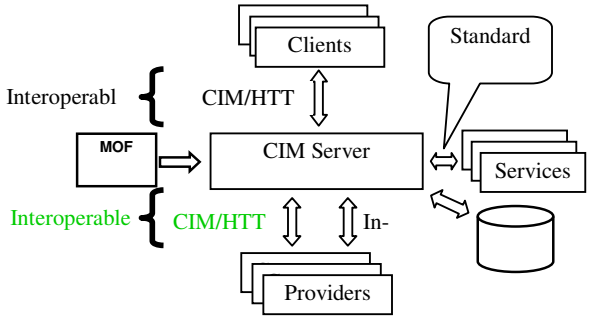

Figure 4.5 – Architecture of Pegasus

The CIM server of Pegasus is implemented in C++. CIM Provider may be implemented in either C++ or Java. The realization in Java is possible because Pegasus contains and embedded JVM. Pegasus provides the set of API for realization client. The realization could be performed by C++ or Java programming language. The Pegasus realization could be executed performed on different types of platforms, such as: Linux, Unix, Windows, Solaris, AIX, HP. The Pegasus realization support all set of operations over HTTP. This fact makes this realization to be widely adopted.

4.5.3 COPS

COPS is the latest among considered network management solutions. But even for COPS Framework is proposed wide variety of solutions. The interesting proposition presented by Intel Corporation. It proposes COPS Client Software Development Kit. The Intel COPS Client SDK enables network device manufacturers to implement a standards-based approach for dynamically configuring equipment such as routers, switches and traffic management devices.

COPS (Common Open Policy Service) Client Software Development Kit (SDK) consists:

- Source code for PIB (Policy Information Base) CIG (Control Interface Generator)
- DiffServ QoS (Quality of Service) PIB Client API (Application Programming Interface)
- COPS-PR Client API

- COPS RSVP Client API
- COPS Client API
- Portability Layer API
- COPS SDK Server Simulator Binaries for testing on Linux, Solaris, and Win32 systems
- Documentation: COPS Client SDK API Reference Guides

The SDK provided by Intel proposes not only create client application and test it on the free of charge server but even emulate translation PIB messages. The PIB Control Interface Generator (CIG) enables network device vendors to generate the client-side PIB-specific code for obtaining policy information for any service-type defined in a PIB. Unfortunately the performance and set of operation supported by free version of Intel server don't give a possibility to test the full set of COPS operation.

4.6 Scalability

In IT terms, scalability is the ability for a computer's application to continue to perform tasks well as the number of its users is growing. An application is scalable if it can meet any future increase in user demand placed on it. Scalability is an important characteristic for network management application to have, since all users access the same copy of the application. The scalability of a business application depends on the scalability of the three underlying components of the business application: software, hardware, and network connection.

The scalability is one of the important factors in developing software application and specially network application. Software application in network management domain has to be scalable, because amount of users in network always increase and network always has to be in well operating condition. The analyses of scalability factor allows to realize how is difficult to extend system. During this analyses we get answer to the question, does each of technologies make provision for extension of storage mechanism.

4.6.1 SNMP

The SNMP is a framework that has the worse scalability characteristic. The main reason of bad scalability of SNMP framework is trying to provision all possible cases. The schema in SNMP framework was created with idea to include as many things as possible. The scheme becomes really huge and difficulty operable. That is why the system could not be able easily extended. The schem of SNMP and CIM framework have a lot of common. Thanks for object oriented approach and better think over solution helps to avoid this problem in CIM framework.

The connection between manager and agent realzed via UDP protocol. The agent could establish connection not only one manager. This ability allows making a backup connection for prevention loosing data.

4.6.2 CIM

According to scalability parameter CIM proposes solution for easy extension. The approach to designing MIB was to allow for flexibility of adding new objects. Extensions can be added to a private subtree. This allows vendors to create objects to manage specific entities on their products and to make those objects visible to management station. But with SNMP a management station can only access information for which it knows how to ask.

Extension in the CIM model involves more work than what is needed for adding another device/component/application in MIB. One has to understand the overall OO schema with the underlying design decisions involved in order to be able to extend the schema to fit his/her purposes. Understanding the different parts of the CIM model as well as the relationships/attributes/methods for each class in the different CIM models can be a substantial task. That is not to say that there will be no reward for understanding where each individual proprietary model fits. Object Oriented Design methodologies were originally invented to cope with issues of complexity and extensibility. So as the CIM model gets extended it will not become incomprehensible. Another point is that depending on what part of management you are interested in (e.g. application management), you will

only need to understand the CIM model that has to do with the specific area of interest. This OO approach into design allows for better reusability and easier understanding as models become bigger and bigger. MIB on the other hand does not allow the same degree of reusability since it does not support inheritance and so it is easier to add up with duplicated schema entries as models grow to support more vendors and more device/application types.

The physical connection between CIM architectures elements organized regarding with client server paradigm. One agent could be connected to the several CIM servers. It helps to avoid loosing connection between client and server.

4.6.3 COPS

According to the work [20], which purpose was to compare scalability and performance characteristics, COPS and one of its extensions has really good results. A low-end machine is shown to handle 800 COPS clients in a controlled way. COPS implementation is compared with a common SNMP implementation, where the former is shown to have the better performance. In the theoretical comparison between COPS-based and SNMP-based systems for policy provisioning, it is strongly suggested that COPS is technically superior, although SNMP might have economical advantages due to its well-established user base.

As a result this work has been concluded that COPS and its extension COPS-PR are indeed feasible protocols for use in policy provisioning systems. It is also technically superior to the main alternative, which is SNMP. However, the question if the technical superiority can motivate eventual increased cost in deployment of this new technology remains unanswered. Each of PEP connected only to a single PDP.

4.7 Security

If commerce over the Internet is to fulfill its potential, customers must be confident that private information (such as credit card numbers) remains secure. Likewise, sensitive strategy and financial information discussed during desktop videoconferencing or available on servers must be protected.

A secure network starts with a strong security policy, which defines the freedom of access to information. The security policy dictates the deployment of security in the network. The security services offer many technologies to choose from in building a custom security solution for Internet, intranet, and remote access networks. These scalable services seamlessly interoperate to deploy enterprise wide network security.

4.7.1 SNMP

The level of security in DiffServ applications should be enough to protect system of fails and preventing making of spoliation by someone. The need for security in SNMP is obvious because the MIB objects being communicated contain critical information about network devices. There is no person, who wants that someone “snooping” into his network to find out IP addresses network computers, or how long machines have been running, or whether the links are down, or pretty much anything else. When it comes to object write operations using SetRequest-PDU, the concerned are magnified even more: we definitely don't want strangers being able to control or interfere with our managed devices by issuing bogus commands to change MIB objects that control device operation. Let’s observe the security aspects in different version of SNMP framework. It was proposed because one of the main problem from the moment of invention this framework has a problem with security. Let’s try to find out does this problem has been changed.

Unfortunately, the security incorporated into SNMPv1 was extremely limited; it really took the form of only one policy and one simple technology:

“Weak Objects”: SNMP was created with the mindset that the MIB objects used in the protocol would be relatively weak. This means that the objects are designed so that any problems in working with them result in minimal damage. The policy of the designers of SNMP was that MIB objects that are normally read should not contain critical information, and objects that are written should not control critical functions.

So, a read-only MIB object containing a description of a machine is fine, but one containing the administrative password is not. Similarly, a read-write MIB object that

controls when the computer next reboots is acceptable, but one that tells the object to reformat its hard disk is (definitely) not.

Community Strings: All the devices in an SNMP network managed by a particular set of network management stations are considered to be in a “community”. A community string that appears in a field in the message header identifies each SNMPv1 message sent between members of the community. This string is like a simple password; the recipient will reject any messages received with the wrong string.

These security features are better than nothing, but not much. The community strings protect against obvious tampering in the form of unauthorized messages. However, the strings are sent in plain open text and can easily be discovered and then used to compromise the “community”.

SNMPv1's security was also sufficient for some users of SNMP. But in newer, larger internetworks, especially ones spanning large distances or using public carriers, SNMPv1 wasn't up to the task.

During the “evolution” of SNMPv2 variants, and eventually the creation of SNMPv3, several new security models were created to improve upon SNMPv1's security:

Party-Based Security Model: This was the security model for the original SNMPv2 standard, now called SNMPv2p. A logical entity called a party is defined for communication that specifies a particular authentication protocol and a privacy (encryption) protocol. The information is used to verify that a particular request is authentic, and to ensure that the sender and receiver agree on how to encrypt and decrypt data.

User-Based Security Model (USM): This was developed in the SNMPv2u variant and used in SNMPv2* (SNMPv2 asterisk); it eventually was adopted in SNMPv3. The idea here is to move away from tying security to the machines and instead use more traditional security based on access rights of a user of a machine. A variety of authentication and encryption protocols can be used to ensure access rights are respected and to protect message privacy.

The method relies on time stamps, clock synchronization and other techniques to protect against certain types of attacks.

View-Based Access Control Model (VACM): VACM is part of SNMPv3, and defines a method where more fine control can be placed on access to objects on a device. A view specifies a particular set of MIB objects that can be accessed by a particular group in a particular context. By controlling these views an administrator can manage which information is accessed by whom.

Security is probably the most complicated subtopic in networking, and describing these methods in detail would require dozens and dozens of topics. You can refer to the relevant standards if you want more information[44], though unless you are well read on security topics, you will likely not be able to make heads or tails out of what is written in them.

Party-based security pretty much died with SNMPv2p; USM and VACM are part of SNMPv3 and provide enhanced security for those who need it (though again, it's interesting to note how many networks continue to use SNMPv1, security warts and all.) SNMPv3 took another important security-related step in redefining the SNMP architecture to seamlessly support multiple security models. This enables different implementations to choose the security model that is best for them. USM is the default model in SNMPv3.

4.7.2 CIM

Security issues of CIM technology base on one of extension http protocol HTTPS. As far as mentioned above CIM uses http for messages interchanging. The safe transferring messages in XML format is covered by Secure Sockets Layer (SSL).

SSL is a protocol designed and implemented by Netscape Communications. Netscape claims it is designed to work, as the name implies, at the socket layer, to protect any higher level protocol built on sockets, such as telnet, ftp, or HTTP. As such, it is ignorant of the details of higher level protocols, and what is being transported. A free reference version of SSL, SSLRef, is available from Netscape. Many of the functions provided by SSL are part of the newly defined IPv6.

SSL provides for encryption of a session, authentication of a server, and optionally a client, and message authentication. The SSL Handshake Protocol and the application protocol both operate on top of the SSL Record Protocol, a simple means of encapsulating authentication information. SSL-Record Layer works on TCP or some other reliable transport mechanism. Session establishment takes from 5 to 8 messages, depending on options used. SSL relies on the existence of a key certification mechanism for the authentication of a server. SSL does not provide for renegotiation of keys within a session. This is not a problem in HTTP, but might be with other protocols. A multitude of ciphers and secure hashes are supported, including some explicitly weakened to comply with export restrictions.

4.7.3 COPS

COPS framework like other network management framework proposes security mechanisms. The COPS protocol provides an Integrity object that can achieve authentication, message integrity, and replay prevention. The Integrity object also provides sequence numbers to avoid replay attacks. Security between the PEP and PDP can be provided by IP Security (IPSec). In this case, the IPSec Authentication Header can be used for the validation of the connection, and additionally, the IPSec Encapsulation Security Payload can be used to provide both validation and secrecy. Transport Layer Security (TLS) can be used for both connection-level validation and privacy.

The Client-Open and Client-Accept messages are also used in security negotiation, if configured so. If the PEP is configured to use COPS security, the very first Client-Open message, which the PEP sends to the PDP, must have the Client Type –field set to zero. In addition to this the Integrity -object must be included. The Integrity –object contains a sequence number – field (32 bits). The PEP assigns to that field the initial sequence number, which the PEP expects the PDP to increment, when the communication continues after the initial Client-Open/Client-Accept exchange. This system is used to prevent replay attacks. Both the PEP and the PDP expect to receive a message with certain sequence number. If the sequence number of the incoming message if not expected one, the receiver sends an error message and closes the connection. There are also two other fields in the

Integrity – object: KeyID (32 bits) and Keyed message digest (96 bits). The Key ID – field is used to identify the shared key between the PEP and the PDP and the cryptographic algorithm to be used. The algorithm the COPS uses to calculate the digest is HMAC (Keyed-Hashing for Message Authentication /7/). The minimum algorithm that the COPS must support is HMAC-MD5-96 algorithm. It's HMAC employing the MD5 Message Digest Algorithm /8/. The normal 128-bit output is truncated to 96 bits Keyed Message Digest –field of the integrity object. The digest is calculated over every object and field in the message except the Keyed Message Digest – field. This is the reason why the integrity object must be always the last object in the message, if it is used. The COPS implementations must at least provide the ability to manually configure the keys and their parameters locally.

CONCLUSIONS

In this paper we have analyzed and compared network management frameworks applicable to DiffServ. All of these three frameworks were founded for the purpose of a network management. Each framework tries to manage issues with collecting and processing network information in a special way. Some issues are common, but some are not. All these peculiarities bring in their advantages and disadvantages to each technology. All of these three technologies could be successfully used in DiffServ.

The consideration and analysis are presented in the order of foundation of these technologies. The first one was SNMP. SNMP is a widely used standard, which has been used by a lot of network management systems. There is a lot of a hardware, which supports the SNMP standard. SNMP becomes the set of standards that has a lot of different versions and extensions. In the process of evolution the SNMP standard has been updated and has improved their capabilities from version to version.

During its history the SNMP standard has created a lot of applications, schemes and utilities. This fact helps it to became such widely used as it is. SNMP was the first technology, which proposed the first protocol using the SMI standard. The Structure of Management Information (SMI) standard is responsible for defining the rules for how MIB objects are structured, described and organized. SMI allows dissimilar devices to communicate by ensuring that they use a universal data representation for all management information. Unlike most protocols, which are command-oriented, SNMP is information-oriented. SNMP operations are implemented using objects called variables that are maintained in managed devices. Rather than issuing commands, a network management station checks the status of a device by reading variables, and controls the operation of the device by changing (writing) variables. The same solution has been used in the creation of CIM and COPS frameworks. The SNMP framework has a lot of different versions and extensions. All of these extensions have been made for solving problems, which this standard has had. Lack of security issues and a low scalability belong to the weak side of the SNMP framework.

The first version of SNMP has a really low security level. These problems were solved in the next version of SNMP. In SNMP v2 USM (User-Based Security Model) was added and in SNMP v3 to already existing security mechanisms VACM (View-Based Access Control Model) has been added. The second problem was made in a process of MIB creation. The structure of MIB was not worked over, that is the reason for scalability problems. In a case, if a system needs to include many devices, produced by different manufacturers, the scheme becomes heavy for processing. The variables containing the same characteristics of devices produced by different manufacturers appear in the totally different places in MIB. It takes much time to find and process this information. It could be a problem, especially in DiffServ networks with a precise restriction for searching and processing time.

These problems force to make this standard retired. However, someone else should take the place of SNMP. As a successor I see two technologies: CIM and COPS. Each of these frameworks could take this place.

The CIM-WBEM framework is a technology in which attempts to solve all disadvantages of the SNMP framework have been carried out. In a process of a design of the CIM technology the experience and knowledge of SNMP have been taken into account. So, for presenting network management information the CIM-MIB format is used. Thanks to one of the characteristics of an object-oriented approach – inheritance, the CIM framework provides good scalability characteristics. The network management information is presented in the xmlCIM format that brings in the feature of heterogeneity and makes an interaction between client and server clearer. The xmlCIM format is an extension of an XML format. Presentation of network management information in an XML format allows using HTTP as a transport protocol. The HTTPS protocol with SSL takes care about security issues in the network management applications. The CIM-WBEM framework presents the widest set of operations, which allow operating with Classes, Instances, Qualifier and even Queries. Such a wide set of operations allows to make the information processing more efficient. The processing time is especially important in the DiffServ network. The “operation over HTTP” mechanism is typical just for the CIM framework. Thanks to “operation over HTTP” the operation could be executed throw one of the extensions of the HTTP protocol.

The most perspective and promising framework is COPS, especially when applying to DiffServ networks. When developing this framework developers were trying to use experience and knowledge from the previous frameworks. They were trying to combine the best features of predecessor and to avoid the bad ones. As a mechanism for information presentation COPS uses PIB (Policy Information Base). PIB is used not only for structuring and storing data, but for presentation information, which is transmitted between PDP and PEP. PIB is based on the model of Structure of Management Information (SMI) and Management Information Bases (MIBs) when used within SMNP. Also PIB gets from CIM such feature as object-oriented paradigm. So, in PIB data are presented as a SNMP-MIB tree using an object-oriented paradigm.

Unlike systems based on SNMP, the systems based on CIM and COPS frameworks use persistent connection and delivery notification, proposed by a TCP protocol.

The paper [26] describes the mechanism for creating a network management system, which allows using different network management frameworks. The UPM (Unified Policy-Based Management) framework allows avoiding scalability problems. One more proposed solution is using the already existing network management infrastructure. Table 5.1 illustrates all characteristics and results of the comparison.

Nowadays, there are many different SDKs for realization of an appropriate network management system. This paper considered several of them. Each of the frameworks has a variety of SDKs for different programming languages and platforms.

The future of this work is practical realization a test systems using each of these frameworks. The test systems allow to accomplish the comparison by getting real characteristics from functioning systems. Also in process of developing could be revealed new aspects and features of each framework.

Table 5.1 Comparison characteristics

Name of a characteristic	**SNMP**	**CIM**	**COPS**
Information presentation	MIB	CIM-MIB, xmlCIM	PIB
Transportation mechanism	SNMP protocol (UDP)	HTTP/HTTPS (TCP)	COPS (TCP)
Set of operations	Get, Set, Trap	Create, Delete, Get, Set for (Class, Instance, Associators, Qualifier), ExecQuery	Open, Close, Accept, Request, Decision, Synchronize, Keep-Alive
Simplicity	DynamicSNMP, jSNMP	Solaris WBEM SDK, Pegasus	Intel COPS Software Development Kit
Scalability	+	++	++
Security	ver.1 weak ver. 2 USM ver. 3 USM + VACM	HTTPS/SSL	IPSec, TSL, (HMAC-MD5-96 and higher)

References

[1] Andersson L., Doolan P., Feldman N., Fredette A. and Thomas R., "Label Distribution Protocol", Internet, Nov. 1998.

[2] Baker F., Chan K., Smith A., Management Information Base for the Differentiated Services Architecture, RFC 3289, May 2002.

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

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

[5] Braden R., Zhang L., Berson S., Herzog S., Jamin S., Resource ReSerVation Protocol (RSVP) RFC 2205, September 1997.

[6] Bodin U., Schelen O., Pink S., “Load-tolerant Differentiation with Active Queue Management”, CDT, SE-971 87.

[7] Boyle J., Cohen R., Durham D., Herzog S., Rajan R., and Sastry A., The COPS (Common Open Policy Service) Protocol, RFC 2748, January 2000.

[8] Chan K., Seligson J., Durham D., Gai S., McCloghrie K., Herzog S., Reichmeyer F., Yavatkar R., Smith A., COPS Usage for Policy Provisioning (COPS-PR), RFC 3084, January 2000.

[9] Clark D., "The Design Philosophy of the DARPA Internet Protocol", ACM SIGCOMM ’88, Aug. 1988.

[10] Comerford R., "State of the Internet: Roundtable 4.0", IEEE Spectrum, October 1998.

[11] Ferrari D., Verma D., "A Scheme for Real-Time Channel Establishment in Wide-Area Networks", IEEE JSAC, vol. 8, no. 3, April 1990, pp. 368-379.

[12] Ferguson P. and Huston G., Quality of Service: Delivering QoS in the Internet and the Corporate Network. Wiley Computer Books, New York, NY, 1998.

[13] Fine M., McCloghrie K., Seligson J., Chan K., Hahn S., Bell C., Smith A., Reichmeyer F., Differentiated Services Quality of Service Policy Information Base, RFC 3317, March 2003.

[14] Floyd S. and Jacobson V., Random early detection gateways for congestion avoidance. IEEE/ACM Transactions on Networking, August 1993.

[15] Grossman D., New Terminology and Clarifications for Diffserv. RFC 3260, April 2002.

[16] Guerin R., Blake S. and Herzog S., "Aggregating RSVP-based QoS Requests", Internet draft <draftguerin - aggreg-RSVP-00.txt>, Nov. 1997.

[17] Guerin R., Peris V., Quality-of-service in packet networks: Basic mechanisms and directions. Computer Networks, 31(3):169–189, February 1999.

[18] Hahne E., Gallager R., “Round Robin Scheduling for Fair low Control in Data Communication Networks,” Tech. Rep. LIDS-TH-1631, Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Dec. 1986.

[19] Heinanen J., Baker F., Weiss W., and Wroclawski J., Assured forwarding PHB group. Request for Comments (Proposed Standard) 2597, Internet Engineering Task Force, June 1999.

[20] Liden E. and Torger A., "Implementation and Evaluation of the Common Open Policy Service (COPS) Protocol and its use for Policy Provisioning", January 2000

[21] Intel’s COPS SDK documentation,

<URL: http://www.intel.com/labs/manage/cops/index.htm>.

[22] Jacobson V., Congestion avoidance and control. ACM Computer Communication Review, 18(4):314– 329, August 1988. Proceedings of the Sigcomm ’88 Symposium in Stanford, CA, August, 1988.

[23] Jacobson V., Nichols K., and Poduri K., An expedited forwarding PHB. Request for Comments (Proposed Standard) 2598, Internet Engineering Task Force, June 1999.

[24] Jain R., "Myths about Congestion Management in High Speed Networks," Internetworking: Research and Experience, Volume 3, 1992, pp. 101-113.

[25] Knightly E. W. and Shroff N. B., “Admission control for statistical QoS: Theory and practice” IEEE Network, 13(2):20–29, March/April 1999.

[26] Law E., Saxena A., “Scalable design of a Policy-based management system and its performance”, June 2003.

[27] Lymberopoulos L., Lupu E. and Sloman M., Using CIM to Realize Policy Validation within the Ponder Framework. Presented at DMTF's 2003 Global Management Conference, San Jose, CA, USA, 16-19 Jun. 2003 (Winner of the Academic Alliance Paper Competition).

[28] Ma Q., "QoS Routing in the Integrated Services networks", Ph.D. thesis, CMU-CS-98-138, Jan. 1998.

[29] McCloghrie K., Fine M., Seligson J., Chan K., Hahn S., Sahita R., Smith A., Reichmeyer F., Structure of Policy Provisioning Information (SPPI), RFC3159, August 2001.

[30] K. McCloghrie and M. T. Rose., Management information base for network management of TCP/IP-based internets: MIB-II. RFC 1213, IETF, March 1991.

[31] Nichols K., Blake S., Baker F., Black D., Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers. Request for Comments (Proposed Standard) 2474, Internet Engineering Task Force, December 1998.

[32] Rajan R., Verma D., Kamat S., Felstaine E., Herzog S., “A policy framework for integrated and differentiated services in the internet” IEEE Network, September/October 1999.

[33] Shenker S., Partridge C. and Guerin R., "Specification of Guaranteed Quality of Service", RFC 2212, Sept. 1997.

[34] Shreedhar M. and Varghese G., “Efficient fair queuing using deficit round robin,” Proc. of ACM SIGCOMM '95 , Aug. 1995.

[35] Technical Specification from Cisco, Distributed Weighted Random Early Detection, <URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ioslI/eel11/wred.pdf>.

[36] Terzis V., Wang L., Ogawa J., Zhang L., “A two-tier resource management model for the internet.” In Proceedings of Global Internet, December 1999.

[37] The Open Pegasus group <URL: http://www.openpegasus.org/>.

[38] Vaananen P. and Ravikanth R., "Framework for Traffic Management in MPLS Networks", Internet draft <draft-vaananen-mpls-tm-framework-00.txt>, Mar. 1998.

[39] Villamizar C. and Li T., "IS-IS Optimized Multipath (IS-IS OMP)", Internet draft <draft-villamizar-isisomp- txt>, Oct. 1998.

[40] Wahl M., Kille S., Howes T., "Lightweight Directory Access Protocol (v3)", RFC 2251, December 1997.

[41] Wroclawski J., "Specification of the Controlled-Load Network Element Service", RFC 2211, Sept. 1997.

[42] Xiao X. and Ni L. M., “Internet QoS: A Big Picture”, IEEE Network Magazine, March/April 1999.

[43] Yeong W., Howes T., and Kille S., "Lightweight Directory Access Protocol", RFC 1777, Mar. 1995.

[44] Zarrington D., Presuhn R. and Wijnen B., "An Architecture for describing SNMP Management Frameworks", RFC 2571, April 1999.

[45] Zhang H., "Service Disciplines For Guaranteed Performance Service in Packet-Switching Networks", Proceedings of the IEEE, 83(10), Oct. 1995.