Application-Driven Networking: Concepts and Architecture for Policy-Based Systems

Jonathan Follows, Detlef Straeten

International Technical Support Organization

http://www.redbooks.ibm.com

SG24-5640-00
Application-Driven Networking: Concepts and Architecture for Policy-Based Systems

December 1999
Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix B, “Special Notices” on page 167.

First Edition (December 1999)

This edition applies to Version 3, Release 3 of “Common Code” for IBM router platforms: Multiprotocol Access Services, Multiprotocol Routing Services and Access Integration Services for the IBM 2216, 2210 and 2212 platforms respectively.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. HZ8 Building 678
P.O. Box 12195
Research Triangle Park, NC 27709-2195

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 1999. All rights reserved.
Note to U.S Government Users - Documentation related to restricted rights - Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.
# Contents

**Preface** ...................................................... vii
The Team That Wrote This Redbook ...................................................... vii
Comments Welcome ...................................................... viii

**Chapter 1. Introduction to Application-Driven Networking** .............. 1
1.1 The Role of This Book ........................................... 2
1.2 Policy-based Management Approach ................................... 3
1.3 Today’s Configuration and Provisioning Mechanisms .................. 6
1.4 Standardization Efforts ........................................... 8
1.4.1 Directory Enabled Networks (DEN) ............................ 8
1.4.2 Distributed Management Task Force (DMTF) ................... 9
1.4.3 Internet Engineering Task Force (IETF) ....................... 10
1.5 Policy-Based Networking Architecture ........................... 11
1.5.1 System Overview ............................................. 12
1.5.2 Administrative Domain Components ........................... 13
1.6 The Overall System View ......................................... 14

**Chapter 2. Policy Enforcement Point** ................................... 17
2.1 IETF Policy Framework ............................................. 18
2.1.1 Policy Enforcement Point (PEP) ............................... 19
2.1.2 Placement of Policy Functions in a Network .................... 20
2.1.3 Example Scenarios ............................................. 21
2.2 The COPS Protocol and the Definition of PIBs .................... 23
2.2.1 Why not SNMP? ............................................... 23
2.2.2 Policy Information Base (PIB) ................................ 25
2.2.3 Details on PEP-PDP Interaction Using COPS .................. 27
2.3 IBM Common Policy Engine (CPE) .................................. 28
2.3.1 CPE System Structure ......................................... 29
2.3.2 Solving the Rule Look-Up Problem ............................ 31
2.4 The Policy Feature in the IBM 221x Router Product Family ....... 33
2.4.1 Policy System Structure ..................................... 34
2.4.2 Policy Objects ................................................. 35

**Chapter 3. Policy Server, including Policy Decision Point** ............. 43
3.1 Policy Framework .................................................. 44
3.2 Policy Rules ........................................................ 46
3.2.1 Policy Trigger and Rule Location ............................. 46
3.2.2 State and Resource Validation ................................. 46

**Chapter 4. Policy Repositories** ........................................ 49
4.1 LDAP Directories .................................................. 50
4.1.1 X.500: The Directory Service Standard ........................ 50
4.1.2 LDAP Server Scenarios ........................................ 51
4.1.3 Directories and Networks ..................................... 52
4.2 Information Models for Policies ................................... 53
4.2.1 LDAP Information Model ...................................... 53
4.2.2 The IBM LDAP Schema ....................................... 55
4.3 Common Information Model for Policies (CIM) ....................... 56
4.3.1 CIM Policy Information Framework ............................ 57
4.3.2 Policy Classification .......................................... 59
4.3.3 Class and Relationship Hierarchies ......................... 60
4.4 Policy Framework Core LDAP Schema ........................................ 62
  4.4.1 Mapping CIM to LDAP .................................................. 63
  4.4.2 Resulting LDAP Directory Structure for Policies .................... 64
  4.4.3 Accessing the LDAP Directory ........................................ 65
4.5 IBM LDAP Directory and Policy Database Interaction ................. 68
  4.5.1 LDAP Overview ....................................................... 68
  4.5.2 Policy Schema ....................................................... 70
  4.5.3 Policy Rules Configuration ........................................ 72

Chapter 5. Configuring Policies ................................................. 75
  5.1 CPE Policy Definition Tool ............................................. 76
    5.1.1 Policy Configuration Overview .................................. 76
    5.1.2 Tool Methodology ................................................ 77
    5.1.3 Configuration Task for Policies ................................. 81
  5.2 System and Network Management ........................................ 83

Chapter 6. User, Application and Implementation Scenarios .............. 85
  6.1 Policy-based QoS in End Systems (OS/390, Windows, etc.) .......... 86
  6.2 CS/390 PEN Implementation ............................................ 87
    6.2.1 Policy Control on CS for OS/390 Release 7 ....................... 88
    6.2.2 Policy Control on CS for OS/390 Release 8 ....................... 90
  6.3 Policy Activities on Windows 2000 ..................................... 92
  6.4 Virtual Private Network and QoS Scenarios ........................... 94
    6.4.1 Configuring Policies via Command Line Interface ................ 94
    6.4.2 Configuring the LDAP Client ..................................... 104

Chapter 7. Putting It All Together ........................................... 109
  7.1 Policy-Controlled QoS and VPN Scenarios ............................. 110

Appendix A. Standards Documents and Sample Files ........................ 117
  A.1 IETF Documents ....................................................... 117
  A.2 DEN Documents ....................................................... 118
  A.3 DMTF Documents ...................................................... 118
  A.4 LDAP Directory Initialization ......................................... 118
    A.4.1 Config Files ....................................................... 119
    A.4.2 Including the Objectclass and Attribute Files in the LDAP Server 120
    A.4.3 Adding and Modifying Policy Entries to LDAP Server via LDIF .... 120
    A.4.4 Configuring the Router to Read Policy Information from LDAP Server 121
    A.4.5 Resetting the LDAP Configuration and Policy Database ............. 122
  A.5 Example LDIF and Schema Files ....................................... 123
    A.5.1 IBM Schema File .................................................. 123
    A.5.2 LDIF File with Example Policy Templates ........................ 152
    A.5.3 LDIF File with Example Policies ................................ 160
    A.5.4 Example Schema File ............................................ 164

Appendix B. Special Notices .................................................. 167

Appendix C. Related Publications ............................................. 169
  C.1 International Technical Support Organization Publications .......... 169
  C.2 IBM Redbooks Collections ............................................ 169
  C.3 Other Publications ................................................... 169
  C.4 Referenced Web Sites ................................................ 170
Preface

This book describes the standards and IBM's implementation of application-driven networking, a term for an architecture and implementation in which network services are driven by rules, global policies and by the applications themselves. This architecture turns the old concept around; applications used to have to conform to the limitations of the network, whereas now networks can be made to conform to the requirements of the applications.

This redbook will help you to understand the overall architecture and concepts of policy-based system control. It covers the standardization aspects as well as IBM-specific implementation concepts. A rather large part of the book is dedicated to policy-based control of communication infrastructures, since networks are today's spinal cords of e-business environments. Policy-based application control is just beginning to evolve. This book covers some initial steps taken in the OS/390 server environment as well as Web traffic control.

---

Fast Start

For those readers who need a rapid understanding of how to configure these new router features immediately, see 2.4, “The Policy Feature in the IBM 221x Router Product Family” on page 33, 6.4, “Virtual Private Network and QoS Scenarios” on page 94, Appendix A.4, “LDAP Directory Initialization” on page 118 and Appendix A.5, “Example LDIF and Schema Files” on page 123.

Do not forget to come back and read the architectural background, since it will help you to understand the much more complex overall picture.

---

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Raleigh Center.

Jonathan Follows is a Networking Specialist at the International Technical Support Organization, Raleigh Center. He writes redbooks on all areas of IBM networking, most recently on ATM and related topics. Before joining the ITSO in 1998, he worked as a technical specialist providing sales and marketing support in the United Kingdom and has 15 years experience in all types of networking. Jonathan read Mathematics at Oxford University, England, and holds a degree in Computing Science from London University, England.

Detlef Straeten is a Certified Consulting I/T Architect in Heidelberg, Germany. He has 10 years experience in network design, implementation and operation. He has worked for IBM Global Services for seven years, including some time spent working for IBM’s Networking Hardware Division. His areas of expertise include campus high speed networking technologies as well as WAN technology topics like voice/data integration. He spent some time on different residencies in IBM research Yorktown (USA), IBM NHD Lab La Gaude (France) and IBM ITSO Raleigh (USA). He is an elected member of the IBM Technical Expert Council (TEC CER), a cross organizational group of technical people in IBM Central Region Europe.
Thanks to the following people for their invaluable contributions to this project:

Martin Murhammer
International Technical Support Organization, Raleigh Center

Shawn Walsh, Gail Christensen
International Technical Support Organization, Raleigh Center

Ed Ellesson
IBM TCP/IP Technology Group

Paul Carter
IBM Networking Hardware Division

Bruce Dillon
IBM Tivoli Systems

Leo Temoshenko
IBM Tivoli Systems

Skip Booth
IBM Networking Hardware Division

Ed Sanchez
IBM Networking Hardware Division

Lap Huynh
IBM CS/390 Design

Silvia Gruben
Heidelberg, Germany

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

• Fax the evaluation form found in “IBM Redbooks Evaluation” on page 185 to the fax number shown on the form.

• Use the online evaluation form found at http://www.redbooks.ibm.com/

• Send your comments in an internet note to redbook@us.ibm.com
Chapter 1. Introduction to Application-Driven Networking

Information technology has evolved: centralized, homogeneous systems have been replaced by distributed processors connected by large and widely disparate networks. The number of users within a corporate network has increased while Web technologies further extend the reach of the network. New and increasingly sophisticated applications require increased server capacity and network bandwidth; collaborative applications enable users to form virtual teams of people residing in remote locations; users initiate large data transfers at unpredictable times during the day.

To create e-business opportunities, companies connect to the Internet to transmit information and to enable product sales. Unfortunately, both connecting IT resources to the Internet and transporting data across it expose a corporation to security violations. To combat unauthorized server access, IT managers install firewalls. To prevent compromising data from traveling across the Internet, IT managers turn to virtual private networks (VPNs), which encrypt data before transmitting it. But what about performance?

Despite their tremendous popularity, IP networks can become hopelessly clogged as traffic increases, causing response times to vary widely, access to be denied and message loss. Nonetheless, IP transport has become a ubiquitous, communication and e-business solution.

To ensure successful e-business ventures, predictable access to business-critical applications is mandatory. But predictability is increasingly difficult to provide when employees work from their homes and when workgroups consist of employees from multiple locations. Corporate acquisitions contribute to predictability problems as IT infrastructures are combined, exposing divergent networking technologies and multiple vendors.

Policies or corporate rules are the means of defining the overall intended behavior of the IT infrastructure.

Application-driven networking (ADN) is the term IBM uses to describe a new concept and architecture to control and provision network node configurations and behaviors based on specific device rules and global policies. ADN involves network hardware components, software modules, communication protocols, database systems and global system- and network-management aspects all at the same time. This makes application-driven networking one of the more interesting new topics in the networking arena both inside and outside IBM.

Note

"Application-driven networking" was sometimes also referred to as "Application Virtual Networking". Since the abbreviation AVN turned out to be very common for "Adult Video News" on the Internet, IBM decided to go with application-driven networking (ADN or AppDrvN).

Application-driven networking is positioned as IBM's approach for policy-based system control in networking. As you can imagine there are a reasonable number of approaches discussed right now in magazines and on the Internet. Cisco's
Application-driven networking allows IT managers to establish network policies directing networks to supply what applications require to perform effectively. Application-driven networking enforces policy decisions established, or driven, by the applications, helping to simplify IT infrastructure management.

1.1 The Role of This Book

Network control and provisioning are the most challenging aspects of modern communication infrastructures. New applications are being installed over time to fulfill the need for higher integration, more scalability and lower cost of ownership, which means a general increase in profitability. Policy-based control systems are a new way to tackle this subject.

Internet technologies - the associated protocols and formats as well as the way people interact with each other in virtual teams or communities - are driving changes in network design and implementation. Quality of service is no longer “nice to have”: it is the basis for all intranet/extranet realizations.

The second major difference in the usage of Internet technologies or transport infrastructures besides QoS is the need for secured communication. Security is a corporate issue covering areas from ethical aspects to authentication and encryption technologies. Users, businesses and service providers are all looking for ways to make their communication paths safe and reliable.

As new technologies in the areas of security and QoS are evolving rapidly it has become clear that the existing network and system management systems are not very well suited to cope with automated and self-learning requirements in user and application-driven environments. There is a need for something between (a) the more offline/remote controlled oriented network and systems management installations and (b) the command line interfaces, handled with high sophistication by very few specialized people inside the company. What we are looking for is an application/user aware provisioning and controlling system which integrates the different abstraction levels of business needs.

Figure 1 on page 3 gives an overview of the typical evolution discussed in today’s customer e-business projects. The three major concerns customers are discussing these days are typically:

1. Quality of Service
2. Security
3. Value Proposition

Value Proposition is the only one which will not be discussed further in this book.

![Figure 1. e-business Evolution Path and Challenges](image)

Business needs are typically defined in **enterprise-wide policies**. Policies are rules which describe the overall company strategic goals as well as derived business/user/application specific behaviors. These intended behaviors have to be engineered at different places in the existing communication infrastructure. The systems and protocols needed to enforce the intended policy-based behaviors are called “policy aware”.

This book contains a description of the most relevant implementation scenarios and policy-based realization steps. These business scenarios will cover both the provisioning of so called Virtual Private Networks (VPN) and the implementation of different traffic classes in IP transport systems.

### 1.2 Policy-based Management Approach

Defining, instituting and managing policy for a large IT infrastructure can be a complex undertaking. Policy complexity can be reduced by adopting a global policy management approach.
Policy-based management means combining all of the procedures and protocols that enable system resource control so that they support business objectives. It begins by understanding corporate policy and then extending that policy to IT resources. Once corporate policy is defined, typically by the chief information officer, the policy is translated into the languages and rules of the systems and network management tools. Figure 2 shows the hierarchical nature of policy definition steps.

Large networks designed to support e-business may only be created efficiently with policy-level configuration. Device-by-device rule establishment is both very time consuming and prone to errors. We estimate that it would take eight years to create VPN tunnels for a 1000-node meshed network using a typical command line interface. Furthermore, “box-level” configurations are likely to introduce policy lapses (for example, backdoors) that compromise policy.

Enforcing policy entails more than managing security, and successful policy enforcement clearly requires elevating and automating the management process. Policy-level management should allow IT managers to specify network performance using several basic parameters. Typical service level agreements (SLAs) include application response times, network availability, notification of security breaches and help desk response times.

Network performance at this definition level is easier to understand, easier to measure and independent of network infrastructure. Corporations that provide SLAs and those that outsource their networks to service providers use these meaningful measurements of policy management to verify fundamental network performance (that is, 99.999% availability, etc.).

Networking strongly influences today’s computing applications. Today, IT managers most often create policies for groups of users rather than for individual users. Typically, these user groups are involved in similar activities and require similar resources. Often, it is the access to specific applications that IT managers use to define the user group.
This application-driven policy concept is not new. For years, corporations have insisted that mission-critical applications be given top priority. Banks give priority to front-office applications such as checking and automatic teller machine management; finance gives priority to the payroll application; sales makes order entry a top priority.

Specifying policy based on the requirements of the application is in accord with corporate objectives and has the potential to simplify management. This approach can also provide significant cost savings. If, for example, employees who download Web-based information are negatively impacting a mission-critical application such as order entry, remedy could either come from the purchase of additional wide area network (WAN) bandwidth or traffic prioritization.

Traffic prioritization based on application would mean order entry traffic would be allocated all the WAN bandwidth it required before allocating any remaining bandwidth for Web downloads. Historically, this prioritization — generally called class-of-service across the WAN — ensures acceptable performance for mission-critical applications without the need to purchase “unlimited” bandwidth.

### COS and QoS

The terms “class of service” and “quality of service” are often used interchangeably, as if to imply that they mean the same thing. Depending on the context, this may or may not be true. If a differentiation between the two terms has to be made, then consider the following definitions:

- “Class of service” refers to differentiating network traffic so that the different traffic classes can be treated differently. This treatment will be one in which one class is given “higher” or “preferential” treatment over another - higher queueing priority, for example.
- “Quality of service” refers to an identifiable metric, such as an absolute time value for transit delay across a network.

The end result may be that a “class of service” mechanism is used to differentiate between types of network traffic, and then a specific “quality of service” is then applied to one of the resulting service classes.

To realize both security needs and QoS management, policy-based networking depends on new Internet technologies such as IPSec and ISAKMP (for security) and on DiffServ and RSVP (for QoS). For further details on the implementation aspects of these technologies see:

- *Application Driven Networking: Class of Service in IP, Ethernet and ATM Networks*, SG24-5384
Application-driven networking is more than a single, centralized decision point at which IT managers make all policy decisions from a single, system-level interface. Application-driven networking's intention is to separate the needs of the application from the capabilities of the network infrastructure.

Simply stated, networks should not be allowed to dictate policy by default. If network connectivity or security limitations are permitted to determine corporate policy, the network — rather than the needs of the corporation — define policy. With application-driven networking, the desired network attributes the application requires are first defined and then compared to the physical network. The network infrastructure is then “adjusted” or augmented with VPNs, Quality of Service (QoS), Differentiated Services (DiffServ) and other services required to support the application.

Separation of the physical network from the logical requirements of the application should enable IT managers to get back to policy management basics such as determining who may access applications, and from where and when will they be allowed access. Decisions regarding application priority needs, security levels and access can be made without regard to infrastructure. For example, a payroll application might require “Class B” priority, “Class A” security and “Class E” access. Requirement tables could be generated for each policy aspect and network infrastructure.

<table>
<thead>
<tr>
<th>Class</th>
<th>Local area network</th>
<th>Dial-in access</th>
<th>ISP access</th>
<th>Internet access</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class A</td>
<td>DES</td>
<td>3-DES</td>
<td>3-DES</td>
<td>not allowed</td>
</tr>
<tr>
<td>Class B</td>
<td>none</td>
<td>DES</td>
<td>3-DES</td>
<td>3-DES</td>
</tr>
<tr>
<td>Class C</td>
<td>none</td>
<td>none</td>
<td>none</td>
<td>DES</td>
</tr>
<tr>
<td>Class D</td>
<td>none</td>
<td>none</td>
<td>none</td>
<td>none</td>
</tr>
</tbody>
</table>

This example clearly points out that all traffic does not require the same level of security, that the encryption used to provide security can vary based on network structure and that costs can be closely controlled. It also points out that corporate policy can simply preclude certain access (such as to the Internet) or not require any security augmentation (for example, for Class D traffic).

To make this conceptual view happen, application-driven networking relies on technology being implemented in different places of the communication infrastructure. As of today, policies related to infrastructure security and performance have the largest impact on simplifying network control. Performance and security policies build the basis for all our discussion throughout this book. Application-driven networking will be a complementary mechanism to the existing configuration and management tools available in the different product lines. What role a policy system can play in the overall system management environment in the long run is of further study in different parts of IBM and not within the scope of this book.

### 1.3 Today's Configuration and Provisioning Mechanisms

Typically there are several options to configure or influence the behavior of a network and its components. Besides management via SNMP (Simple Network
Management Protocol) most of the configuration tasks are performed today either online via a command line interface accessing the individual device or via an offline configuration tool. The offline tool enables the administrator to generate a definition file which is finally downloaded into the individual nodes.

IBM has recently made a number of efforts to harmonize configuration tool development across product lines.

Initially in 1997 there was an initiative called PAC (Policy Administration and Configuration) in which the goal was to define a global corporate-wide configuration tool across IBM products. This goal soon turned out to be a very aggressive one.

The first repository-based approach in IBM was LIDIA. LIDIA’s goal was to provide a raw LDAP-based (Lightweight Directory Access Protocol) repository interface to store schemata and related object information. This code is part of the current SecureWay Directory Software.

All of these activities were proprietary in nature in one way or another.

Since 1998 the IBM Nways Manager Development group has been working on a policy definition tool supporting the new functions and features controlled by policies in IBM Common Router Code MRS/MAS/AIS V3.3. The Nways Network Management tool started with VPN monitoring tools, which is currently shipped and will develop over time to a standards-based Policy Definition Tool called the Nways Policy Manager for QoS and VPN management.

During 1999 the Nways Manager Development group was integrated with Tivoli Systems, a business unit in IBM dealing exclusively with all problems of network and system management. Tivoli Systems is currently also looking into a more generic platform for policy management based on the new Tivoli Millennium Framework.

The Nways Policy Manager will be a first release of policy-based management for IBM networking hardware components. The Nways Management Platform, on which the Nways Policy Manager resides, has already been available for some time. We expect to control policies in networks with of several hundred of nodes using IBM-specific policy schemata in the first version.

Tivoli Systems is looking for a more general multi-vendor strategy to support policy definitions which reflect the install base of current customer environments. The Nways Policy Manager function will be integrated over time into the Tivoli Millennium Framework: GEM (Global Enterprise Manager) and SAM (Service Agreement Manager).
Very often policy-based management is viewed as just an additional way to configure a system remotely. This book will describe a much wider approach to system control based on the end-to-end modelling of I/T infrastructure using policies.

Since this area of technology is at its starting phase we will often refer to drafts or standards currently under development.

1.4 Standardization Efforts

Different standardization organizations have been discussing the changing need for end-to-end control of I/T infrastructures for several months. Although initially trying to develop a system management API (Application Programming Interface) including an appropriate information model, the DMTF (Desktop Management Task Force) has shifted one of its focus areas to Directory Enabled Networks (DEN).

The “DEN Ad Hoc Working Group” was actually initiated by Cisco and Microsoft at the end of 1997 without any relation to the DMTF. Since the DMTF has much broader vendor support and a settled agreement process between parties, the “DEN Ad Hoc Working Group” goal became part of the DMTF, renaming itself as the “Distributed Management Taskforce”. At the same time the IETF (Internet Engineering Task Force) has initiated several different working groups dealing with VPN (IPsec, ISAKMP), QoS (IntServ, DiffServ) topics and networking policy frameworks.

Today both the DMTF and IETF are working closely together in generating the specification base for policy-based networking. The current focus of activities is mainly QoS related; VPN policy-based control is rather unique to IBM today.

1.4.1 Directory Enabled Networks (DEN)

The Directory Enabled Network (DEN) initiative and related specification work is an effort to build intelligent networks and networked applications that can associate users and applications with services available from the network according to a consistent and rational set of policies.
DEN defines a directory as a centralized repository that coordinates information storage and retrieval, enabling other data- and application-specific repositories to be united. Eventually, intelligent network applications will be able to make use of appropriate information about the network and the services that the repository offers on behalf of its users and the particular context that the application is running in.

The DEN Ad Hoc Working Group had over 70 member companies contributing to the development of the DEN spec. The final edits to the DEN specification were finished in late June 1998 before its submission to the DMTF. The DEN work is being folded into the CIM Networks working group of the DMTF. The Networks Working Group will produce a CIM Network Schema and a DEN Extension Schema. The DEN Extension Schema contains optimizations that are necessary and specific to directory services, while the CIM Network Schema is an information model that is not specific to any particular implementation. The fundamental purpose is to create a standard information model and schema which enables vendors to provide interoperable network services.

For example, when a subscriber of an ISP asks for a service, that service must be delivered in an end-to-end fashion. This implies that the data using that service will span multiple vendors’ network devices. DEN is a template for exchanging information that enables each vendor to have a common definition of the service while maintaining the ability to implement that service in their own way (thereby providing added value).

1.4.2 Distributed Management Task Force (DMTF)

The “Distributed Management Task Force” initially named “Desktop Management Task Force” was founded as a vendor consortium to develop a Common Information Model (CIM) for all kinds of desktop- or workstation-related management information. The information model covers all kinds of information from application administration to operating system control.

The information stored in an SNMP MIB can be seen as one source of information which is represented in a CIM entry. CIM is a conceptual information model that is not bound to a particular implementation. This allows for the interchange of management information between management systems and applications. This can be either “agent-to-manager” or “manager-to-manager” communications which provides for distributed system management.

The DMI (Distributed Management Interface) was one of the first results of the DMTF activities. DMI generates a standard framework for managing and tracking components in a desktop computer, notebook or server. In IBM, Tivoli is adapting to this model. The current implementation of a Tivoli agent on your workstation can be seen as an early implementation of a proprietary nature to model information describing the status of an end system. This proprietary nature has changed over time. Information on the current standards-compliant product implementation of DMI can be found at:

http://www.dmtf.org/memb/list?info=product

You will also find IBM-related DMI certified product implementation references at this location.
**CIM (Common Information Model)** consists of two parts, the language to define the model and the model description itself.

Namely:

- 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 **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 formal definition of the CIM schema is expressed in a Managed Object File (MOF) which is an ASCII file that can be used as input into an MOF editor, parser or compiler for use in an application. The Unified Modeling Language (UML) is used to portray the structure of the schema. Some readers may already be familiar with an SNMP MIB browser or ASN.1 compiler, both of which have a similar concept.

In CIM, networking components were not initially part of the specification scope. This changed, as already indicated above, when the “DEN Ad hoc working Group”, initially started by Microsoft and Cisco, made a submission of the DEN Specifications to the DMTF and had them accepted.

Today there are two groups in the DMTF dealing with the overall subject:

- The Service Level Agreement (SLA) group dealing with representation of SLA policies and their representation in CIM. This group was headed by Ed Ellesson, IBM.
- The network modelling group led by John Strassner, Cisco (former DEN chair).

**1.4.2.1 DMI and SNMP Management Frameworks**

The Distributed Management Interface (DMI) and the Internet-standard Network Management Framework, commonly known as “SNMP Management Framework”, are standard management frameworks widely deployed to manage computer systems and network devices respectively. The two frameworks are similar in concept and function. However, while the two frameworks may co-exist on the same system, the two are not inherently interoperable.

Despite this, applications that span the heterogeneous nature of system and network management must access management information using both frameworks.

**1.4.3 Internet Engineering Task Force (IETF)**

As mentioned earlier the IETF (Internet Engineering Task Force) was not part of the DEN and DMTF efforts in defining policy-based control mechanisms. Nevertheless it was always the plan to move the DEN activities over to the IETF. Since the IETF was not, at that time, actively considering policies or schemata there was no working group specifically dealing with them.
Parallel QoS approaches developed inside the different IETF working groups (DiffServ, IntServ). The RSVP Resource Allocation Protocol group (RAP) was developing a protocol called COPS (Common Open Policy Service) to be able to perform admission control for certain RSVP requests outside individual routers. COPS was viewed mainly as a rather simple protocol to support “yes/no” decisions; nobody was actually talking about policy schemata using COPS at that time.

The are current efforts to enhance the COPS protocol to support DiffServ’s more complex scenarios (compared to RSVP’s). This effort is called RAP (Resource Allocation Protocol).

The IETF suddenly realized that information representation and admission control had became a more important part of their work. Today there is a Policy Framework group in place trying to develop policy schemata in a similar manner to the way the IETF has developed MIB specifications and the associated information model in the past. There is even something already defined which is called PIB (Policy Information Base) where a certain type of policy information is stored.

The main focus for the development of schemata are IPSec and QoS. The initial purpose of the RAP working group was to establish a scalable policy control model for RSVP. The working group will specify a protocol for use among RSVP-capable network nodes and policy servers. In addition, the working group will define usage directives for use of the COPS (Common Open Policy Service) base protocol to support policy information exchange transactions within the framework being standardized in the Policy Framework working group. This will evolve over time also in the definition of COPS extensions for VPN and DiffServ.

Current plans show the intention to bring both groups (RAP and Policy Framework) under one umbrella: the Operations and Maintenance group inside the IETF, since this group is dealing with configuration issues like those solved by SNMP V1/V2/V3.

This final observation refers back to the discussion above on the relation of policy-based control to offline configuration mechanisms.

Future Outlook
Since the IETF Operations and Maintenance group was dealing in the past with different versions of SNMP (V1, V2, V2*, V3) we may ask the question:

Is there something like SNMP V4, a SNPP Simple Network Policy Protocol on the way? Or is COPS already the answer? Time will tell - we will have to wait and see.

1.5 Policy-Based Networking Architecture

The following section gives a high-level overview of the different environments and components which constitute an overall policy-based solution. The definition of administrative domain and the associated policy scopes are rather intuitive and reuse a common understanding of the term domain in the Internet world (SNMP domain, routing domain, etc.).
1.5.1 System Overview

Networks and the configuration and provisioning tasks related to them are typically divided into different geographically dispersed areas. These areas are defined based on the distribution of the administrative control responsibilities. One representative separation of control responsibilities is that between a service provider and associated customer sites. Figure 4 on page 12 shows an overview of three interconnected administrative domains. The middle domain could represent a service provider domain attached to two different user domains. Each domain consists of a number of communication devices contributing to the establishment of a connection between a source and a destination of a user session.

![Diagram of Policy-Based Networking Environment](image)

Figure 4. Policy-Based Networking Environment

A user may require the establishment of a connection with appropriate security which also provides certain QoS (Quality of Service) characteristics. Depending on the path the data traffic is using to reach the appropriate destination, the connection segments can belong to a different administrative domain. Devices connecting different administrative domains are called Edge Devices.

Inside an administrative domain each communication device can have a different level of policy capabilities, starting from no policy awareness at all, up to one with sophisticated integrated Policy Enforcement Points and Policy Decision Points. Depending on the level of sophistication, one administrative domain can provide a different granularity and type of service to users or to other administrative domains. The term Service Level Agreement is often used to describe the intended overall behavior by defining a policy for a specific segment or
administrative domain. SLAs are typically between service providers and users or between service providers themselves.

An example of different SLAs can be:
- A Best Effort service with no security
- A VPN tunnel with associated DiffServ priority traffic

### 1.5.2 Administrative Domain Components

To fulfill a certain policy-based behavior a domain has to implement a number of functions distributed across different devices and system control platforms. Figure 5 on page 13 shows an overview.

A network node is configured to provide the correct data forwarding behavior for specific user and/or application data by implementing policy rules which are derived from the SLA.

![Figure 5. Components Inside an Administrative Domain](image)

The configuration task of a network node is distributed across different platforms. The first one, already available in all systems today, is a network and system management platform. This platform provides the basic configuration settings, and are not typically dynamic in nature. Policies, on the other hand, are expected to vary over time and are typically application/user oriented and not device-specific.

This is where the Policy Server comes in. The policy server instructs the network node to behave according to the policy rules which the server itself received from a repository; this repository is often called a directory. The network node takes care of the actual implementation of the rule.
A Policy Definition Tool is used to populate the policy directory. This tool enables the system administrator to specify the policies for the entire network. The definition process can be supported by a graphical user interface and some configuration methodology which enable the administrator to specify the rules in an efficient and secure way. Over time, the policy definition tool will be integrated into a global system and network management platform.

### 1.6 The Overall System View

The following figure shows how all the components of application-driven networking interrelate. We will use figures similar to this one repeatedly in this book to position each topic under discussion.

![Overall System View Diagram](image)

**Figure 6. Overall System View**

To get a thorough understanding of the complete picture, this redbook will explain each component shown in much more details. Figure 6 gives an overview of the overall system components and the technologies involved. After reading this book you should be able to compare different policy-based solutions inside IBM and in the market by positioning the different solutions according to the reference above.

Since this area of policy-based control is evolving rapidly over time there will be changes in the details of the technologies and techniques used; the current belief is that the functional composition shown in Figure 6 will provide a common ground for the near future.
Please refer to Figure 7 to find the appropriate chapters covering the different components. The numbers inside this figure refer to the subsequent chapters in this redbook which discuss each of the individual components in detail.

Figure 7. Application-Driven Networking Content Overview

Figure 7 shows also a reference to companion redbooks. These redbooks deal with the technical details of QoS and security technologies. In particular, please consult the following ITSO redbooks to find appropriate information:

- A Comprehensive Guide to Virtual Private Networks, Volume I: IBM Firewall, Server and Client Solutions, SG24-5201
- Application Driven Networking: Class of Service in IP, Ethernet and ATM Networks, SG24-5384

Chapter 7, “Putting It All Together” on page 109 will give you some conclusions and closing remarks.
Chapter 2. Policy Enforcement Point

The network node actually transporting the user and/or application data is the central component of the policy system. It is the place where policies and the associated behavior rules have to be implemented and monitored. However, not every network node is functionally equal. Some network nodes have the ability to identify and classify specific user/application data streams; others may be able to perform prioritization based on their own classifications or on classifications performed by others and indicated inside a data packet with a certain flag (TOS, DSCP,...). Other network nodes may simply be able to apply certain priorities based on the interpretation of these flags. Some devices will implement security capabilities and some will not.

The bottom line is that a communication system consists of a heterogeneous collection of devices with different capabilities in size, scalability, function and policy awareness. A generic policy system has to take this into account and provide the flexibility to integrate the different granularity of sophistication of policy awareness into a device.

Figure 8 shows an overview of the components of a network node showing its functional parts and the dependencies on other external policy instances.

Figure 8. Network Node Overview

The numbered relationships shown in the figure are:

1. The applications use existing interface protocols to signal their needs. This can be "plain IP" or with standard enhancements such as TOS/DSCP marking or RSVP.
2. A centralized repository (directory) can be used by a router to configure its policy control system, using LDAP (Lightweight Directory Access Protocol).

3. A policy server can be used to avoid implementing a Policy Decision Point in the network node itself by offloading the decision taking process to an external device.

4. A network management platform is used to monitor and configure the network node via SNMP, or nowadays also by using HTTP and JAVA.

This chapter concentrates on the Policy Enforcement Point (PEP) and on its relationship with the Policy Decision Point (PDP): the relationship numbered “3” in the preceding figure.

A network node has to implement a means to differentiate between user/application data streams, which relates to the implementation of IntServ (RSVP) or DiffServ (DSCP) functions. The network node may also have the ability and requirement to perform security functions, which relates to IKE and/or IPSec functions.

The details of those data manipulation functions are not part of this discussion. We will focus mainly on the implementation of policy-related functions inside a network node. To do that we will first start with a description of the IETF policy framework followed by IBM’s implementation of a Common Policy Engine.

### 2.1 IETF Policy Framework

In terms of the IETF Policy Framework specifications, a network node consists of two functional components: the PEP (Policy Enforcement Point) and the PDP (Policy Decision Point). The PEP is always located inside a network node whereas the PDP may be implemented elsewhere. The communication protocol used to communicate between the PDP and PEP is called the COPS (Common Open Policy Service) protocol and is based on the specification work already done for the provisioning of RSVP reservations. We will discuss this in more detail later; see 2.2, “The COPS Protocol and the Definition of PIBs” on page 23. Figure 9 on page 19 shows the two different implementation scenarios of policy-aware network nodes such as routers and switches.
Policy-based configuration information is stored inside a network node in the PIB (Policy Information Base). The method of storing and using this information is akin to the SNMP MIB (Management Information Base); in fact we are reusing the same information model structure and coding which is based on SMI (Structure of Management Information\(^1\)) and BER (Basic Encoding Rules for ASN.1).

The following paragraphs will discuss the relationship between the PEP and PDP components in more detail.

### 2.1.1 Policy Enforcement Point (PEP)

As already stated, the two main architectural elements for policy control in a network node are the PEP (Policy Enforcement Point) and the PDP (Policy Decision Point). Figure 9 on page 19 shows two configurations involving these two elements; PEP is always a component of a network node and PDP is an entity that may reside in a separate policy server. The PEP is the point at which policy decisions are actually enforced; policy decisions are primarily made at the PDP. The PDP itself may make use of additional mechanisms and protocols to implement additional functions such as user authentication, accounting and policy information storage. The PDP is likely to use an LDAP-based directory service for storage and retrieval of policy information.

The basic interaction between the components begins with the PEP. The PEP will receive a notification or a message that requires a policy decision. Given such an event, the PEP then formulates a request for a policy decision and sends it to the PDP. The request for policy control from a PEP to the PDP can contain a variety of different types of information, such as a flowspec or amount of bandwidth requested in the original message or the event that triggered the policy decision request. The PDP returns the policy decision and the PEP then enforces the policy decision by appropriately accepting or denying the request. The PDP may also return additional information to the PEP which need not be associated with

---

\(^1\) See Appendix A.1, “IETF Documents” on page 117 for references to more information on SMI.
an admission control decision but can be used to formulate an error message or outgoing/forwarded message.

The PEP module at a network node uses the following steps to reach a policy decision:

1. When a local event or message invokes PEP for a policy decision, the PEP creates a request that includes information from the message that describes the admission control request.

2. The PEP may consult a local configuration database to identify the policy conditions that are to be evaluated locally, in which case the PEP passes the request to the Local Policy Decision Point (LPDP) and collects the result from the LPDP.

3. If necessary, the PEP passes the request to the PDP, possibly including the partial result from the LPDP. The PDP applies policies based on all information it received, combining its result with the partial result of the LPDP (if applicable) to reach a final decision.

4. The PDP returns the final policy decision to the PEP.

Note that in the above model the PEP must contact the PDP even if no specific resource-related information is received in the admission control request. This requirement ensures that a request cannot bypass policy control by omitting policy elements in a reservation request.

It must also be noted that the PDP may, at any time, send an asynchronous notification to the PEP to change its earlier decision or to generate a policy error/warning message.

### 2.1.2 Placement of Policy Functions in a Network

By allowing division of labor between an LPDP and a PDP, the policy control architecture allows staged deployment by enabling routers of varying degrees of sophistication (as far as policy control is concerned) to communicate with policy servers. Figure 10 on page 21 depicts an example set of nodes belonging to three different administrative domains, AD-1, AD-2 and AD-3: each AD could perhaps correspond to a different service provider. Nodes A, B and C all belong to administrative domain AD-1, advised by PDP PS-1, while nodes D and E each belong to a separate administrative domain: domains AD-2 and AD-3 respectively. Node E communicates with PDP PS-3. It is expected that there will be at least one PDP per administrative domain.

Policy-capable network nodes could range from unsophisticated nodes such as node E, which has no LPDP (and thus has to rely on an external PDP for every policy processing operation) to self-sufficient nodes such as node D, which essentially implements both LPDP and PDP functions locally.
2.1.3 Example Scenarios

Let us now discuss some scenarios which will be rather common in future policy-based systems:

- Admission control policies based on factors such as time of day, user identity or other credentials

Policy control must be able to express and enforce rules with temporal dependencies. For example, a group of users might be allowed to make reservations at certain levels only during off-peak hours. In addition, the policy control must also support policies that take into account identity or credentials of users requesting a particular service or resource. For example, an RSVP reservation request may be denied or accepted based on the credentials or identity supplied in the request.

- Bilateral agreements between service providers

Until recently, usage agreements between service providers for traffic crossing their boundaries have been quite simple. For example, two ISPs might agree to accept all traffic from each other, often without performing any accounting or billing for the foreign traffic carried. However, with the availability of QoS mechanisms based on Integrated and Differentiated Services, traffic differentiation and quality of service guarantees are being phased into the Internet environment. As ISPs start to sell their customers different grades of service and can differentiate between different types of traffic flows, they will...
also need mechanisms for charging each other for traffic (and reservations) crossing their networks.

One additional incentive in establishing such mechanisms is the potential asymmetry in terms of the customer base that different providers will exhibit: ISPs focused on servicing corporate traffic are likely to experience much higher demand for reserved services than those that service the consumer market.

Lack of sophisticated accounting schemes for inter-ISP traffic could lead to inefficient allocation of costs among different service providers. Bilateral agreements could fall into two broad categories: local or global. Due to the complexity of the problem, it is expected that initially only the former will be deployed. In these, providers which manage a network cloud or administrative domain contract with their closest point of contact (neighbor) to establish ground rules and arrangements for access control and accounting. These contracts are mostly local and do not rely on global agreements; consequently, a policy node maintains information about its neighboring nodes only.

Referring again to Figure 10 on page 21, this local model implies that provider AD-1 has established arrangements with AD-2 but not with AD-3 for the use of each other’s network. Provider AD-2, in turn, has agreements in place with AD-3. Thus, when forwarding a reservation request from AD-1 to AD-2, provider AD-2 will charge AD-1 for use of all resources beyond AD-1’s network.

This information is obtained by recursively applying the bilateral agreements at every boundary between (neighboring) providers until the recipient of the reservation request is reached.

- A model of increasing popularity in the telephone network is that of the pre-paid calling card. This concept could also be applied to the Internet; users purchase tokens which can be redeemed at a later time for access to network services. When a user makes a reservation request through, say, an RSVP RESV message, the user supplies a unique identification number of the token.

Referring once more to Figure 10 on page 21, suppose receiver R1 in the administrative domain AD-3 wants to request a reservation for a service originating in AD-1. Two issues have to be resolved here:

1. What is the scope of these charges?
2. When are charges (in the form of decrementing the remaining credit) first applied?

The answer to the first question is dependent on the bilateral agreement model in place. If, on the one hand, provider AD-3 has established agreements with both AD-2 and AD-1, it could charge for the cost of the complete reservation up to sender S1. On the other hand, if AD-3 has no bilateral agreements in place, it will simply charge for the cost of the reservation within AD-3. Subsequent PDPs in other administrative domains will charge for their respective reservations.

Since multiple entities are both reading (remaining credit) and writing (decrementing credit) to the same database, some coordination and concurrency control might be needed.

Another problem in this scenario is determining when the credit is exhausted. The PDPs should contact the database periodically to submit a charge against the credit; if the remaining credit reaches zero, there must be a mechanism to
detect that and to cause revocation or termination of privileges granted based on the credit.

As you can imagine, in reality today only very simple scenarios are implemented - rules along the lines of:

- Prioritize SAP traffic between 08:00 in the morning and 18:00 in the afternoon.
- Establish a VPN tunnel with the following parameters between the following end points.

Near the end of this chapter we will discuss more details of the first release of IBM’s Common Policy Engine.

### 2.2 The COPS Protocol and the Definition of PIBs

The IETF RSVP Admission Policy (RAP) working group has defined the COPS (Common Open Policy Service) protocol as a scalable protocol that allows policy servers (PDPs) to communicate policy decisions to network devices (PEPs).

COPS was not designed just to support one type of client; multiple types of COPS policy clients may be defined. Today there are different specifications for a COPS-RSVP, COPS-DiffServ or COPS-Provisioning (VPN related) client.

COPS is a query/response protocol that supports two common models for policy control: Outsourcing and Provisioning.

**The Outsourcing model** addresses the kinds of events at the PEP that require instantaneous policy decisions (authorization). The PEP, being aware that it must perform a policy decision and being unable to carry out the task itself, delegates responsibility to an external policy server (PDP). For example, in COPS-RSVP when a reservation message arrives, the PEP is aware that it must decide whether to admit or reject the request; it sends a specific query to the PDP, and in most case, waits for a decision before admitting or rejecting the pending reservation.

**The Provisioning model**, on the other hand, makes no assumptions of such direct 1:1 correlation between PEP events and PDP decisions. The PDP may proactively provision the PEP in reaction to external events (such as user input) as well as in reaction to PEP events, and any ratio between these (N:M correlation). Provisioning may be performed in bulk (for example, entire router QoS or VPN configuration) or in portions (such as updating a DiffServ marking filter).

Network resources are provisioned based on relatively static SLAs (Service Level Agreements) at network boundaries. While the outsourcing model is dynamically paced by the PEP in real-time, the provisioning model is paced by the PDP in somewhat flexible timing over a wide range of configurable aspects of the PEP.

#### 2.2.1 Why not SNMP?

SNMP is a very popular network management protocol. We should question the use of COPS rather than extending SNMP for policy provisioning.
There are several aspects intrinsic to SNMP that prevent it from being a successful policy protocol:

- SNMP uses a transactional model and does not support the concept of long-term client/server connections. This can mean that servers may not know that devices have failed and vice-versa. A “hello” poll is an ugly and incomplete solution because it will not solve the problem if a device reboots in between polling intervals.

- The SNMP transactional model allows multiple servers to modify the state of a single network device simultaneously. Given that SNMP does not have resource locking facilities, a policy server would have to poll and verify that no other network management software or humans changed any of the configured resources.

- SNMP is based on UDP and is thus unreliable. The lack of reliability is unacceptable for a policy protocol.

- Lastly, SNMP was not designed as a real-time operations protocol. Its trap mechanism is inefficient and cumbersome and there are no performance guarantees.

![Figure 11. The COPS Protocol](image)

Referring to Figure 11, COPS was designed to overcome these shortcomings. It has a single connection between client and server and it guarantees that only one server updates the policy configuration at any given time (and these configurations are locked, even from console configuration, while COPS is connected to a server).

COPS uses reliable TCP transport and thus uses a state sharing/synchronization mechanism and exchanges differential updates only. If either the server or client is rebooted (or restarted), the other will know about it quickly. Last, it is defined as a high priority (real-time) mechanism for the PEP device.

The COPS protocol is already used for policy control over RSVP. It is highly desirable to use a single policy control protocol for Quality of Service (QoS).
mechanisms (if possible) and VPN control, rather than invent a new one for each type of policy problem.

At the same time, useful mechanisms from SNMP were adopted. COPS uses a Policy Information Base (PIB) with SMI, MIB and BER data encoding. This allows reuse of experience, knowledge, tools and some code from the SNMP world.

### 2.2.2 Policy Information Base (PIB)

COPS data is a collection of policy rules, each identified by Policy Rule Identification (PRID). The PRID is a globally-unique name which describes the representation (format) and semantics of the policy rule.

COPS uses a Policy Information Base (PIB) as its global name space of provisioning policy. The PIB name space is common to both the PEP and the PDP. The PIB can be described as a tree, where the branches of the tree represent classes (types) of policy rules (PRC) and the leaves represent instances (contents) of policy rules (PRI). See Figure 12 on page 25 for an overview.

There may be multiple instances of rules (PRIs) for any given rule type (PRC). For example, if one wanted to install multiple access control filters, the PRC would represent a generic access control filter type, and each PRI would represent an actual access control filter to be installed.

![Figure 12. PIB Structure](image)

The PIB is based on SMI and MIBs. The decision to use this format as a basis opens up the possibility of reusing SMI and MIB knowledge, experience, and tools.
PRIs and PRCs are uniquely identified by PRIDs. PRIDs have a hierarchical structure in dotted-decimal format (for example, 1.3.4.2.7), where the first part identifies the PRC (in this case, “1.3.4”) and the last part identifies the PRI (“2.7” here).

![Figure 13. Policy Rule Identification](image)

The policy tree names all the policy rule classes and instances and this creates a common view of the policy organization between the client (PEP) and the server (PDP).

Consider the example of a set of filters for marking traffic with a certain DiffServ code point (DSCP). Each filter has the following attributes to set:

- Protocol number
- Source address
- Source port
- Destination address
- Destination port
- DSCP value

Let’s assume that the class Filter’s PRID is “$.1”, where “$” represents some prefix in the policy tree to which the class Filter belongs. The first filter would have a PRID of $.1.1, the second $.1.2, and so on.

As mentioned before the PIB is described using SMI and MIBs. SMI and MIBs are defined based on the ASN.1 data definition language. To simplify the implementation and retain the ability to re-use the SNMP encoding/decoding code, the representation of the policy information must follow BER encoding.
2.2.3 Details on PEP-PDP Interaction Using COPS

This section describes typical exchanges between a PDP and Policy Provisioning COPS client.

First, a TCP connection is established between the client and server and the PEP identifies itself as a Policy Provisioning client. If the PDP supports the provisioning client type the connection is established. If the client type is not supported, a message is returned by the PDP to the PEP, possibly identifying an alternate server that is known to support the policy for the provisioning client type.

The PEP can now send a configuration request to the server. The configuration request message from the client serves two purposes. First, it is a request to the PDP for any provisioning configuration data which the PDP may currently have for the PEP, such as access control filters, etc. Secondly, the configuration request is a request to send policy data to the PEP asynchronously, whenever the PDP decides it is necessary. This asynchronous data may be new policy data or an update to policy data sent previously.

If the PDP has Policy Provisioning policy configuration information for the client, that information is returned to the client in a decision (DEC) message. If no policy rules are defined for the client, the DEC message will simply specify that there is no policy information to be returned.

The PDP can add new policy data or update existing state by sending subsequent DEC message(s) to the PEP.

For Policy Provisioning purposes, access state, and access requests to the policy server can be initiated by other sources such as users requesting network services via a Web interface into a central management application or H.323 servers requesting resources on behalf of a user for a video conferencing application. When such a request is accepted, the device affected by the decision must be informed of the decision, and since the PEP in the device did not initiate the request, the specifics of the request (for example, the flowspec, packet filter, and PHB to apply) must be communicated to the PEP by the PDP.

The PEP acknowledges the DEC message and action taken by sending a report (RPT) message with a “Commit” or “No-Commit” indication. This serves as an indication to the PDP that the requestor (such as an H.323 server) can be notified that the request has been accepted by the network. If the PEP needs to reject the DEC operation for any reason, an RPT “No-Commit” message is sent, optionally with client-specific information specifying the policy data that was rejected. The PDP can then respond to the requestor accordingly.

When communication is lost between PEP and PDP, the PEP attempts to re-establish the TCP connection with the PDP it was last connected to. If that server cannot be reached, then the PEP attempts to connect to a secondary PDP if such a thing is configured at the PEP. In any case, when a connection is finally re-established either with the primary PDP or a secondary PDP, the PEP should provide the last PDP address of the PDP for which it is still caching decisions. Based on this information, the PDP may request the PEP to re-sync its current state information. If no decisions are being cached on the PEP (due to reboot or time-out) the PEP must not include the last PDP address information. If a PEP
re-connects to a PDP and the PDP does not request synchronization, the client can assume the server recognizes it and the current state at the PEP is correct.

Any state changes which occurred at the PEP while the connection was lost must be reported to the PDP in an RPT message. If re-synchronization is requested, the PEP should reissue its configuration requests and the PDP should delete the appropriate PRCs (policy rule classes) on the PEP (thus removing all previous decisions below the PRC, effectively resetting all states and reverting to some preconfigured decisions).

After this brief overview of the IETF policy specification framework we will now discuss IBM’s first actual implementation of a policy engine.

### 2.3 IBM Common Policy Engine (CPE)

Before going into details let us briefly reassess the structure of network nodes with regard to policy-based control.

---

**Figure 14. Standard versus IBM Policy Engine**

Figure 14 on page 28 shows a comparison between the policy architecture described by the IETF and IBM’s initial implementation. It is obvious that IBM took the decision to implement a combined PEP/PDP architecture, thus avoiding the need to implement COPS. The COPS standard was not stable enough for inclusion at the time the actual router code was designed and implemented; in any case a combined approach has some nice side effects which we will discuss later in this book.

Even in cases where vendors claim to be “COPS-based” it is always a proprietary implementation; it is only in the area of RSVP processing (COPS-RSVP) that a standards-based implementation is available today on some vendor router
platforms. Since policy-based control is more than just RSVP, IBM did not choose to develop its own extensions to COPS-RSVP.

The Policy Information Base (PIB) is the implementation of an information model used by COPS to identify certain policy rule classes (PRC) and policy rule instances (PRI). Since we are not using COPS, we have no PIB and the IBM Common Policy Engine concentrated on the interaction with a policy repository using LDAP (Lightweight Directory Access Protocol) and related schemata (the information model).

Details on LDAP and LDAP schemata will be discussed in Chapter 3, “Policy Server, including Policy Decision Point” on page 43.

2.3.1 CPE System Structure

The principal function of the combined policy engine is to search through a set of policy rules in order to retrieve the set of actions that the router needs to perform on any given packet.

Each policy rule has two parts. The condition describes the stream of packets to which this rule applies. Although the router deals with only one packet at a time, the concept of a stream is a useful one; it represents the flow of packets over time, each of which is to be treated in the same manner by the actions resulting from the policy rules. Most router implementations will identify a stream of data in order to cache the policy rules for this data flow to speed up handling of later packets in the same stream. The condition element consists of a number of fields or attributes which identify the stream in terms of attributes such as the source address, the destination address, the port numbers being used, the protocol identifier and the router interfaces on which this stream arrives and departs. The second part of the policy rule is a defined set of actions corresponding to the stream thus identified, such as “mark TOS” or “deny”.

These rules may be retrieved from a directory using the LDAP protocol or may be configured directly on the router. In either event, the policy engine receives a set of rules that pertain to the processing of packets by various enforcement modules.

The differentiated services (DiffServ) module, the RSVP module, the IPSec module and the IKE module are all instances in the IBM router code which use policy control. Each of these modules queries the Common Policy Engine (CPE) with a packet and receives a decision appropriate to that packet.

Any module may issue a query for just one or for all of the packets of a policy stream; modules may choose to cache initial policy decisions and use the information to handle the processing of subsequent packets.

When a module queries the policy engine it may have to pass additional information that is not contained in the packet header itself. For instance, a fragmented packet may not contain the port numbers, which may have to be remembered from the first fragment by a fragment handler.
The policy engine is combined or unified in the sense that each query returns the decisions for all modules [DiffServ (DSCP), IntServ (RSVP) and VPN (IPSec, IKE)], thus eliminating the need for multiple queries to multiple policy engines. The policy engine itself is unified in its design and retrieves multiple actions in a single search.

The decision is not the same as the action. For example, if we consider an action “Admit no more than 10 RSVP flows”, then the decisions of “accept” or “deny” are taken for each requested RSVP flow depending on how many flows have already been admitted.

The Policy Manager module provides tools to the network policy administrator to define, validate and store policy rules. There are a number of different possibilities in the way that policies can be retrieved by the router, such as:

- Via LDAP (Lightweight Directory Access Protocol) client. This method will be further discussed in 4.5, “IBM LDAP Directory and Policy Database Interaction” on page 68.
- Via Talk 6, which means by using the command line interface. This method will be discussed further in 2.4, “The Policy Feature in the IBM 221x Router Product Family” on page 33.
The Policy Matcher module fetches a number of policy rules from the Policy Manager and organizes them into an efficient search structure. We will elaborate on this in 2.3.2, “Solving the Rule Look-Up Problem” on page 31. The Policy Matcher’s primary function is to provide traffic classification service to the Policy Enforcement Entity, which is responsible for differential treatment of traffic flows based on the results of classification.

The Policy Enforcement Entity is the actual component of the policy engine which sits in the actual traffic path. There are generally several service-specific modules within such a device that implement the actions specified by the applicable policies. The decisions returned by the Policy Matcher module determine the actual treatment that a packet receives within the Policy Enforcement Entity and the order in which the service-specific modules process it.

To summarize:

The decisions returned by the Policy Matcher module depend on the following:

1. The set of policy rules obtained from the Policy Manager
2. The attributes or condition (see 2.3.1, “CPE System Structure” on page 29), specified by the Policy Enforcement Entity in its query
3. Some internal state within the Policy Matcher that may be initiated, set or modified by the various service-specific modules within the Policy Enforcement Entity. For example, the policy at a firewall may dictate that all packets matching specified conditions be directed to use the same IPSec tunnel. In this case, after IPSec signaling has been completed and tunnel state installed in the router, the IPSec service module instructs the Policy Matcher to return the tunnel handle as part of the decision returned for all succeeding data packets matching the policy conditions.

2.3.2 Solving the Rule Look-Up Problem

Looking up hundreds of rules and matching them to the conditions given by the query is one of the major challenges in the implementation of an efficient design. Conditions and policies are typically expressed as applying to different ranges of attribute values, for example: “Any packet with port numbers between xx and yy should have priority, but only between hh:mm a.m. in the morning and hh:mm p.m. in the afternoon.”

That means the number of attributes used for policy decisions determines the number of dimensions a lookup algorithm has to implement. Important attributes used in our CPE include:

- Source address
- Destination address
- Protocol type
- Source port
- Destination port
- TOS (Type Of Service) field content
- Ingress interface on which the packet was received
- Egress interface over which the packet will be transmitted
• The time the packet was received

For the purpose of this discussion, imagine that we only have two attributes (source address and source port, for example) and want to derive a rule based on conditions expressed in terms of these two attributes. To find a matching rule, we have to find the area where both conditions are met.

![Figure 16. Policy Search Example with 2/3 Dimensions](image)

With only two attributes (or dimensions) under consideration we have a solution "area" which can be represented as shown in Figure 16, if we are matching a rule on a single value or range each of the two attributes. If we have three attributes instead of two, we get a "cube" or "box", and in general we can express policy rules as applying to D-dimensional boxes and abstract the classification or matching problem as a search for the set of boxes that an incoming packet belongs to.

That means each incoming packet might belong to at least one D-dimensional box. To search for all the matching boxes we build a binary search tree based on a subset of D-dimensions. Since IBM has some patents pending on this subject we cannot go into more details, but we claim that our patented approach is 25 times faster than a more conventional approach:

The CPE creates a binary search tree. Each packet requires up to the order of \(\log_2(n)\) comparisons for each attribute. For 200 entries and 5 selectors, that's \(\log_2(200)\) multiplied by 5 or about 40 comparisons; a linear search algorithm under the same circumstances requires 200 x 5 comparisons, or 1000 comparisons, which means that the IBM approach is some 25 times faster.

For 200 rule entries, our approach is 25 times faster than all other vendors who implement linear searches on access control lists (ACLs): this is what all of the other major vendors do. Note that the decision tree has to be recalculated each time a policy changes, and this can be a significant burden for the router compared to the relatively simple process of adding a rule to a list. Assuming a reasonably stable policy status for QoS and VPN configurations, this aspect is relatively academic in terms of overall performance.
Figure 17 summarizes the advantage of the IBM CPE search approach over the linear search algorithm often implemented elsewhere.

2.4 The Policy Feature in the IBM 221x Router Product Family

The policy feature of the IBM router code (MRS/MAS/AIS 3.3) facilitates the management of IPv4 traffic in a network. You may configure policies for very simple filter rules (drop or pass) or for complex security and QoS implementations. The combination of policies determines how routers handle IPv4 traffic in a network.

The policy implementation in this family of routers constitutes the basis for policy decisions and the means of enforcing them. These concepts, as shown above, are often referred to as a policy decision point (PDP) and a policy enforcement point (PEP). The policy database which resides in the router's memory is comprised of the set of policies loaded from local configuration and policies that have been read from LDAP. The policy database is built under the following conditions:

- Device reload or restart
- Talk 5 reset database command
- Automatic refresh
- SNMP set request
2.4.1 Policy System Structure

The policy database serves as the PDP and consists of a set of policies that determine how the policy feature-related components handle packets. When a policy results in a decision (based on information such as the time of day, IP packet information, and protocol-specific information such as identification), the decision is passed to the enforcement component (PEP) to carry out the action. Figure 18 shows the relationships:

![Figure 18. IP Packet Flow in a Router](image)

IP packets first must pass the input packet filter before any other actions can be taken. If the input packet filter has rules present then the packet may have some action taken on it. If there is a filter match that excludes the packet or there is no match found in the input packet filter then the packet is dropped.

If the packet passes the input packet filter then it goes to a demultiplexing filter, which checks to see whether the packet is locally destined. If it is, then depending on the type of packet, it is passed to other modules. These modules include those which process IPSec, IKE, and RSVP.

If the packet is locally destined for IPSec, IKE, or RSVP then those modules may query the policy database to determine which action to take. The number of queries can be very significant. Since our current implementation is one of using a co-located policy engine with a binary tree-based decision function inside each router itself, the volume of queries should present no implementation problems.
If the packet is not locally-destined then it is given to the forwarding engine and a routing decision is made, and here global access control rules may modify the packet and even drop it completely. If the routing decision does not drop the packet, then the packet goes to the output packet filter. If filter rules are present in the output packet then the packet may have address translation performed (NAT), may be passed or may be dropped. If no filter rules are present then the packet is passed. If filter rules are present and no match is found then the packet is dropped.

If the packet passes the output packet filter then the IP engine queries the policy database to determine whether any other actions should be performed on this packet. The policy database is populated with policy objects defining the conditions under which a certain action has to take place.

### Note

The triggering mechanisms which cause the modules to query the policy engine are different:

- For DiffServ it is packets.
- For RSVP it is signalling messages including RSVP refresh messages which occur approx. every 30 sec.
- For IKE/IPSec it is packets and IKE signalling.

Each module may implement some efficiency “hooks” to avoid going to the policy engine for each and every packet. In particular, when using DiffServ, the router implements an “IP flow cache” and a mechanism for identifying individual packets as being members of a stream or flow in order to minimise the number of requests directed to the policy engine.

If the input and output packet filters are enabled for an interface(s), and packets that are to be controlled by the policy database are expected to traverse these interfaces, then a filter rule that includes these packets must be present in the input and output packet filters so they will not be dropped before the policy database is queried. One suggestion is to use the policy database to configure all the pass/drop rules and not to use the packet filters.

### 2.4.2 Policy Objects

A policy definition in a router is made up of a number of different components:

1. A policy
2. A profile
3. A validity period
4. An action (DiffServ, RSVP, IPSec)
5. Security descriptions

The distinct parts that make up a policy are named objects. Policy objects may refer to one another, and as a group of related items they comprise a policy. By
separating configuration information into separate distinct objects, you can reuse many of them across multiple policy definitions, thus saving time and reducing maintenance efforts.

![Diagram of Definition Tree]

**Figure 19. Definition Tree**

Individual policy objects are discussed in detail in the following topics.

### 2.4.2.1 Policy

The policy object describes which conditions should be checked against, and if the checks match, which actions are to be enforced. The policy makes named references to the validity period and the profile. For the policy to be valid, these references are required.

The policy must also make a named reference to one or more of the following actions: an **IPSec manual-keyed tunnel object**, an **IPSec action**, an **ISAKMP action**, an **RSVP action**, or a **DiffServ action**, valid combinations of which are:

**IPSec only**
- IPSec manual-keyed tunnel
- IPSec action to drop packets
- IPSec action to pass packets (no security)
- IPSec action to secure packets, ISAKMP action

**DiffServ and IPSec**
- DiffServ action (drop)
- IPSec manual-keyed tunnel and DiffServ action (pass)
- IPSec action to secure packets, ISAKMP action, DiffServ action (pass)

**IntServ(RSVP) and DiffServ**
- RSVP action
- RSVP action and DiffServ action (pass)
Each policy also has a priority number associated with it (the higher the number in the priority attribute, the higher the priority). The priority determines whether or not this policy takes precedence over another policy. Typically, you only have to set this if two or more policies’ profiles conflict with each other in some way. The policy with the more specific profile should have a higher priority. For example, suppose that one policy specifies that traffic from subnet A to subnet B is to be secured with IPSec (DES) and another policy specifies that traffic from point a (a particular host inside of subnet A) to subnet B is to be secured with IPSec (3DES). The more specific policy (a to B) should have a higher priority than the policy referring to traffic from A to B.

It is a good idea to designate initial priority values which are 5 or more digits apart to allow room for specifying additional priority values for conflicting policies later. Each policy also has an enabled attribute, which determines whether the policy is to be enabled when loaded into the policy database. If a policy match is found during a policy database search but the policy is disabled, then the next most specific policy is enforced.

2.4.2.2 Profile
The profile determines exactly which attributes of a packet are to be used to perform conditional testing to determine whether or not a packet is included in a particular policy; there is only one profile corresponding to each policy. In addition to a name, the profile consists of applicable values for the packet’s source and destination addresses, protocol information, and source and destination port information.

---

**Note**

When defining policies for IPSec/ISAKMP, each router providing the security must have a policy to define the security association. The profile on each router must associate the source with the destination and the destination with the source. The profile for an IPSec policy must specify the source address as the traffic to be encapsulated into the tunnel and the destination address must be at the remote end of the tunnel.

The profile also can be used to select packets based on the type-of-service (TOS) byte and can be associated with specific ingress and egress IP port addresses. This allows the specification of combinations such as “any packet arriving on IPaddrX and leaving on any interface OR any packet coming in on any interface and leaving on IPaddrX”, which is particularly useful for the definition of a general drop rule for a public interface.
If you want to use the profile to select an IPSec/ISAKMP policy then you have the option of specifying the local ID to be sent during Phase 1 and the list of acceptable remote IDs during Phase 1 negotiations. For details on IPsec/ISAKMP configuration see:

- A Comprehensive Guide to Virtual Private Networks, Volume I: IBM Firewall, Server and Client Solutions, SG24-5201

For information on DiffServ and IntServ configuration details, see:

- Application Driven Networking: Class of Service in IP, Ethernet and ATM Networks, SG24-5384

2.4.2.3 Validity Period
The validity period specifies the life of the policy, the year, the months of the year, the days of the week, and the hours of the day that it is valid. This flexibility enables the network administrator to specify when a policy is valid, for example “all the time”, or “only this year, during the months of January, February and March, on Monday through Friday, from 9 a.m. to 5 p.m.”

When a policy in the policy database becomes invalid, the next most specific policy will be enforced. Thus you could define a policy that specifies on Monday through Friday from 9 a.m. to 5 p.m. to secure all traffic from subnet A to subnet B, and at any other time drop all traffic from subnet A to subnet B. In this case the first policy must have a higher priority, as mentioned before.

2.4.2.4 DiffServ Action
The DiffServ action describes the quality of service that is to be provided to packets that match a policy that specifies a DiffServ action.

You may configure the DiffServ action to drop packets. You may also use the DiffServ action to map packets into relative qualities of service.

You may configure the bandwidth allocated as a percentage of output bandwidth or as an absolute value in Kbps. You must specify whether the best effort/assured queue or the premium queue is to provide the bandwidth allocation.

The DiffServ action also specifies how to mark the TOS byte before it is sent on the egress interface. It is useful to mark the packets at some point in the network based on the information in the IP packet header; once the classification has been made then the rest of the hops in the network can simply look at the TOS byte to determine which QoS to apply to the packet. Looking at the TOS byte alone is much more efficient than looking up values such as IP address ranges and may be necessary to achieve high performance in the DiffServ backbone.

2.4.2.5 RSVP Action
The RSVP action specifies whether to permit or deny RSVP flows if an RSVP reservation occurs and the reservation request matches the profile of the policy. If you want to permit the reservation, then the RSVP action also states the allowed duration of the reservation, the allowed maximum bandwidth, and optionally, a
reference to a DiffServ action. The reference to the DiffServ action enables RSVP to determine how to mark the TOS byte before the packet leaves the router. This is useful when packets pass from an RSVP network into a DiffServ network. RSVP can provide the QoS up to the RSVP boundary and then mark the TOS byte appropriately so the DiffServ network can apply the correct treatment there.

2.4.2.6 IPSec Action
The IPSec action may specify either a drop, pass, or secure action. If the action is drop, then all packets matching this policy are dropped. If the action is pass with no security, then all packets are passed in the clear. If the action is pass with security, then all packets are secured by means of the security association (SA) specified by this action.

The IPSec action also contains the IP addresses of the tunnel end points for the IPSec tunnel and IKE security associations. The attributes of the SA are determined by the IPSec proposals that the IPSec action references. The IPSec action may specify multiple IPSec proposals which are sent and checked in the order they are specified. Having multiple proposals in an IPSec action allows the configuration to contain all the acceptable combinations of security, thereby reducing the number of potential configuration mismatches between VPN gateways.

IPSec Proposal
The IPSec proposal contains the information about which transform to propose or check against during Phase 2 ISAKMP negotiations. If you require perfect forward secrecy (a fresh Diffie Hellman calculation), then the IPSec proposal identifies which DH group to use.

The transforms that the IPSec proposal references are sent or checked against in the order in which they are specified. The first ESP transform (encryption, authentication, or both) or AH transform (authentication only) must be the one that is most appropriate to use. If more than one transform is in the list, then each one is compared to the peer’s list of transforms to find a match. If none of the configured transforms match the peer’s list then the negotiation fails.

The IPSec proposal may list a combination of AH and ESP transforms, but the only valid combinations are:

- List of AH only (tunnel or transport mode)
- List of ESP only (tunnel or transport mode)
- List of AH (transport mode) and list of ESP (tunnel mode)

IPSec Transform
The attributes of the IPSec transform contain information about the IPSec encryption and authentication parameters and also specify how often the keys are refreshed. The transform is either AH (authentication only) or ESP (encryption, authentication, or both) and may be configured to operate in either tunnel or transport mode.

2.4.2.7 ISAKMP Action
The ISAKMP action specifies the key management information for Phase 1. It specifies whether the Phase 1 negotiations are to start in main mode (provides identity protection) or in aggressive mode. It also specifies whether the Phase 1 security association is to be negotiated at device startup or on demand. The
ISAKMP action also must reference one or more ISAKMP proposals. The first reference must be to the most acceptable ISAKMP proposal.

**ISAKMP Proposal**
The ISAKMP proposal specifies the encryption and authentication attributes of the Phase 1 security association. It also specifies which Diffie Hellman group to use to generate the keys, and the life of the Phase 1 security association. You must select the authentication method in the ISAKMP proposal. It can be either pre-shared key or certificate mode.

2.4.2.8 User
You must configure a USER for any policy that uses an ISAKMP negotiation with pre-shared key as the authentication method. The USER configuration identifies the pre-shared key to use for the ISAKMP peer. The user object contains the identifying information for a remote ISAKMP peer, that is IP address, FQDN, user FQDN or key ID, and which method the user wants to use for authentication.

You may select either pre-shared key or certificate mode. If you select pre-shared key, then you must also specify whether the pre-shared key must be entered in ASCII or hexadecimal, and the value of the key. USERS may be grouped together by assigning them to the same group name. This group can then optionally be associated with a policy's profile to perform a more strict policy lookup for Phase 1.

2.4.2.9 IPSec Manual-Keyed Tunnel
The IPSec manual-keyed tunnel is a static configuration of the encryption and authentication parameters. No negotiation is performed for the tunnel so both peers must have exactly the same configuration. The keys are actually entered as part of this configuration and must match on both sides of the tunnel. Since no negotiations performed in this mode, the keys are never refreshed.
Figure 20 on page 41 gives a summary of the configuration components. For a series of example configurations see Chapter 6, “User, Application and Implementation Scenarios” on page 85.
Chapter 3. Policy Server, including Policy Decision Point

The term “Policy Server” has already been used a number of times in this book to describe the functional entity which actually performs the decision on what action has to be taken. Chapter 2, “Policy Enforcement Point” on page 17 took a much more network node-centric view in this respect.

From a router’s point of view, a policy server or a decision point can be co-located with its policy enforcement function or it can be separated and implemented in a remote entity which will be accessed via the COPS (Common Open Policy Service) protocol.

To be able to address the conceptual and architectural aspects of a policy server more accurately we will discuss its functional decomposition in more detail in the following paragraphs. To do this we will refer back to the discussion of COPS in 2.2, “The COPS Protocol and the Definition of PIBs” on page 23.

A major part of this chapter will focus on the relationship numbered “1” in Figure 21. Relationship “3” was discussed in detail in 2.2, “The COPS Protocol and the Definition of PIBs” on page 23. Relationship “2” will be discussed later in this book.

A policy server, as described above, is a central entity in the decision making process of a policy system. To make it easier to understand all aspects of a decision point we will start with a short discussion of the basic policy framework as discussed in the IETF.
A policy server interacts with:

1. A directory server to retrieve the necessary policy information based on LDAP (Lightweight Directory Access Protocol). LDAP will be explained in detail in Chapter 4, “Policy Repositories” on page 49.

2. The network and system management platform to retrieve additional information outside the content scope of a directory. This can be an accounting system or an AAA (Administration, Authorization and Authentication) service. Protocols used are SNMP, HTTP or Java.

3. The Policy Enforcement Point to reply to policy queries via COPS (Common Open Policy Service) protocol.

---

**Note**

The term *policy* is very confusing to understand since it is used under different circumstances. We may refer to a *policy* which states that “ERP traffic has to be prioritized during working hours” and to another *policy* in which “port number XX has to be assigned DSCP YY on IP address pairs (x.y.z.k, q.w.e.r)”. The first is a *corporate policy*, the second a *router interface policy*. Keep this in mind when reading repeated definitions of *policy* throughout this book.

---

### 3.1 Policy Framework

Policy-based management involves the use of administratively formulated rules that specify actions in response to defined criteria. Rules may be general and abstract, requiring interpretation by the policy framework, or very specific.

Rules are condition/action pairs. Several components are critical to the definition of a policy architecture:

1. The ability to define and update policy rules.
2. The ability to store and retrieve rules.
3. The ability to decipher the conditional criteria of a rule.
4. The ability to take the specified actions of a rule if the conditional criteria are met.

At the center of this architecture is the policy rule. Its format and storage are critical to a policy-based management framework and addressed by something which is called the *policy schema*.

The retrieval and execution of the policy rules can take many forms. No single implementation is automatically correct; however all implementations of a policy-based management architecture must address several key components. These are:

1. An interface to define and update rules - either graphical or command line/script.
2. A repository to retain and retrieve rules.
3. A decision point to assess the conditional criteria of a rule.
4. An enforcement point to execute the actions of a rule, when the conditions evaluate to “true”.

---

44 Application-Driven Networking: Concepts and Architecture for Policy-Based Systems
Other concepts must also be addressed in a policy architecture: the scope to which policy rules extend, rule conflict detection and resolution, and how to achieve interoperability.

![Model Overview](image)

The model shown in Figure 22 is another description of the policy framework as already shown in Figure 6 on page 14. It permits the implementation of three different policy abstractions using the functional components shown. The policy framework as described in the standardization bodies does not require that all functional components be implemented nor does it specify implementation packaging of functional components.

The different abstraction levels of policies are:

1. The **administrator’s view of policy** is an abstraction of general configuration and operational characteristics of the resources in a policy domain and of service level objectives (SLOs) on these resources. SLOs are frequently derived from contractual service level agreements (SLAs). The Policy Management Tool may provide significant value-add in the level of abstraction and degree of sophistication of the GUI presentation and the mapping to and from the lower-level policy rule representation.

2. The **policy rules**, as stored in the policy repository using defined schemata, provide a deterministic set of policies for managing resources in the policy domain. These policy rules are produced by the Policy Management Tools and used by the Policy Decision Points (PDP). Policy rules are network device-independent.

3. The **policy mechanisms** are policy discipline-specific and may be implementation-specific. They are used for configuring the necessary logic in the respective network components - the Policy Enforcement Points (PEP) used to deliver the services as prescribed at the administrative interface of the Policy Management Tool.

In Chapter 2, “Policy Enforcement Point” on page 17 we focussed on the third aspect of policy abstraction. We called the representation a PIB (Policy Information Base). This chapter will focus on the second abstraction level: the policy rule and the related policy repositories.
3.2 Policy Rules

As previously noted, the Policy Management Tool functional component produces the policy rules and the Policy Decision Point components are users of the policy rules. The policy rules specify the logic used to deliver the ordained service and levels of service. The PDP components interpret the policy rules and then map these rules to the underlying policy mechanisms.

Generally, policy rules are of the form: if <condition> then <action>. The <condition> expression may be a compound expression and it may be related to entities such as hosts, applications, protocols, users, etc. The <action> may be a set of actions that specify services to grant or deny, or other parameters to be input to the provision of some service.

Although other policy repositories are permitted in this framework, LDAP is specifically referenced in this architecture as the access protocol and scheme used to store the policy rules. In LDAP, the policy rules are represented as a set of object entries that include, amongst others, object classes for policy rules, policy conditions and policy actions. For LDAP implementations, PDPs may detect changes in policy rules by periodic polling, but use of the (currently draft) LDAP event notification mechanisms would be preferable.

3.2.1 Policy Trigger and Rule Location

Triggers to evaluate one or more policy rules include events, polling, and system/component requests. The detection of an event, the receipt and processing of a request, or the monitoring of a polling interval are the functions of the Policy Trigger function. Using the data or parameters in the trigger, this function locates applicable rules in the policy repository. The State and Resource Validation function performs the policy condition evaluation itself.

Event triggers could be the time of day, a repository update, a receipt of an SNMP trap, or notification of user logon.

Requests could be a query from a PEP component for a policy decision as described in Chapter 2, “Policy Enforcement Point” on page 17.

The PDP (Policy Decision Point):

1. Receives the query
2. Locates rules which match the parameters in the query
3. Invokes the State and Resource Validation to evaluate the “matched” rule conditions (such as authenticating with the AAA service and/or checking resource settings)
4. Sends a decision back to the initiator of the query, perhaps in a device-specific format as specified by the PEP interface (using COPS)

3.2.2 State and Resource Validation

Evaluation of policy conditions may involve multiple components (for example, conditions placed on the current state of both the policy server and the device itself). When evaluation applies to a single device this function may be co-located with the PEP as IBM has implemented in its CPE (Common Policy Engine).
The Validation components gather, (optionally) store, and monitor network state and resource information. Upon a request to evaluate a set of policy rules, the Validation function returns a determination of “true” or “false”.

Authentication and authorization checking are often functions provided by the Validation components. Examples include checking the current time of day against the authorized times that a user or application can access certain resources, or checking against the level of service that a user or application can request.

State and resource validation is also concerned with the current availability of network resources. If requested resources are available, then the actions of a policy rule may be executed. Resource availability is dynamic and depends on how resources are currently provisioned in the network and what resources are presently in use by, or reserved for, other traffic.

The state of the network and its individual resources may be obtained by applying measurement, estimation, protocol-specific query, or other techniques. In addition, standard management interfaces, like SNMP, will be utilized. Regardless of how the information is obtained, state and resource data can optionally be stored/cached locally by the Validation components for quick retrieval and processing.

The Device Adapter inside the PDP function takes the canonical representation of policy rules (as stored in the Policy Repository) and maps it into device-specific mechanisms. The device adaptation allows different protocols (e.g., COPS, Telnet/command-line interface, SNMP, network manager proprietary interfaces) to be used to communicate with the network systems and devices.

It is the responsibility of the Device Adapter component to determine which network devices are affected by a policy rule, and then to make decisions based on the actual network devices’ capabilities.
Since there is no stand-alone policy server implemented in IBM’s current product family we will not discuss it further here. For more information on IBM’s current implementation of a co-located PDP/PEP (Policy Decision Point/Policy Enforcement Point), see 2.3, “IBM Common Policy Engine (CPE)” on page 28.

Do NOT confuse a PDP as discussed above with IBM’s SecureWay Policy Director, which is a central control point for defining and coordinating security policies for Web server/browsers as well as specific user authentication needs. For more details see Chapter 5, “Configuring Policies” on page 75.

It certainly would be a very good idea to integrate both policy systems onto one general policy platform and there is work under way at Tivoli Systems as well as in the SecureWay FirstSecure group to do just that.
Chapter 4. Policy Repositories

This chapter discusses the concepts and architecture of a policy repository needed to store the policy rules discussed above. Again there are different instances involved in populating and querying this structured database.

Figure 23. Repository Positioning

The policy repository may have a number of different interfaces enabled, in order to allow different types of users to manipulate the contents of the policy database. The entities which interact with the policy database include:

- Users
- Applications
- Policy administrators
- Network nodes (combined PDP/PEP)
- System and network management administrators

A repository may interact via LDAP with several components as shown in Figure 23:

1. An end-system/application LDAP client, like the one implemented in the CS/390 policy agent or application-oriented extensions.
2. An LDAP client implemented in a co-located PDP/PEP, like the IBM 221x router.
3. A system-management platform populating the LDAP directory, as proposed for the Tivoli Systems Millennium Framework.
4. A policy definition tool which populates the directory based on the definitions created by a policy administrator via a graphical user interface.
For example, a user may indicate that he is going to start a video conference the next day at a specific time. The policy definition tool will allow the appropriate policies to be defined to make sure that he will get the appropriate QoS at that time.

Another scenario is one in which an application asks what kind of parameters it should use for specific traffic patterns before going into service by performing an LDAP query. This will enable the application to set the DSCP or TOS bits according to the network configuration rather than using settings predefined in the application itself. We will talk about this example in detail in 6.1, “Policy-based QoS in End Systems (OS/390, Windows, etc.)” on page 86.

Another example might be a router querying its QoS and VPN policy configuration from the directory, as IBM is doing in its 221x router family.

Figure 23 on page 49 shows the positioning of the repository in our overall system framework.

We will start our discussion of directories with a short introduction to LDAP. After that we will discuss the LDAP information model and protocol related to policy-based networking in more detail.

### 4.1 LDAP Directories

LDAP defines a communication protocol: the transport and format of messages used by a client to access data in an “X.500-like” directory. LDAP does not define the directory service itself. People often talk about LDAP directories; others say that LDAP is only a protocol and that there is no such thing as an LDAP directory.

LDAP has evolved to meet the needs of providing access to a common directory infrastructure. LDAP is an open industry standard that is supported by many system vendors on a variety of platforms. It is being incorporated into software products and is quickly becoming the directory access protocol of choice. LDAP allows products from different vendors on different platforms to interoperate and provide a global directory infrastructure in much the same way as HTTP enabled the deployment of the World Wide Web.

What is an LDAP directory in detail? Let us first consider some basics of the X.500 standard, from which LDAP is derived.

#### 4.1.1 X.500: The Directory Service Standard

The CCITT (Comité Consultatif International Telephonique et Telegraphique or Consultative Committee on International Telephony and Telegraphy, which is nowadays ITU-T, International Telecommunications Union - Telecommunication Standardization Sector) defined the X.500 standard in 1988, which then became ISO 9594, Data Communications Network Directory, Recommendations X.500/X.521 in 1990, though it is still commonly referred to as X.500.

X.500 organizes directory entries in a hierarchical name space capable of supporting large amounts of information. It also defines powerful search capabilities to make information retrieval easier. Because of its functionality and scalability, X.500 is often used together with add-on modules for interoperability between otherwise incompatible directory services.
X.500 specifies that communication between the directory client and the directory server uses the directory access protocol (DAP). However, as an application layer protocol, the DAP requires the entire OSI protocol stack to operate. Supporting the OSI stack requires more resources than are available in many small environments. Therefore, an interface to an X.500 directory server using a less resource-intensive or “lightweight” protocol was desired.

4.1.2 LDAP Server Scenarios

An application client program initiates an LDAP message by calling an LDAP API. But an X.500 directory server does not understand LDAP messages; in fact the LDAP client and X.500 server even use different communication protocols (TCP/IP vs. OSI).

The LDAP client actually communicates with a gateway process that translates and forwards requests to the X.500 directory server. This gateway is known as an LDAP server because it services requests from the LDAP client. It does this by becoming a client of the X.500 server. The LDAP server must communicate using both TCP/IP and OSI. Clients can now access the X.500 directory without dealing with the overhead and complexity which X.500 itself requires.

This scenario is shown in Figure 24 on page 51 and labeled as “RFC 1777” or “LDAP Version 2”.

As the use of LDAP grew and its benefits became apparent, people who did not have X.500 servers or the environments to support them wanted to build directories that could be accessed by LDAP clients. Why not, therefore, have the
LDAP server store and access the directory itself instead of only acting as a gateway to X.500 servers? This eliminates any need for the OSI protocol stack.

Of course, this makes the LDAP server much more complicated since it must now itself be capable of storage and retrieval of directory entries. These types of LDAP servers are often called “stand-alone LDAP servers” because they do not depend on an X.500 directory server. Since LDAP does not support all X.500 capabilities, a stand-alone LDAP server only needs to support the capabilities required by LDAP.

This scenario is shown in Figure 24 on page 51 and labeled as “RFC 2251” or “LDAP Version 3”.

To summarize: RFC 1777 (LDAP Version 2) discusses how to provide access to the X.500 directory. RFC 2251 (LDAP Version 3) discusses how to provide access to directories supporting the X.500 model.

This change in language between Version 2 and Version 3 of the LDAP protocol standard reflects the idea that an LDAP server can implement the directory itself or can be a gateway to an X.500 directory. From the client’s point of view, any server that implements the LDAP protocol is an LDAP directory server, whether the server actually implements the directory or is a gateway to an X.500 server.

In addition, some extensions to LDAP Version 2 were made in Version 3, for example, in the area of “LDAP client referrals from servers”. That means that LDAP Version 3 permits servers to return client referrals to other servers. This allows servers to offload the work of contacting other servers themselves. LDAP extensions become more and more important by using them in a policy-controlled environment, since the repository becomes more and more an integral part of the system operations.

The directory that is accessed can be called an LDAP directory whether the directory is implemented by a stand-alone LDAP server or by an X.500 server.

Current LDAP products support at least the LDAP Version 2 protocol level. Many products already support parts or all of LDAP Version 3. Further enhancements beyond Version 3 are being discussed by the IETF (Internet Engineering Task Force).

4.1.3 Directories and Networks

The Directory Enabled Networks (DEN) specification and its evolution performed by the DMTF, which we have discussed in 1.4, “Standardization Efforts” on page 8, allows information about network configuration, protocol information, router characteristics, policies and so on to be stored in an LDAP directory.

The availability of this information in a standardized format from many equipment vendors will allow for the intelligent management and provisioning of network resources.
4.2 Information Models for Policies

To understand how policies are stored in LDAP directories (accepting the usage of the term) let us first explain how an LDAP directory stores information in general. After that we will discuss the specific differences between the CIM model and the generic LDAP approach.

4.2.1 LDAP Information Model

The LDAP information model is based on a subset of the X.500 information model but is extensible and modifiable. Complex attribute types which are supported in X.500 are not supported in LDAP.

Information is stored in entries which contain attributes. The set of allowed characters for object and attribute names is defined in Lightweight Directory Access Protocol (V3): Attribute Syntax Definitions, RFC 2252 (for example, the underscore character is invalid).

Attributes are typed in the form of <type>=<value> pairs in which the <type> is defined by an object identifier (OID) and the <value> has a defined syntax. Attributes can be single-valued or multi-valued. There is no ordering for multiple values or multi-valued attributes. An entry can contain multiple attributes.

Entries are organized by their distinguished name (DN). Entries whose distinguished name contains the distinguished name of another entry as a suffix are considered to reside under the latter entry in the hierarchy: the namespace is thus hierarchical.

A schema defines rules for distinguished names and what attributes an entry must and/or may contain. To organize the information stored in LDAP directory entries, the schema defines object classes.

An object class consists of a set of mandatory and optional attributes. Every entry in the LDAP directory has an object class associated with it. Thus, every entry in the LDAP directory contains a set of mandatory attributes and (possibly) a subset of the optional attributes based on the entry’s object class and that object classes’ defined schema.

The object class definition also defines where (with respect to other objects in the namespace) it may appear. An object class is declared as abstract, structural, or auxiliary. An abstract object class is used as a template for creating other object classes.

Note

There are a number of different directory vendors in the market: Novell, Netscape, Microsoft and IBM to name just a few. In reality all existing policy systems work (are tested) only with a specific directory, and even when interoperability is a technical possibility you should check very carefully before choosing another vendor’s directory.
The collection of directory entries forms the Directory Information Tree (DIT). The way the DIT is stored is implementation-dependent and hidden from the user of the LDAP directory. For example, the IBM SecureWay Directory uses DB2 as its data store, but this is completely transparent to the LDAP user.

The simplest association between objects is established by their structure in the directory information tree. This is analogous to the structure of a file system directory tree. The DIT hierarchy is a natural method for aggregate associations (for example, “part of” being established by being lower down the branch of a particular tree), but it can also be used for other associations that are one-to-many. Associations encoded in the DIT can provide retrieval of a collection of entries in a single LDAP call. This use must be balanced against the requirement for flexibility in the directory administrator’s choices for DIT structure.

A distinguished name (DN) can be thought of as a directory pointer and, as such, can provide associations between directory entries. The value of an attribute that is defined as having a syntax “DN” can be said to point to an entry named by that DN. The entry that the DN pointer names may be in the same partition or in a different partition as the directory entry that holds the DN pointer. For example, an account entry may contain the DN of the white pages (person) object for the user’s account entry. This is called a forward pointer, and it represents an association from the account object to the person object.

Additionally, the object pointed to by the DN pointer may also contain a pointer back to the originating object. This is called a backward pointer (or simply back pointer). A backpointer in our example represents an association from the person object back to the account object. Both forward and backward pointers are considered valid if there is a method by which referential integrity can be maintained (a guarantee that the values of each pointer are always valid).

There are two kinds of reuse in developing a directory schema:

- Reuse by subclassing from existing directory object classes
- Reuse by applying previously-defined attributes in additional (or new subclass) object classes.

Reuse of object class and attribute definitions helps to eliminate redundancy and, hence, helps to produce a more understandable and usable schema.

The base schema that IBM ships with the IBM SecureWay Directory already defines many object classes and attributes. In developing any extensions to the common schema, one should use the definitions already in the schema where possible. These include the industry-standard definitions that IBM supports plus definitions that IBM has developed for additional use. This statement is also valid for the IBM Common Policy Engine LDAP schema used today.
Whenever possible one should then subclass the objects and define new objects only when the current ones do not meet the needs. This is because the object class is just a mechanism for defining a collection of attributes for the instantiation of a directory entry.

4.2.2 The IBM LDAP Schema

IBM provides an extensive schema with the IBM SecureWay Directory called the IBM schema. The IBM schema is comprised of object classes and attributes defined by industry standards as well as object classes and attributes defined by IBM that are either derived from industry standards or new.

The IBM schema is not a static schema, and new classes and attributes will be added as requirements dictate. The IBM SecureWay Directory schema is based on industry standards where standards exist. It also extends the industry schema as necessary so that IBM products can exploit the directory service and management utilities can manage the data. As the schema for exploiting products is defined, it will be folded into the base schema shipped with the IBM SecureWay Directory.

The industry standards on which the IBM schema is based include:

1. Desktop Management Task Force’s Common Information Model, CIM ([http://www.dmtf.org](http://www.dmtf.org)).
   
   CIM defines schemata for managed elements in a system. These consist of physical objects such as computer systems, software objects, devices, network elements, user related elements, security elements, policies, and so on. IBM incorporates these definitions and creates subclasses from them when reasonable. The Directory Enabled Networks (DEN) schema is incorporated by CIM.

2. Internet Engineering Task Force, IETF ([http://www.ietf.org](http://www.ietf.org)).
   
   There are several RFCs that define schema for LDAP V2 and LDAP V3:
   
   - RFC 1778, 2252, 2256
   
   These RFCs are derived from industry experience and X.500 standards. The set IBM supports in its IBM SecureWay Directory are mainly the objects and attributes related to country, organization, person, group, and their subclasses. There are some object classes and attributes that IBM lists in its schema from RFC 1274, although many of them are superseded by the RFCs listed above.

   
   LIPS is a schema proposed for the storage of user properties. It includes information pertaining to white pages, organization, residence, phone numbers, IDs and passwords. This schema has also been adopted and standardized by The Open Group and specified in the Internet White Pages profile (a profile that directory servers may support). The LIPS object class, along with its attributes, is a subclass of the X.500 person object class.

The schema defined is stored in a special directory entry. The schema contains the following information:
• **Object class** is a collection of attributes. A class can inherit attributes from one or more parent classes.

• **Attribute types** contains information about the attribute, such as name, OID, and matching rules.

• **IBM attribute types** are the implementation-specific attributes, such as database table name, column name, SQL type, and maximum length of each attribute.

• **Syntaxes include:**
  - Distinguished name
  - Telephone number
  - Binary
  - IA5 string
  - Directory string
  - Boolean
  - Integer
  - Generalized time and UTC time
  - Matching Rules supported in the IBM SecureWay Directory V3.1 server are those defined in the LDAPv3 specifications for each syntax.
  - The syntax for schema definitions complies with what is defined in RFC 2252.

The schema is logically divided into several categories for ease of understanding and discussion:

• **Directory server objects** such as top and subschema

• **White pages**, which includes objects such as person, group, country, organization, organizational unit and role, locality, state.

• **Security**, which includes the objects necessary for authorization, authentication, accounting, and audit.

• **Policy and profile**, which include objects for security- and non-security-related policies and profiles.

• **System**, which includes objects such as application, computer, and operating system.

• **Software**, which includes objects such as software inventory.

• **Services**, which includes objects such as services and access points.

• **Other**, which includes objects that are IBM-defined and not yet categorized.

Since we are discussing policy-based control systems, only the schemata with categories policy and profile will be discussed further.

### 4.3 Common Information Model for Policies (CIM)

Let us first sit back again and try to understand what we are trying to achieve here. We are talking about policies, represented by rules, which again name conditions and actions. We need a new language to talk about those rules consistently and a way to represent them in some kind of structured manner. We
need a vocabulary and the associated grammar. This is what an information model and a schema is all about.

The CIM model defines the vocabulary and grammar for abstract policies. LDAP defines the structuring, population and retrieval of information.

4.3.1 CIM Policy Information Framework

The CIM policy information framework is an object-oriented information model for representing policy information currently under development as part of the Common Information Model (CIM) activity in the Distributed Management Task Force (DMTF). This CIM model defines two hierarchies of object classes:

- **Structural classes** represent policy information and control of policies.
- **Relationship classes** indicate how instances of the structural classes are related to each other.

Section 4.4, “Policy Framework Core LDAP Schema” on page 62 shows how to map this information model to a directory that uses LDAP V3 as its access protocol.

The policy classes and relationships defined in the CIM model are generic in order to allow them to represent policies related to anything. Their initial application in the IETF is anticipated to be for the representation of policies related to QoS (DiffServ and IntServ) and to IPSec.

One way to think of a policy-controlled network is to model the network as a state machine and then use policy to control which state a policy-controlled device should be in or is allowed to be in at any given time. Given this approach, policy is applied using a set of policy rules. Each policy rule consists of a set of conditions and a set of actions. Policy rules may be aggregated into policy groups. These groups may be nested, to represent a hierarchy of policies.

The set of conditions associated with a policy rule specifies when the policy rule is applicable. The set of conditions can be expressed as either an ORed set of ANDed sets of condition statements or an ANDed set of ORed sets of statements. Individual condition statements can also be negated. These combinations are termed, respectively, Disjunctive Normal Form (DNF) and Conjunctive Normal Form (CNF) for the conditions.

---

**Note**

Those of us who remember the “logic” or “boolean algebra” classes at school know that there is a whole theory behind the CNF or DNF to allow these forms to represent of all kinds of logical relationships including “implication (=>)” and “equivalence (==)” in a very efficient way.

For those who do not remember, just believe that CNF/DNF are two equally suitable ways to describe logical relationships - such as policy rules - in a standard format.

If the set of conditions associated with a policy rule evaluates to TRUE, then a set of actions that either maintains the current state of the object or changes the object’s state to a new state will be executed.
For the set of actions associated with a policy rule, it is possible to specify an order of execution as well as an indication of whether the order is required or merely recommended or even in fact immaterial.

Policy rules themselves can also be prioritized. One common reason for doing this is to express an overall policy that has a general case with a few specific exceptions; the exceptions would then be prioritized above the general policy rule. For example, a general QoS policy rule might specify that traffic originating from members of the engineering group is to get "bronze" service. A second policy rule might express an exception: traffic originating from John, a specific member of the engineering group, is to get "gold" service. Since traffic originating from John satisfies the conditions of both policy rules, and since the actions associated with the two rules are incompatible, a priority needs to be established. By giving the second rule (the exception) a higher priority than the first rule (the general case), a policy administrator can get the desired effect: traffic originating from John gets "gold" service and traffic originating from all the other members of the engineering group gets "bronze" service.

Policies can either be used in a stand-alone fashion or aggregated into policy groups to perform more elaborate functions. Stand-alone policies are called policy rules; policy groups can be used to model intricate interactions between objects that have complex interdependencies.

---

**Note**

There is a difference between "complex" and "intricate". A car is a complex piece of machinery but its individual components are not very intricate.

---

An example of the use of policy groups is a sophisticated user logon policy that sets up application access, security, and reconfigures network connections based on a combination of user identity, network location, logon method and time of day. A policy group represents a unit of reusability and manageability in that its management is handled by an identifiable group of administrators and its policy rules apply equally to the scope of the policy group.

Stand-alone policies are those that can be expressed in a simple statement. They can be represented effectively in schemata or MIBs. Examples of these are:

- VLAN assignments
- Simple YES/NO QoS requests
- IP address allocations

A specific design goal of CIM is to support both stand-alone and aggregated policies.

Policies really represent business goals and objectives. A translation must be made between these goals and objectives and their realization in the network. An example of this could be a Service Level Agreement (SLA), and its objectives and metrics (Service Level Objectives, or SLOs) that are used to specify services that the network will provide for a given client. The SLA will usually be written in high-level business terminology. SLOs address more specific metrics in support of the SLA. These high-level descriptions of network services and metrics must be translated into lower-level, but also vendor- and device-independent

---

1 This example assumes the "Olympic" model of gold/silver/bronze levels of service, often applicable to service providers.
specifications. The Policy Core Information Model classes are intended to serve as the foundation for these vendor- and device-independent specifications.

It is envisioned that the definition of policy is generic in nature and is applicable to Quality of Service (QoS), to non-QoS networking applications (such as IPSec), and to non-networking applications (backup policies, auditing access, etc.).

4.3.2 Policy Classification

Policy groups and rules can be classified by their purpose and intent; this classification is useful in querying or grouping policy rules. It indicates whether the policy is used to determine when or how an action occurs, or to characterize services (that can then be used, for example, to bind clients to network services).

The following classification is given in the IETF draft:

- **Motivational policies** are solely targeted at whether or how a policy's goal is accomplished. Configuration and Usage policies are specific kinds of Motivational policies. Another example is the scheduling of file backup based on disk write activity from 8 a.m. to 3 p.m., M-F.

- **Configuration policies** define the default (or generic) setup of a managed entity (for example, a network service). Examples of Configuration policies are the setup of a network forwarding service or a network-hosted print queue.

- **Installation policies** define what can and cannot be put on a system or component, as well as the configuration of the mechanisms that perform the install. Installation policies typically represent specific administrative permissions, and can also represent dependencies between different components (for example, to complete the installation of component A, components B and C must be previously successfully installed or uninstalled).

- **Error and Event policies**. For example, if a device fails between 8 a.m. and 9 p.m., call the system administrator, otherwise call the HelpDesk.

- **Usage policies** control the selection and configuration of entities based on specific “usage” data. Configuration policies can be modified or simply re-applied by Usage policies. Examples of Usage policies include upgrading network forwarding services after a user is verified to be a member of a “gold” service group, or reconfiguring a printer to be able to handle the next job in its queue.

- **Security policies** deal with verifying that the client is actually who the client purports to be, permitting or denying access to resources, selecting and applying appropriate authentication mechanisms, and performing accounting and auditing of resources.

- **Service policies** characterize network and other services. For example, all wide-area backbone interfaces shall use a specific type of queuing. Service policies describe services available in the network. Usage policies describe the particular binding of a network client to services available in the network.

These categories are represented in the Policy Core Information Model by special values.

---

2 Policy Framework Core Information Model <draft-ietf-policy-core-info-model-00.txt>; Friday, June 25, 1999, 1:27 PM
4.3.3 Class and Relationship Hierarchies

Figure 25 illustrates the inheritance hierarchy for the core policy classes. It shows that each “Policy” has to be described by a “PolicyGroup”, a “PolicyRule”, a “PolicyCondition” and a “PolicyAction”.

For each class shown above, CIM defines its properties. For example:

- **For “Policy”:**
  
<table>
<thead>
<tr>
<th>NAME</th>
<th>Policy</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION</td>
<td>An abstract class with five properties for describing a policy-related instance.</td>
</tr>
<tr>
<td>DERIVED FROM</td>
<td>Top</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>TRUE</td>
</tr>
<tr>
<td>PROPERTIES</td>
<td>CreationClassName[key]</td>
</tr>
<tr>
<td></td>
<td>CommonName (CN)</td>
</tr>
<tr>
<td></td>
<td>Caption</td>
</tr>
<tr>
<td></td>
<td>Description</td>
</tr>
<tr>
<td></td>
<td>PolicyKeywords[ ]</td>
</tr>
</tbody>
</table>

- **For “PolicyRule”:**

<table>
<thead>
<tr>
<th>NAME</th>
<th>PolicyRule</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION</td>
<td>The central class for representing the “If Condition then Action” semantics associated with a policy rule.</td>
</tr>
<tr>
<td>DERIVED FROM</td>
<td>Policy</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>FALSE</td>
</tr>
<tr>
<td>PROPERTIES</td>
<td>PolicyRuleName[key]</td>
</tr>
<tr>
<td></td>
<td>Enabled</td>
</tr>
<tr>
<td></td>
<td>ConditionListType</td>
</tr>
<tr>
<td></td>
<td>RuleUsage</td>
</tr>
<tr>
<td></td>
<td>Priority</td>
</tr>
<tr>
<td></td>
<td>Mandatory</td>
</tr>
</tbody>
</table>
**For "ConditionList":**

<table>
<thead>
<tr>
<th>NAME</th>
<th>ConditionListType</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION</td>
<td>Indicates whether the list of policy conditions associated with this policy rule is in disjunctive normal form (DNF) or conjunctive normal form (CNF).</td>
</tr>
<tr>
<td>SYNTAX</td>
<td>uint16</td>
</tr>
<tr>
<td>VALUES</td>
<td>DNF(1), CNF(2)</td>
</tr>
<tr>
<td>DEFAULT VALUE</td>
<td>DNF(1)</td>
</tr>
</tbody>
</table>

What is missing is the inheritance hierarchy for relationships to represent the connections between objects:

![Figure 26. CIM Inheritance Hierarchy for Relationships (Grammar)](image)

Relationships are a central feature of information models. A relationship represents a physical or conceptual connection between objects. CIM and DEN define the general concept of an association between two or more objects. Two types of relationships in CIM are aggregations (which express whole-part relationships) and associations, such as those that express dependency.

Figure 26 shows an overview of the inheritance hierarchy as used in CIM today.

Here are some more examples:

**For "ContainedPolicyGroup":**

<table>
<thead>
<tr>
<th>NAME</th>
<th>ContainedPolicyGroup</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION</td>
<td>A class representing the aggregation of PolicyGroups by a higher-level PolicyGroup.</td>
</tr>
<tr>
<td>DERIVED FROM</td>
<td>Top</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>FALSE</td>
</tr>
<tr>
<td>PROPERTIES</td>
<td>ContainingGroup [ref PolicyGroup[0..n]]</td>
</tr>
</tbody>
</table>
The ContainedPolicyGroup aggregation enables policy groups to be nested. This is critical for scalability and manageability, as it enables complex policies to be constructed from multiple simpler policies for administrative convenience. For example, a policy group representing policies for the US might have nested within it policy groups for the Eastern and Western US.

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relationships refer to structural classes not relationships!</td>
</tr>
</tbody>
</table>

- **For "ContainedPolicyRule":**
  
<table>
<thead>
<tr>
<th>NAME</th>
<th>ContainedPolicyRule</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION</td>
<td>A class representing the aggregation of PolicyRules by a PolicyGroup.</td>
</tr>
<tr>
<td>DERIVED FROM</td>
<td>Top</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>FALSE</td>
</tr>
<tr>
<td>PROPERTIES</td>
<td>ContainingGroup[ref PolicyGroup[0..n]]</td>
</tr>
<tr>
<td></td>
<td>ContainedRule[ref PolicyRule[0..n]]</td>
</tr>
</tbody>
</table>

- **For "GroupWithSettingJurisdiction":**
  
<table>
<thead>
<tr>
<th>NAME</th>
<th>GroupWithSettingJurisdiction</th>
</tr>
</thead>
<tbody>
<tr>
<td>DESCRIPTION</td>
<td>A class representing the fact that a PolicyGroup is applicable to a resource represented by a Setting object.</td>
</tr>
<tr>
<td>DERIVED FROM</td>
<td>Top</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>FALSE</td>
</tr>
<tr>
<td>PROPERTIES</td>
<td>GroupScope[ref Setting[0..n]]</td>
</tr>
<tr>
<td></td>
<td>ApplicableGroup[ref PolicyGroup[0..n]]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Properties&quot; in GroupWithSettingJurisdiction associates the policy with its applicable environment (resources, setting).</td>
</tr>
</tbody>
</table>

There are actually four CIM associations that link objects representing resources to which policies apply with the PolicyGroup objects that represent these policies. The fact that there are four associations rather than one reflects how CIM has modeled the resources, not how it has modeled policies.

For those who may need more details please see the latest IETF Information Model Drafts (http://www.ietf.org).

Now, that we have some feeling for the information model, its purpose and specification we will discuss its realization in an LDAP directory.

### 4.4 Policy Framework Core LDAP Schema

To be able to store the policy information model in an LDAP directory we have to map CIM onto LDAP schema, again a different object model.

There are a significant number of differences between CIM and LDAP class specifications:
Instead of LDAP's three class types (abstract, auxiliary, structural), CIM has only two: abstract and instantiable. CIM uses the term "property" for what LDAP terms an "attribute".

CIM uses the array notation "[ ]" to indicate that a property is multi-valued. As is the case with LDAP, multi-valued properties in CIM are unordered.

There is no distinction in a CIM class between mandatory and optional properties. Aside from the key properties (designated for naming instances of the class), all properties are optional.

CIM classes and properties are identified by name, not by OID.

In LDAP, attribute definitions are global, and the same attribute may appear in multiple classes. In CIM, a property is defined within the scope of a single class definition. The property may be inherited into subclasses of the class in which it is defined, but otherwise it cannot appear in other classes. One side effect of this difference is that CIM property names tend to be much shorter than LDAP attribute names, since they are implicitly scoped by the name of the class in which they are defined.

4.4.1 Mapping CIM to LDAP

As mentioned earlier CIM defines two hierarchies of object classes:
1. Structural classes representing policy information and control of policies
2. Relationship classes that indicate how instances of the structural classes are related to each other

In general, both of these class hierarchies will need to be mapped to a particular data store.

The IETF draft defines the mapping of these information model classes to a directory that uses LDAP V3 as its access protocol. Since the mapping of the relationship classes could be done in a number of different ways, there is the risk of non-interoperable implementations. To avoid this possibility, the IETF has defined a single mapping that all implementations using an LDAP directory as their policy repository have to use.

Two types of mappings are involved:
1. For the **structural classes** in the information model, the mapping is basically one-to-one: information model classes map to LDAP classes, and information model properties map to LDAP attributes.
2. For the **relationship classes** in the information model, different mappings are possible. The information model's relationship classes and their properties are mapped in three ways:
   1. To LDAP auxiliary classes
   2. To attributes representing DN pointers
   3. To "composite" attributes representing DN pointers with additional data elements

Let us begin by describing the mapping of the structural classes:
In Figure 27 we see the eight CIM classes mentioned in Figure 25 on page 60 and their mappings to LDAP classes. What is still missing are the mappings for the relationship classes. In overview:

<table>
<thead>
<tr>
<th>CIM</th>
<th>LDAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Policy</td>
<td>policy</td>
</tr>
<tr>
<td>PolicyGroup</td>
<td>policyGroup</td>
</tr>
<tr>
<td>PolicyRule</td>
<td>policyRule</td>
</tr>
<tr>
<td>PolicyCondition</td>
<td>policyCondition</td>
</tr>
<tr>
<td>PolicyAction</td>
<td>policyAction</td>
</tr>
<tr>
<td>VendorPolicyCondition</td>
<td>vendorPolicyCondition</td>
</tr>
<tr>
<td>VendorPolicyAction</td>
<td>vendorPolicyAction</td>
</tr>
<tr>
<td>PolicyTimePeriodCondition</td>
<td>policyTimePeriodCondition</td>
</tr>
</tbody>
</table>

Notice the difference uppercase versus lowercase letter

Figure 28. CIM Relationship Mappings to LDAP Classes and Attributes

4.4.2 Resulting LDAP Directory Structure for Policies

As mentioned in 4.2.1, “LDAP Information Model” on page 53, instances in a directory are identified by distinguished names (DNs), which provide the same type of hierarchical organization that a filesystem provides in a computer system. A distinguished name is a sequence of relative distinguished names (RDNs), where an RDN provides a unique identifier for an instance within the context of its immediate superior, in the same way that a filename provides a unique identifier for a file within the context of the folder in which it resides.

To preserve maximum naming flexibility for policy administrators, each of the structural classes defined in this schema has its own naming attribute. Since the
naming attributes are different, a policy administrator can, by using these attributes, guarantee that there will be no name collisions between instances of different classes.

A key feature of the LDAP core schema is the use of auxiliary classes for modeling policy conditions and policy actions. These auxiliary classes make it possible to model a policy rule in two different ways:

1. **Simple policy rule**: The conditions and/or the actions for the rule are attached to the rule object itself.

2. **Complex policy rule**: The conditions and/or the actions for the rule are attached to instances of the structural class policyInstance; that is, they are not attached directly to the rule object itself.

The trade-offs between simple and complex policy rules are between the efficiency of simple rules and the flexibility and greater potential for reuse of complex rules. With a simple policy rule, the semantic options are limited:

1. All conditions are ANDed together. This combination can be represented in two ways in the DNF/CNF expressions characteristic of policy conditions:
   - As a DNF expression with a single AND group
   - As a CNF expression with multiple single-condition OR groups
2. If multiple actions are included, no order can be specified for them.

Thus if a policy administrator needs to combine conditions in some other way, or if there is a set of actions that must be ordered, then the only option is to use a complex policy rule. The cost of a complex rule lies in the overhead of following DN pointers from the rule object to condition and/or action objects.

4.4.3 Accessing the LDAP Directory

When a PDP goes to an LDAP directory to retrieve the policy object instances relevant to the PEPs it serves, it is faced with two related problems:

1. How does it locate and retrieve the directory entries that apply to its PEPs? These entries may include instances of the core schema classes, instances of domain-specific subclasses of these classes, and instances of other classes modeling such resources as user groups, interfaces, and address ranges.

2. How does it retrieve the directory entries it needs in an efficient manner, so that retrieval of policy information from the directory does not become a roadblock to scaleability?

The placement of objects in the Directory Information Tree (DIT) involves considerations other than how the policy-related objects will be retrieved by a PDP. Consequently, all that the core schema can do is to provide a “toolkit” of classes to assist the policy administrator as the DIT is being designed and built. A PDP should be able to take advantage of any tools that the policy administrator is able to build into the DIT, but it must be able to use a less efficient means of retrieval if that is all it has available to it.

The basic idea behind the LDAP optimization classes is a simple one: make it possible for a PDP to retrieve all the policy-related objects it needs, and only those objects, using as few LDAP calls as possible.
The PDP is initially configured with a DN pointer to some entry in the DIT. The structural class of this entry is not important; the PDP is interested only in the class called `policySubtreesPtrAuxClass` attached to it.

This auxiliary class contains a multi-valued attribute with DN pointers to objects that anchor subtrees containing policy-related objects of interest to the PDP. Since `policySubtreesPtrAuxClass` is an auxiliary class, it can be attached to an entry that the PDP would need to access anyway - perhaps an entry containing initial configuration settings for the PDP, or for a PEP that uses the PDP.

Once it has retrieved the DN pointers, the PDP will direct an LDAP request to each of the objects identified that all entries in its subtree be evaluated against the selection criteria specified in the request. The LDAP-enabled directory then returns all entries in that subtree that satisfy the specified criteria.

The selection criteria always specify that `object class = “policy”`. Since all classes representing policy rules, policy conditions, and policy actions, both in the core schema and in any domain-specific schema derived from it, are subclasses of the abstract class policy, this criterion evaluates to TRUE for all instances of these classes.

To accommodate special cases where a PDP needs to retrieve objects that are not inherently policy-related (for example, an IP address range object pointed to by a subclass of `policyAction` representing the DHCP action “assign from this address range”), the auxiliary class `policyElement` can be used to “tag” an entry, so that it will be found by the selection criterion `object class = “policy”`.

Unfortunately this approach will not work for certain directory implementations, because these implementations do not support matching of auxiliary classes in the `objectClass` attribute. For environments where these implementations are expected to be present, the “tagging” of entries as relevant to policy can be accomplished by inserting the special value “POLICY” into the list of values contained in the `policyKeywords` attribute.

If a PDP needs only a subset of the policy-related objects in the indicated subtrees, then it can be configured with additional selection criteria based on the previously mentioned `policyKeywords` attribute defined in the policy class. This attribute supports both standardized and administrator-defined values. Thus a
PDP could be configured to request only those policy-related objects containing the keywords “DHCP” and “Eastern US”.

To optimize what is expected to be a typical case, the initial request from the client includes not only the object to which its “seed” DN pointer points, but also the subtree contained under this object. The filter for searching this subtree is whatever the client is going to use later to search the other subtrees: “object class = policy”, presence of the keyword “POLICY”, or presence of a more specific policy keyword.

Returning to the example in Figure 29 on page 66, we see that in the best case a PDP can get all the policy-related objects it needs, and only these objects, with exactly three LDAP requests:

1. To its starting object A to get the pointers to B and C, as well as the policy-related objects it needs from the subtree under A
2. and 3. One request each to B and C to get all the policy-related objects that pass the selection criteria with which it was configured

Once it has retrieved all of these objects, the PDP can then traverse their various DN pointers locally to understand the semantic relationships among them. The PDP should also be prepared to find a pointer to another subtree attached to any of the objects it retrieves, and to follow this pointer first, before it follows any of the semantically significant pointers it has received.

This recursion permits a structured approach to identifying related policies. In Figure 29 on page 66, for example, if the subtree under B includes departmental policies and the one under C includes divisional policies, then there might be a pointer from the subtree under C to an object D that roots the subtree of corporate-level policies and so forth.

Since a PDP has no guarantee that the entity that populates the directory won't use the policySubtreesPtrAuxClass, a PDP should understand this class, should be capable of retrieving and processing the entries in the subtrees it points to, and should be capable of doing all of this recursively.

The same requirements apply to any other entity needing to retrieve policy information from the directory. Thus a policy-management tool that retrieves policy entries from the directory in order to perform validation and conflict detection should also understand and be capable of using the policySubtreesPtrAuxClass.

Note

All of these requirements are “should”s rather than “must”s because an LDAP client that doesn't implement them can still access and retrieve the directory entries it needs. The process of doing so will just be less efficient than it would have been if the client had implemented these optimizations.

Finally when it is serving as a tool for creating policy entries in the directory, a policy-management tool should support creation of policySubtreePrtAuxClass entries and their DN pointers.
4.5 IBM LDAP Directory and Policy Database Interaction

As we have extensively discussed in 2.4, “The Policy Feature in the IBM 221x Router Product Family” on page 33, there is a proprietary policy database in each IBM 221x router. Since each router code implements an LDAP client, we can populate this router policy database via an LDAP server.

This avoids the rather uncomfortable task of having to define each policy using the command line interface (which is, however, discussed in 6.4, “Virtual Private Network and QoS Scenarios” on page 94).

The following paragraphs discuss the use of an LDAP server in more detail.

4.5.1 LDAP Overview

The IBM family of 221x routers allows a Lightweight Directory Access Protocol (LDAP) server to be the repository of policy information (the policy database).

The routers support the ability to search for (but not modify) information in the directory server. The policy search agent in the router retrieves all the policy information in the directory server that is intended for that particular device. Any LDAP server operating at LDAP Version 2 or 3 works with the implementation in the router.

An important advantage of using a directory server to store policy information as opposed to more traditional methods of locally-stored configuration is the ability to make a change in one place and have that change applied across all the devices in the extended network.

For example, suppose you have an IPSec definition that resides in the directory. If you want to change the corporate policy for encryption from DES to 3DES, this would normally require a change in every device configuration across each network boundary. If you use the directory to deploy the policies, then you only have to change one IPSec definition. Each policy-enabled device in your network would then need to rebuild the database.

As another example, suppose you need to change a DiffServ action named “GoldService” to increase the bandwidth value from 40% to 45% of bandwidth. The LDAP server and policy infrastructure allow these types of configuration changes to scale much better and they reduce configuration mismatches.

If you are the network administrator, you may also take advantage of the ability to refresh the database automatically at a specified time each day. This option can be set from the command line using the policy feature’s SET REFRESH command.

You may specify whether refreshing is enabled or not and, if enabled, the time at which the database is to refresh. This option is useful for making automated changes. For example, suppose that you must add a new policy so that the marketing department in the U.S. can talk to the development department in Japan across the Internet, and that the security gateways are SG1 and SG2. You can simply enter this information into the directory at any time during the day and have the routers automatically refresh their policy databases at midnight.
The LDAP policy search engine enables you to specify the security level to be used while building the policy database. You define these security options with the policy feature's `SET DEFAULT` command. The options are:

- Pass all traffic during the search (default).
- Drop all traffic except LDAP policy search requests and results.
- Drop all traffic except LDAP policy search requests and results protected by IPSec.

In some situations either of the first two options is sufficient. However, if the LDAP traffic will traverse the public infrastructure, you should secure and authenticate the information by selecting the third option. If you do this, you must select Phase 1 and Phase 2 authentication and encryption options and you must enter the IP addresses for the tunnel endpoints (primary and secondary LDAP servers). This bootstrap IKE/IPSec tunnel will be negotiated before any LDAP traffic is sent. This feature allows you to establish the configuration shown in Figure 30 on page 69.

![Figure 30. Securing LDAP Traffic across the Internet](image)

This figure shows an LDAP server on Subnet A in the corporate network. Three devices, SG1, SG2, and SG3, are fetching their policies from the LDAP server. The policy search for SG2 and SG3 occurs across the Internet and is protected through IPSec.

For all servers, the configuration information required for the policy database to successfully retrieve the policies from the directory is:

- Primary server IP address (a backup secondary server may also be configured)
• Port number on which the server is listening
  (Note: SSL and TLS are not supported.)
• Username and password information if required
• Base distinguished name of the DeviceProfile object for this router or class of routers
• Default policy information

After you have entered this configuration information, the next time the policy database is refreshed an attempt is made to interrogate the directory server for policy information. The policy database allows for a combination of locally configured policies and rules read from the LDAP server. If two rules are found to be conflicting and they are at the same priority, then the rule read from the local configuration takes precedence over the rule read from the directory server.

4.5.2 Policy Schema

The LDAP schema is the set of rules and information making up the class and attribute definitions that determine the contents of entries in the directory.

Typically the LDAP schema is written in ASN.1 syntax, similar to SNMP MIBs. The policy schema that the IBM family of routers supports is based on pre-standard efforts under discussion in the IETF. It is based on the standards track work being done by the IPSec and Policy Working Groups in the IETF and the Policy Working Group in the DMTF. The policy schema closely matches the existing configuration objects in the policy feature on the router. The policy schema definition files and LDAP server configuration files may be found by accessing the following URL:


They are also shown in Appendix A.5, “Example LDIF and Schema Files” on page 123.
Figure 31 shows the overall structure of the policy schema used. The `DeviceProfile` and `DevicePolicyRules` are two key objects in the policy schema. They enable the policy search agent to locate the policies needed for the device. The `DeviceProfile` contains information about the device’s administrative IP address and a mandatory `DevicePolicyRules` reference.

You may group devices together into one `DeviceProfile` or each device in the network can have its own `DeviceProfile`. The choice you make depends on whether more than one device in the network must fetch the same set of rules. Typically, for security gateways this is not the case because every gateway has a different tunnel endpoint. For QoS-only devices, it is conceivable that all devices in a group would all read the same set of policies.

The `DevicePolicyRules` object is retrieved based on the value in the `DeviceProfile` that is fetched for the device. Once the `DevicePolicyRules` object has been retrieved, then the list of `PolicyRules` for that device can be retrieved. If any object is not found or if an error is detected during a consistency check on a object then the search is aborted and messages are displayed to the ELS (Event logging) identifying the error. If an error occurs, the network administrator may configure one of the following choices to handle it:

- Delete all locally-read policies and revert to a “drop all” or “pass all” rule.
- Keep all locally-read policies. Specify this option with the policy feature’s `SET DEFAULT` command.
In either case, the search is attempted again at the configured retry interval. If the primary LDAP server cannot be contacted, then after 5 attempts the secondary server is tried. If the secondary server cannot be reached, then after 5 attempts the primary server is tried again. You can specify the retry interval with the policy feature’s `SET LDAP RETRY-INTERVAL` command. If a search is failing because of network latency, you may change the search timeout from the default of 3 seconds using the policy feature’s `SET LDAP SEARCH-TIMEOUT` command.

### 4.5.3 Policy Rules Configuration

Configure a policy to specify how you want the network to operate. The router translates the policy information into a set of rules that it compares to traffic flows. In the past you may have done this manually by defining inbound and outbound packet filters for each traffic pattern. The policy database eliminates this, because with it you only configure a single policy.

Most of the work is done internally each time the policy database is built. In some cases a router translates a policy directly into a single rule, but in the case of ISAKMP/IPSec it translates a policy into five rules; five rules are needed to account for the traffic directions (in and out) and for the control flows that occur during Phase 1 and Phase 2 of IKE negotiations.

The relationship between policies and rules is as follows:

- One DiffServ policy leads to one DiffServ rule.
- One RSVP policy leads to one RSVP rule.
- One ISAKMP/IPSec policy leads to five ISAKMP/IPSec rules, as in the following example:

#### Example:

Secure the traffic from subnet A to subnet B; the tunnel endpoints are SGa and SGb.

1. Phase 1 inbound (Profile = SGb to SGa, protocol UDP, source port 500, destination port 500):
   
   This rule is needed to filter incoming Phase 1 negotiations from the remote ISAKMP peer if the device is functioning as an ISAKMP responder.

2. Phase 1 outbound (Profile = SGa to SGb, protocol UDP, source port 500, destination port 500):
   
   This rule is needed to filter the Phase 1 information needed if traffic initiates ISAKMP Phase 1 negotiations. In this case the device is functioning as an ISAKMP initiator.

3. Phase 2 inbound (Profile = SGb to SGa, protocol UDP, source port 500, destination port 500):
   
   This rule is needed to filter incoming Phase 2 traffic from the remote ISAKMP peer. This traffic is the result of the remote peer initiating a Phase 2 refresh or initial negotiation. A Phase 2 outbound rule is not needed since the outbound traffic (rule 5) always starts the negotiations if needed.

4. Traffic into the secure tunnel (Profile = subnet A to subnet B):
This rule is needed to put unprotected traffic into a secure tunnel. If the security association has not been negotiated, then the Phase 1 rule is also gathered and IKE starts Phase 1 and Phase 2. Once the SAs have been established, then packets matching this rule are given to IPSec for encapsulation and transmission.

5. Traffic from the secure tunnel (Profile = subnet B to subnet A):

This rule is needed to ensure that packets that should have arrived in a secure tunnel did indeed arrive in a secure tunnel. If the packet was not decapsulated by IPSec and encounters this rule, then the packet is dropped. This rule handles any traffic that is spoofed into the network.

- One IPSec manual-keyed tunnel leads to two IPSec rules, as in the following example:

   **Example:**

   Secure the traffic from subnet A to subnet B; the tunnel endpoints are SGa and SGb.

1. Traffic into the secure tunnel (Profile = subnet A to subnet B):

   This rule is needed to put unprotected traffic into a secure tunnel. This is a statically configured tunnel so it is always available, and packets matching this rule are given directly to IPSec for encapsulation and transmission.

2. Traffic from the secure tunnel (Profile = subnet B to subnet A):

   This rule is needed to ensure that packets that should have arrived in a secure tunnel did indeed arrive in a secure tunnel. If the packet was not decapsulated by IPSec and encounters this rule, then the packet is dropped. This rule handles any traffic that is spoofed into the network.

Actual configuration examples using an LDAP repository can be found in 6.4, “Virtual Private Network and QoS Scenarios” on page 94.
Chapter 5. Configuring Policies

In the preceding chapters we discussed the architecture and concepts behind policy based system controls. We discovered different information models that are necessary to represent policies at different levels of abstraction.

Each of the defining authorities has a different view on the subject. As the complexity and scope of corporate policies become increasingly intricate (going up the chain of business operations), standardization bodies and vendors are starting to implement policy definition tools at the very bottom of the policy value chain.

These initial approaches are typically guided by optimizing the business value and applying these new concepts to an existing corporate infrastructure.

Today's most promising business values enabled with policy-based control are:
1. Realization of QoS mechanisms for new Internet services implemented by ISPs.
2. Provisioning of VPNs (Virtual Private Networks), both to solve the security issue of using the public Internet for corporate traffic and at the same time simplifying the process of configuration.
3. Controlling Web access based on content scope and user domain.

Inside IBM today, at the end of 1999, there are two policy definition tools either available or in the process of becoming available:
1. The Nways Policy Definition Tool for the Common Policy Engine for QoS and VPNs for our 221x router family running MRS/MAS/AIS V3.3
2. The SecureWay Policy Director

Today, these offerings are complementary - the SecureWay Policy Director focuses on coordinatung overall security policy, with the enforcement of this policy limited to access control of Web server resources, while the Common Policy Engine will provide enforcement for VPN and related networking policy.

IBM's intention is to evolve to a state in which:
1. IBM's SecureWay Policy Director provides definition and coordination of security policies for e-business applications, as well as enforcement for Web server resource authorization (access control) policies today. In the coming years, it will expand to define and coordinate related network policy, and integrate with numerous additional policy enforcement engines.
2. IBM's Common Policy Engine will provide network and VPN policy enforcement for customers who require a hardware solution, with policy definition and coordination provided by the Nways Manager, as well as the SecureWay Policy Director and/or Tivoli over time.

Since this book is discussing NHD's “Application Driven Networking” approach, we will only cover the Nways Policy Definition Tool for the IBM router CPE (Common Policy Engine).
5.1 CPE Policy Definition Tool

Coming back to our overall picture, we will next discuss the reasoning and approach taken by IBM behind the implementation of its policy definition tool.

Figure 32. Policy Definition Tool

Figure 32 shows the relationships between a policy definition tool and:

1. A user or administrator accessing the graphical user interface directly or via a browser interface.
2. The policy directory which gets populated by the tool.
3. The system- and network-management platform which retrieves information about the system environment and uses it as an alternate path to influence the policy-aware devices.

The following discussion is one of IBM policy definition tools in general rather than of a specific product. This is because IBM’s actual policy definition tool has not yet been formally announced.

5.1.1 Policy Configuration Overview

Policy configuration tools tackle the problem of populating the LDAP policy repository which is then used to distribute rules to network devices or PDPs (Policy Decision Points).

As we have seen in previous chapters an LDAP directory provides a specific way to store information: a schema. The IETF and DMTF are working on an information model (including the mapping to LDAP) which will define a common base for specifying generic policies independent of specific policy scenarios like...
QoS or VPNs. We have discussed this in 4.2, “Information Models for Policies” on page 53.

The goal of a policy definition tool is to offload the burden of defining policies according to the formal LDAP schema and instead to provide a graphical user interface and methodology to simplify the definition task. That said, a policy tool is always implemented under the assumption that a specific information model for policies is being used; the same assumption is also true for PDPs (Policy Decision Points) querying the same repository. Put another way, the LDAP repository has to be able to store IBM-specific schema information, and therefore the IBM policy tool has to be based on an IBM pre-standard schema too.

For an overview of how to customize specific LDAP directories to be able to store policies, see 4.5, “IBM LDAP Directory and Policy Database Interaction” on page 68, A.4, “LDAP Directory Initialization” on page 118 and A.5, “Example LDIF and Schema Files” on page 123.

5.1.2 Tool Methodology

The Nways Policy Configuration application provides a set of functions needed to configure Virtual Private Network (VPN) and Quality of Service (QoS) policy definitions. The policy definitions are stored in a Lightweight Directory Access Protocol (LDAP) server and subsequently downloaded to the appropriate devices.

The Policy Configuration application has been integrated into the existing Nways VPN Management application which provides monitoring, event reporting, troubleshooting, operational control and application launching functions.

The initial release of the application supports the VPN/QoS capabilities contained in the V3.3 release for the IBM 221x Router.

5.1.2.1 Modelling Policies

A policy consists of a condition and an action. When a VPN/QoS device receives network traffic, the policy conditions are searched for a match. If a condition match is found, the associated action is performed.

The following IF-THEN statement illustrates the enforcement of a policy:

\[
\text{IF } \text{network traffic MATCHES condition } \rightarrow \text{THEN perform action}
\]

The condition contains a validity period and a traffic profile. The validity period defines the time frame in which the action should be performed. The traffic profile defines the type of network traffic for which the action should be performed.

The action is actually composed of one or more sub-actions. These sub-actions consist of:

- An IPSec action
- A Differentiated Services (DiffServ) action
- A ReSerVation Protocol (RSVP) action
The IPSec action is further composed of:

- An IPSec key management action
- An IPSec data management action

Once we consider this proliferation of actions even in a relatively small network such as that shown in Figure 33 on page 78, the total number of rules which needs to be specified can be rather large; depending on the granularity of QoS classes (by subnet, by IP-address or by user/application) we may in fact have hundreds of rules to specify. Another example is the provisioning of VPN tunnels. Assuming a 4-node network and tunnels between each node (as shown in the core of Figure 33) we require 6 tunnels, which then needs to be multiplied by 5 to make 30 rules which need to be defined (see 4.5.3, “Policy Rules Configuration” on page 72).

Expanding this thought to a network of 16 nodes in an rather small ISP-VPN would require 120 tunnels multiplied by 5, or 600 rules.

The question now is: What defines a policy in detail and how can we structure this specification problem to limit the amount and number of separate pieces of information we need to specify?

---

**Policy Definitions: How Many?**

<table>
<thead>
<tr>
<th>IPSec Policies/Rules:</th>
</tr>
</thead>
<tbody>
<tr>
<td>- #Tunnels x 2 if manual keyed tunnel are used</td>
</tr>
<tr>
<td>- #Tunnels x 5 if ISAKMP is used</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DiffServ/RSVP Policies/Rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>- n, where n is the number of QOS classes/flows</td>
</tr>
</tbody>
</table>

*Figure 33. Total Number of Policies in a Simple Network*

To reiterate, a policy must contain at least one policy condition consisting of a validity period and a traffic profile and at least one policy action.

We can classify those configuration elements as containing either device-dependent or device-independent information as follows:

- The *validity period* component may be referred to from multiple policies as it does not contain any device-specific information.
• The traffic profile component is unique to each policy as it contains device-specific IP address information.

• The key management action and key management proposal components of the IPSec action may both be used by multiple policies as neither contains any device-specific information.

• The data management action component of the IPSec action is unique to the policy as it contains device-specific IP address information.

• The data management action, data management proposal, authentication header (AH) transform and encapsulated security payload (ESP) transform components of the IPSec action may be referred to from multiple policies as none of them contains device-specific information.

• The RSVP action and the DiffServ action component may be used by multiple policies as neither contains any device-specific information.

The following table summarizes these relationships:

Table 2. Policy Component Relationships

<table>
<thead>
<tr>
<th>Policy Components</th>
<th>Device Specific Data Relationships</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Conditions:</strong></td>
<td></td>
</tr>
<tr>
<td>Validity period</td>
<td>May be shared by multiple policies</td>
</tr>
<tr>
<td>Traffic profile</td>
<td>Unique to policy</td>
</tr>
<tr>
<td><strong>IPSec actions:</strong></td>
<td></td>
</tr>
<tr>
<td>Key management action (KMA)</td>
<td>May be shared by multiple policies</td>
</tr>
<tr>
<td>Key management proposal (KMP)</td>
<td>May be shared by KMAs</td>
</tr>
<tr>
<td>Data management action (DMA)</td>
<td>Unique to policy</td>
</tr>
<tr>
<td>Data management proposal (DMP)</td>
<td>May be shared by DMAs</td>
</tr>
<tr>
<td>AH transform</td>
<td>May be shared by DMPs</td>
</tr>
<tr>
<td>ESP transform</td>
<td>May be shared by DMPs</td>
</tr>
<tr>
<td><strong>QoS actions:</strong></td>
<td></td>
</tr>
<tr>
<td>RSVP</td>
<td>May be shared</td>
</tr>
<tr>
<td>DiffServ</td>
<td>May be shared</td>
</tr>
</tbody>
</table>

5.1.2.2 Policy Configuration

As we have discussed in 1.5, “Policy-Based Networking Architecture” on page 11, a network is represented as the sum of all the administrative domains, in which an administrative domain can consist of multiple policy segments. A policy segment, in turn, comprises a group of devices which need to enforce a group of common policy components.

Initially, there will be policy segments for IPSec, DiffServ and for RSVP. Other types of policy segments may be added to the definition tool. An individual device may be a part of any number of policy segments, which means that each individual device can have no policies applied to it or can have many policies applied to it; this depends on the role the device plays in the network’s security and/or QoS implementation.
1. IPSec policy segments

An IPSec policy segment is used to define a VPN network composed of IPSec tunnels between devices, based on a group of common policy components. Individual IPSec policy segments may or may not be connected. A specific IPSec policy segment consists of the following:

- IPSec device list
- IPSec policy template
- Connection type (mesh, star, device pair)
- Hub name (for star connection types)

2. Differentiated Services (DiffServ) policy segments

An DiffServ policy segment is used to define a QoS network composed of a collection of devices which share a group of common policy components. DiffServ is a unidirectional connectionless hop-by-hop QoS protocol. A specific QoS policy segment consists of the following:

- QoS device list
- Differentiated Services policy template
- QoS traffic template

3. RSVP policy segments

An RSVP policy segment is used to define a QoS network composed of RSVP connections between devices based on a group of common policy components. RSVP is a unidirectional end-to-end QoS signalling protocol. It should be noted however that the policies for RSVP do not contain information about the actual devices which may form the RSVP connection; the policies
only define what scope of RSVP reservations can be granted by the network device when reservation requests are received from elsewhere. Individual RSVP policy segments may or may not be connected. A specific RSVP policy segment consists of the following:

- QoS device list
- RSVP policy template
- QoS traffic template

5.1.3 Configuration Task for Policies

Table 3. Configuration Components per Discipline

<table>
<thead>
<tr>
<th>Elements</th>
<th>IPSec</th>
<th>RSVP</th>
<th>DiffServ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment type</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Device list</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Policy template</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Connection type</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Hub name (if connection type is star)</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>QoS traffic template</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

Table 3 shows that there are only a few components which distinguish the policy configuration tasks between each discipline. The following short paragraphs give a brief overview of the possible configuration parameters. To understand their significance an understanding of RSVP, DiffServ and IPSec is assumed; without this understanding this list should still provide a flavor of the contents of policy definitions.

To learn about DiffServ, IPSec and RSVP in detail see:

- A Comprehensive Guide to Virtual Private Networks, Volume I: IBM Firewall, Server and Client Solutions, SG24-5201
- Application Driven Networking: Class of Service in IP, Ethernet and ATM Networks, SG24-5384

5.1.3.1 Device List

This is the list of devices to which the policy should be applied. A device can be part of many different device lists.

5.1.3.2 Policy Template

A policy template represents the if-then-action structure, defining:

1. A priority
2. A status (enabled, disabled)
3. A condition, referring to a validity period and a traffic profile
4. References to several actions for IPSec, RSVP and/or DiffServ

5.1.3.3 Policy Condition
A policy condition is typically described by the following parameters:

Validity Period
- Start time, stop time
- Valid months, valid days
- Start each day at time, stop each day at time

Traffic Profile
- Protocol number range (low, high)
- Source traffic port range (low, high)
- Destination traffic port range (low, high)
- Type of service (TOS mask, TOS match)
- IP address range

5.1.3.4 Policy Actions
Policy actions define the actions to take after the conditions are met for a specific device list and traffic profile in a policy discipline.

For DiffServ the following parameters can be defined:
- Permission (permit, deny traffic)
- TOS byte handling
- Bandwidth reservation handling (absolute, relative)
- Share of bandwidth to reserve
- Queuing priority

For RSVP the following parameters can be defined:
- Permission (permit, deny traffic)
- Maximum rate (in kbps)
- Maximum flow duration (in seconds)
- A DiffServ action

The task of defining a VPN tunnel between a significant number of nodes with the associated correct key management information is not a trivial task.

Note:
To ease the burden of defining similar but slightly different configurations per VPN node, the configuration tool has introduced the concept of an IPSec segment as being meshed, a device pair or a star. That means the configuration tool will generate all necessary IPSec parameters for all IPSec device list members.

For IPSec the following parameters can be defined:
- Key management actions
• Auto start (enable, disable)
• Exchange mode (main, aggressive)
• Security association lifetime (in seconds)
• Security association life size (in kBps)
• Security association refresh percentage
• (1 to 5) key management proposals
  • Authentication methods (pre-shared, RSA-signalled)
  • Hash algorithms (MD5, SHA)
  • Cipher algorithm (DES, 3DES)
  • Diffie Hellmann group (1 or 2)
  • SA lifetime
  • SA lifesize

• Data management actions
  • Permission (permit, deny)
  • Auto start (enable, disable)
  • SA refresh percent
  • SA refresh threshold
  • Copy don’t fragment bit (copy, set, clear, none)
  • Replay prevention (enable, disable)
  • (1 to 5) data management proposals
  • Perfect-Forward-Secrecy (enabled, disabled)
  • Diffie Hellmann group (1 or 2)
  • AH transforms (1 to 5) with SA lifetime, SA lifesize, encapsulation mode (tunnel, transport) and integrity algorithm (HMAC-MD5, HMAC-SHA)
  • ESP transforms (1 to 5) with SA lifetime, SA lifesize, encapsulation mode (tunnel, transport), integrity algorithm (none, HMAC-MD5, HMAC-SHA) and cipher algorithm (none, DES, 3DES, CDMF)

5.2 System and Network Management

In this chapter we have so far discussed an overview of the methods used by a dedicated configuration tool to populate an LDAP directory with policy information. FirstSecure SecureWay and Tivoli Systems are also working on the next generation of repository, security and system and network management technologies and frameworks.

As you can imagine there is always a mismatch between time-to-market optimization and integration efforts across brands and technology areas. Today IBM is working on the definition of a global security and policy framework across all brands and platforms.

As of today in both areas (security, policy networking (QoS, VPN)) only dedicated tools are available or in the process of becoming available.
In addition, Tivoli Systems is working on the Millennium Management Framework. Part of this framework includes two umbrella functions called:

- GEM, Global Enterprise Manager
- SAM, Service Level Manager

Both will play a significant role in integrating both aspects of policy-based system control, namely:

- The ease of configuration of complex communication systems (VPN, security,...)
- The flexible and dynamic implementation of end-to-end QoS mechanisms.
Chapter 6. User, Application and Implementation Scenarios

Application-driven networking is about providing end-to-end policy-based control. Applications and users in a system are the entities that define the communication needs which the building blocks of the network are then configured to implement.

The following paragraphs give an overview of a policy agent implemented for IBM CS/390, some Windows-related activities in this arena and finally some implementation scenarios for IPSec and QoS provisioning via policy systems using both the command line interface to the router and the LDAP repository.

For those readers primarily interested in some router example configurations, go immediately to 6.4, “Virtual Private Network and QoS Scenarios” on page 94.

As shown in Figure 36, applications and users have different means to communicate or signal their communication needs. Most of these mechanisms complement each other.

There may be means for a user or an application to interface to the System Management environment to indicate certain actions to be performed in the near future (relationship 3 above). For example, this may be a planned video conference or a new software version which will need more communication resources. There are other methods by which policies are created and modified: by a direct connection to the policy definition tool (2) or even to the LDAP directory (1) via some functionally reduced instruction set using HTML, Java, and so on.
The previous relationships are concerned with users and applications modifying the policies themselves; many network nodes are configured to react in “real time” to DiffServ Code Points (DSCPs) or RSVP reservation signals sent through the network by the application (4).

End systems may choose a mixture of these approaches; for example, the policy agent (PAGENT) in CS/390 is based on a mixed approach where the end system retrieves information from the LDAP repository and uses the information to determine how to signal QoS parameters in its IP traffic: RSVP signals or DiffServ Code Points.

Windows is taking a slightly different route by using an ACS (Admission Control Service) between the end system and the LDAP directory. This approach is very much like the PDP approach in the IETF, but extensions to RSVP are used instead of using COPS. For more details about this approach see draft-ietf-issll-is802-sbm-08.txt

6.1 Policy-based QoS in End Systems (OS/390, Windows, etc.)

Contrary to existing TCP/IP networks, which treat all traffic (IP datagrams) equally as “best effort” traffic, QoS provisioning or service differentiation is a mechanism to provide different service levels to different traffic types based on their requirements and importance in an enterprise network.

For example, it may be critical to provide ERP (Enterprise Resource Planning) traffic with better service during peak hours (in order to cope with high network load) compared to FTP traffic or traffic generated by Web browsers. For a service differentiation to be effective, it is required that all network elements (end-to-end) implement mechanisms to differentiate traffic (by providing different levels of priority, for example) based on a consistent service level policy.

A directory is typically used as a mechanism to manage and distribute service level policies to network elements/devices across different vendors’ platforms.

In addition to enabling QoS policy in the network, it is essential to monitor the performance of each QoS policy to guarantee if the QoS requirement is met. This brings up the need for service level management information to be kept by the network elements (Policy Decision Point/Policy Enforcement Points - PDP/PEP), which can then be used and analyzed for a variety of reasons: from traffic trend analysis to network capacity planning, and dynamic QoS level tuning.

It is often suggested that adding bandwidth is one way to provide enhanced network performance. There are several shortcomings in this approach, not the least of which is the cost efficiency. Bandwidth cost will always remain a significant factor in network planning as new applications continually generate more traffic than the increase in bandwidth (for a given cost) can accommodate. This trend will continue as more powerful processing capability becomes available and more devices are connected to the network.

Furthermore, QoS does not involve only bandwidth but also the local node processing and resources (buffers, I/O etc.). Decisions to buffer an outbound packet or discard it are just as important for a router/switch/host as the decision on which packet to transmit next on the link.

1 SBM (Subnet Bandwidth Manager): A Protocol for RSVP-based Admission Control over IEEE 802-style networks
In order to address the wide range of application service requirements, there are two distinct forms of services being defined in the IETF. The first, Integrated Services (IntServ), is an end-to-end reservation-based service that uses the Resource reSerVation Protocol (RSVP is a signaling protocol) to request an appropriate level of service for specific “sessions/flows”.

Services of the second form, Differentiated Services (DiffServ), are intended to provide service differentiation between broad classes of users and applications. DiffServ is a form of aggregation of traffic with the same network service provision. The IETF has recently proposed a standard format for the Differentiated Services byte in the IP packet header (the same as the Type of Service - TOS- byte that has been defined in earlier standards) to carry different classes of service along with some initial defined classes (Assured Forwarding, Expedited Forwarding etc).

RSVP does require network nodes (including edge/host nodes) to maintain individual reservation states; DiffServ only requires each network node to provide a service level according to the DS field (called a “per hop behavior”). RSVP has a problem with scalability when it comes to carrier/backbone networks. This problem is being addressed and different mechanisms have been proposed for it. One is to use RSVP to request a reservation for a group of flows (a form of aggregation), while another is to use it to provide admission control only for the access network (that is, RSVP is used at the edge and DiffServ is used in the carrier/service provider networks). In any case, RSVP and DiffServ can complement each other in providing end-to-end QoS.

As mentioned above, service level policy is required to control QoS levels for different users, hosts, and applications at different times. With RSVP, a way to control who can reserve, how much to reserve, and how many reservations are allowed is required. With DiffServ, the definition of a policy for each class of traffic is needed, for example, the priority of all FTP traffic from one subnet to another subnet, how much bandwidth can be used by the traffic class and so on.

### 6.2 CS/390 PEN Implementation

Terminology: System/390 (S/390) mainframes can implement the Operating System/390 (OS/390) operating system; Communications Server for OS/390 (CS/390) is the “networking” software component of OS/390 which runs on the mainframe and encompasses products such as VTAM (Virtual Telecommunications Access Method) and TCP/IP (Transmission Control Protocol/Internet Protocol). Other operating systems (such as those abbreviated as VM, VSE and TPF) also exist for the S/390 hardware platform; these operating systems will not be discussed here.

The objective of implementing QoS and policy-enabled network (PEN) functions for S/390 mainframes is to make it the “platform of choice” for e-business. This requires S/390 to support mixed traffic types: from ERP traffic, interactive traffic in general to multimedia traffic and “batch” file transfer traffic, all of which require different QoS levels.

Furthermore, each user specifies certain service level agreements that need to be satisfied. Accomplishing this objective means the following functions need to be supported:
• **Definition**: enable network administrators to define service level policies that are simple and flexible in specifying end-to-end service levels.

• **Enforcement**: provide hardware and software infrastructure to enforce the QoS service level policies.

• **Monitoring**: monitor policy performance to make sure service levels are met.

• **Tuning**: provide automatic mechanisms to tune the policies in order to correct the policy performance and to notify system administrators of discrepancies.

The next two sections will describe the QoS functions that are supported in different releases of OS/390 (from V2R7 to V2R8) and will talk about potential future functions in later releases.

**6.2.1 Policy Control on CS for OS/390 Release 7**

Figure 37 shows the structure of CS/390 components that provide the QoS and policy function:

As shown in Figure 37, there are several software components required to support QoS and policy functions. We will describe them in the context of the two service types mentioned before: DiffServ and IntServ (RSVP). Some of the software components are common to both types of service.

A policy agent (PAGENT) in CS/390 Release 7 supports the retrieval of policies either from a file or from an LDAP server. Each instance of PAGENT can manage policies for multiple TCP/IP stacks (images) defined within an S/390 logical

---

2 OS/390 releases are scheduled for March and September of each year; Version 2 Release 8 was released in September 1999. All OS/390 releases under discussion here are “Version 2”.

---

Figure 37. CS/390 QoS and Policy Structure
partition (LPAR). It is possible for PAGENT to read policies from a configuration file for one stack and retrieve policies from an LDAP server for another.

PAGENT does not support reloading policies when they are updated; PAGENT must be terminated and restarted for the updated policies to take effect. When PAGENT is active, it adds or deletes policies as they become active or inactive based on their original time-of-day specification.

6.2.1.1 Policy Overview
Initially only the QoS policy schema version originally defined by IBM research is supported.

The packet classification (how to map a packet to a policy rule definition in order to find a corresponding service for it) is based on a collection of parameters which includes:

- The source address (range)
- The destination address (range)
- The protocol identifier
- The source port number (range)
- The destination port number (range)
- The local outbound interface (subnet identifier)

Policies can also be used to re-define the default mapping of TOS/DS to S/390 QDIO priorities. This is akin to a “local transmission priority” which is used for queueing outbound traffic over QDIO devices; the only current implementation of QDIO is with the Gigabit Ethernet Open Systems Adapter (OSA-Express). Note also that, in addition, users can define the QDIO blocking or non-blocking option for each TOS/DS value. This blocking option is not documented.

It is important to note that the policy support is designed for S/390 as a host and not as a router. This enables us to make significant performance optimization in QoS policy mapping and classification. Specifically, all policy handling is performed at the transport layer instead of at the network (IP) layer; per-packet processing is therefore avoided. Each TCP connection is assigned a policy handle which is kept for its duration. Each UDP/RAW session/socket is assigned a policy handle as long as its packets are sent to the same destination.

6.2.1.2 DiffServ
PAGENT can be configured to specify the TOS/DS byte for appropriate hosts/applications (such as for specific IP addresses or port numbers).

In addition, service level policy can be used to control TCP minimum and maximum throughput (amount of data transfer over one round-trip delay) and admissions - “the number of active TCP connections that are allowed at any given time for a given set of applications”. Admission control is useful in limiting application access during certain times (for example, “no more than 50 Web connections from a particular subnet to the server between 8 a.m. and 10 a.m.”).

By specifying “minimum throughput”, a TCP connection will not allow its throughput to go below this value (assuming there is application data to be sent) unless a time-out is experienced (a time out which causes the TCP transmission window to be reset to 1, an indication that congestion is taking place in the
network). A TCP connection with a minimum throughput specified will take priority over other TCP connections which either have no minimum throughput specification or a lower minimum throughput specification under circumstances of network congestion/saturation.

Specifying “maximum throughput”, conversely, allows users to control how fast a TCP connection can transmit its data even when the network is not congested. There are a couple of cases in which this may be useful;

1. When users access the network with a low-speed modem: by specifying the maximum throughput (such as 28 kbps or 56 kbps), TCP will not attempt to push its throughput beyond this maximum and therefore can avoid the “sawtooth” nature of its congestion window, resulting in higher overall throughput.

2. When there is a desire to limit some application’s throughput (FTP, for example) during peak hours to reserve bandwidth for other applications.

The TOS/DS byte will determine what priority queue (one of four) a packet will be sent to if it is destined to a QDIO device. The TOS/DS value of a packet will be respected as it traverses the network which supports DiffServ.

6.2.1.3 IntServ (RSVP)
There is no IntServ support in the formal OS/390 V2R7 release; a prototype of RSVP that can run with R7 is available for download at no cost. This version of RSVP has support for marking the TOS/DS byte and for making reservations over ATM networks.

6.2.2 Policy Control on CS for OS/390 Release 8
RSVP (IntServ) becomes a formal part of this OS/390 release. It also uses the new Policy API (PAPI) to issue queries for RSVP policies when processing reservation requests.

An SNMP SLA MIB subagent is added for policy performance monitoring.

6.2.2.1 PAGENT Enhancements
PAGENT can now allow dynamic policy updates in the following ways:

- An update period can be defined, and at the beginning of each period, PAGENT will check to see if either the policy configuration file has been updated or policies from the LDAP server have been changed since last time; if so, PAGENT will update its local and stack policies accordingly;

- Another option is for PAGENT to be triggered following an update so that as soon as the policy configuration file is updated, PAGENT will immediately check for changes.

- In addition, PAGENT will monitor for a TCP/IP image being recycled. Upon detecting that an image is recycled, PAGENT will re-install all policies to that image.

PAGENT is also enhanced to accept policy requests via a Policy API (PAPI). The first user of this API will be RSVPD (the RSVP “daemon”).
6.2.2.2 PAPI (Policy API)
The Policy API enables applications themselves to issue policy queries (as opposed to simply being able to configure the TCP/IP “stack” itself). The first user is RSVPD, which queries RSVP policies to admit/deny incoming and/or outgoing reservation requests. This new PAPI utilizes the local socket for communication between a client application and the server PAGENT. As a result, remote clients will not be able to access the PAGENT server (in other words, the clients can not use TCP or UDP services to access a PAGENT server residing in a different OS/390 image).

6.2.2.3 Policy Schemata Enhancements
Additional policy schemata to support RSVP (IntServ) are supported in the service policy. These new schemata enable specifications such as:

- The total number of RSVP flows allowed
- The maximum token bucket size
- The mean rate for each flow

6.2.2.4 DiffServ
No additional support for DiffServ is provided in this release.

6.2.2.5 IntServ
RSVP is now supported in this release. The RSVP API (RAPI) has been updated with the latest RAPI specification from the Open Group standard. RSVPD is also enhanced to use PAPI for processing reservation requests. Depending on the RSVP policy specification, RSVP can change the reservation parameters or deny a reservation altogether.

Note that users can specify in an RSVP service policy the number of RSVP flows allowed and how much each flow can reserve. An RSVP reservation request that exceeds the number of flows allowed will be denied.

As with the RSVP prototype in CS/390 V2R7, reservation itself is only made over ATM subnets, in which a separate SVC is established for each reservation request. Otherwise, RSVP signalling will cause network devices such as routers to establish resource reservations elsewhere in the network.

RSVP-reserved traffic can be specified to carry a specific TOS/DS value so that it can be treated accordingly as it traverses the network.

6.2.2.6 SNMP SLA MIB Subagent
Statistics such as number of outbound/inbound/discarded packets, number of active/denied/total TCP connections etc. are now collected for each policy in the TCP/IP stack. They are retrieved by the SNMP SLA MIB subagent and monitored for possible SLA (policy performance specification) performance deviations.

For instance, if policy specification indicates that the minimum rate of a TCP connection is 256 kbps, upon detecting that a TCP connection belonging to this policy falls below this minimum (for example, due to network congestions), an SNMP trap is sent out for processing by any network management platform.

The subagent also monitors the minimum/maximum delays. Thresholds can be set, together with some of the subagent internal filtering, to minimize the number of traps sent (note that a trap is also sent when policy performance returns to the
specified level) and false notifications (a TCP connection can be idled during the monitoring period and will result in low throughput).

With respect to RSVP policy statistics, only the number of inbound/outbound packets for which an actual reservation is made are collected along with policy timestamps. It's important to note that RSVP is unidirectional; that is a reservation is only made in one direction. Therefore, an actual reservation (such as one reserving bandwidth) is only made for the direction of the data flow. As a result, the sender will have an actual reservation record kept in the TCP/IP stack, but it doesn't exist at the receiver. Only the PAGENT and the RSVPD have any knowledge of the inbound reservation.

### 6.3 Policy Activities on Windows 2000

Unlike existing Windows platforms (Windows 95, 98 and NT), Windows 2000 will provide support for making RSVP reservations across the network to which the Windows 2000 machine is connected. The following charts from Microsoft give some idea of the types of QoS (layer 2 and layer 3) which will be supported on Windows 2000.

#### 2 QoS Models

- **Reservation Model - e.g. RSVP, ATM, etc.**
  - Signaling of service request thru network
    - IETF RSVP protocol
  - Switch/Router based per-flow Queuing
    - Some media has HW support - ATM

- **Precedence/Priority/Differentiated Model**
  - Packet is marked
    - IP Precedence, TOS (layer 3)
    - IEEE 802.1p (layer 2)
  - Switch/Router fabric preferentially handles packet based on markings
    - several models- priority queuing, preferential drop policies, WFQ, etc.

- **Microsoft's QoS Plan supports and integrates both models**

*Figure 38. QoS Models Supported by Microsoft*
Admission Control Services

- Admission Control Servers in the Network
  - ACS can also run in routers or switches
- User ID carried in signaling messages
- Policies are maintained in the Directory
  - ACS uses LDAP to retrieve Policy Information
- ACS Policy is per subnetwork/per user
  - Can be abstracted to per Enterprise/Per Group
- Enables approval/denial of resources based on user ID, service requested, etc.
- Can map reservation requests into priority

Figure 39. Admission Control Service, Policy Server

ACS Management Model

- Network Admin uses Microsoft Management Console to Administer Policy in the Directory Service
  - User Object is extended to permit a mapping from a User to a Profile Group
    - e.g. Redmond\PeterF -> ProgManagers
  - Default policies at Enterprise Level
    - “All users can reserve up to 500 Kbps”
    - Enterprise-wide User, Profile policies
- Per Subnetwork Policies
  - Individual Users and Profiles

Figure 40. Policy Configuration Environment
6.4 Virtual Private Network and QoS Scenarios

The following examples show the actual use of IBM routers and the methods used to configure them.

6.4.1 Configuring Policies via Command Line Interface

First, access the policy feature as shown:

* `talk 6`
  `Config>feature policy`
  `IP Network Policy configuration`

The following examples will use the network shown in Figure 42 on page 95, for which the policy statement is:

---

**The policy statement:**

Secure the traffic from subnet 11 to subnet 12 with the tunnel endpoints being SG1 and SG2, and provide a QoS for the traffic in this tunnel by means of DiffServ GoldService.

---

You may enter policy information in either of two ways. The first way is to define the individual policy objects and then group them together. To use this method, first define the IPSec transforms, then the IPSec proposal (which refers to the IPSec transforms). Then define the IPSec action (which refers to the IPSec proposals), and so forth until you completely define the policy.
The second approach, which you may find easier, is to define the high-level policy options first, and - as you are prompted - enter the definitions for the individual policy objects as you go along. If an object was defined previously that meets your needs, then you can reuse it instead of creating a new definition. For example, if a validity period for “allTheTime” was configured for a previous policy, then you may reuse it. The following procedure shows the entire process, but does not demonstrate the reuse of previously-defined policy information.

1. Add the policy.

```
Policy config> add policy
Enter a Name (1-29 characters) for this Policy []? examplePolicySecure11to12
Enter the priority of this policy (This number is used to determine the policy to enforce in the event of policy conflicts) [5]? 10
```

2. No profiles are configured so you must define a new one.

```
List of Profiles:
0: New Profile

Enter number of the profile for this policy [0]? 0
```

3. New profile definition; in this case the traffic we are interested in is from subnet 11 to subnet 12.

```
Enter a Name (1-29 characters) for this Profile []? trafficFrom11NetTo12Net
Source Address Format (1:NetMask, 2:Range, 3:Single Addr) [1]? Enter IPV4 Source Address [0.0.0.0]? 11.0.0.0
Enter IPV4 Source Mask [255.0.0.0]? Enter IPV4 Destination Address [0.0.0.0]? 12.0.0.0
Enter IPV4 Destination Mask [255.0.0.0]?

Protocol IDs:
1) TCP
2) UDP
3) All Protocols
4) Specify Range

Select the protocol to filter on (1-4) [3]? Enter the Starting value for the Source Port [0]? Enter the Ending value for the Source Port [65535]? Enter the Starting value for the Destination Port [0]? Enter the Ending value for the Destination Port [65535]? Enter the Mask to be applied to the Received DS-byte [0]? Enter the value to match against after the Mask has been applied to the Received DS-byte [0]? Configure local and remote ID©s for ISAKMP? [No]:
```

Note that the code release used for this example is V3R3. The latest level of code, V3R3.1, changes some of the detail although the configuration process is essentially the same.
Limit this profile to specific interface(s)? [No]:

Here is the Profile you specified...

Profile Name = trafficFrom11NetTo12Net
sAddr:Mask= 11.0.0.0 : 255.0.0.0 sPort= 0 : 65535
dAddr:Mask= 12.0.0.0 : 255.0.0.0 dPort= 0 : 65535
proto = 0 : 255
TOS = x00 : x00
Remote Grp=All Users
Is this correct? [Yes]:

4. We have finished with the profile definition and have returned to the policy configuration menu.

List of Profiles:
0: New Profile
1: trafficFrom11NetTo12Net

Enter number of the profile for this policy [1]? 1

5. No validity periods are configured so you must define a new one.

List of Validity Periods:
0: New Validity Period

Enter number of the validity period for this policy [0]? 0

6. Validity period configuration questions; in this example the validity period is from 9 AM to 5 PM, Monday through Friday, every month of 1999.

Enter a Name (1-29 characters) for this Policy Valid Profile []? MonToFri-9am:5pm-1999
Enter the lifetime of this policy. Please input the information in the following format: yyyyyymmddhhmmss:yyyyymmddhhmmss OR ©*© denotes forever. [*]? 19990101000000:19991231000000
During which months should policies containing this profile be valid. Please input any sequence of months by typing in the first three letters of each month with a space in between each entry, or type ALL to signify year round.
ALL?
During which days should policies containing this profile be valid. Please input any sequence of days by typing in the first three letters of each day with a space in between each entry, or type ALL to signify all week
ALL? mon tue wed thu fri
Enter the starting time (hh:mm:ss or * denotes all day) [*]? 00:00:00
Enter the ending time (hh:mm:ss) [00:00:00]? 17:00:00

Here is the Policy Validity Profile you specified...

Validity Name = MonToFri-9am:5pm-1999
Duration = 19990101000000 : 19991231000000
Months = ALL
Days = MON TUE WED THU FRI
Hours = 09:00:00 : 17:00:00
Is this correct? [Yes]:
7. We have finished with the validity period definition and have returned to the policy configuration menu.

List of Validity Periods:
0: New Validity Period
1: MonToFri-9am:5pm-1999

Enter number of the validity period for this policy [1]? 1
Should this policy enforce an IPSEC action? [No]: yes

8. You should always define a new IPsec action because the tunnel endpoint will always be different. The exceptions to this are if there are multiple tunnels between the same two gateways, and in the “wildcarded” remote access configurations where the tunnel endpoint is unknown.

IPSEC Actions:
0: New IPSEC Action
Enter the Number of the IPSEC Action [0]? 

9. IPsec action menu.

Enter a Name (1-29 characters) for this IPsec Action []? secure11NetTo12Net
List of IPsec Security Action types:
1) Block (block connection)
2) Permit

Select the Security Action type (1-2) [2]? 2
Should the traffic flow into a secure tunnel or in the clear:
1) Clear
2) Secure Tunnel
[2]? Enter Tunnel Start Point IPV4 Address
[11.0.0.5]? 1.1.1.1
Enter Tunnel End Point IPV4 Address (0.0.0.0 for Remote Access)
[0.0.0.0]? 1.1.1.2
Does this IPSEC tunnel flow within another IPSEC tunnel? [No]: Percentage of SA lifesize/lifetime to use as the acceptable minimum [75]?

Security Association Refresh Threshold, in percent (1-100) [85]? Options for DF Bit in outer header (tunnel mode):
1) Copy
2) Set
3) Clear

Enter choice (1-3) [1]? Enable Replay prevention (1=enable, 2=disable) [2]?
Do you want to negotiate the security association at system initialization (Y-N)? [No]:
You must choose the proposals to be sent/checked against during phase 2 negotiations. Proposals should be entered in order of priority.

10. No IPSec proposals have been defined so you must define a new one. Note that once the IPSec proposal has been defined it can be reused across multiple IPSec actions.

List of IPSEC Proposals:
0: New Proposal

Enter the Number of the IPSEC Proposal [0]? 

11. IPsec proposal configuration.

Enter a Name (1-29 characters) for this IPSec Proposal []? genP2Proposal
Does this proposal require Perfect Forward Secrecy? (Y-N)? [No]:
Do you wish to enter any AH transforms for this proposal? [No]:
Do you wish to enter any ESP transforms for this proposal? [No]: yes

12. No ESP transforms are configured so you must define a new one. Once the ESP transform has been defined it may be reused by any IPSec proposal.

List of ESP Transforms:
0: New Transform

Enter the Number of the ESP transform [0]? 0

13. IPSec transform configuration.

Enter a Name (1-29 characters) for this IPSec Transform []? esp3DESwSHA
List of Protocol IDs:
1) IPSEC AH
2) IPSEC ESP

Select the Protocol ID (1-2) [1]? 2

List of Encapsulation Modes:
1) Tunnel
2) Transport

Select the Encapsulation Mode (1-2) [1]? 1

List of IPSec Authentication Algorithms:
0) None
1) HMAC-MD5
2) HMAC_SHA

Select the ESP Authentication Algorithm (0-2) [2]? 2

List of ESP Cipher Algorithms:
1) ESP DES
2) ESP 3DES
3) ESP CDMF
4) ESP NULL

Select the ESP Cipher Algorithm (1-4) [1]? 2

Security Association Lifesize, in kilobytes (1024-65535) [50000]? 50000
Security Association Lifetime, in seconds (120-65535) [3600]? 3600

Here is the IPSec transform you specified...

Transform Name = esp3DESwSHA
Type =ESP Mode =Tunnel LifeSize= 50000 LifeTime= 3600
Auth =SHA Encr =3DES
Is this correct? [Yes]:

14. Return to the IPSec proposal menu.

List of ESP Transforms:
0: New Transform
1: esp3DESwSHA

Enter the Number of the ESP transform [1]?
Do you wish to add another ESP transform to this proposal? [Yes]: no

Here is the IPSec proposal you specified...

Name = genP2Proposal
Pfs = N
ESP Transforms:
esp3DESwSHA
Is this correct? [Yes]:

15. Return to the IPSec action menu.

List of IPSEC Proposals:
0: New Proposal
1: genP2Proposal

Enter the Number of the IPSEC Proposal [1]?
Are there any more Proposal definitions for this IPSEC Action? [No]:

Here is the IPSec Action you specified...

IPSECAction Name = secure11NetTo12Net
Tunnel Start:End = 1.1.1.1 : 1.1.1.2
Tunnel In Tunnel = No
Min Percent of SA Life = 75
Refresh Threshold = 85 %
Autostart = No
DF Bit = COPY
Replay Prevention = Disabled
IPSEC Proposals:  
   genP2Proposal
Is this correct? [Yes]:

16. Return to the policy menu.

IPSEC Actions:
0: New IPSEC Action
1: secure11NetTo12Net

Enter the Number of the IPSEC Action [1]?

17. You have specified a secure IPSec action type, so you must identify an ISAKMP action for the Phase 1 negotiations. None are defined, so you must enter a new one. In most cases, one ISAKMP action and proposal are sufficient for all of the security policies.

ISAKMP Actions:
0: New ISAKMP Action

Enter the Number of the ISAKMP Action [0]?

18. ISAKMP action configuration.

Enter a Name (1-29 characters) for this ISAKMP Action []? genPhase1Action
List of ISAKMP Exchange Modes:
1) Main
2) Aggressive
Enter Exchange Mode (1-2) [1]?
Percentage of SA lifesize/lifetime to use as the acceptable minimum [75]?

ISAKMP Connection Lifesize, in kilobytes (100-65535) [5000]?
ISAKMP Connection Lifetime, in seconds (120-65535) [30000]?
Do you want to negotiate the security association at system initialization(Y-N)? [Yes]: no
You must choose the proposals to be sent/checked against during phase 1 negotiations. Proposals should be entered in order of priority.

19. No ISAKMP proposals are configured, so you must create a new one.

List of ISAKMP Proposals:
0: New Proposal

20. ISAKMP proposal configuration.

Enter the number of the ISAKMP Proposal [0]?
Enter a Name (1-29 characters) for this ISAKMP Proposal []? genP1Proposal

List of Authentication Methods:
1) Pre-Shared Key
2) RSA SIG

Select the authentication method (1-2) [1]? 2

List of Hashing Algorithms:
1) MD5
2) SHA

Select the hashing algorithm (1-2) [1]? 2

List of Cipher Algorithms:
1) DES
2) 3DES

Select the Cipher Algorithm (1-2) [1]? 2

Security Association Lifesize, in kilobytes (100-65535) [1000]?
Security Association Lifetime, in seconds (120-65535) [15000]?

List of Diffie Hellman Groups:
1) Diffie Hellman Group 1
2) Diffie Hellman Group 2

Select the Diffie Hellman Group ID from this proposal (1-2) [1]?

Here is the ISAKMP Proposal you specified...

Name = genP1Proposal
AuthMethod = Pre-Shared Key
LifeSize = 1000
LifeTime = 15000
DHGroupID = 1
Hash Algo = SHA
Encr Algo = 3DES CB
Is this correct? [Yes]:

21. Return to the ISAKMP action configuration.

List of ISAKMP Proposals:
0: New Proposal
1: genP1Proposal

Enter the number of the ISAKMP Proposal [1]?
Are there any more Proposal definitions for this ISAKMP Action? [No]:

Here is the ISAKMP Action you specified...

ISAKMP Name = genPhase1Action
Mode = Main
Min Percent of SA Life = 75
Conn LifeSize/LifeTime = 5000 : 30000
Autostart = No
ISAKMP Proposals:
genP1Proposal
Is this correct? [Yes]:

22. Return to the policy configuration.

ISAKMP Actions:
0: New ISAKMP Action
1: genPhase1Action

Enter the Number of the ISAKMP Action [1]?
Do you wish to Map a DiffServ Action to this Policy? [No]: yes

23. Define the DiffServ GoldService action.

DiffServ Actions:
0: New DiffServ Action

Enter the Number of the DiffServ Action [0]?

24. DiffServ action configuration.

Enter a Name (1-29 characters) for this DiffServ Action []? GoldService
Enter the permission level for packets matching this DiffServ Action (1. Permit, 2. Deny) [2]? 1
List of DiffServ Queues:
1) Premium
2) Assured/BE
Enter the Queue Number(1-2) for outgoing packets matching this DiffServ Action [2]? 2
How do you want to specify the bandwidth allocated to this service?
Enter absolute kbps(1) or percentage of output bandwidth(2) [2]? 2
Enter the percentage of output bandwidth allocated to this service [10]? 40

Transmitted DS-byte mask [0]?
Transmitted DS-byte modify value [0]?

Here is the DiffServ Action you specified...

DiffServ Name = GoldService Type = Permit
TOS mask:modify=x00:x00
Queue:BwShare = Assured : 40 %
Is this correct? [Yes]:

25. Return to the policy configuration.

DiffServ Actions:
0: New DiffServ Action
1: GoldService

Enter the Number of the DiffServ Action [1]? 1
Policy Enabled/Disabled (1. Enabled, 2. Disabled) [1]?

Here is the Policy you specified...

Policy Name = examplePolicySecure11to12
State:Priority = Enabled : 10
Profile = trafficFrom10NetTo12Net
Valid Period = MonToFri-9am:5pm-1999
IPSEC Action = secure11NetTo12Net
ISAKMP Action = genPhase1Action
DiffServ Action = GoldService
26. If DiffServ or IPSec are not enabled, then you are alerted that before the policy can be enforced, you must enable DiffServ, IPSec, or both (DiffServ feature or IPSec feature).

You must enable and configure DiffServ in feature DS before QoS can be ensured for this policy.

27. The final step in this process is to add a USER profile definition for the remote ISAKMP peer. This step is not needed if the ISAKMP negotiations are to authenticate the peer with public certificates. However, in the preceding example we chose pre-shared key as the authentication method, so we must identify the user and enter the pre-shared key that we expect the peer to use.

Policy config> add user
Choose from the following ways to identify a user:
1: IP Address
2: Fully Qualified Domain Name
3: User Fully Qualified Domain Name
4: Key ID (Any string)
Enter your choice (1-4) [1]?
Enter the IP Address that distinguishes this user [0.0.0.0]? 1.1.1.2
Group to include this user in []? peers
Authenticate user with 1: pre-shared key or 2: Public Certificate [1]?
Mode to enter key (1=ASCII, 2=HEX) [1]?
Enter the Pre-Shared Key (an even number of 2-128 ascii chars):
Enter the Pre-Shared Key again (10 characters) in ascii:

Here is the User Information you specified...

Name = 1.1.1.2
Type = IPV4 Addr
Group = peers
Auth Mode = Pre-Shared Key
Key(Ascii)=exampleKey
Is this correct? [Yes]:

28. The policy configuration steps are now complete. If you want to configure DiffServ, IPSec, or any network or IP configuration, then you must do that before the IPSec tunnel will be functional. The following list command example shows the configuration that was just completed. To activate these changes, either reload the device or enter the policy feature’s Talk 5 reset database command.

Policy config> list all

Configured Policies....

Policy Name = examplePolicySecure11to12
State:Priority = Enabled : 10
Profile = trafficFrom11NetTo12Net
Valid Period = MonToFri-9am:5pm-1999
IPSEC Action = secure11NetTo12Net
ISAKMP Action = genPhase1Action
DiffServ Action = GoldService

--More--
Configured Profiles....

Profile Name = trafficFrom11NetTo12Net
sAddr:Mask= 11.0.0.0 / 255.0.0.0 sPort= 0 : 65535
dAddr:Mask= 12.0.0.0 / 255.0.0.0 dPort= 0 : 65535
proto = 0 : 255
TOS = x00 : x00
Remote Grp=All Users

--More--

Configured Validity Periods
Validity Name = MonToFri-9am:5pm-1999
Duration = 19990101000000 : 19991231000000
Months = ALL
Days = MON TUE WED THU FRI
Hours = 09:00:00 : 17:00:00

--More--

Configured DiffServ Actions....
DiffServ Name = GoldService Type =Permit
TOS mask:modify=x00:x00
Queue:BwShare =Assured : 40 %

--More--

Configured IPSec Actions....
IPSecAction Name = secure11NetTo12Net
Tunnel Start:End = 1.1.1.1 : 1.1.1.2
Tunnel In Tunnel = No
Min Percent of SA Life = 75
Refresh Threshold = 85 %
Autostart = No
DF Bit = COPY
Replay Prevention = Disabled
IPSec Proposals:
genP2Proposal

--More--

Configured IPSec Proposals....
Name = genP2Proposal
Pfs = N
ESP Transforms:
esp3DESwSHA

--More--

Configured IPSec Transforms....
Transform Name = esp3DESwSHA
Type =ESP Mode =Tunnel LifeSize= 50000 LifeTime= 3600
Auth =SHA Encr =3DES

--More--

Configured ISAKMP Actions....
ISAKMP Name = genPhase1Action
Mode = Main
Min Percent of SA Life = 75
Conn LifeSize:LifeTime = 5000 : 30000
Autostart = No
ISAKMP Proposals:
genP1Proposal

--More--

Configured ISAKMP Proposals....

Name = genP1Proposal
AuthMethod = Pre-Shared Key
LifeSize = 1000
LifeTime = 15000
DHGroupID = 1
Hash Algo = SHA
Encr Algo = 3DES CB

--More--

Configured Policy Users....

Name = 1.1.1.2
Type = IPV4 Addr
Group =peers
Auth Mode =Pre-Shared Key
Key(Ascii)=exampleKey

--More--

Configured Manual IPSEC Tunnels....

IPv4 Tunnels

For more information about configuring policy features manually see “Chapter 20. Configuring and Monitoring the Policy Feature” in Nways Multiprotocol Access Services, Using and Configuring Features Version 3.3, SC30-3993-01.

6.4.2 Configuring the LDAP Client

This example shows how to configure and enable the LDAP policy search engine.

In this example there are two LDAP directories (a primary and a secondary) with IP addresses of 11.0.0.2 and 13.0.0.1 respectively. They are both listening on TCP port 389 and the device must bind to the LDAP server as cn=router, password myPassWord.

The base entry in the directory tree for the router's policies is cn=RouterDeviceProfile, o=ibm,c=us.

This manual is applicable to the IBM 2216; similar manuals are available for the software running on the 2210 and 2212 routers.
This example also shows how to set the default policy so that LDAP communications are secured through IPSec. This example uses pre-shared key for the ISAKMP authentication, and SHA and 3DES for the authentication and encryption parameters for Phase 1 and Phase 2. The tunnel start point is 1.1.1.4 for the device performing the LDAP policy search, and the tunnel endpoints are 1.1.1.1 for the 11.0.0.1 LDAP server, and 1.1.1.3 for the 13.0.0.1 LDAP server.

1. Configure and enable the LDAP policy search engine, and list the results.

   Policy config> set ldap primary-server 11.0.0.1
   Policy config> set ldap secondary-server 13.0.0.1
   Policy config> set ldap port 389
   Policy config> set ldap bind-name cn=router
   Policy config> set ldap bind-pw myPassWord
   Policy config> set ldap anonymous-bind no
   Policy config> set ldap policy-base cn=RouterDeviceProfile,o=ibm,c=us
   Policy config> enable ldap policy-search
   Policy config> list ldap

   LDAP CONFIGURATION information:
   
   Primary Server Address: 11.0.0.1
   Secondary Server Address: 13.0.0.1
   
   Search timeout value: 3 sec(s)
   Retry interval on search failures: 1 min(s)
   Server TCP port number: 389
   Server Version number: 2
   
   Bind Information:
   Bind Anonymously: No
   Device Distinguished Name: cn=router
   Device Password: myPassWord
   
   Base DN for this device’s policies: cn=RouterDeviceProfile,o=ibm,c=us
   
   Search policies from LDAP Directory: Enabled

2. Set the default policy.

   Policy config> set default-policy

   List of default policy rules:
   1) Accept and Forward all IP Traffic
   2) Permit LDAP traffic, drop all other IP Traffic
   3) Permit and Secure LDAP traffic, drop all other IP Traffic

   Select the default policy rule to use during policy refresh periods [1]? 3

   List of default error handling procedures:
   1) Reset Policy Database to Default Rule
   2) Flush any rules read from LDAP, load local rules

   Select the error handling behavior for when loading Policy Database [1]?
Please enter the set of Security Information for encrypting and authenticating the LDAP traffic generated by the device when retrieving policy information from the LDAP Server.

Enter phase 1 ISAKMP negotiation parameters:

List of Diffie Hellman Groups:
1) Diffie Hellman Group 1
2) Diffie Hellman Group 2

Select the Diffie Hellman Group ID from this proposal (1-2) [1]?

Select the hashing algorithm (1-2) [1]? 2

List of Cipher Algorithms:
1) DES
2) 3DES

Select the Cipher Algorithm (1-2) [1]? 2

Authentication: (1) Pre-shared Key or (2) Certificate (RSA Sig) [2]? 1
Enter the Pre-Shared Key []? test

Enter phase 2 IPSec negotiation parameters:
List of IPSec Authentication Algorithms:
0) None
1) HMAC-MD5
2) HMAC_SHA

Select the ESP Authentication Algorithm (0-2) [1]? 2
List of ESP Cipher Algorithms:
1) ESP DES
2) ESP 3DES
3) ESP CCMF
4) ESP NULL

Select the ESP Cipher Algorithm (1-4) [1]? 2
Tunnel Start IPV4 Address (Primary LDAP Server) [0.0.0.0]? 1.1.1.4
Tunnel End Point IPV4 Address (Primary LDAP Server) [0.0.0.0]? 1.1.1.1
Tunnel Start IPV4 Address (Secondary LDAP Server) [1.1.1.4]? Tunnel End Point IPV4 Address (Secondary LDAP Server) [1.1.1.1]? 1.1.1.3
Policy config> list default-policy

Default Policy Rule: Drop All IP Traffic except secure LDAP
Default error handling procedure: Reset Policy Database to Default Rule

Phase 1 ISAKMP negotiation parameters:
Diffie Hellman Group ID: 1
Hashing Algorithm: SHA
ISAKMP Cipher Algorithm: ESP 3DES CBC

Per-shared key value: test

Phase 2 IPSEC negotiation parameters:
IPSec ESP Authentication Algorithm: HMAC SHA
ESP Cipher Algorithm: 3DES
Local Tunnel Addr (Primary Server): 1.1.1.4
Remote Tunnel Addr (Primary Server): 1.1.1.1
Local Tunnel Addr (Secondary Server): 1.1.1.4
Remote Tunnel Addr (Secondary Server): 1.1.1.3

At this point you are ready to manage the routers in your network with the policy feature.
Chapter 7. Putting It All Together

We have discussed the concepts, architectures, information models, business needs and initial implementation scenarios needed to make application-driven networking happen. We have done this by having different initial applications such as VPN or QoS management in our minds. Figure 43 shows the overall picture, which was our starting point, showing the product-level elements marked with a star.

Figure 43. Overall System Approach

The following paragraphs serve to re-state the main ideas and motivations by giving some thought to today’s customer requirements; we will discuss the relevance of certain marketing messages about policy-based networking and system design in general.
Confucius said:

Tell me - And I will forget
Show me - And I will remember
Involve me - And I will understand

Figure 44. Citation

So, let us try to discuss the question:

The question

**When would we use and how would we design a policy-based system today?**

### 7.1 Policy-Controlled QoS and VPN Scenarios

Those of us reading magazine articles and having discussions with I/T staff about QoS issues often come to believe that bandwidth increase outperforms sophistication of bandwidth management. Each time a very sophisticated bandwidth management technology was introduced into the market it was challenged by potentially cheaper, much easier to handle, and nearly unchanged evolutions of traditional technology.

This was especially true for the campus environment where no monthly fees for bandwidth utilization enabled an increased tolerance of inefficient bandwidth usage.

Unfortunately this approach is only reasonable as long as the communication system is always over-designed. Big numbers came to the rescue here. In traditional systems with hundreds of users and applications, the overall picture is one of a rather smooth statistical behavior in which bandwidth utilization problems occur very rarely and - if so - only for a few seconds. So a network which is moderately over-designed was often appropriate. The relatively simple buffering and scheduling requirements at each end system found this sort of network quite acceptable.

Today's applications and user behaviors no longer have the nice capability of smoothing out inadequacies in the network. The “sawtooth” nature of today's communication protocols' bandwidth requirements combined with the completely network-unaware distributed application design and development lead to problems.
Today’s “server farm” solutions for mail, ERP or business intelligence applications are demanding a significant reduction in reaction time when adapting the infrastructure to the needs of the user.

Failures to react in time map directly to a reduction in profitability for all users/applications, and not only for those who produced the additional demand either. Today we are looking for a graceful degradation behaviors within complex and heterogeneous communication systems. “Graceful” in this case means intentionally down-grading some users or applications whilst at the same time keeping others up to speed.

What we are looking for is an intelligent infrastructure which enables us (I/T architects), the company and the user to survive over a period of time giving us the time to enhance the infrastructure in a cost-effective manner. We are actually looking for the implementation of different service classes with a high degree of dynamics in the mapping of users and applications to those classes.

In other words we need policy-based (which means “intelligent”), fast-adapting service differentiation end-to-end from one network user to another.

Having written that:

**Conclusion 1**

Quality-of-Service-related policy-based system control will likely be first implemented in I/T environments where bandwidth is not considered an indefinite resource, such as:

1. WAN environments in large geographically-dispersed corporations
2. Internet service providers.

Having said that, we need also take into account that new concepts can only survive if the investment in introduction and handling for the customer is low enough to be termed “acceptable”.

**Conclusion 2**

It is nice to have more “intelligence” in the communication system, but the same people operating today’s environments need to continue to do this in the future.

Keep it simple, resist the urge to be too creative in complex policy generation.

It is obvious that ISPs (Internet Service Providers) and large corporate IP networks have the biggest need for such self-controlled, policy-guided multi-service systems. Over-designing the network means immediate lost profit. Underestimating the demand means loss in profitability or “unhappy” users, which is actually another side of the same coin.

By using a QoS mechanism to control bandwidth it becomes essential for the scalable design of policy systems to understand the granularity and dynamics of service classes to implement.
Internet Service Providers tend to implement today the “Olympic” model, which means that very few service classes are mapped to users (into “Gold”, “Silver” and “Bronze” types of service). DiffServ is currently the preferred mechanism for ISPs to implement Class-of-Service.

The service classes themselves and the user associations with these classes are rather stable; they do not tend to change during a given day, for example. This means we can expect that the number of policy updates and policy lookups will be rather low. This lookup frequency pattern matches the existing usage behavior of AAA servers in operation in ISPs today.

If an ISP is looking to support RSVP-based bandwidth management, the lookup frequency pattern tends to be much higher.

In both cases, DiffServ and RSVP, it is likely that the ISP will want to include in its policy decision certain types of “external” information. PDP external information can be provided by an accounting system or application service provisioning system (for example, a video conference system).

This discussion leads to two questions:

1. Should I use co-located or separated PDP/PEP systems?
2. If a separated PDP/PEP system is used, how many PDPs do I need?

![Figure 45. Policy Decision Taking Process](image)

Figure 45 tries to gives an overview of the dependencies between policy complexity, change/access frequency of policies and policy scope (local decisions versus global corporate decisions). The two curves indicate the distribution of policy complexity (on the right) and policy access frequency (on the left) in a specific policy system. The resulting distribution of quadrants (each size) shows the distribution of co-located versus separated PDP/PEPs in the policy system.

Let us give some examples:
1. In a policy system where the majority of policies of high access frequency can be addressed locally and only a few of the more complex rules are local, most of the policy-aware devices should have a co-located PDP/PEP approach. The access frequency demands fast decisions and the rules have local scope which can be decided in the device itself.

A typical scenario for this is in edge devices for simple DiffServ applications.

2. In a policy system where, most of the time, global dependencies are checked in conjunction with rather medium frequency a separated PDP-PEP approach is more feasible.

A typical scenario for this is in VPN solutions for corporations provided by an ISP.

3. In a policy system where there is a low frequency of policy decisions with local relevance, a co-located PDP/PEP approach should be used.

These are typically LDAP-based configuration/provisioning solutions.

Our current assumption is that the costs of a co-located versus separated approach for policy-aware devices is not very high. In real life, this assumption holds only for rather complex devices. In cases where price-per-box is an issue, especially in networks with a large number of devices (an ISP’s POP boxes, for example) or the individual price point per box is very low (Campus), a separated PDP-PEP approach is much more suitable.

---

**Conclusion 3**

The type of policy-aware device depends heavily on the policy scenario we would like to implement. We will initially see a lot of rather simple (provisioning-oriented or simple service class-oriented) applications. This suggests co-located PDP/PEP implementations, such as IBM has implemented today, especially in corporate, non-carrier environments. See **Conclusion 2**!

Over time, application-driven networking will evolve into much more complex environments utilizing both models at the same time.

Let us finally add some thoughts on Virtual Private Network scenarios.

The configuration process for VPNs tends to be complex and error-prone. The number of tunnels and the associated security associations is high, creating the challenge of setting up consistent IP-VPN networks. At the same time, a VPN configuration changes more often than other configuration elements on a router or firewall platform. This situation makes LDAP-based VPN configuration approaches very promising compared to SNMP-based or tool-based configuration tasks.

This is in parallel to an increasing interest in the use of LDAP-based repositories to store semi-stable system information (user info, security info, etc.), so there is often no extensive additional effort necessary to build the LDAP repository.
Finally some comments on the number of PDPs and LDAP servers is needed in a reasonable policy-controlled environment.

Figure 45 on page 112 tried to visualize the different scalability issues in a generic policy environment. As you can imagine, the more lookups you have to do between any server and the associated clients under specific minimal response time requirements, the closer the server has to be placed to the clients. In the PDP case the closest solution is to integrate client (PEP) and server (PDP). This will add some cost but relieves the burden of having a policy server co-located.

Let us assume we have a big star-shaped corporate network (such as for an insurance company or for most banks). In this case, a separated PDP would mean - more or less - one PDP per branch, which is ridiculous. An LDAP approach with co-located PDP/PEPs is much more suitable.

Additionally you have to be careful what kind of operational aspects you may migrate to an LDAP-based solution, since today there are only limited redundancy concepts for LDAP repositories. The IETF and others are working on LDAP extensions to handle this kind of challenge.

All of this discussion comes down to the number of geographically dispersed hierarchies in the basic network design. ISPs may decide to go for a number of separated policy servers, corporations may not.
Final Conclusion

Keep the number of policy servers down. There are still other SNMP or JAVA/HTML system- and network-management platforms available, which can today and will in the future perform specific tasks very well. Look at application-driven networking as an additional mechanism to control communication system elements and as the base for much more sophisticated approaches in the future.

The more integrated your e-environments get, the more distributed the intelligence of the system should be, keeping a centralized overall control.

Like in real life: A company hires intelligent people (“distributed intelligence”) but reduces the number of control instances (“centralized and lean control”).
Appendix A. Standards Documents and Sample Files

A.1 IETF Documents

The IETF (Internet Engineering Task force) has a very good Web presence. You can find all the latest documents at:

http://www.ietf.org

In particular, for details on the Policy Framework and related issues please see the following (especially those documents highlighted in **bold** text):

**IETF Working Groups Drafts:**
- Differentiated Services (diffserv) -- 4 Internet-Drafts
- Integrated Services (intserv) -- 2 Internet-Drafts
- IP Security Protocol (ipsec) -- 36 Internet-Drafts
- Integrated Services over Specific Link Layers (issll) -- 10 Internet-Drafts
- LDAP Extension (ldapext) -- 22 Internet-Drafts
- LDAP Duplication/Replication/Update Protocols (ldup) -- 5 Internet-Drafts
- **Policy Framework (policy)** -- 6 Internet-Drafts
- Resource Allocation Protocol (rap, **cops**) -- 8 Internet-Drafts
- Resource Reservation Setup Protocol (rsvp) -- 6 Internet-Drafts
- Schema Registration (schema) -- 8 Internet-Drafts

**Drafts used in this book:**
- draft-ietf-diffserv-framework-02.txt
- draft-ietf-diffserv-mib-00.txt
- draft-ietf-diffserv-model-00.txt
- draft-ietf-diffserv-trafcon-format-00.txt
- draft-ietf-ipsec-policy-model-00.txt
- draft-ietf-ipsec-policy-schema-00.txt
- draft-ietf-ipsec-vpn-policy-schema-00.txt
- draft-ietf-issll-is802-sbm-04.txt
- draft-ietf-lldap-model-01.txt
- draft-ietf-policy-core-info-model-00.txt
- draft-ietf-policy-core-schema-03.txt
- draft-ietf-policy-core-schema-04.txt
- draft-ietf-policy-framework-pfdl-00.txt
- draft-ietf-policy-terms-00.txt
- draft-ietf-policy-terms-02.txt
- draft-ietf-rap-cops-06.txt
- draft-ietf-rap-cops-rsvp-05.txt
A.2 DEN Documents

Since the DEN (Directory Enabled Networks) initiative was integrated into the DMTF (Distributed Management Task Force) efforts, there is only one URL available specifically for DEN:

http://www.murchiso.com/den/

A.3 DMTF Documents

The DMTF (Distributed Management Task Force) has a very good Web presence. You can find the latest documents at:

http://www.dmtf.org

For details on the Network Policy Framework and related issues please see:

http://www.dmtf.org/educ/conf1999/pres.html

A.4 LDAP Directory Initialization

The following discussion uses material available for download at the following URLs:


The router-specific examples were taken from the product manual *Access Integration Services, Using and Configuring Features Version 3.3*, SC30-3989-01, which applies explicitly to the IBM 2212 router. Similar manuals exist for the software running on the 2210 and 2216 routers.
A Version 2 or a Version 3 LDAP server is required. IBM’s SecureWay LDAP server is the recommended implementation, however any LDAP server should work. There are a couple of LDAP server implementations freely available on the Internet (OpenLDAP and the University of Michigan implementation).

The LDAP schema is the set of rules and information that comprise the class and attribute definitions that define the entries that ultimately exist in the directory. The LDAP schema is typically written in ASN.1 syntax similar to SNMP MIBs. The policy schema developed for the IBM routers is based on early efforts being worked on in the IETF by the IPSec and Policy working groups.

Every class definition consists of a set of attributes that define the class. The attributes are either required attributes or allowed attributes. Required attributes are ones which must be present when the object is added or modified in the LDAP server. Allowed attributes are ones which may or may not be included in the object definition. If they are not included then the default value will be used by the policy search agent in the router when parsing the object from the LDAP server. The default values for each attribute can be found in the ibmPolicySchema.txt file.

Two key objects in the policy schema that allow the policy search agent to search for and to find the necessary policies for the device are the DeviceProfile and the DevicePolicyRules. The DeviceProfile has information about the device’s mandatory DevicePolicyRules reference. Devices can be grouped together into one DeviceProfile or each device in the network can have its own DeviceProfile. This really will depend on whether more than one box in the network needs to fetch the same set of rules. Typically for security gateways this will not be true since every gateway will have a different tunnel endpoint. For QoS-only boxes, it would be conceivable that a group of devices would all read the same set of policies.

The DevicePolicyRules object will be retrieved based on the value in the DeviceProfile that is fetched for the device. Once the DevicePolicyRules object has been retrieved, then the list of policy rules for that device can also be retrieved. If any of the objects are not found or if an error is detected during a consistency check on an object then the search is ended and messages will be displayed for the PLCY event logging subsystem (ELS) denoting the error detected.

The policy schema closely matches the configuration objects in the policy feature on the router. Perform the configuration steps to add a policy using "talk 6", “feature policy” to acquaint yourself with the policy objects and their relationships.

A.4.1 Config Files

The following LDAP configuration files are very useful and can be downloaded from the Web via the following URL:


Files:
1. policyTemplates.ldif with predefined policy objects
2. policyExamples.ldif with some examples of security policies
A.4.2 Including the Objectclass and Attribute Files in the LDAP Server

LDAP servers typically have the ability to enforce “schema checking”. Schema checking is the ability of the LDAP server to understand the attributes of a class, which attributes are required for the object to be added to the directory and the attributes that are optionally allowed with the object.

The LDAP server will not allow the object to be added if a required attribute is not included or if an unknown attribute is included within the object. Since IBM’s policy schema is not standardized yet, the schema will not come pre-installed with the LDAP server.

The instructions for adding the objectclass file (policySchema_oc.conf) and the attribute file (policySchema_<platform>_at.conf) to the LDAP server configuration file are detailed below:

1. Locate the LDAP server configuration file. This typically is the slapd.conf file. On UNIX machines this may be in the /etc directory.
2. Edit the slapd.conf file and add the following lines (these lines should be added after any other include statements in the config file)

   include <path>/policySchema_<platform>_at.conf
   include <path>/policySchema_oc.conf

The oc and at files were tested and verified to work with the versions of the respective LDAP servers at the time of this writing. Conceivably there may be changes by these vendors that will cause the files to be modified. In the event that installing these files does not work and you cannot determine the problem, please contact someone in router support and every attempt will be made to resolve the issue.

A.4.3 Adding and Modifying Policy Entries to LDAP Server via LDIF

Depending on your platform, LDAP server choice and growth of products which interact with the LDAP server there may be numerous ways to modify the information stored in your LDAP server. Currently the ubiquitous method of editing the content of the directory server is via the command line utilities which parse LDIF (LDAP Data Interchange Format).

These command line utilities are ldapsearch, ldapmodify, ldapdelete and ldapadd. Every LDAP server we have tested included versions of these client utilities.

Two files are included in this package to help the administrator get started deploying policies in the LDAP server:

1. policyTemplates.ldif

   See http://www.openldap.org/openldap
This file contains predefined objects for various combinations of IPSec transforms and proposals and ISAKMP actions and proposals. It also contains definitions for common validity periods and defines the DiffServ code points AF1-AF4 and EF. This suite of definitions should be adequate for most configurations to use as a starting point for their security and QoS policies.

2. policyExamples.ldif

This file contains several example policies which use the objects defined in the policyTemplates.ldif file. This file is intended to give the user a sample which may be modified to match their own requirements.

The instructions for how to add the objects defined in these files to the LDAP server are described below:

1. Add policy templates first:
   a. Remove comments from file using the following command
      
      ```
      grep -v '^#' policyTemplates.ldif > out.ldif
      ```
   b. Add the entries in the new out.ldif file using the `ldapmodify` command
      
      ```
      ldapmodify -h <hostname> -D <user dn> -w <password> -rac -f out.ldif
      ```

2. Make any modifications to the example policies supplied and then perform the following steps:
   a. Remove comments from the file using the following command
      
      ```
      grep -v '^#' policyObjects.ldif > out.ldif
      ```
   b. Add the entries in the new out.ldif file using the `ldapmodify` command
      
      ```
      ldapmodify -h <hostname> -D <user dn> -w <password> -rac -f out.ldif
      ```

These same instructions can be used to perform modifications to the objects in the directory. Note, if you do not care to put comments in the “.ldif” files you can just make modifications directly in the file produced by the `grep` command.

An easy way to manage the information residing in the LDAP server is to keep a master LDIF file which represents all the definitions residing in the LDAP server. Any modifications or additions are made directly to this file and the `ldapmodify` command may be used to apply these changes to the LDAP server. This also allows the administrator to have a text file backup in the event that the LDAP server needs to be restored.

To make modifications use the following syntax of the `ldapmodify` command:

```
ldapmodify -h <hostname> -D <user dn> -w <password> -rc -f out.ldif
```

A.4.4 Configuring the Router to Read Policy Information from LDAP Server

Now that policies actually exist in the LDAP server, you must enable the routers to retrieve their policy information from the LDAP server. To do this perform the following steps:

1. Attach to the router using an ASCII terminal or a telnet session.
   
   At the “*” prompt, type `talk 6` and press Enter.
2. Proceed to the policy feature by typing `feature policy` and pressing Enter.
3. Enter the IP address of the LDAP server which contains the policy information needed by this device.

   Policy config> set ldap primary-server 11.0.0.1

4. If the LDAP server requires authentication then set the bind parameters. First enable sending of authentication parameters by issuing the following command:

   Policy config> set ldap anonymous-bind no
   Policy config> set ldap bind-name cn=root
   Policy config> set ldap bind-pw password

5. Configure the distinguished name of the device profile object in the LDAP server for this device.

   Policy config> set ldap policy-base cn=deviceProfileForSG1, o=ibm, c=us

6. Enable retrieving policies from the LDAP server.

   Policy config> enable ldap policy-search

7. List the LDAP parameters and verify they look similar to the following:

   Policy config> list ldap
   LDAP CONFIGURATION information:
   
   Primary Server Address: 11.0.0.1
   Secondary Server Address: 0.0.0.0
   Search timeout value: 3 sec(s)
   Retry interval on search failures: 1 min(s)
   Server TCP port number: 389
   Server Version number: 2
   Bind Information:
   Bind Anonymously: No
   Bind Name: cn=root
   Base DN for this device's policies: cn=deviceProfileForSG1, o=ibm, c=us

   Retrieve policies from LDAP Server Only

A.4.5 Resetting the LDAP Configuration and Policy Database

For the changes made above to take affect, the user must either restart/reload the router or go into talk 5 and use the dynamic reconfiguration feature of the router to activate the changes. This procedure is shown in the following steps:

1. Turn on ELS messages for PLCY subsystem in talk 5.

2. In feature policy (talk 5), reset the ldap configuration.

3. In feature policy (talk 5), reset the policy database.

A screen capture showing these steps is shown below:

* 
*talk 5
+event
Event Logging System user console
ELS> nodisplay subsystem all all
Complete
ELS> display subsystem plcy error
ELS> display event plcy.22
ELS> display event plcy.26
An alternative way to cause the policy database in the router to reload the policies from the LDAP server is to perform an SNMP set.

Please refer to the `vpnpolicy.mib` file, which can be found under the MIB download page:

```
http://www.networking.ibm.com/support/code.nsf/mibscode
```

The name of the object in the `vpnpolicy.mib` to set is the `vpSysRefreshConfig`. It may be useful to build a script which sets this object for each policy-enabled router in the network.

---

**A.5 Example LDIF and Schema Files**

**A.5.1 IBM Schema File**

```
IBM 221x Policy Schema Description
Date: Tue May 25 15:32:41 EDT 1999
Draft: 1.0

Policy Class Structure
```

```
+---------------+
| DeviceProfile |
+---------------+
```

```
+-------------------+
| DevicePolicyRules |
+-------------------+
```

```
+------------+
| PolicyRule |
+------------+
```

```
+----------------+
| TrafficProfile |
+----------------+
```

```
+-----------------------+
| PolicyValidityPeriod |
+-----------------------+
```

```
+-------------------+
| DiffServAction* |
+-------------------+
```

```
+-------------+
| RSVPAction* |
+-------------+
```

```
+-------------------+
| DiffServAction* |
+-------------------+
```

---

Standards Documents and Sample Files 123
Notes:
1: > = single reference, >> = multiple reference
2: * indicates optional reference
3: When defining Security Policies for IPSEC/ISAKMP, the traffic profile should define the traffic flowing into the secure tunnel.

Class Name: DeviceProfile
Requires
objectClass
cn
devicerulesreference
Allows

Attribute Definitions:

NAME: objectClass
REQUIRED
MULTI-VALUE
DESC:
The Class type for this object
EQUALITY: objectIdentifierMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES: DeviceProfile, top
DEFAULT VALUE: DeviceProfile, top

NAME: cn
REQUIRED
MULTI-VALUE
DESC:
The Common Name for this object
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES: Common name for object
DEFAULT VALUE: None, must be specified

NAME: devicerulesreference
REQUIRED
SINGLE-VALUE
DESC:
The DN of the DevicePolicyRules object in the directory that contains the policy rules for this device
EQUALITY: distinguishedNameMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.12
VALID VALUES: DN of DevicePolicyRules Object
DEFAULT VALUE: None, must be specified

Class Name: DevicePolicyRules
Requires
objectClass
cn
policyrulereference
Allows

Attribute Definitions:

NAME: objectClass
REQUIRED
MULTI-VALUED
DESC:
The Class type for this object
EQUALITY: objectIdentifierMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES: DevicePolicyRules, top
DEFAULT VALUE: DevicePolicyRules, top

NAME:
REQUIRED
MULTI-VALUED
DESC:
The Common Name for this object
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES: Common name for object
DEFAULT VALUE: None, must be specified

NAME: policyrulereference
REQUIRED
MULTI-VALUED
DESC:
The DN of a PolicyRule object in the directory
EQUALITY: distinguishedNameMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.12
VALID VALUES: DN of PolicyRule Object
DEFAULT VALUE: None, must be specified

Class Name: PolicyRule
DESC:
The PolicyRule class describes what conditionals should be checked against and if the checks match then what actions should be enforced. The policy makes named references to the validity period and the profile. These are required references for the policy to be considered valid. The policy must make reference to one or more actions.
The valid action combinations are:
- IPSEC Action (Drop)
- DIFFSERV Action (Drop)
- DIFFSERV Action (Pass with QOS)
- IPSEC Action (Pass in Clear)
- IPSEC Action (Pass in Clear) and DIFFSERV
- IPSec and ISAKMP Action for security
- IPSec and ISAKMP Action (security) and DIFFSERV
- RSVP Action
- RSVP and DIFFSERV Action

Requires
- ObjectClass
cn
- PolicyScope
- TrafficProfileReference
- PolicyValidityPeriodReference

Allows
- RulePriority
- PolicyRuleEnabled
- RSVPActionReference
- DiffServActionReference
- IPSecISAKMPActionReference
- IPSecSecurityActionReference

Attribute Definitions:

---
NAME: ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY: objectIdentifierMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES: PolicyRule, top
DEFAULT VALUE: PolicyRule, top
---
NAME: cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this should be also be the rdn for this object.
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES: Any string
DEFAULT VALUE: NA
---
NAME: PolicyScope
REQUIRED
MULTI-VALUED
DESC:
Specifies the scope of the policy.
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: ipsec isakmp diffserv rsvp
DEFAULT VALUE: None, must be specified

-----------

**NAME:** RulePriority
**ALLOWED** SINGLE-VALUED
**DESC:** Specifies the priority of the policy. The priority is needed to resolve conflicts between policies with overlapping profiles. A higher number indicates a higher priority policy. In general, policies with more specific profiles should have higher priority and vice versa for policies with less specific profiles. Note, values 0 thru 5 are reserved for system use.
**EQUALITY:** integerMatch
**SYNTAX:** 1.3.6.1.4.1.1466.115.121.1.27
**VALID VALUES:** 5 to 65535
**DEFAULT VALUE:** 5

-----------

**NAME:** PolicyRuleEnabled
**ALLOWED** SINGLE-VALUED
**DESC:**
This attribute is used to specify whether the policy rule is currently enabled or disabled from an Administrative point of view. Its purpose is to allow a policy to be disabled without having to remove it from the directory. By default, if this attribute is not specified the policy is enabled.
**EQUALITY:** integerMatch
**SYNTAX:** 1.3.6.1.4.1.1466.115.121.1.27
**VALID VALUES:** 0 (disabled) to 1 (enabled)
**DEFAULT VALUE:** 1 (enabled)

-----------

**NAME:** TrafficProfileReference
**REQUIRED** SINGLE-VALUED
**DESC:** Specifies the DN of the profile entry in the directory. The profile object defines what traffic/users should use this policy.
**EQUALITY:** distinguishedNameMatch
**SYNTAX:** 1.3.6.1.4.1.1466.115.121.1.12
**VALID VALUES:** The DN of a profile in the directory
**DEFAULT VALUE:** None, must be specified

-----------

**NAME:** PolicyValidityPeriodReference
**REQUIRED** SINGLE-VALUED
**DESC:** Specifies the DN of the validity period entry in the directory for when this policy should be valid.
**EQUALITY:** distinguishedNameMatch
**SYNTAX:** 1.3.6.1.4.1.1466.115.121.1.12
**VALID VALUES:** The DN of a policy validity period object in the directory
NAME: RSVPActionReference
ALLOWED
SINGLE-VALUED
DESC:
Specifies the DN of the RSVP Action entry in the directory that should be enforced by this policy.
EQUALITY: distinguishedNameMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.12
VALID VALUES: The DN of a RSVP Action object in the directory
DEFAULT VALUE: None, must be specified

NAME: DiffServActionReference
ALLOWED
SINGLE-VALUED
DESC:
Specifies the DN of the DIFFSERV Action entry in the directory that should be enforced by this policy.
EQUALITY: distinguishedNameMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.12
VALID VALUES: The DN of a DIFFSERV Action object in the directory
DEFAULT VALUE: None, must be specified

NAME: IPSecISAKMPActionReference
ALLOWED
SINGLE-VALUED
DESC:
Specifies the DN of the ISAKMP (Key Management) Action entry in the directory that should be enforced by this policy. The IPSecSecurityActionReference attribute must be present if this attribute is specified.
EQUALITY: distinguishedNameMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.12
VALID VALUES: The DN of a ISAKMP Action object in the directory
DEFAULT VALUE: None, must be specified

NAME: IPSecSecurityActionReference
ALLOWED
SINGLE-VALUED
DESC:
Specifies the DN of the IPSEC Security Action entry in the directory that should be enforced by this policy. If the IPSEC Action referred to by this attribute specifies a Secure Connection, then the IPSecISAKMPActionReference must be initialized to the DN of a valid IPSEC Action.
EQUALITY: distinguishedNameMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.12
VALID VALUES: The DN of a IPSEC Action object in the directory
DEFAULT VALUE: None, must be specified
ClassName: TrafficProfile

DESC:
The profile determines the set of information that should be used to select a particular policy. The profile consists of source address and destination address information, protocol information, and source and destination port information. The profile also may be defined to select on the TOS byte and the ingress and egress interface addresses. The profile for an IPSEC security policy should state the source address(s) as the traffic to encapsulated into the tunnel and the destination address(s) should be on the remote side of the tunnel.

Requires
ObjectClass
\cn
Allows
Interface
SourceAddressRange
DestinationAddressRange
SourcePortRange
DestinationPortRange
ProtocolNumber
ReceivedTOSByteCheck
LocalID
RemoteID

Attribute Definitions:

-----------
NAME:ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY:objectIdentifierMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES:TrafficProfile, top
DEFAULT VALUE:TrafficProfile, top
-----------
NAME:cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this should be also be the rdn for this object.
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES:Any string
DEFAULT VALUE:NA
-----------
NAME:Interface
ALLOWED
MULTI-VALUED
DESC:
Specifies for which ingress and egress interfaces the profile should match against. If this attribute is missing then the default is any interface.
NAME: SourcePortRange
ALLOWED
SINGLE-VALUED
DESC: Specifies the range of source ip addresses the profile should match against. If this attribute is missing then the default is any source ip address will match this profile. If the range format is used then the ipaddr end address must be greater than or equal to the starting ipaddr.

Format: 1:<ipaddr start>-<ipaddr-end>
Example: 1:1.1.1.1-255.255.255.255 #Single Addr
Example: 2:1.1.1.1-1.1.1.1 #Range-Single Addr
Example: 3:1.1.1.1-1.1.1.1 #Range
EQUALITY: caseExactIA5Match
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any correctly formatted source address mask/range
DEFAULT VALUE: Any source ip address

NAME: DestinationAddressRange
ALLOWED
SINGLE-VALUED
DESC: Specifies the range of destination ip addresses the profile should match against. If this attribute is missing then the default is any destination ip address will match this profile. If the range format is used then the ipaddr end address must be greater than or equal to the starting ipaddr.

Format: 1:<ipaddr start>-<ipaddr-end>
Example: 1:1.1.1.1-255.255.255.255 #Single Addr
Example: 2:1.1.1.1-1.1.1.1 #Range-Single Addr
Example: 3:1.1.1.1-1.1.1.1 #Range
EQUALITY: caseExactIA5Match
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any correctly formatted destination address mask/range
DEFAULT VALUE: Any destination ip address

NAME: SourcePortRange
ALLOWED
SINGLE-VALUED
DESC:
Specifies the range of source ports that should match this profile. The ending port number must be greater than or equal to the starting port number.
Format:<starting Port Number>:<ending Port Number>
Example:23:23 (telnet traffic)
EQUALITY:caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES:Any correctly formatted port range
DEFAULT VALUE:All Source Ports

NAME:DestinationPortRange
ALLOWED
SINGLE-VALUED
DESC:
Specifies the range of destination ports that should match this profile. The ending port number must be greater than or equal to the starting port number.
Format:<starting Port Number>:<ending Port Number>
Example:23:23 (telnet traffic)
EQUALITY:caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES:Any correctly formatted port range
DEFAULT VALUE:All Destination Ports

NAME:ProtocolNumber
ALLOWED
SINGLE-VALUED
DESC:
Specifies the range of IP Protocols that should match this profile. The ending protocol number must be greater than or equal to the starting protocol number.
Format:<starting protocol>:<ending protocol>
Example:17:17 (UDP traffic)
EQUALITY:caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES:Any correctly formatted protocol range
DEFAULT VALUE:All protocols

NAME:ReceivedTOSByteCheck
ALLOWED
SINGLE-VALUED
DESC:
Specifies the value(s) of the IP TOS byte that should match this profile. This attribute is formatted with a mask that should be applied to the incoming TOS byte and then the value that it should match. The mask and match values are a string of 0's and 1's that specifies the 8-bit field. The received TOS byte is ANDed with the MASK and the result is compared against the Match.
Format:<xxxxxxxx>:<yyyyyyyy> (Mask:Match)
Example:11111111:00000001 (TOS Byte of 0x01 matches)
EQUALITY:caseExactIA5Match
NAME: LocalID
ALLOWED
SINGLE-VALUED
DESC:
Specifies the local ID information that should be used during ISAKMP Phase 1 negotiations. The local ID is optional. If specified, then the local ID sent will be of the type idType with the specified value.
If the localID attribute is not specified then the local tunnel IP address will be sent in the IDii.
Format: idType:value
ID Types: 2 (FQDN), 3 (USER FQDN), 11 (KEYID)
Example: 2: foo@raleigh.ibm.com
EQUALITY: caseExactIA5Match

NAME: RemoteID
ALLOWED
MULTI-VALUED
DESC:
Specifies the remote ISAKMP Peers that may initiate ISAKMP/IPSEC to the local entity. The remoteID attribute is optional. If specified then the IDii received as a responder will be matched in the profile to determine which Phase 1 policy to negotiate with the with the remote peer.
If the IDii cannot be matched, then the remote peers Phase 1 negotiations will fail. Note this attribute is multi-valued any thus allows a list of remote peers to be configured that should be allowed access. This attribute does not authenticate the user, it just may be used to further determine which users SHOULD be authenticated.
Format: idType:value
ID Types: 2 (FQDN), 3 (USER FQDN), 11 (KEYID)
Example: 2: foo@raleigh.ibm.com
EQUALITY: caseExactIA5Match

ClassName: PolicyValidityPeriod
DESC:
The validity period specifies the life of the policy, the months that it should be valid, the days of the week it should be valid, and the hours of the day it should be valid. When a policy becomes invalid the next most specific policy will be enforced. This is useful to define a policy that says on Monday thru
Friday from 9 to 5, secure all traffic from Subnet A to Subnet B and any other time drop all traffic.

Requires
ObjectClass
cn
Allows
PolicyValidityTime
PolicyValidityMonthMask
PolicyValidityDayOfWeekMask
PolicyValidityTimeOfDayRange

Attribute Definitions:

__________
NAME:ObjectClass
REQUIRED
MULTI-VALUED
DESC: The class type for this object.
EQUALITY:objectIdentifierMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES:PolicyValidityPeriod, top
DEFAULT VALUE:PolicyValidityPeriod, top

----------

NAME:cn
REQUIRED
MULTI-VALUED
DESC: The common name for this object, this should be also be the rdn for this object.
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES:Any string
DEFAULT VALUE:NA

----------

NAME:PolicyValidityTime
ALLOWED
SINGLE-VALUED
DESC: Specifies the duration of the policy. The attribute states when the policy becomes valid, when it expires and optionally what time zone it should apply in. If the Time zone is omitted then the is local time at the Policy Decision point.
Format: yyyyMMddhhmmss:yyyyMMddhhmmss:timezone
Example: 19980101000000:19981231235959 (1998 only)
Example: 19980101000000:19981231235959:GMT (1998-GMT time)
EQUALITY:caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES:Any correctly formatted duration
DEFAULT VALUE:No expiration, duration is forever

----------

NAME:PolicyValidityMonthMask
ALLOWED
SINGLE-VALUED
DESC:
Specifies the months during which the policy is valid. The format is a string denoting a mask of 12 zeros and ones. A 1 states that month should be considered valid. The first bit in the mask is the month of January.
Format:xxxxxxxxxxxx
Example:111000000000 (January, February, March)
EQUALITY:caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any correctly formatted month mask
DEFAULT VALUE: Every month
----------
__________
NAME: PolicyValidityDayOfWeekMask
ALLOWED SINGLE-VALUED
DESC:
Specifies the days of the week during which the policy is valid. The format is a string denoting a mask of 7 zeros and ones. A 1 states that day should be considered valid. The first bit in mask is Monday.
Format:xxxxxxxx
Example:1111100 (Monday thru Friday)
EQUALITY: caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any correctly formatted day mask
DEFAULT VALUE: Every day
----------
__________
NAME: PolicyValidityTimeOfDayRange
ALLOWED SINGLE-VALUED
DESC:
Specifies the time of the day during which the policy is valid. The format is a start time: end time, 24 hr format. If the end time is less than the start time then wrap around midnight is assumed.
Format:<start hhmmss>:<end hhmmss>
Example: 090000:170000 (9:00am to 5:00pm)
Example: 170000:090000 (5:00pm to 12:00am to 9:00am)
EQUALITY: caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any correctly formatted day mask
DEFAULT VALUE: Every day
----------

ClassName: IPSecTransform
DESC:
The attributes of the IPSEC Transform contain information about the IPSEC Encryption and Authentication parameters and also specify how often the keys are refreshed. The transform is either AH (authentication only) or ESP (Encryption and/or Authentication) and may be specified to operate in Tunnel or Transport mode.
Requires
ObjectClass
cn
IPSecProtocolID
Allows
EncapsulationMode
AHIntegrityAlgorithm
ESPIntegrityAlgorithm
ESPCipherAlgorithm
SecurityAssociationLifeTimeSec
SecurityAssociationLifeTimeKBytes

Attribute Definitions:

__________
NAME:ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY:objectIdentifierMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES:IPSecTransform, top
DEFAULT VALUE:IPSecTransform, top

__________
NAME:cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this may also be the rdn for this object.
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES:Any string
DEFAULT VALUE:NA

__________
NAME:IPSecProtocolID
REQUIRED
SINGLE-VALUED
DESC:
Specifies the type of IPSEC phase2 transform.
Format:integer
Example:3 (ESP)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:2 (AH), 3 (ESP)
DEFAULT VALUE:Required, Must be specified

__________
NAME:EncapsulationMode
ALLOWED
SINGLE-VALUED
DESC:
Specifies the encapsulation mode for the transform.
Format:integer
Example:1 (Tunnel)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
<table>
<thead>
<tr>
<th>NAME</th>
<th>ALLOWED</th>
<th>SINGLE-VALUED</th>
<th>DESC</th>
<th>FORMAT</th>
<th>EXAMPLE</th>
<th>EQUALITY</th>
<th>SYNTAX</th>
<th>VALID VALUES</th>
<th>DEFAULT VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH Integrity Algorithm</td>
<td></td>
<td></td>
<td>Specifies the type of integrity transform in AH.</td>
<td>integer</td>
<td>3 (SHA)</td>
<td>integerMatch</td>
<td>1.3.6.1.4.1.1466.115.121.1.27</td>
<td>2 (HMAC-MD5), 3 (HMAC-SHA)</td>
<td>2 (HMAC-MD5)</td>
</tr>
<tr>
<td>ESP Integrity Algorithm</td>
<td></td>
<td></td>
<td>Specifies the type of integrity transform in ESP.</td>
<td>integer</td>
<td>2 (SHA)</td>
<td>integerMatch</td>
<td>1.3.6.1.4.1.1466.115.121.1.27</td>
<td>0 (None), 1 (HMAC-MD5), 2 (HMAC-SHA)</td>
<td>2 (HMAC-SHA)</td>
</tr>
<tr>
<td>ESP Cipher Algorithm</td>
<td></td>
<td></td>
<td>Specifies the type of cipher algorithm to use in ESP.</td>
<td>integer</td>
<td>2 (DES)</td>
<td>integerMatch</td>
<td>1.3.6.1.4.1.1466.115.121.1.27</td>
<td>2 (DES), 3 (3DES), 11 (NULL), 37 (IBM CDMF)</td>
<td>2 (DES)</td>
</tr>
<tr>
<td>Security Association LifeTime Sec</td>
<td></td>
<td></td>
<td>Specifies the lifetime of the Security Association in seconds.</td>
<td>integer</td>
<td>3600 (1 hour)</td>
<td>integerMatch</td>
<td>1.3.6.1.4.1.1466.115.121.1.27</td>
<td>120 to 2147483647</td>
<td>3600</td>
</tr>
<tr>
<td>Security Association LifeTime KBytes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

136  Application-Driven Networking: Concepts and Architecture for Policy-Based Systems
ALLOWED
SINGLE-VALUED
DESC:
Specifies the lifesize of the Security Association in KB
Format: integer
Example: 50000 (50,000 KB)
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 1024 to 2147483647
DEFAULT VALUE: 50000

ClassName: IPSecProposal
DESC:
The IPSEC Proposal contains the information about which
ESP and/or AH transform should be proposed/checked against
during phase 2 ISAKMP negotiations. If Perfect Forward
Secrecy is required (A Fresh Diffie Hellman calculation),
then the IPSEC Proposal contains which DH Group to use.
The transforms referenced by the IPSEC Proposal are
sent/checked again in the order in which they are
specified. The first ESP/AH transform in the list should
be the one that is most appropriate to use. If there is
more than one transform in the list, then each one is
compared against the peers list to find a match. If none
of the configured transforms matches the peers list of
transforms then the negotiation will fail. The IPSEC
Proposal may list a combination of AH and ESP transforms,
but the only valid combinations are:
1: List of AH Only (Tunnel or Transport Mode)
2: List of ESP Only (Tunnel or Transport Mode)
3: List of AH (Transport Mode) and ESP (Tunnel Mode)

Requires
ObjectClass

cn
PerfectForwardSecrecy

Allows
DefaultDiffHellmanGroupId
AHProtocolTransformReference
ESPProtocolTransformReference

Attribute Definitions:

NAME: ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY: objectIdentifierMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES: IPSecProposal, top
DEFAULT VALUE: IPSecProposal, top

NAME: cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this may also be the rdn for this object.
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES: Any string
DEFAULT VALUE: NA
----------
NAME: PerfectForwardSecrecy
REQUIRED
SINGLE-VALUED
DESC:
Specifies the whether Perfert Forward Secrecy is required. PFS denotes whether a fresh Diffie Hellman exchange is required for the phase2 quick mode negotiation.
Format: integer
Example: 1
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 0 (PFS Not Required), 1 (PFS Required)
DEFAULT VALUE: 0
----------
NAME: DefaultDiffHellmanGroupId
ALLOWED
SINGLE-VALUED
DESC:
If the PerfectForwardSecrecy attribute is set to 1 (Required), then this attribute specifies which DH Group to use. The default, if not specified, is to use Group 1.
Format: integer
Example: 1
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 1 (Group 1), 2 (Group 2)
DEFAULT VALUE: 1
----------
NAME: AHProtocolTransformReference
ALLOWED
MULTI-VALUED
DESC:
If AH Transforms should be negotiated during phase 2, then this attribute specifies which AH transforms should be sent in this proposal. This attribute is multi-valued and lists in priority order which transforms should be sent (initiator) or checked against (responder). Note: At least one transform must be specified in the proposal for the proposal to be valid.
Format: <pref>:<DN of IPSecTransform>
Example: 1:cn=ahVeryStrong,o=ibm,c=us
Example: 2:cn=ahStrong,o=ibm,c=us
EQUALITY: caseExactIA5Match
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any properly formatted AH transform reference
DEFAULT VALUE: No AH Transforms in proposal

---

NAME: ESPProtocolTransformReference
ALLOWED
MULTI-VALUED
DESC:
If ESP Transforms should be negotiated during phase 2, then this attribute specifies which ESP transforms should be sent in this proposal. This attribute is multi-valued and lists in priority order which transforms should be sent (initiator) or checked against (responder). Note: At least one transform must be specified in the proposal for the proposal to be valid.
Format: <pref>:<DN of IPSecTransform>
Example: 1: cn=espVeryStrong, o=ibm, c=us
Example: 2: cn=espStrong, o=ibm, c=us
EQUALITY: caseExactIA5Match
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any properly formatted ESP transform reference.
DEFAULT VALUE: No ESP Transforms in proposal

---

ClassName: IPSecSecurityAction
DESC:
The IPSEC Action may specify a Pass, Drop or Secure action. If the action is drop then all packets matching this policy will be dropped. If the action is pass with no security then all packets will be passed in the clear. If the action is pass with security then all packets will be secured via the security association negotiated by the information specified by this action. The IPSEC Action also contains the IP addresses of the tunnel endpoints for the IPSEC tunnel and IKE SAs. The attributes of the security association will be determined by the IPSEC proposals that are referenced by the IPSEC Action. Multiple IPSEC Proposals may be specified in the IPSEC action and they are sent/checked against in the order they are specified. Having multiple proposals in a IPSEC action allows the configuration to contain all the acceptable combinations of security thereby reducing the number of potential configuration mismatch between VPN gateways.
Requires
ObjectClass
cn
SecurityAction
Allows
IPSecTunnelStart
IPSecTunnelEnd
IPSecProposalReference
MinSARefreshPercentage
SecurityAssociationRefreshThreshold
IPSecAutoStartFlag
IPSecCopyDFBit
IPSecReplayPrev
IPSecTunnelInTunnel
Attribute Definitions:

____________________________
NAME:ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY:objectIdentifierMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES:IPSecSecurityAction, top
DEFAULT VALUE:IPSecSecurityAction, top

____________________________
NAME:cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this may also be the rdn for this object.
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES:Any string
DEFAULT VALUE:NA

____________________________
NAME:SecurityAction
REQUIRED
SINGLE-VALUED
DESC:
This attribute specifies what type of security action should be applied to this packet. A value of Deny states: packets matching this action should be dropped.
A value of Permit means one of two things:
1: If the IPSecProposalReference attribute IS NOT present in this object, then send the packet in the clear.
2: If the IPSecProposalReference attribute IS present in this object, then the packet must be secured by IPSEC.
EQUALITY:caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES:Permit, Deny
DEFAULT VALUE:Permit

____________________________
NAME:IPSecTunnelStart
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies the local IP Address of the IPSEC tunnel. This is the IP Address that should be used during the phase 1/2 negotiations and in the IP header for the secured traffic. This must be a valid IP Address on the box retrieving this action.
Example:1.1.1.1
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES:Any valid IP Address (dotted decimal IPv4 Address)
DEFAULT VALUE:Must be specified if action is permit and secure
Standards Documents and Sample Files

NAME: IPSecTunnelEnd
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies the remote IP Address of the IPSec tunnel. This is the IP Address that packets should be sent to or received from during the phase 1/2 negotiations and in the IP header for the secured traffic. If the IP Address of the remote peer is unknown, i.e. Remote Access Users, then a value of 0.0.0.0 may be specified. If 0.0.0.0 is specified, then the device retrieving will only be allowed to respond to ISAKMP negotiations.
Example:1.1.1.1
Example:0.0.0.0 (Unknown, Remote Access User)
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any valid IP Address (dotted decimal IPv4 Address)
DEFAULT VALUE: 0.0.0.0

NAME: IPSecProposalReference
ALLOWED
MULTI-VALUED
DESC:
If the SecurityAction type is Permit and the traffic should be secured, then this attribute specifies which proposals should be sent/checked against during quick mode negotiations. This attribute is multi-valued and lists in priority order which IPSecProposals should be sent (initiator) or checked against (responder). Note: At least one proposal must be specified if the SecurityAction is permit and the action should secure traffic. If the IPSecProposalReference is not present then the action will be assumed to be permit traffic in the clear.
Format: <pref>: <DN of IPSecProposal>
Example: 1: cn=veryStrongProposal,o=ibm,c=us
Example: 2: cn=strongProposal,o=ibm,c=us
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any valid formatted proposal reference
DEFAULT VALUE: No proposals (clear)

NAME: MinSARefreshPercentage
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies percentage of the configured lifetime and lifesize values in the IPSecTransform to accept. There may be situations where the peer policy contains lifetime and lifesize values that are smaller than the local values. If this is true, then this attribute specifies what percentage of the local values should be acceptable when comparing the remote values to the local values. This attribute gives the user the flexibility to accept a range of values, but not accept a
value so small that it hurts processing performance.
Format:<integer>
Example:75 (percent)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:1 to 100
DEFAULT VALUE:75

---

NAME:SecurityAssociationRefreshThreshold
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies percentage of the
negotiated lifetime and lifetsize for the SA before the
refresh is actually started. For instance, if the
negotiated lifetime is 3600 seconds (1 hour) and the
value for SecurityAssociationRefreshThreshold is 75
percent then the SA refresh will actually occur after 2700
seconds. This allows for some overlap between refreshes
so that no data is lost.
Format:<integer>
Example:85 (percent)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:1 to 100
DEFAULT VALUE:85

---

NAME:IPSecAutoStartFlag
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies whether the phase 2 negotiations
should automatically start at device initialization.
Format:<integer>
Example:1 (Autostart)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:0 (autostart off), 1 (autostart on)
DEFAULT VALUE:0

---

NAME:IPSecCopyDFBit
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies whether to copy the Dont
Fragment bit into the outer tunnel header or to
set it or to clear it.
Format:<integer>
Example:0 (copy)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:0 (copy), 1 (set), 2 (clear)
DEFAULT VALUE:0

---
NAME: IPSecReplayPrev
ALLOWED
SINGLE-VALUED
DESC:
The attribute specifies whether the local side should enforce replay prevention.
Format:<integer>
Example:0 (disabled)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:0 (disabled), 1 (enabled)
DEFAULT VALUE:0

----------

NAME: IPSecTunnelInTunnel
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies whether the traffic secured by this IPSEC action will be encrypted again by another IPSEC tunnel on the same device. Note: The 22xx only support a maximum of 2 cascaded tunnels.
If a policy is found with an IPSEC Action with the tunnelInTunnel attribute set to 1, then there must be another policy that describes the second tunnel.
Format:<integer>
Example:1 (Yes)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:0 (No Further Tunnels), 1 (Yes, Further tunnels)
DEFAULT VALUE:0

----------

ClassName: ISAKMPProposal
DESC:
The ISAKMP Proposal specifies the encryption and authentication attributes of the phase 1 security association. It also specifies which Diffie Hellman group should be used to generate the keys and the life of the phase 1 security association.
The authentication method must be selected in the ISAKMP proposal and can be either pre-shared key or Certificate mode.
Requires
ObjectClass
 cn
ISAKMPAuthenticationMethod
ISAKMPHashAlgorithm
ISAKMPCipherAlgorithm
Allows
DefaultDiffHellmanGroupId
SecurityAssociationLifetimeSec
SecurityAssociationLifetimeKBytes

Attribute Definitions:
NAME: ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY: objectIdentifierMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES: ISAKMPProposal, top
DEFAULT VALUE: ISAKMPProposal, top

----------

NAME: cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this may also be the rdn for this object.
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES: Any string
DEFAULT VALUE: NA

----------

NAME: ISAKMPAuthenticationMethod
REQUIRED
SINGLE-VALUED
DESC:
This attribute specifies the authentication method to use to authenticate the ISAKMP phase 1 peer.
Format: <integer>
Example: 1 (pre-shared key)
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 1 (Pre-shared key), 3 (Cert - RSA Signature)
DEFAULT VALUE: 1

----------

NAME: ISAKMPSignatureAlgorithm
REQUIRED
SINGLE-VALUED
DESC:
This attribute specifies the hash algorithm to use during ISAKMP phase 1 negotiations.
Format: <integer>
Example: 1 (MD5)
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 1 (MD5), 2 (SHA)
DEFAULT VALUE: 1

----------

NAME: ISAKMPCipherAlgorithm
REQUIRED
SINGLE-VALUED
DESC:
This attribute specifies the cipher algorithm to use during ISAKMP phase 1 negotiations.
Format: <integer>
Example: 1 (DES-CBC)
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 1(DES-CBC), 5(3DES-CBC)
DEFAULT VALUE: 1

----------

NAME: DefaultDiffHellmanGroupId
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies which DH Group to use. The default, if not specified, is to use Group 1.
Format: integer
Example: 1
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 1 (Group 1), 2 (Group 2)
DEFAULT VALUE: 1

----------

NAME: SecurityAssociationLifetimeSec
ALLOWED
SINGLE-VALUED
DESC:
Specifies the lifetime of the Security Association in seconds.
Format: integer
Example: 3600 (1 hour)
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 120 to 2147483647
DEFAULT VALUE: 15000

----------

NAME: SecurityAssociationLifetimeKBytes
ALLOWED
SINGLE-VALUED
DESC:
Specifies the life size of the Security Association in KBytes
Format: integer
Example: 50000 (50,000 KB)
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 100 to 2147483647
DEFAULT VALUE: 1000

----------

ClassName: IPSecISAKMPAction
DESC:
The ISAKMP Action specifies the Key Management information for phase 1. It specifies whether the phase 1 negotiations should be started in Main Mode (Provides Identity Protection) or Aggressive Mode. It also specifies whether the phase 1 Security Association should be negotiated a box startup or on demand. The ISAKMP action also must reference one or more
ISAKMP proposals. The first reference should be to the most acceptable ISAKMP Proposal.

Requires
ObjectClass
  cn
ISAKMPExchangeMode
ISAKMPProposalReference
Allows
MinSARefreshPercentage
ISAKMPConnectionLifetimeSec
ISAKMPConnectionLifetimeKBytes
ISAKMPAutoStartFlag

Attribute Definitions:

-----------
NAME:ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY:objectIdentifierMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES:IPSecISAKMPAction, top
DEFAULT VALUE:IPSecISAKMPAction, top
-----------

NAME:cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this may also be the rdn for this object.
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES:Any string
DEFAULT VALUE:NA
-----------

NAME:ISAKMPExchangeMode
REQUIRED
SINGLE-VALUED
DESC:
This attribute specifies whether identity protection is required. If Main mode is chosen then the Identification information is guaranteed to be secured during phase 1. If Aggressive mode is chosen the identification is received earlier in the negotiations however it is unsecured.
Format:<integer>
Example:2 (Main Mode)
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:2 (Main Mode), 4 (Aggressive Mode)
DEFAULT VALUE:2
-----------

NAME:ISAKMPProposalReference
REQUIRED
MULTI-VALUED
DESC:
This attribute specifies which ISAKMPProposals should be
sent/checked against during Phase 1 negotiations.
This attribute is multi-valued and lists in priority order
which ISAKMPProposals should be sent (initiator) or checked
against (responder). Note: At least one proposal must be
specified.
Format:<pref>:<DN of ISAKMPProposal>
Example:1:cn=veryStrongProposal,o=ibm,c=us
Example:2:cn=strongProposal,o=ibm,c=us
EQUALITY:caseIgnoreMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any valid formatted proposal reference
DEFAULT VALUE: Must be specified
-----------

NAME: MinSARefreshPercentage
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies percentage of the
configured lifetime and lifesize values in the
ISAKMPProposal to accept. There may be situations where
the peer policy contains lifetime and lifesize values
that are smaller than the local values. If this is true,
then this attribute specifies what percentage of the local
values should be acceptable when comparing the remote values
to the local values. This attribute gives the user the
flexibility to accept a range of values, but not accept a
value so small that it hurts processing performance.
Format:<integer>
Example:75
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 1 to 100
DEFAULT VALUE: 75
-----------

NAME: ISAKMPConnectionLifetimeSec
ALLOWED
SINGLE-VALUED
DESC:
Specifies the amount of time, in seconds, that the phase 1
SA should automatically refresh. Once this time expires
then some other event must occur to restart the phase 1
tunnel.
Format: integer
Example: 30000
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 120 to 2147483647
DEFAULT VALUE: 30000
-----------

NAME: ISAKMPConnectionLifetimeKBytes
ALLOWED
SINGLE-VALUED
DESC:
Specifies the total number of KBytes exchanged over the phase 2 exchanges protected by this phase 1 may exchange while automatically refreshing the phase 1.
Format: integer
Example: 5000
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 100 to 2147483647
DEFAULT VALUE: 5000

NAME: ISAKMPAutoStartFlag
ALLOWED
SINGLE-VALUED
DESC:
This attribute specifies whether the phase 1 negotiations should automatically start at device initialization.
Format: <integer>
Example: 1 (Autostart)
EQUALITY: integerMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 0 (autostart off), 1 (autostart on)
DEFAULT VALUE: 0

ClassName: DiffServAction
DESC:
The DiffServ Action describes the quality of service that should be provided to packets matching a policy that specifies a DiffServ Action. The DiffServ Action may be configured to drop packets thus providing a QOS of 0 percent. It may also be used to map packets into relative qualities of service. The bandwidth allocated may be configured as a percentage of output bandwidth or as an absolute value in kbps. The user must specify whether the best effort/assured queue or the premium queue should provide the bandwidth allocation. The DiffServ Action also specifies how the TOS byte should be marked before it is sent out the egress interface. By default the TOS byte is not marked. It is useful to mark the packets at some point in the network based on the information in the IP packet header. Once the classification has been determined, since the TOS byte marking has already been performed, then the rest of the hops in the network can just look at the new TOS byte to determine the QOS that should be applied to this packet.
Requires
ObjectClass
cn
DiffServPermission
Allows
DiffServOutTOSByte
DiffServBandwidthShare
DiffServQueuePriority
Attribute Definitions:

__________
NAME:ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY:objectIdentifierMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES:DiffServAction, top
DEFAULT VALUE:DiffServAction, top

__________
NAME:cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this should be
also be the rdn for this object.
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES:Any string
DEFAULT VALUE:NA

__________
NAME:DiffServPermission
REQUIRED
SINGLE-VALUED
DESC:
Accept or Deny DiffServ Flow. If the permission is
Deny then all packets matching this action will be
dropped. If the permission is Accept then the
packets matching this action will have the service
described by this action applied to them.
EQUALITY:caseIgnoreString
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES:Accept, Deny
DEFAULT VALUE:Accept

__________
NAME:DiffServOutTOSByte
ALLOWED
SINGLE-VALUED
DESC:
Specifies the marking of the IP TOS byte that should
be applied to packets be forwarded by this device.
This attribute is formatted with a mask that
should be applied to the outgoing TOS byte and then the
value to be ORed into TOS Byte. The mask and mark values
are a string of 0's and 1's that specifies the 8-bit field.
Zeros in the mask imply that the corresponding bit should
not change. A one implies that the bit should be marked
with the bit value in the mark byte. The operation is:
newTOSByte = (Mask^ & receivedTOSByte) | (Mask&Mark)
The ^ is a bitwise complement
Format:<xxxxxxxx>:<yyyyyyyy>
Example:11111101:00000001
Using the example, a received value 0x07 would be sent
out with a value of 0x02
EQUALITY:caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any correctly formatted mask: mark
DEFAULT VALUE: Do not mark outgoing tos byte (00000000:00000000)

---------

NAME: DiffServBandwidthShare
ALLOWED
SINGLE-VALUED
DESC:
Specifies the amount of bandwidth packets matching that action should be allocated. The syntax for the value of this attribute is the type of allocation and the amount of allocation. The bwType can be a 1 (indicating absolute allocation in kbps) or a 2 (indicating a percentage of output bandwidth).
Format: <bwType>:<allocation amount>
Example: 1:100000 (100000 kbps)
Example: 2:60 (60 percentage)
EQUALITY: caseExactIA5Match
SYNTAX:1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Any correctly formatted bandwidth share
DEFAULT VALUE: Must be specified if the permission is permit

---------

NAME: DiffServQueuePriority
ALLOWED
SINGLE-VALUED
DESC:
Specifies the queue that packets matching this action should be put into.
Format: Integer
Example: 1
Example: 2
EQUALITY: integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES: 1 (Premium), 2 (Assured/Best Effort)
DEFAULT VALUE: 2

---------

ClassName: RSVPAction
DESC:
The RSVP Action specifies whether RSVP flows should be permitted or denied when a RSVP reservation occurs and the reservation request matches the profile of the policy. If the reservation is to be permitted then the RSVP Action also states the allowed duration of the reservation, the allowed bandwidth, and optionally a reference to a DiffServ Action. The reference to the DiffServ Action allows RSVP to determine how to mark the TOS byte before the packet leaves the router. This is useful when packets leave an RSVP network into a DiffServ network. RSVP can provide the QOS up to the RSVP boundary and then mark the TOS byte appropriately so the DiffServ network can now apply the correct bandwidth.
Requires
ObjectClass
cn
RSVPPermission
Allows
RSVPMaxRatePerFlow
RSVPMaxFlowDuration
RSVPtoDiffServReference

Attribute Definitions:

NAME: ObjectClass
REQUIRED
MULTI-VALUED
DESC:
The class type for this object.
EQUALITY: objectIdentifierMatch
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.38
VALID VALUES: RSVPAction, top
DEFAULT VALUE: RSVPAction, top

NAME: cn
REQUIRED
MULTI-VALUED
DESC:
The common name for this object, this should be also be the rdn for this object.
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.15
VALID VALUES: Any string
DEFAULT VALUE: NA

NAME: RSVPPermission
REQUIRED
SINGLE-VALUED
DESC:
Accept or Deny RSVP requests for flows matching this action. If the permission is Deny then all RSVP reservations matching this action will be denied. If the permission is Accept then the RSVP flows matching this action will be accepted with the limitations/guarantees described by this action.
EQUALITY: caseIgnoreString
SYNTAX: 1.3.6.1.4.1.1466.115.121.1.26
VALID VALUES: Accept, Deny
DEFAULT VALUE: Accept

NAME: RSVPMaxRatePerFlow
ALLOWED
SINGLE-VALUED
DESC:
The maximum amount of bandwidth (in kbs) that RSVP should be allowed to allocate for one individual flow.
EQUALITY: integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:integer
DEFAULT VALUE:Must specify if permission is Accept

NAME:RSVPMaxFlowDuration
ALLOWED
SINGLE-VALUED
DESC:
The maximum amount of time RSVP should allow a reservation to be active (in seconds), 0 means no limit
EQUALITY:integerMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.27
VALID VALUES:integer
DEFAULT VALUE:600

NAME:RSVPtoDiffServReference
ALLOWED
SINGLE-VALUED
DESC:
Optional. The name of a configured diffserv action to map RSVP flows onto. RSVP will use the information from the diffserv action to mark the TOS byte for the next diffserv enabled upstream device. This is intended for use in networks where packets leave an RSVP enabled network into a DIFFSERV enabled network.
EQUALITY:distinguishedNameMatch
SYNTAX:1.3.6.1.4.1.1466.115.121.1.12
VALID VALUES:The DN of any valid DiffServAction
DEFAULT VALUE:None, no relationship to DiffServ

A.5.2 LDIF File with Example Policy Templates

### ORGANIZATION - this is not a policy object, but there must be some node created under which the policy objects can be placed. Most likely this will not be the organization ibm.###
### If there is already a node in the DIT that is suitable for this information then remove this organization object and change the suffix (o=ibm, c=us) to the existing node you wish to use. If you want to create a new organization then change the "o=ibm, c=us" to something else and then change the suffixes in the other policy objects.###

dn: o=ibm, c=us
objectclass: organization
#1 All the time
dn: cn=allTheTime, o=ibm, c=us
objectclass: policyvalidityperiod
cn: allTheTime

#2 All hours during the week
dn: cn=allTheTimeMonThruFri, o=ibm, c=us
objectclass: policyvalidityperiod
cn: allTheTimeMonThruFri
policyvaliditydayofweekmask: 1111100

#3 Working hours 9am to 5pm
dn: cn=9to5MonThruFri, o=ibm, c=us
objectclass: policyvalidityperiod
cn: 9to5MonThruFri
policyvaliditydayofweekmask: 1111100
policyvaliditytimeofdayrange: 090000:170000

#4 After hours 5pm to 9am
dn: cn=5to9MonThruFri, o=ibm, c=us
objectclass: policyvalidityperiod
cn: 5to9MonThruFri
policyvaliditydayofweekmask: 1111100
policyvaliditytimeofdayrange: 170000:090000

# DIFFSERV ACTIONS

dn: cn=EF, o=ibm, c=us
objectclass: diffservaction
cn: EF
diffservpermission: accept
diffservbandwidthshare: 2:19
diffservqueuepriority: 1
diffservouttosbyte: 11111100:10111000

dn: cn=AF1, o=ibm, c=us
objectclass: diffservaction
cn: AF1
diffservpermission: accept
diffservbandwidthshare: 2:15
diffservqueuepriority: 2
diffservouttosbyte: 11111100:01010000

dn: cn=AF2, o=ibm, c=us
objectclass: diffservaction
cn: AF2
diffservpermission: accept
diffservbandwidthshare: 2:10
diffservqueuepriority: 2
diffservouttosbyte: 11111100:01001000

dn: cn=AF3, o=ibm, c=us
objectclass: diffservaction
cn: AF3
diffservpermission: accept
diffservvbandwidthshare: 2:10
diffservqueuepriority: 2
diffservvouttosbyte: 1111100:01101000
dn: cn=AF4, o=ibm, c=us
objectclass: diffservaction
cn: AF4
diffservpermission: accept
diffservvbandwidthshare: 2:5
diffservqueuepriority: 2
diffservvouttosbyte: 1111100:10001000

############################
##### ISAKMP PROPOSALS #####
############################

#1 strong p1 proposal - pre-shared key, MD5 Auth, DES encr, DH Grp 1
dn: cn=strongP1PropSharedKey, o=ibm, c=us
objectclass: isakmpproposal
cn: strongP1PropSharedKey
isakmpauthenticationmethod: 1
isakmphashalgorithm: 1
isakmpcipheralgorithm: 1
defaultdiffhellmangroupid: 1

#2 strong p1 proposal - RSA Cert, MD5 Auth, DES encr, DH Grp 1
dn: cn=strongP1PropRSACert, o=ibm, c=us
objectclass: isakmpproposal
cn: strongP1PropRSACert
isakmpauthenticationmethod: 3
isakmphashalgorithm: 1
isakmpcipheralgorithm: 1
defaultdiffhellmangroupid: 1

#3 very strong p1 proposal - Pre-shared key, SHA Auth, 3DES encr, DH Grp 1
dn: cn=veryStrongP1PropSharedKey, o=ibm, c=us
objectclass: isakmpproposal
cn: veryStrongP1PropSharedKey
isakmpauthenticationmethod: 1
isakmphashalgorithm: 2
isakmpcipheralgorithm: 5
defaultdiffhellmangroupid: 1

#4 very strong p1 proposal - RSA Cert, SHA Auth, 3DES encr, DH Grp 1
dn: cn=veryStrongP1PropRSACert, o=ibm, c=us
objectclass: isakmpproposal
cn: veryStrongP1PropRSACert
isakmpauthenticationmethod: 3
isakmphashalgorithm: 2
isakmpcipheralgorithm: 5
defaultdiffhellmangroupid: 1

############################
##### ISAKMP ACTIONS #####
############################
#1 Phase 1 action, main mode, strong pre-shared key and cert proposals and very strong
# pre-shared key and cert proposals
dn: cn=generalPhase1Action, o=ibm, c=us
objectclass: ipsecisakmpaction
cn: generalPhase1Action
isakmpexchangemode: 2
isakmpproposalreference: 1: cn=veryStrongP1PropRSACert, o=ibm, c=us
isakmpproposalreference: 2: cn=strongP1PropRSACert, o=ibm, c=us
isakmpproposalreference: 3: cn=veryStrongP1PropSharedKey, o=ibm, c=us
isakmpproposalreference: 4: cn=strongP1PropSharedKey, o=ibm, c=us
isakmpconnectionlifetimesec: 30000
isakmpconnectionlifetimekbytes: 5000
isakmpautostartflag: 0

#########################################################
##### IPSEC TRANSFORMS #####
#########################################################
#1 AH Transform, Transport Mode, MD5 authentication
dn: cn=ahTransportMD5, o=ibm, c=us
objectclass: ipsectransform
cn: ahTransportMD5
ipsecprotocolid: 2
ahintegrityalgorithm: 2
encapsulationmode: 2

#2 AH Transform, Transport Mode, SHA authentication
dn: cn=ahTransportSHA, o=ibm, c=us
objectclass: ipsectransform
cn: ahTransportSHA
ipsecprotocolid: 2
ahintegrityalgorithm: 3
encapsulationmode: 2

#3 AH Transform, Tunnel Mode, MD5 authentication
dn: cn=ahTunnelMD5, o=ibm, c=us
objectclass: ipsectransform
cn: ahTunnelMD5
ipsecprotocolid: 2
ahintegrityalgorithm: 2
encapsulationmode: 1

#4 AH Transform, Tunnel Mode, SHA authentication
dn: cn=ahTunnelSHA, o=ibm, c=us
objectclass: ipsectransform
cn: ahTunnelSHA
ipsecprotocolid: 2
ahintegrityalgorithm: 3
encapsulationmode: 1

#5 ESP Transform, Tunnel Mode with MD5 and DES
dn: cn=espTunnelMD5andDES, o=ibm, c=us
objectclass: ipsectransform
cn: espTunnelMD5andDES
ipsecprotocolid: 3
espintegrityalgorithm: 1
espcipheralgorithm: 2
#6 ESP Transform, Tunnel Mode with SHA and DES
dn: cn=espTunnelSHAandDES, o=ibm, c=us
objectclass: ipsectransform
cn: espTunnelSHAandDES
ipsecprotocolid: 3
encapsulationmode: 1
espintegrityalgorithm: 2
espcipheralgorithm: 2

#7 ESP Transform, Tunnel Mode with MD5 and 3DES
dn: cn=espTunnelMD5and3DES, o=ibm, c=us
objectclass: ipsectransform
cn: espTunnelMD5and3DES
ipsecprotocolid: 3
encapsulationmode: 1
espintegrityalgorithm: 1
espcipheralgorithm: 3

#8 ESP Transform, Tunnel Mode with SHA and 3DES
dn: cn=espTunnelSHAand3DES, o=ibm, c=us
objectclass: ipsectransform
cn: espTunnelSHAand3DES
ipsecprotocolid: 3
encapsulationmode: 1
espintegrityalgorithm: 2
espcipheralgorithm: 3

#9 ESP Transform, Tunnel Mode with no authentication and DES
dn: cn=espTunnelDES, o=ibm, c=us
objectclass: ipsectransform
cn: espTunnelDES
ipsecprotocolid: 3
encapsulationmode: 1
espintegrityalgorithm: 0
espcipheralgorithm: 2

#10 ESP Transform, Tunnel Mode with no authentication and 3DES
dn: cn=espTunnel3DES, o=ibm, c=us
objectclass: ipsectransform
cn: espTunnel3DES
ipsecprotocolid: 3
encapsulationmode: 1
espintegrityalgorithm: 0
espcipheralgorithm: 3

#11 ESP Transform, Transport Mode with MD5 and DES
dn: cn=espTransportMD5andDES, o=ibm, c=us
objectclass: ipsectransform
cn: espTransportMD5andDES
ipsecprotocolid: 3
encapsulationmode: 2
espintegrityalgorithm: 1
espcipheralgorithm: 2

#12 ESP Transform, Transport Mode with SHA and DES
dn: cn=espTransportSHAandDES, o=ibm, c=us
objectclass: ipsectransform
cn: espTransportSHAandDES  
ipsecprotocolid: 3  
encapsulationmode: 2  
espcipheralgorithm: 2

#13 ESP Transform, Transport Mode with MD5 and 3DES  
dn: cn=espTransportMD5and3DES, o=ibm, c=us  
objectclass: ipsectransform  
cn: espTransportMD5and3DES  
ipsecprotocolid: 3  
encapsulationmode: 2  
espcipheralgorithm: 2  
espcipheralgorithm: 3

#14 ESP Transform, Transport Mode with SHA and 3DES  
dn: cn=espTransportSHAand3DES, o=ibm, c=us  
objectclass: ipsectransform  
cn: espTransportSHAand3DES  
ipsecprotocolid: 3  
encapsulationmode: 2  
espcipheralgorithm: 2

#15 ESP Transform, Transport Mode with DES, no Auth  
dn: cn=espTransportDES, o=ibm, c=us  
objectclass: ipsectransform  
cn: espTransportDES  
ipsecprotocolid: 3  
encapsulationmode: 2  
espcipheralgorithm: 2

#16 ESP Transform, Transport Mode with 3DES, no Auth  
dn: cn=espTransport3DES, o=ibm, c=us  
objectclass: ipsectransform  
cn: espTransport3DES  
ipsecprotocolid: 3  
encapsulationmode: 2  
espcipheralgorithm: 2

###########################
##### IPSEC PROPOSALS ######
###########################

#1 Strong Phase 2 Proposal, ESP Only in Tunnel Mode, combinations of DES,MD5,SHA  
dn: cn=strongP2EspProp, o=ibm, c=us  
objectclass: ipsecproposal  
cn: strongP2EspProp  
perfectforwardsecrecy: 0  
espprotocoltransformreference: 1: cn=espTunnelMD5andDES, o=ibm, c=us  
espprotocoltransformreference: 2: cn=espTunnelSHAandDES, o=ibm, c=us

#2 Strong Phase 2 Proposal - AH-ESP in Tunnel Mode, Combinations of DES,3DES,MD5,3DES-SHA  
dn: cn=strongP2EspAhProp, o=ibm, c=us  
objectclass: ipsecproposal
#3 Very Strong Phase 2 Proposal, ESP only in Tunnel Mode, combinations of 3DES, MD5, SHA
dn: cn=veryStrongP2EspProp, o=ibm, c=us
objectclass: ipsecproposal
cn: veryStrongP2EspProp
perfectforwardsecrecy: 0
espprotocoltransformreference: 1: cn=espTransportSHAand3DES, o=ibm, c=us
espprotocoltransformreference: 2: cn=espTransportMD5and3DES, o=ibm, c=us

#4 Very Strong Phase 2 Proposal, AH-ESP in Tunnel Mode with 3DES and SHA or MD5

dn: cn=veryStrongP2EspAhProp, o=ibm, c=us
objectclass: ipsecproposal
cn: veryStrongP2EspAhProp
perfectforwardsecrecy: 0
espprotocoltransformreference: 1: cn=espTransportSHAand3DES, o=ibm, c=us
espprotocoltransformreference: 2: cn=espTransportMD5and3DES, o=ibm, c=us

#5 Very Strong Phase 2 Proposal, ESP only in Tunnel Mode, combinations of 3DES, MD5, SHA, with PFS
dn: cn=veryStrongP2EspPropPFS, o=ibm, c=us
objectclass: ipsecproposal
cn: veryStrongP2EspPropPFS
perfectforwardsecrecy: 1
espprotocoltransformreference: 1: cn=espTransportSHAand3DES, o=ibm, c=us
espprotocoltransformreference: 2: cn=espTransportMD5and3DES, o=ibm, c=us

#6 Very Strong Phase 2 Proposal, AH-ESP in Tunnel Mode with 3DES and SHA or MD5, with PFS

dn: cn=veryStrongP2EspAhPropPFS, o=ibm, c=us
objectclass: ipsecproposal
cn: veryStrongP2EspAhPropPFS
perfectforwardsecrecy: 1
espprotocoltransformreference: 1: cn=espTransportSHAand3DES, o=ibm, c=us
espprotocoltransformreference: 2: cn=espTransportMD5and3DES, o=ibm, c=us

#7 Strong Phase 2 Proposal, ESP Only in Transport Mode, combinations of DES, MD5, SHA

dn: cn=strongP2EspPropXport, o=ibm, c=us
objectclass: ipsecproposal
cn: strongP2EspPropXport
perfectforwardsecrecy: 0
espprotocoltransformreference: 1: cn=espTransportMD5and3DES, o=ibm, c=us
espprotocoltransformreference: 2: cn=espTransportSHAand3DES, o=ibm, c=us

#8 Strong Phase 2 Proposal - AH-ESP in Transport Mode, Combinations of DES, 3DES, MD5, 3DES-SHA

dn: cn=strongP2EspAhPropXport, o=ibm, c=us
objectclass: ipsecproposal
cn: strongP2EspAhPropXport
perfectforwardsecrecy: 0
#9 Very Strong Phase 2 Proposal, ESP only in Transport Mode, combinations of 3DES, MD5, SHA

\[ \text{dn: cn=veryStrongP2EspPropXport, o=ibm, c=us} \]
\[ \text{objectclass: ipsecproposal} \]
\[ \text{cn: veryStrongP2EspPropXport} \]
\[ \text{perfectforwardsecrecy: 0} \]
\[ \text{espprotocoltransformreference: 1: cn=espTransportSHAand3DES, o=ibm, c=us} \]
\[ \text{espprotocoltransformreference: 2: cn=espTransportMD5and3DES, o=ibm, c=us} \]

#10 Very Strong Phase 2 Proposal, AH-ESP in Transport Mode with 3DES and SHA or MD5

\[ \text{dn: cn=veryStrongP2EspAhPropXport, o=ibm, c=us} \]
\[ \text{objectclass: ipsecproposal} \]
\[ \text{cn: veryStrongP2EspAhPropXport} \]
\[ \text{perfectforwardsecrecy: 0} \]
\[ \text{espprotocoltransformreference: 1: cn=espTransport3DES, o=ibm, c=us} \]
\[ \text{ahprotocoltransformreference: 1: cn=ahTransportSHA, o=ibm, c=us} \]
\[ \text{ahprotocoltransformreference: 2: cn=ahTransportMD5, o=ibm, c=us} \]

#11 Very Strong Phase 2 Proposal, ESP only in Transport Mode, combinations of 3DES, MD5, SHA, with PFS

\[ \text{dn: cn=veryStrongP2EspPropPFSXport, o=ibm, c=us} \]
\[ \text{objectclass: ipsecproposal} \]
\[ \text{cn: veryStrongP2EspPropPFSXport} \]
\[ \text{perfectforwardsecrecy: 1} \]
\[ \text{espprotocoltransformreference: 1: cn=espTransportSHAand3DES, o=ibm, c=us} \]
\[ \text{espprotocoltransformreference: 2: cn=espTransportMD5and3DES, o=ibm, c=us} \]

#12 Very Strong Phase 2 Proposal, AH-ESP in Transport Mode with 3DES and SHA or MD5, with PFS

\[ \text{dn: cn=veryStrongP2EspAhPropPFSXport, o=ibm, c=us} \]
\[ \text{objectclass: ipsecproposal} \]
\[ \text{cn: veryStrongP2EspAhPropPFSXport} \]
\[ \text{perfectforwardsecrecy: 1} \]
\[ \text{espprotocoltransformreference: 1: cn=espTransport3DES, o=ibm, c=us} \]
\[ \text{ahprotocoltransformreference: 1: cn=ahTransportSHA, o=ibm, c=us} \]
\[ \text{ahprotocoltransformreference: 2: cn=ahTransportMD5, o=ibm, c=us} \]

####################################################
##### GENERIC IPSEC ACTIONS ######
####################################################

#1 IPSec action to drop packets (filter rule)

\[ \text{dn: cn=ipsecDrop, o=ibm, c=us} \]
\[ \text{objectclass: IPSecSecurityAction} \]
\[ \text{cn: ipsecDrop} \]
\[ \text{securityaction: block} \]

#2 IPSec action to pass packets in clear (filter rule)

\[ \text{dn: cn=ipsecPassClear, o=ibm, c=us} \]
\[ \text{objectclass: IPSecSecurityAction} \]
\[ \text{cn: ipsecPassClear} \]
\[ \text{securityaction: permit} \]
### Example 1###
### Branch office policy for securing traffic from 11.0.0.0 to 12.0.0.0 ###
### This policy consists of the information needed to setup the ###
### security association for the Security Gateway (SG1 - public IP ###
### Address = 1.1.1.1) protecting the 11.0.0.0 network and the###
### information needed for the Security Gateway (SG2 - public IP ###
### Address = 1.1.1.2) protecting the 12.0.0.0 network. ###

# Profile for SG1
dn: cn=11to12, o=ibm, c=us
objectclass: trafficprofile
  cn: 11to12
  sourceaddressrange: 1:11.0.0.0-255.255.255.0
  destinationaddressrange: 1:12.0.0.0-255.255.255.0

#IPSEC Action for SG1
dn: cn=secure11to12, o=ibm, c=us
objectclass: IPSecSecurityAction
  cn: secure11to12
  securityaction: permit
  ipsectunnelstart: 1.1.1.1
  ipsectunnelend: 1.1.1.2
  ipsecproposalreference: 1: cn=strongP2EspProp, o=ibm, c=us
  ipsecproposalreference: 2: cn=strongP2EspAhProp, o=ibm, c=us
  ipsecproposalreference: 3: cn=veryStrongP2EspProp, o=ibm, c=us
  ipsecproposalreference: 4: cn=veryStrongP2EspAhProp, o=ibm, c=us

#Policy for SG1
dn: cn=policySecure11to12, o=ibm, c=us
objectclass: policyrule
  cn: policySecure11to12
  rulepriority: 20
  policiescope: isakmp
  policiescope: ipsec
  trafficprofilereference: cn=11to12, o=ibm, c=us
  policyvalidityperiodreference: cn=allTheTime, o=ibm, c=us
  ipsecsecurityactionreference: cn=secure11to12, o=ibm, c=us
  ipsecisakmpactionreference: cn=generalPhase1Action, o=ibm, c=us

#Profile for SG2
dn: cn=12to11, o=ibm, c=us
objectclass: trafficprofile
  cn: 12to11
  sourceaddressrange: 1:12.0.0.0-255.255.255.0
Standards Documents and Sample Files

destinationaddressrange: 1:11.0.0.0-255.255.255.0

#IPSEC Action for SG2
dn: cn=secure12to11, o=ibm, c=us
objectclass: IPSecSecurityAction
cn: secure12to11
securityaction: permit
ipsectunnelstart: 1.1.1.2
ipsectunnelend: 1.1.1.1
ipsecproposalreference: 1: cn=strongP2EspProp, o=ibm, c=us
ipsecproposalreference: 2: cn=strongP2EspAhProp, o=ibm, c=us
ipsecproposalreference: 3: cn=veryStrongP2EspProp, o=ibm, c=us
ipsecproposalreference: 4: cn=veryStrongP2EspAhProp, o=ibm, c=us

#Policy for SG2
dn: cn=policySecure12to11, o=ibm, c=us
objectclass: policyrule
cn: policySecure12to11
rulepriority: 20
policyscope: isakmp
policyscope: ipsec
traffiprofilereference: cn=12to11, o=ibm, c=us
policyvalidityperiodreference: cn=allTheTime, o=ibm, c=us
ipsecsecurityactionreference: cn=secure12to11, o=ibm, c=us
ipsecisakmpactionreference: cn=generalPhase1Action, o=ibm, c=us

###########################################################################
### Example 2:###
### Remote Access policy for securing traffic from Any User to 11.0.0.0 ###
### This policy definition is for remote access users connecting to ###
### their local ISP and being assigned a public IP address. This ###
### address is unknown to the corporate security gateway and would be ###
### considered invalid in the trusted network. As a result, the ###
### mode of operation is to run IPSEC in transport mode and run L2TP ###
### on top of IPSEC. L2TP will provide the user authentication and IP ###
### address assignment for the private network. IPSEC will protect and ###
### authentication the L2TP traffic. This policy must be one of the ###
### lowest priority policies since the remote address information is ###
### causing overlaps with the more specific policies. Note, you must ###
### still enable L2TP, configure L2TP for fixed port mode and PPP users ###
### to finish setting up the configuration. ###
###########################################################################

#Profile for SG1 - Any traffic going in/out 1.1.1.1 from/to 11.0.0.0 network
dn: cn=remoteUserto11, o=ibm, c=us
objectclass: trafficprofile
cn: remoteUserto11
sourceaddressrange: 1:11.0.0.0-255.255.255.0
destinationaddressrange: 1:0.0.0.0-0.0.0.0
sourceportrange: 1701:1701
destinationportrange: 1701:1701
protocolnumber: 17
interface: 1:-1.1.1.1

#IPSEC Action for SG1, Transport Mode
# Note: ipsectunnelend value of 0.0.0.0 denotes remote access
# users with an unknown IP Address.
dn: cn=secureRemoteAccessUsersTo11, o=ibm, c=us
objectclass: IPSecSecurityAction
cn: secureRemoteAccessUsersTo11
securityaction: permit
ipsectunnelstart: 1.1.1.1
ipsectunnelend: 0.0.0.0
ipsecproposalreference: 1: cn=strongP2EspPropXport, o=ibm, c=us
ipsecproposalreference: 2: cn=strongP2EspAhPropXport, o=ibm, c=us
ipsecproposalreference: 3: cn=veryStrongP2EspPropXport, o=ibm, c=us
ipsecproposalreference: 4: cn=veryStrongP2EspAhPropXport, o=ibm, c=us

#Policy for SG1
dn: cn=remoteAccesssto11, o=ibm, c=us
objectclass: policyrule
cn: remoteAccesssto11
rulepriority: 10
policyscope: ipsec
policyscope: isakmp
trafficprofilereference: cn=remoteUserto11, o=ibm, c=us
policyvalidityperiodreference: cn=allTheTime, o=ibm, c=us
ipsecsecurityactionreference: cn=secureRemoteAccessUsersTo11, o=ibm, c=us
ipsecisakmpanctionreference: cn=generalPhase1Action, o=ibm, c=us

###########################################################################
### Example 3###
### This policy provides the Bootstrap Policy needed to terminate IPSEC ###
### tunnels being setup by remote security gateways fetching their ###
### policy information from a local ldap server protected by this###
### SG. In this example the LDAP server IP Address is 11.0.0.1 and the ###
### port number that the LDAP Server is listening on is 389.###
###########################################################################
#Profile for SG1 protecting LDAP server, remote address and port information
#is unknown.
dn: cn=gatewaysToLDAPServer, o=ibm, c=us
objectclass: trafficprofile
cn: gatewaysToLDAPServer
sourceaddressrange: 1:11.0.0.1-255.255.255.255
sourceportrange: 389:389
protocolnumber: 6:6
interface: 1:-1.1.1.1

#IPSEC Action for SG1
dn: cn=secureLdapServer, o=ibm, c=us
objectclass: IPSecSecurityAction
cn: secureLdapServer
securityaction: permit
ipsectunnelstart: 1.1.1.1
ipsectunnelend: 0.0.0.0
ipsecproposalreference: 1: cn=strongP2EspProp, o=ibm, c=us
ipsecproposalreference: 2: cn=strongP2EspAhProp, o=ibm, c=us
ipsecproposalreference: 3: cn=veryStrongP2EspProp, o=ibm, c=us
ipsecproposalreference: 4: cn=veryStrongP2EspAhProp, o=ibm, c=us

#Policy for SG1
dn: cn=gatewayToLdapServerPolicy, o=ibm, c=us
objectclass: policyrule
cn: gatewayToLdapServerPolicy
rulepriority: 15
policyscope: ipsec
policyscope: isakmp
### Example 4: Policy to Drop All Other Traffic Going in or Out Public Interface ###

#### This example has the rules for SG1 and SG2 ####

### Profile for SG1 ###

dn: cn=inOutPublicSG1, o=ibm, c=us
objectclass: trafficprofile
cn: inOutPublicOn11
interface: 1:1.1.1.1-
interface: 1:-1.1.1.1

### Policy for SG1 ###

dn: cn=dropPublicTrafficSG1, o=ibm, c=us
objectclass: policyrule
cn: dropPublicTrafficTo11
rulepriority: 5
policyscope: ipsec
trafficprofilereference: cn=inOutPublicSG1, o=ibm, c=us
policyvalidityperiodreference: cn=allTheTime, o=ibm, c=us
ipsecsecurityactionreference: cn=ipsecDrop, o=ibm, c=us

### Profile for SG2 ###

dn: cn=inOutPublicSG2, o=ibm, c=us
objectclass: trafficprofile
cn: inOutPublicOn12
interface: 1:1.1.1.2-
interface: 1:-1.1.1.2

### Policy for SG2 ###

dn: cn=dropPublicTrafficSG2, o=ibm, c=us
objectclass: policyrule
cn: dropPublicTrafficTo12
rulepriority: 5
policyscope: ipsec
trafficprofilereference: cn=inOutPublicSG2, o=ibm, c=us
policyvalidityperiodreference: cn=allTheTime, o=ibm, c=us
ipsecsecurityactionreference: cn=ipsecDrop, o=ibm, c=us

### DEVICEPOLICYRULES LIST for SG1 ###

dn: cn=rulesForSG1, o=ibm, c=us
objectclass: devicepolicyrules
cn: cn=rulesForSG-On-11Net
policyrulereference: cn=policySecure11to12, o=ibm, c=us
policyrulereference: cn=gatewaysToLDAPServerPolicy, o=ibm, c=us
policyrulereference: cn=remoteAccessTo11, o=ibm, c=us
policyrulereference: cn=dropPublicTrafficSG1, o=ibm, c=us

### DEVICEPROFILE for SG1 ###

---

#Standards Documents and Sample Files#
dn: cn=deviceProfileForSG1, o=ibm, c=us
objectclass: deviceprofile
cn: deviceProfileForSG1
devicerulesreference: cn=rulesForSG1, o=ibm, c=us

####################################
## DEVICEPOLICYRULES LIST for SG2 ##
####################################
dn: cn=rulesForSG2, o=ibm, c=us
objectclass: devicepolicyrules
cn: rulesForSG2
policyrulereference: cn=policySecure12to11, o=ibm, c=us
policyrulereference: cn=dropPublicTrafficSG2, o=ibm, c=us

###########################
## DEVICEPROFILE for SG2 ##
###########################
dn: cn=deviceProfileForSG2, o=ibm, c=us
objectclass: deviceprofile
cn: deviceProfileForSG2
devicerulesreference: cn=rulesForSG2, o=ibm, c=us

### A.5.4 Example Schema File ###

# Policy Schema attribute definitions for IBM SecureWays LDAP server
# Add this file to the include definitions in the slapd.conf file
# include /local/ldapServer/slapd-blueridge/config/policySchema_ibm_at.conf
# include /local/ldapServer/slapd-blueridge/config/policySchema_oc.conf
attribute devicerulesreference
cn=devicerulesreference128normal
attribute policyrulereference
cn=policyrulereference128normal
attribute policyscope
cn=policyscope128normal
attribute policyruleenabled
cn=policyruleenabled128normal
attribute rulepriority
cn=rulepriority128normal
attribute trafficprofilereference
cn=traffic profilereference128normal
attribute policyvalidityperiodreference
cn=policyvalidityper128normal
attribute rsvpactionreference
cn=rsvpactionreference128normal
attribute ipsecisakmpactionreference
cn=ipsecisakmpactionreference128normal
attribute ipsecsecurityactionreference
cn=ipsecsecurityactionreference128normal
attribute interface
cn=interface128normal
attribute sourcedestination
cn=sourcedestination128normal
attribute localid
cn=localid128normal
attribute remoteid
cn=remoteid128normal
attribute destinationportrange
cn=destinationportrange128normal
attribute protocolnumber
cn=protocolnumber128normal
attribute receivedtosbytecheck
cn=receivedtosbytecheck128normal
attribute policyvaliditytime
cn=policyvaliditytime128normal
attribute policyvaliditymonthmask
cn=policyvaliditymonthmask128normal
attribute policyvaliditydayofweekmask
cn=policyvaliditydayofweekmask128normal
attribute ipsecprotocolid
cn=ipsecprotocolid128normal
attribute ahintegrityalgorithm
cn=ahintegrityalgorithm128normal
attribute encapsulationmode
cn=encapsulationmode128normal
attribute espintegrityalgorithm
cn=espintegrityalgorithm128normal
attribute espcipheralgorithm
cn=espcipheralgorithm128normal
attribute defaultdiffhellmangroupid
cn=defaultdhgroupid128normal
Appendix B. Special Notices

This publication is intended to help people who want to understand the current concepts, implementations and trends behind “policy-based networking” in which a network is controlled through the application of network-wide policies rather than through the configuration of individual building blocks. The information in this publication is not intended as the specification of any programming interfaces that are provided by IBM routers and IBM policy servers. See the PUBLICATIONS section of the IBM Programming Announcements for IBM routers (2210, 2212, 2216) for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (“vendor”) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer's ability to evaluate and integrate them into the customer's operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other operating
environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

<table>
<thead>
<tr>
<th>Term</th>
<th>Trademark Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS/400</td>
<td>IBM</td>
</tr>
<tr>
<td>IBM</td>
<td></td>
</tr>
<tr>
<td>Nways</td>
<td>DB2</td>
</tr>
<tr>
<td>P2P</td>
<td>Netfinity</td>
</tr>
<tr>
<td>S/390</td>
<td>OS/390</td>
</tr>
<tr>
<td>System/390</td>
<td>RS/6000</td>
</tr>
<tr>
<td>SecureWay</td>
<td>VTAM</td>
</tr>
</tbody>
</table>

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc. in the United States and/or other countries.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and/or other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States and/or other countries.

PC Direct is a trademark of Ziff Communications Company in the United States and/or other countries and is used by IBM Corporation under license.

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States and/or other countries.

UNIX is a registered trademark in the United States and/or other countries licensed exclusively through The Open Group.

SET and the SET logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.
Appendix C. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

C.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get IBM Redbooks” on page 171.

- A Comprehensive Guide to Virtual Private Networks, Volume I: IBM Firewall, Server and Client Solutions, SG24-5201
- Application Driven Networking: Class of Service in IP, Ethernet and ATM Networks, SG24-5384

C.2 IBM Redbooks Collections

Redbooks are also available on the following CD-ROMs. Click the CD-ROMs button at http://www.redbooks.ibm.com/ for information about all the CD-ROMs offered, updates and formats.

<table>
<thead>
<tr>
<th>CD-ROM Title</th>
<th>Collection Kit Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>System/390 Redbooks Collection</td>
<td>SK2T-2177</td>
</tr>
<tr>
<td>Networking and Systems Management Redbooks Collection</td>
<td>SK2T-6022</td>
</tr>
<tr>
<td>Transaction Processing and Data Management Redbooks Collection</td>
<td>SK2T-8038</td>
</tr>
<tr>
<td>Lotus Redbooks Collection</td>
<td>SK2T-8039</td>
</tr>
<tr>
<td>Tivoli Redbooks Collection</td>
<td>SK2T-8044</td>
</tr>
<tr>
<td>AS/400 Redbooks Collection</td>
<td>SK2T-2849</td>
</tr>
<tr>
<td>Netfinity Hardware and Software Redbooks Collection</td>
<td>SK2T-8046</td>
</tr>
<tr>
<td>RS/6000 Redbooks Collection (BkMgr Format)</td>
<td>SK2T-8040</td>
</tr>
<tr>
<td>RS/6000 Redbooks Collection (PDF Format)</td>
<td>SK2T-8043</td>
</tr>
<tr>
<td>Application Development Redbooks Collection</td>
<td>SK2T-8037</td>
</tr>
<tr>
<td>IBM Enterprise Storage and Systems Management Solutions</td>
<td>SK3T-3694</td>
</tr>
</tbody>
</table>

C.3 Other Publications

These publications are also relevant as further information sources:

- Nways Multiprotocol Access Services, Using and Configuring Features Version 3.3, SC30-3993
- Access Integration Services, Using and Configuring Features Version 3.3, SC30-3989
- Nways Multiprotocol Routing Services: Using and Configuring Features Version 3.4, SC30-3992
C.4 Referenced Web Sites

- http://www.openldap.org/openldap
- http://www.ietf.org
- http://www.murchiso.com/den
- http://www.dmtf.org
- http://www.redbooks.ibm.com
- http://w3.itso.ibm.com
- http://w3.ibm.com
- http://www.dmtf.org/memb/list?info=product
- http://www.netapps.org
How to Get IBM Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- Redbooks Web Site  http://www.redbooks.ibm.com/
  Search for, view, download, or order hardcopy/CD-ROM redbooks from the redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this redbooks site.
  Redpieces are redbooks in progress; not all redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- E-mail Orders
  Send orders by e-mail including information from the redbooks fax order form to:

  | In United States | e-mail address | usib6fpl@ibmmail.com |
  | Outside North America | Contact information is in the “How to Order” section at this site: |

- Telephone Orders
  United States (toll free)  1-800-879-2755
  Canada (toll free)  1-800-IBM-4YOU
  Outside North America  Country coordinator phone number is in the “How to Order” section at this site:

- Fax Orders
  United States (toll free)  1-800-445-9269
  Canada  1-403-267-4455
  Outside North America  Fax phone number is in the “How to Order” section at this site:

This information was current at the time of publication, but is continually subject to change. The latest information may be found at the redbooks Web site.

---

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and redbooks by accessing the IBM Intranet Web site at http://w3.itso.ibm.com/ and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may access MyNews at http://w3.ibm.com/ for redbook, residency, and workshop announcements.
<table>
<thead>
<tr>
<th>Title</th>
<th>Order Number</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

First name

Last name

Company

Address

City

Postal code

Country

Telephone number

Telefax number

VAT number

Invoice to customer number

Credit card number

Credit card expiration date

Card issued to

Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.
Glossary

Administrative Domain
Any separately managed network, whether departmental, divisional, or company-wide. Although there are several different kinds of domains, such as NT domains and IP domains, for quality of service enforcement purposes, a network domain refers to any domain that shares a common QoS policy. Usually managed by a single corporate entity.

Admission Control
Any technique for controlling the admission of network traffic from outside a given administrative domain.

Asynchronous Transfer Mode (ATM)
A data framing and transmission architecture designed to carry voice, video and data, which has built-in QoS capabilities. Operates at Layer 2 of the OSI model. Although relatively few native ATM applications exist, TCP/IP traffic can be sent over an underlying ATM layer. This practice is now prevalent for the implementation of Wide-Area Networks and the Internet backbones operated by major telecommunications companies. ATM-based technology is also finding its way into systems for improving TCP/IP connectivity on corporate networks. Although generally considered too costly to be an end-to-end (desktop-to-desktop) QoS solution, ATM services are likely to co-exist with QoS-enabled IP networks for many years to come.

Backbone
The superstructure of the Internet where the national Internet Service Providers link directly to each other. Sometimes used to refer to the trunkline of any network. Usually the backbone speed is significantly higher than that of the networks that it connects. A connecting network highway for major networks including large enterprise networks and the Internet.

Bandwidth
A measure of data transmission capacity, usually expressed in kilobits per second (kbps) or megabits per second (Mbps). Bandwidth indicates the theoretical maximum capacity of a connection, but as the theoretical bandwidth is approached, negative factors such as transmission delay can cause deterioration in quality. If you increase bandwidth, you can transfer more data. Network bandwidth can be visualized as a pipe that transfers data. The larger the pipe, the more data can be sent through it.

Bandwidth Manager
A rudimentary traffic management solution deployed at congestion points that limits access to network resources (it might prevent Pointcast-like “push” traffic from entering the network). Because a bandwidth manager solution often requires locating a proprietary hardware device directly on the network, it may be an additional point-of-failure. Not an end-to-end solution, a bandwidth manager cannot coordinate multiple traffic flows or resolve conflicting QoS requests made by multiple clients. See also Point QoS.

Best-effort Service
The default behavior of TCP/IP networks in the absence of QoS measures. TCP/IP nodes will make their best effort to deliver a transmission but will drop packets indiscriminately in the event of congestion.

Common Information Mode (CIM)
The information model used by the policy framework to specify policies. CIM is defined by the Distributed Management Task force.

Class-based Queuing (CBQ)
A methodology for classifying packets and queueing them according to criteria defined by an administrator. The queuing system is designed to prevent any one application from monopolizing the system. Also known as Custom Queueing. See also Weighted Fair Queueing.

Class of Service (CoS)
A category based on type of user, type of application, or some other criteria that QoS systems can use to provide differentiated classes of service.

Congestion Avoidance
An attempt to head off congestion before it can occur. Random Early Detection (RED) is an example. See also Congestion Management.

Congestion Management
A mechanism that imposes order when traffic exceeds network capacity. It determines whether some packets must be discarded, and, if so, it preserves the more important packets. Queueing, scheduling, and traffic shaping are among the most popular techniques. See Also Congestion Avoidance.

Controlled-load Service
A high level but not guaranteed service. In the proposed IETF Integrated Service model, this level of service is designed for multimedia applications where time delay is not critical but quality of the delivery is important. This service is appropriate for applications such as one-way voice or video, but not for real-time applications, such as teleconferencing. See also Guaranteed Service.

Converged Network
A network that combines varied traffic types such as data, voice, and multimedia. Most analysts expect the converged network of the future to be based on
Internet protocols. This trend is evident in corporate networks, which are starting to combine videoconferencing on their traditional data networks, as well as in the merging of the telephone, cable television, and Internet service industries.

**Common Open Policy Service (COPS)**

An IETF proposed standard for implementing QoS policies as an end-to-end service. Common Open Policy Service allows a policy server to control the devices on the network, such as routers and switches, so that a cohesive policy based on business priorities can be achieved. Common Open Policy Service is a companion protocol to Resource ReSerVation Protocol. The QoS policy exchange mechanism.

**Custom Queuing (CQ)**

See Class-based Queuing

**Differentiated Services (DiffServ)**

An emerging Quality of Service standard. A superset of IP Precedence/CBQ. By utilizes an entire TOS byte in the IP header, it offers up to 256 levels of priority. This protocol is expected to be used predominantly in the IP backbone environments.

**Edge Device**

A device such as a router or a gateway that is deployed at the border of an administrative domain. Such devices control traffic through one point only. Contrast with End-to-End QoS.

**End-to-end QoS**

A system that enforces consistent Quality of Service policies throughout a network. Has the ability to provide both class of service and reserved bandwidth for different types of network traffic. End-to-end QoS coordinates and enforces predefined traffic management policies across multiple network devices. See also ThruQoS

**Guaranteed Service**

A service level that attempts to guarantee a minimal delay for traffic delivery. In the proposed IETF Integrated Service model, guaranteed service is intended for real-time applications, such as teleconferencing. The guarantee is not absolute, but such traffic is a level above controlled-load service. See also Controlled-load Service

**H.323**

A standard approved by the International Telecommunication Union (ITU) that defines how audiovisual conferencing data is transmitted across networks. In theory, H.323 should enable users to participate in the same conference even though they are using different videoconferencing applications. Although most videoconferencing vendors have announced that their products will conform to H.323, it's too early to say whether such adherence will actually result in interoperability.

This recommendation specifies a packet-oriented multiplexing protocol for low bit rate multimedia communication. This protocol can be used between two low bit rate multimedia terminals, or between a low bit rate multimedia terminal and a multipoint control unit or an interworking adapter. The protocol allows the transfer of any combination of digital voice/audio, digital video/image and data information over a single communication link. This protocol provides low delay and low overhead by using segmentation and reassembly and by combining information from different logical channels in a single packet. The control procedures necessary to implement this multiplexing protocol are specified in Recommendation H.245.

**IEEE 802.1p Standard**

An IEEE standard that governs the prioritization of packets in the Ethernet and Token Ring networks. Offers eight discrete priority levels, ranging from the default of best effort, through excellent effort (a business-critical application, but tolerant of some delay), interactive multimedia (sensitive to delay or jitter), and reserved (highest priority). This is a Layer 2 (Data Link) priority setting, as opposed to the ToS and IP Precedence/CBQ bits, which are Layer 3 (Network Level) settings carried in the IPv4 header. Because it must be implemented in the hardware of network devices, existing switches and routers need to be replaced with ones supporting this technology. See also IP Precedence/CBQ, Type of Service.

**Integrated Services**

The practice of supporting audio, video, and real-time data within a single network infrastructure. The IETF’s Integrated Services (IntServ) working group has proposed a variety of QoS standards to support such systems. See also Controlled-load Service, Guaranteed Service.

**Internet Engineering Task Force (IETF)**

The international organization that defines Internet protocols and standards, including refinements to improve QoS.

**Internet Service Provider (ISP)**

A telecommunications company that provides dialup or leased-line connections to the Internet. Local and regional ISPs forward traffic to backbone ISPs, the large carrier-class ISPs that own the national and international Internet backbone infrastructure. Many ISPs offer additional services, such as Web site hosting, Voice over IP (VoIP), or virtual private networks (VPN).

**IPv4 (Internet Protocol)**

The most widely deployed version of the Internet Protocol, IPv4 provides some basic traffic classification mechanisms with its IP Precedence/CBQ and Type of Service header fields. However, network
hardware and software traditionally have not been configured to use them.

**IPv6 (Internet Protocol)**

An update to the Internet Protocol that is in the early phases of adoption. Most of the refinements concentrate on basics such as expanding the IP address numbering scheme to accommodate the growth of the Internet. However, IPv6 does include a Class header field that is explicitly intended to designate a Class of Service (an extension of IPv4’s IP Precedence/CBQ field).

**IP Multicast**

A technique for making a single transmission fan out to multiple recipients. Instead of sending a copy of each packet to multiple destinations, the sender transmits one packet to a multicast group address. Specialized multicast systems then retransmit each packet to the individuals in the group. IP Multicast makes more efficient use of bandwidth by minimizing duplication, particularly in multipoint applications such as broadcast video traffic (for example, distance learning).

**IP Precedence/CBQ**

One of the key QoS transmission standards. A 3-bit value in the IP packet header meant to designate the relative priority of a packet. Offers eight levels of priority from 0 to 7. For example, a brokerage firm might assign a higher IP Precedence/CBQ value to real-time stock trades than to e-mail to ensure that the trading gets expedited delivery. Same as TOS bits – see IETF RFC 1349

**Jitter**

A type of distortion that is caused by packets arriving at irregular intervals. This distortion is particularly damaging to multimedia traffic. For example, the playback of audio or video data may have a jittery or shaky quality.

**Leaky Bucket**

A traffic-shaping mechanism in which only a fixed amount of traffic is admitted to the network. Excess traffic is held in a queue until either it can be accommodated or must be discarded. The analogy is with water flowing into a leaky bucket. If the water continues to flow in faster than it can leak out the bottom, the bucket eventually overflows. See also Token Bucket

**Lightweight Directory Access Protocol (LDAP)**

A standard for repositories that store user profiles and other information about the network. LDAP repositories make it easier for users to find the e-mail addresses and public key encryption codes of other users. Currently, there is no standard governing storing the QoS policy information in the LDAP format. However, such support is expected in the future. LDAP repositories are not capable of issuing dynamic policy authorizations and do not have policy enforcement capabilities.

**Over-provisioned Bandwidth**

An expensive approach to addressing current limitations of “best effort” networks by provisioning more bandwidth than expected network peak requirements. over-provisioning increases the probability, but does not guarantee the quality, of transmission of time-sensitive and bandwidth-intensive applications.

**Peering Agreement**

A reciprocal agreement that lets Internet Service Providers share backbone links, so traffic can reach destinations beyond the ISP’s management domain.

**PointQoS**

An interim solution between “best-effort” and ThruQoS-enabled networks. PointQoS offers proprietary hardware-based solutions that address traffic flow congestion on a single point in the network, usually at the network edge. Based on the negative principle of blocking and filtering specific data types or applications (e.g., PointCast), PointQoS introduces an extra point of failure, does not guarantee the quality of the data transmission across the entire data path, and does not address the needs of time-critical applications. See also ThruQoS.

**Policy Decision Point**

The instance in a policy system which check all policy rules. Queried by a Policy Enforcement Point and based on the policy information gathered from a LDAP directory.

**Policy Enforcement Point**

The point a policy system where the actual policy actions are performed or enforced. Today typically DiffServ, IntServ or VPN functions.

**Policy Information Base**

Like a MIB the internal representation of policies inside a Policy Enforcement Point.

**Policy Server**

It is a server that authorizes QoS requests received from Common Open Policy Service-enabled routers or Policy Gateways and coordinates bandwidth usage on multiple network devices to ensure consistent end-to-end service throughout the data-path. The Policy Server ensures that packets receive the appropriate Quality of Service, based on a set of policies defined by the network administrator.

**Quality of Service (QoS)**

Quality of Service is the network’s ability to match user and application requirements to network capabilities. It is based on a set of intelligent network protocols and services used to efficiently control the movement of information through local or wide area networks. QoS
software sorts and classifies IP packet requests into different traffic classes and allocates the proper resources to direct traffic based on various criteria including application type, user or application ID, source or destination IP address, time of day, and other user-specified variables. See also ThruQoS.

**QoS Signaling**

Any system for transmitting QoS requests and parameters between devices or applications. Resource ReSerVation Protocol is an example of a QoS signaling system.

**Queuing**

A method for metering the flow of traffic by placing packets in holding queues, and retransmitting them according to a sorting algorithm, typically a simple first-in-first-out (FIFO) formula. Queues of different sizes can be used to assign levels of importance according to Class of Service designations. Queues that overflow typically discard packets to reduce network congestion.

**Random Early Detection (RED)**

A basic congestion avoidance technique built on the base-level TCP behavior of automatically slowing transmissions when packet loss is detected. RED tries to anticipate congestion by monitoring a queue. When the specified threshold is reached, it randomly discards packets. This is an implicit signal that the originating applications should slow their transmissions before congestion becomes severe. See also Weighted Random Early Detection.

**Resource ReSerVation Protocol (RSVP)**

One of the key IETF protocols that communicates the QoS requirements for a given application to a device in the path of the transmission. A reservation for the required bandwidth is allowed or denied depending on the current network conditions. By itself, Resource ReSerVation Protocol provides a way to reserve capacity one device at a time. In a centrally managed QoS system, Resource ReSerVation Protocol can be implemented according to policies that apply across the network. Resource ReSerVation Protocol is expected to be utilized predominantly in the campus-level networks. See also Common Open Policy Service.

**SLA (Service Level Agreement)**

An agreement between a business or residential customer and a network service provider or between two independent service providers that specifies the minimum service levels that must be provided. Cooperative service level agreements can be written to ensure consistent handling of differentiated traffic flows across multiple network domains, allowing end-to-end quality of service for Internet traffic.

**TCP/IP (Transmission Control Protocol/Internet Protocol)**

The standard protocol suite for the Internet and an increasing number of corporate networks. IP is the base network protocol (on which multiple transport protocols have been implemented) and TCP controls the behavior of packet transmission.

**TCP Rate Control**

A technology implemented at network end points that attempts to regulate the introduction of traffic into the network.

**ThruQoS**

A comprehensive QoS solution that transforms today's unpredictable “best-effort” IP networks into smart, priority-driven environments that reliably deliver time-critical and bandwidth-intensive information. ThruQoS offers complete end-to-end Quality of Service. All traffic flows are controlled by predefined policies and receive guaranteed bandwidth across the entire data path. Based on widely accepted IETF standards like IP Precedence/CBQ, Diff-Serv, Common Open Policy Service and Resource ReSerVation Protocol, ThruQoS is a non-intrusive software-only solution that integrates with existing network hardware and doesn’t add another point of failure to your network. See also PointQoS.

**Token Bucket**

A traffic-shaping mechanism in which a predetermined amount of tokens in a bucket represent the capacity allowed to each class of traffic. Packets are forwarded until they exhaust their supply of tokens. When the token supply is exhausted, packets may be discarded or delayed until the bucket is replenished. In some systems, a customer's token supply might correspond to a service fee. See also Leaky Bucket.

**Traffic Shaping**

A group of techniques that attempt to regulate or meter the flow of packets through the network. See also Leaky Bucket, Token Bucket

**Type of Service (ToS)**

A 4-bit value in an IP packet's message header that identifies the type of application generating a given traffic flow. Like the IP Precedence/CBQ field, the ToS value can be used for traffic classification. Functionally, TOS and IP Precedence/CBQ are identical.

**Virtual Private Network (VPN)**

A way of duplicating the security and reliability of a dedicated network connection over a less expensive Internet link. This requires that each network participating in the VPN deploy a compatible firewall for encrypting messages and for permitting authorized access from remote locations. A combination of VPN and ThruQoS offers guaranteed quality and reliability of traffic flow and allows for the migration of customers from dedicated lines to shared IP infrastructure.
Voice over IP (VoIP)
An emerging technology for carrying phone conversations over the Internet and intranets. VoIP is sometimes provided as part of a package with other collaborative applications, such as text-based chat. ThruQoS is important for VoIP because the audio signal must come through in a steady stream, just as it would over a traditional telephone switch.

Weighted Fair Queuing (WFQ)
A methodology for segmenting traffic into multiple queues, giving greater weight to certain traffic types by assigning larger queues. Like class-based queuing (CBQ), WFQ is designed to prevent any one traffic type from entirely eclipsing another. By default, WFQ favors lower-volume traffic flows over higher-volume ones (for example, a routine e-mail over a large FTP download). See also Class-based Queuing.

Weighted Random Early Detection (WRED)
A congestion avoidance technique that takes advantage of TCP's interpretation of packet loss as a sign to slow transmissions. WRED monitors a queue until it fills to a specified threshold. It then begins discarding packets, starting with those that have the lowest IP Precedence/CBQ. See also Random Early Detection.
# List of Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3DES</td>
<td>Triple DES</td>
</tr>
<tr>
<td>AAA</td>
<td>Administration, Authorization, and Authentication</td>
</tr>
<tr>
<td>ACL</td>
<td>Access Control List</td>
</tr>
<tr>
<td>ACS</td>
<td>Admission Control Service</td>
</tr>
<tr>
<td>ADN</td>
<td>Application Driven Networking</td>
</tr>
<tr>
<td>AH</td>
<td>Authentication Header</td>
</tr>
<tr>
<td>AIS</td>
<td>Access Integration Services (software for IBM 2212 router)</td>
</tr>
<tr>
<td>ASN.1</td>
<td>Abstract Syntax Notation One</td>
</tr>
<tr>
<td>BER</td>
<td>Basic Encoding Rules (for ASN.1)</td>
</tr>
<tr>
<td>CDMF</td>
<td>Commercial Data Masking Facility</td>
</tr>
<tr>
<td>CIM</td>
<td>Common Information Model</td>
</tr>
<tr>
<td>CLI</td>
<td>Command Line Interface</td>
</tr>
<tr>
<td>CNF</td>
<td>Conjunctive Normal Form</td>
</tr>
<tr>
<td>COPS</td>
<td>Common Open Policy Server</td>
</tr>
<tr>
<td>CPE</td>
<td>Common Policy Engine</td>
</tr>
<tr>
<td>CS/390</td>
<td>Communications Server/390</td>
</tr>
<tr>
<td>DAP</td>
<td>Directory Access Protocol</td>
</tr>
<tr>
<td>DEC</td>
<td>Decision (response to COPS REQUEST)</td>
</tr>
<tr>
<td>DEN</td>
<td>Directory Enabled Networks</td>
</tr>
<tr>
<td>DES</td>
<td>Data Encryption Standard</td>
</tr>
<tr>
<td>DH</td>
<td>Diffie Hellman (method for key agreement)</td>
</tr>
<tr>
<td>DiffServ</td>
<td>Differentiated Services</td>
</tr>
<tr>
<td>DIT</td>
<td>Directory Information Tree</td>
</tr>
<tr>
<td>DMA</td>
<td>Data Management Action</td>
</tr>
<tr>
<td>DMP</td>
<td>Data Management Proposal</td>
</tr>
<tr>
<td>DMTF</td>
<td>Desktop Management Task Force</td>
</tr>
<tr>
<td>DN</td>
<td>Distinguished Name</td>
</tr>
<tr>
<td>DNF</td>
<td>Disjunctive Normal Form</td>
</tr>
<tr>
<td>DSCP</td>
<td>Differentiated Services Code Point</td>
</tr>
<tr>
<td>ESP</td>
<td>Encapsulating Security Payload</td>
</tr>
<tr>
<td>GEM</td>
<td>Global Enterprise Manager</td>
</tr>
<tr>
<td>HMAC</td>
<td>Hashed Message Authentication Code</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IBM</td>
<td>International Business Machines Corporation</td>
</tr>
<tr>
<td>IKE</td>
<td>Internet Key Encryption</td>
</tr>
<tr>
<td>IntServ</td>
<td>Integrated Services</td>
</tr>
<tr>
<td>IPSec</td>
<td>Secure Internet Protocol</td>
</tr>
<tr>
<td>ISAKMP</td>
<td>Internet Security Association and Key Management Protocol</td>
</tr>
<tr>
<td>ISP</td>
<td>Internet Service Provider</td>
</tr>
<tr>
<td>ITSO</td>
<td>International Technical Support Organization</td>
</tr>
<tr>
<td>KMA</td>
<td>Key Management Action</td>
</tr>
<tr>
<td>KMP</td>
<td>Key Management Proposal</td>
</tr>
<tr>
<td>LDAP</td>
<td>Lightweight Directory Access Protocol</td>
</tr>
<tr>
<td>LDIF</td>
<td>LDAP Data Interchange Format</td>
</tr>
<tr>
<td>LIPS</td>
<td>Lightweight Internet Person Schema</td>
</tr>
<tr>
<td>MAS</td>
<td>Multiprotocol Access Services (software for IBM 2216 router)</td>
</tr>
<tr>
<td>MD5</td>
<td>Message Digest 5 (a one-way hash algorithm)</td>
</tr>
<tr>
<td>MIB</td>
<td>Management Information Base</td>
</tr>
<tr>
<td>MOF</td>
<td>Managed Object File</td>
</tr>
<tr>
<td>MRS</td>
<td>Multiprotocol Routing Services (software for IBM 2210 router)</td>
</tr>
<tr>
<td>NAT</td>
<td>Network Address Translation</td>
</tr>
<tr>
<td>NHD</td>
<td>Networking Hardware Division</td>
</tr>
<tr>
<td>OID</td>
<td>Object Identifier</td>
</tr>
<tr>
<td>OS/390</td>
<td>Open Systems Interconnection</td>
</tr>
<tr>
<td>OSI</td>
<td>Open Systems Interconnection</td>
</tr>
<tr>
<td>PAC</td>
<td>Policy Administration and Configuration</td>
</tr>
<tr>
<td>PDP</td>
<td>Policy Decision Point</td>
</tr>
<tr>
<td>PEN</td>
<td>Policy-Enabled Network</td>
</tr>
<tr>
<td>PEP</td>
<td>Policy Enforcement Point</td>
</tr>
<tr>
<td>PIB</td>
<td>Policy Information Base</td>
</tr>
</tbody>
</table>
PRC  Policy Rule Class
PRI  Policy Rule Instance
PRID  Policy Rule Identifier
QoS  Quality of Service
RAP  (RSVP) Resource Allocation Protocol
RDN  Relative Distinguished Name
RFC  Request For Comments
RPT  Report (a specific type of COPS operation)
RSA  An encryption algorithm named after Ron Rivest, Adi Shamir and Leonard Adleman
RSVP  Resource Reservation Protocol
SA  Security Association
SAM  Service Agreement Manager
SHA  Secure Hash Algorithm
SLA  Service Level Agreement
SLO  Service Level Objective
SMI  Structure of Management Information
SNPP  Simple Network Policy Protocol
TCP  Transmission Control Protocol
TOS  Type Of Service
UDP  User Datagram Protocol
UML  Unified Modelling Language
UTC  Universal Time, Coordinated (a "politically correct" alternative for Greenwich Mean Time)
VPN  Virtual Private Network
## Index

### Numerics

<table>
<thead>
<tr>
<th>Numeral</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>221x 49</td>
<td></td>
</tr>
<tr>
<td>3DES 105</td>
<td></td>
</tr>
</tbody>
</table>

### A

<table>
<thead>
<tr>
<th>Term</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAA 44, 112</td>
<td></td>
</tr>
<tr>
<td>ACS 86</td>
<td></td>
</tr>
<tr>
<td>administrative domain 20</td>
<td></td>
</tr>
<tr>
<td>admission control policies 21</td>
<td></td>
</tr>
<tr>
<td>Admission Control Service 86</td>
<td></td>
</tr>
<tr>
<td>AH 79</td>
<td></td>
</tr>
<tr>
<td>AIS 33</td>
<td></td>
</tr>
<tr>
<td>application response times 4</td>
<td></td>
</tr>
<tr>
<td>ASN.1 70, 119</td>
<td></td>
</tr>
<tr>
<td>ASN.1 compiler 10</td>
<td></td>
</tr>
<tr>
<td>asynchronous data 27</td>
<td></td>
</tr>
</tbody>
</table>

### B

<table>
<thead>
<tr>
<th>Term</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Basic Encoding Rules for ASN.1 19</td>
<td></td>
</tr>
<tr>
<td>BER 19, 25</td>
<td></td>
</tr>
<tr>
<td>bilateral agreements 21, 22</td>
<td></td>
</tr>
<tr>
<td>binary search tree 32</td>
<td></td>
</tr>
<tr>
<td>business objectives 4</td>
<td></td>
</tr>
</tbody>
</table>

### C

<table>
<thead>
<tr>
<th>Term</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>cache 29</td>
<td></td>
</tr>
<tr>
<td>calling card 22</td>
<td></td>
</tr>
<tr>
<td>CCITT 50</td>
<td></td>
</tr>
<tr>
<td>CDMF 83</td>
<td></td>
</tr>
<tr>
<td>chief information officer 4</td>
<td></td>
</tr>
<tr>
<td>CIM 9, 10, 53, 54, 55, 56, 57, 58, 61, 62, 63</td>
<td></td>
</tr>
<tr>
<td>aggregation 61</td>
<td></td>
</tr>
<tr>
<td>association 61, 62</td>
<td></td>
</tr>
<tr>
<td>CIM Network Schema 9</td>
<td></td>
</tr>
<tr>
<td>CIM Schema 10</td>
<td></td>
</tr>
<tr>
<td>CIM Specification 10</td>
<td></td>
</tr>
<tr>
<td>CiscoAssure 2</td>
<td></td>
</tr>
<tr>
<td>CNF 57, 65</td>
<td></td>
</tr>
<tr>
<td>co-located PDP/PEP 49</td>
<td></td>
</tr>
<tr>
<td>Common Information Model 9, 10, 55</td>
<td></td>
</tr>
<tr>
<td>Common Open Policy Service 11, 18, 23, 43, 44</td>
<td></td>
</tr>
<tr>
<td>Common Policy Engine 18, 23, 29</td>
<td></td>
</tr>
<tr>
<td>complex policy rule 65</td>
<td></td>
</tr>
<tr>
<td>compromise 4</td>
<td></td>
</tr>
<tr>
<td>condition 31</td>
<td></td>
</tr>
<tr>
<td>configuration request 27</td>
<td></td>
</tr>
<tr>
<td>Conjunctive Normal Form 57</td>
<td></td>
</tr>
<tr>
<td>COPS 11, 18, 23, 24, 25, 29, 43, 44, 86</td>
<td></td>
</tr>
<tr>
<td>with RSVP 28</td>
<td></td>
</tr>
<tr>
<td>corporate policy 4</td>
<td></td>
</tr>
<tr>
<td>CPE 29, 32</td>
<td></td>
</tr>
<tr>
<td>credentials 21</td>
<td></td>
</tr>
<tr>
<td>CS/390 86</td>
<td></td>
</tr>
<tr>
<td>admission control 89</td>
<td></td>
</tr>
<tr>
<td>PAGENT 86</td>
<td></td>
</tr>
<tr>
<td>Policy Agent 49</td>
<td></td>
</tr>
<tr>
<td>Policy API 90</td>
<td></td>
</tr>
</tbody>
</table>

### D

<table>
<thead>
<tr>
<th>Term</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAP 51</td>
<td></td>
</tr>
<tr>
<td>DB2 54</td>
<td></td>
</tr>
<tr>
<td>DEC 27</td>
<td></td>
</tr>
<tr>
<td>DEN 8, 9, 10, 52, 55, 61</td>
<td></td>
</tr>
<tr>
<td>DEN Ad Hoc Working Group 8, 9</td>
<td></td>
</tr>
<tr>
<td>DEN Extension Schema 9</td>
<td></td>
</tr>
<tr>
<td>DES 68, 83</td>
<td></td>
</tr>
<tr>
<td>Desktop Management Taskforce 8, 9</td>
<td></td>
</tr>
<tr>
<td>device adapter 47</td>
<td></td>
</tr>
<tr>
<td>device-by-device rule establishment 4</td>
<td></td>
</tr>
<tr>
<td>DH 39</td>
<td></td>
</tr>
<tr>
<td>Differentiated Services 6, 21</td>
<td></td>
</tr>
<tr>
<td>DiffServ 11, 18, 30, 35, 36, 38, 57, 79, 82, 86, 87, 94, 112</td>
<td></td>
</tr>
<tr>
<td>action 36, 68, 77, 79, 82</td>
<td></td>
</tr>
<tr>
<td>adjusted by ADN 6</td>
<td></td>
</tr>
<tr>
<td>code points 86</td>
<td></td>
</tr>
<tr>
<td>configuration details 38</td>
<td></td>
</tr>
<tr>
<td>CS/390 91</td>
<td></td>
</tr>
<tr>
<td>IETF 11</td>
<td></td>
</tr>
<tr>
<td>in detail 81</td>
<td></td>
</tr>
<tr>
<td>Internet technologies 5</td>
<td></td>
</tr>
<tr>
<td>module 29</td>
<td></td>
</tr>
<tr>
<td>Olympic service model 112</td>
<td></td>
</tr>
<tr>
<td>policy and rule 72</td>
<td></td>
</tr>
<tr>
<td>policy segment 80</td>
<td></td>
</tr>
<tr>
<td>SLA example 13</td>
<td></td>
</tr>
<tr>
<td>triggering mechanism 35</td>
<td></td>
</tr>
<tr>
<td>directory 13</td>
<td></td>
</tr>
<tr>
<td>Directory Enabled Network 8, 52</td>
<td></td>
</tr>
<tr>
<td>Directory Information Tree 54, 65</td>
<td></td>
</tr>
<tr>
<td>Disjunctive Normal Form 57</td>
<td></td>
</tr>
<tr>
<td>Distinguished Name 53, 54</td>
<td></td>
</tr>
<tr>
<td>Distributed Management Interface 9, 10</td>
<td></td>
</tr>
<tr>
<td>Distributed Management Task Force 8, 9, 57</td>
<td></td>
</tr>
<tr>
<td>DIT 54, 65, 66</td>
<td></td>
</tr>
<tr>
<td>DMI 9, 10</td>
<td></td>
</tr>
<tr>
<td>DMTF 8, 10, 52, 57, 70, 76</td>
<td></td>
</tr>
<tr>
<td>DN 53, 54, 64, 65, 66, 67</td>
<td></td>
</tr>
<tr>
<td>DNF 57, 65</td>
<td></td>
</tr>
<tr>
<td>DSCP 17, 18, 30, 86</td>
<td></td>
</tr>
</tbody>
</table>

### E

<table>
<thead>
<tr>
<th>Term</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>edge devices 12</td>
<td></td>
</tr>
<tr>
<td>ELS 71, 119</td>
<td></td>
</tr>
<tr>
<td>enforcing policy 4</td>
<td></td>
</tr>
<tr>
<td>enterprise wide policies 3</td>
<td></td>
</tr>
<tr>
<td>ESP 79</td>
<td></td>
</tr>
</tbody>
</table>

### F

<table>
<thead>
<tr>
<th>Term</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>flow 29</td>
<td></td>
</tr>
</tbody>
</table>

© Copyright IBM Corp. 1999
flowspec

G
Global Enterprise Manager 7
global policy management approach 3

H
H.323 27
help desk response times 4
HMAC 83
HTML 85

I
IBM 53
IBM Common Policy Engine 54
IBM schema 55
IBM SecureWay Directory 54, 55
identity 21
IETF 8, 10, 18, 23, 52, 54, 55, 57, 70, 76, 86, 87, 119
Policy Framework 18
IKE 18, 29, 30, 34, 35, 39, 73
Integrated Services 21
Internet Engineering Task Force 8, 10
Internet Service Providers 112
Internet-standard Network Management Framework 10
IntServ 11, 18, 30, 36, 38, 57, 87, 90, 91
IPSec 18, 30, 35, 36, 38, 57, 59, 69, 73, 77, 78, 79, 82
action 39, 79, 82
definition 68
IETF 11, 70
in detail 81
IPSec action 36
manual-keyed tunnel 36
module 29, 34
policy and rules 72
policy segment 80
policy working group 119
signaling 31
technology 5
transform 94
triggering mechanism 35
ISAKMP 5, 38, 39, 72
ISAKMP action 36
ISO 9594 50
ISP 21, 75
ISPs 112
ITU-T 50

J
Java 85

L
LDAP 33, 46, 49, 53, 54, 65, 68, 69, 76
attributes 63
client 30, 49, 68
core schema 62, 65
CS/390 86
directory 19, 50, 53
directory server 44
IETF 55
LIDIA 7
policy repository 76
policy search engine 69, 104
repository 18
requests 67
schema 70
schemata 29
server 77
transparent storage of data 54
Version 2 52
Version 3 52, 57
LIDIA 7
Lightweight Directory Access Protocol 7, 18, 29, 30, 44
Lightweight Internet Person Schema 55
LIPS 55
LPAR 89
LPDP 20

M
Managed Object File 10
Management Information Base 19
MAS 33
MD5 83
MIB 9, 19, 25, 91
Microsoft 53
MOF 10
MRS 33

N
Netscape 53
network and system management platform 13
network availability 4
network node 17
network policy administrator 30
Network Policy Framework 118
Networks Working Group 9
Novell 53
Nways 7
Nways Network Management tool 7
Nways Policy Configuration application 77
Nways Policy Manager 7
Nways VPN Management application 77

O
object class 53
object identifier 53
OID 53
Olympic 112
Operations and Maintenance group 11
Optivity Policy Server 2
OSA-Express 89
OSI 51
Outsourcing 23
policy
administration and configuration 7
Core Information Model 59
definition steps 4
definition tool 7, 14, 49, 75
Enforcement Entity 31
framework 11
local Policy Decision Point 20
management tool 45, 67
manager 30, 31
matcher 31
objects 35
Policy API 91
Policy Decision Point 12, 18, 19, 45, 76
Policy Enforcement Entity 31
Policy Enforcement Point 12, 18, 19, 44, 45
Policy Framework Working Group 11
Policy Information Base 11, 19, 25
policy-based management 4
priority number 37
repository 47
rule classes 25
rule identification 25
rule instances 25
schema 44, 89
server 13, 18
trigger 46
policySubtreesPtrAuxClass 67
PRC 25, 28, 29
pre-paid 22
PRI 25, 29
PRID 25
provisioning 23
quality of service 2, 59
guarantees 21

R
RAP 11, 23
RDN 64
repository 13, 18
Resource Allocation Protocol 11
RESV 22
reuse 54
RFC 1274 55
RFC 1777 51, 118
RFC 1778 55
RFC 2251 52, 118
RFC 2252 53, 55, 56
RFC 2256 55
RFC 2474 118
RFC 2475 118
RFC 2578 118
RFC 2597 118
RFC 2598 118
RFC 2638 118
RPT 27, 28
RSA 83
RSVP 11, 17, 18, 30, 35, 79, 82, 86, 90, 112
action 36, 38, 77, 79
Admission Policy Working Group 23
bandwidth management 112
COPS 24, 28
CS/390 90, 91
in detail 81
module 29, 34
policy segments 80
policy statistics 92
protocol 87
RAP 11
reservation request 21
RESV message 22
RSVP action 36
rule 72
signaling 86
technology 5
triggering mechanism 35

S
SAP 23
schema 53
Secure Way Policy Director 48
SecureWay 7
SecureWay FirstSecure 48
SecureWay Policy Director 75
security breaches 4
Service Agreement Manager 7
service level agreement 4, 10, 12, 45, 58, 87
service level objectives 45, 58
SHA 83, 105
Simple Network Policy Protocol 11
simple policy rule 65
SLA 10, 13, 45, 58
SLO 45, 58
SMI 19, 25
SNMP 9, 18, 23, 24, 25, 33, 91
trap 91
SNMP Management Framework 10
SNMP MIB browser 10
SNMP V4 11
SNPP 11
staged deployment 20
state and resource validation 46
stream 29
Structure of Management Information 19
successful policy enforcement 4
T
TCP/IP 51
(temporal dependencies 21
Tivoli 7, 48
Tivoli Millennium Framework 7
Tivoli Systems Millenium Framework 49
TOS 17
traffic differentiation 21
transactional model 24
Transcend Ware 2

U
UML 10
Unified Modeling Language 10

V
video conferencing application 27
Virtual Private Network 75
VLAN 58
VPN 5, 11, 13, 15, 23, 25, 30, 38, 50, 75, 77, 81, 82, 169

X
X.500 50, 53, 55

184 Application-Driven Networking: Concepts and Architecture for Policy-Based Systems
IBM Redbooks Evaluation

Application-Driven Networking: Concepts and Architecture for Policy-Based Systems
SG24-5640-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. Please complete this questionnaire and return it using one of the following methods:

- Use the online evaluation form found at [http://www.redbooks.ibm.com/](http://www.redbooks.ibm.com/)
- Fax this form to: USA International Access Code + 1 914 432 8264
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
- Customer
- Business Partner
- Solution Developer
- IBM employee
- None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction

Please answer the following questions:

Was this redbook published in time for your needs? Yes___ No___

If no, please explain:

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

What other redbooks would you like to see published?

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

Comments/Suggestions: (THANK YOU FOR YOUR FEEDBACK!)

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________

________________________________________________________________________