Network Working Group
Internet Draft
Expires June 2000

M. Fine
K. McCloghrie
Cisco Systems
J. Seligson
K. Chan
Nortel Networks
S. Hahn
Intel
A. Smith
Extreme Networks

Oct 22 1999

Quality of Service Policy Information Base

draft-mfine-cops-pib-02.txt

Status of this Memo

Disclaimer

This draft is preliminary and is known to be inconsistent in some respects with the Diffserv Conceptual Model [MODEL]. It is intended to correct this prior to the next version, as well as checking for full consistency with RFC 2474 and RFC 2475.

1. Glossary

PRC Policy Rule Class. A type of policy data.
PRI Policy Rule Instance. An instance of a PRC.
PIB Policy Information Base. The database of policy information.
PDP Policy Decision Point. See [RAP-FRAMEWORK].
PEP Policy Enforcement Point. See [RAP-FRAMEWORK].
PRID Policy Rule Instance Identifier. Uniquely identifies an instance of a a PRC.

2. Introduction

This document defines a set of policy rule classes for describing quality of service (QoS) policies.

This document structures QoS policy information as instances of policy rule classes. A policy rule class (PRC) is an ordered set of scalar attributes. Policy rule classes are arranged in a hierarchical structure similar to tables in SNMP's SMIv2 [SNMP-SMI]. As with SNMP tables, they are identified by a sequence of integer identifiers (an Object Identifier).

For each policy rule class a device may have zero or more policy rule instances. Each policy rule instance is also identified by a sequence of integers where the first part of the sequence is the ID of the PRC. Collections of policy rule classes are defined in PIB modules. These modules are written using a structure designed for policy information which is described in [COPS-PR].

3. General PIB Concepts

3.1. Roles

The policy to apply to an interface may depend on many factors such as immutable characteristics of the interface (e.g., ethernet or frame

relay), the status of the interface (e.g., half or full duplex), or user configuration (e.g., branch office or headquarters interface). Rather than specifying policies explicitly for each interface in the QoS domain, policies are specified in terms of interface functionality.

To describe these functionalities of an interface we use the concept of "roles". A role is simply a string that is associated with an interface. A given interface may have any number of roles simultaneously. Policy rule classes have an attribute called a "role-combination" which is an unordered set of roles. Instances of a given policy rule class are applied to an interface if and only if the set of roles in the role combination is identical to the set of the roles of the interface.

Thus, roles provide a way to bind policy to interfaces without having to explicitly identify interfaces in a consistent manner across all network devices. (The SNMP experience with ifIndex has proved this to be a difficult task.) That is, roles provide a level of indirection to the application of a set of policies to specific interfaces. Furthermore, if the same policy is being applied to several interfaces, that policy need be pushed to the device only once, rather than once per interface, as long as the interfaces are configured with the same role combination.

We point out that, in the event that the administrator needs to have unique policy for each interface, this can be achieved by configuring each interface with a unique role.

The PEP reports all its role combinations to the PDP at connect time or whenever they change.

The comparing of roles (or role combinations) must be case insensitive. For display purposes, roles (or role combinations) should preserve the case specified by the user.

The concept and usage of roles in this document is consistent with that specified in [POLICY]. Roles are currently under discussion in the IETF's Policy WG; as and when that discussion reaches a conclusion, this PIB will be updated in accordance with that conclusion.

3.2. Reporting of Device Capabilities

Each network device providing policy-based services has its own inherent capabilities. These capabilities can be hardware specific, e.g., an ethernet interface supporting input classification, or can be statically configured, e.g., supported queuing disciplines. These capabilities are

communicated to the PDP when initial policy is requested by the PEP. Knowing device capabilities, the PDP can send the policy rule instances (PRIs) relevant to the specific device, rather than sending the entire PIB.

4. DiffServ PIB Concepts

4.1. Filters, Filter Groups and Classifiers

The basis of differential QoS treatment of packets is a filter. This is simply a general specification for matching a pattern to appear in packets belonging to flows, e.g. microflows or bandwidth aggregates. Associated with each filter is a permit/deny flag which effectively gives a negation operation.

Sets of these filters are used to create classifiers. Classifiers are applied to interfaces with a direction flag to indicate an ingress or egress classifier. Filters are combined, in order, into filter groups; filter groups are then combined, in order, to build a classifier. This allows a rudimentary classification grammar to be defined. On input, each packet is checked against the ingress classifier on the interface. Similarly, on output each packet is checked against the egress classifier on the interface. The result of the classifier then feeds into appropriate meters and actions to be applied to packets.

For each classifier, the packet is checked against the set of filter groups in the appropriate order. The detailed operation of the PIB syntax is as follows. If a packet matches a filter in the first filter group of a classifier and the sense is "permit" then the subsequent meters and actions associated with that classifier are applied to that packet and no further filters are compared. If the sense is "deny" then the rest of the filters in the current filter group are skipped and operation proceeds with the first filter of the next filter group. If the packet does not match any of the filters in the filter group then the next filter group is tried. This process is continued until a definitive match is obtained. Each classifier must cover all possible matches i.e., it must be complete.

4.2. Applying QoS Policy Using Targets

The task of applying QoS policy within a network requires the specification of several components. The flows to which QoS policy should be applied must be identified. The interfaces of the device on which the policy should be enforced must be known. A certain set of

parameters to support flow metering is also required. The combination of these components provides the target against which QoS policy is to be applied. Within the context of the QoS PIB, the association between these components is defined efficiently using the Target class.

The Target class serves to logically link several other QoS policy classes. Flow classification rules, specifying behavior aggregate (BA) or multi-field (MF) classification parameters, are indirectly identified using the PRC for the appropriate classification class (e.g., IP, 802) coupled with an identifier for a specific classifier. Interface information is specified using the role combination tag, defined in the Interface Type class, to identify the group of interfaces on which classification is to be performed. The direction of packet flow on the identified interfaces is provided as well. A link to the metering component is provided using the PRC for the appropriate metering class instance.

Once a target has been defined, actions based on the classification and metering phases must be specified. Action class instances are linked with the Target entry through the associated Meter class instance. A precedence component is also provided so that a definitive order of evaluation may be defined for Target class instances being applied to the same interface role and flow direction targets. The Target class thus functions as the integration point for the range of components used for the application of QoS policy.

4.3. Queue Modeling with Queue Sets

The traffic processing capabilities of an interface are determined by the queuing resources that are associated with the interface. These capabilities are represented abstractly using queue sets. A queue set is comprised of one or more individual queues and facilitates treating the collection of queues as a single unit based on their combined behaviors. A device may support a number of different queue sets. The number of queue sets supported by a device is typically related to the number of unique combinations of interface properties within that device. The queue set abstraction is not limited to modeling physical interface properties, however, and can be used to represent logical and dynamic queuing behavior as well.

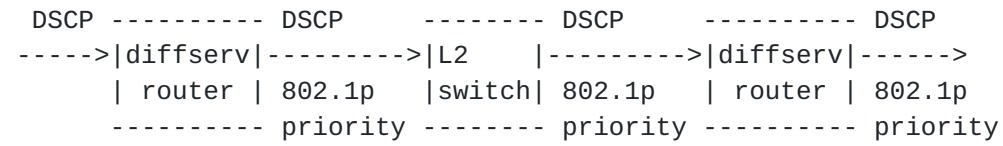
Each individual queue in a set is characterized by the interface bandwidth it can consume, the queuing discipline it employs and it's relationship with other queues in the set. Interface bandwidth allocation per queue can be represented in either relative or absolute terms. A queue's drain size (i.e., the maximum number of bytes that may

be drained from the queue in one cycle) can be used to determine the relative bandwidth allocation. The sum of the drain sizes of all of the related queues in a set is used to compute the percentage of interface bandwidth allocated to a specific queue based on its drain size. The maximum interface bandwidth that is available may also be described in absolute terms.

The traffic processing paradigm employed by a given queue is represented by queue discipline attributes. Several general purpose and well-known queuing disciplines (e.g., priority, fifo, weighted fair queuing) are supported and a mechanism to define additional paradigms in an extensible fashion is provided. The relationship among queues within a set is specified using a service order attribute. This attribute provides an additional level of service precedence among queues. This is required for describing the behavior of queues utilizing the same processing discipline (e.g., a series of priority queues) and when the various queues that comprise a queue set are serviced using a mix of queuing disciplines (e.g., priority and weighted round robin queues). These individual queue attributes, when combined, support the representation of (potentially) complex queuing systems associated with an interface type (i.e., role combination).

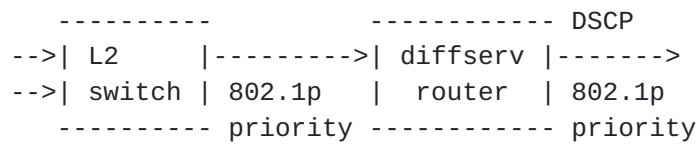
4.4. IP Mapping to and from Layer 2

The PIB specifies QoS policy by assigning DSCP values to specific queues, but in order to provide a complete QoS picture, the PIB must consider that not all devices on the network are diffserv capable, i.e., capable of setting/inspecting a packet's DSCP value. Specifically, the network might include layer 2 devices (switches) that can only support IEEE 802.1p classes of service. In order to support network configuration that consists of diffserv capable devices and devices that can only support IEEE 802.1p, the PIB has included a mapping table that can allow the DSCP values to be mapped to specific IEEE 802.1p tag values.



A second case exists where packets coming into the network are arriving from a non-diffserv enabled device and no DSCP exists with in the

packet, but an 802.1p tag does exist. In the case where the diffserv device has the ability to set a DSCP in the packet, the diffserv router can map the layer 2 tag into a DSCP value. The PIB supports a mapping table that can be used to map from the layer 2 tag to a DSCP value. This mapping would be configured to apply to those ports where the upstream device marks packets using a L2 802.1p tag as shown in the figure below.



Alternatively, the diffserv router can have policies applied to it that cause it to reclassify the incoming packet using a MF classifier, ignoring the incoming 802.1p tag.

5. Summary of the PIB Modules

This section gives a brief summary of the top-level groups in the three modules defined in this document.

Device Configuration Group
: This group contains device configuration information. This configuration is either set by management or reflects the physical configuration of the device.

QoS Interface Group
: This group is used to indicate to the PDP the types of interface configured on the PEP. Note that this group indicates the types of interfaces, not the configuration of each and every interface on the device.

QoS Metering Group
: This group contains configuration of meters. These meters can then be used to by target classes to specify metering policy.

QoS Action Group
: This group contains the policies that define the action to be taken after the result of the classification and metering. This group also contains the policies that associate the classifiers, meters and actions.

IP Classification and Policing Group
This group contains the policies that define the IP classifier elements.

802 Classification and Policing Group
This group contains the policies that define the IEEE 802 classifier elements.

6. PIB Operational Overview

This section provides an operation overview of how the three modules are used in concert to provide policy to the PEP.

After initial PEP to PDP communication setup, using [COPS-PR] for example, the PEP will provide to the PDP the PIB Policy Rule Classes (PRCs), interface types, and interface type capabilities it supports.

The PRCs supported by the PEP are reported to the PDP in the PRC Support Table, qosPrcSupportTable. Each instance of the qosPrcSupportTable class indicates a PRC that the PEP understands and for which the PDP can send class instances as part of the policy information.

The interface types the PEP supports are reported to the PDP in the Interface Type Table, qosInterfaceTypeTable. Each instance of this class describes the characteristics of an interface type. Each interface type is identified by a role combination. Each interface type's inherent capability is reported to the PDP using the Interface Type Table. Examples of interface capabilities are classification, policing, dropping, queuing, and shaping. An interface type is associated with a queue set which indicates the number of queues that interface supports and its queuing disciplines.

The PDP, with knowledge of the PEP's capabilities, will provide the PEP with:

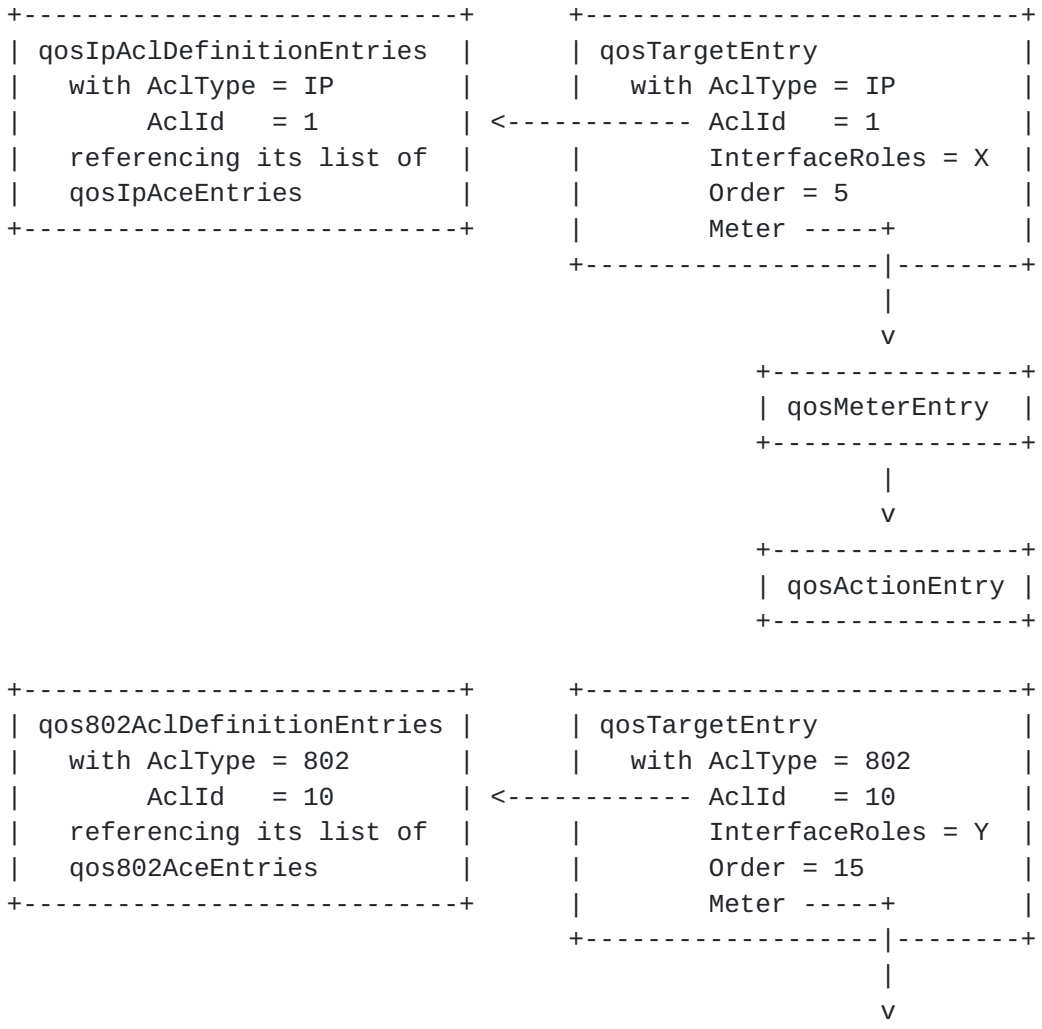
(1) Administration domain policy information in
qosIfDscpAssignmentTable
qos802DscpMappingTable
qos802CosToDscpTable

(2) Interface type and role specific IP policy information in
qosIpAceTable
qosIpAclDefinitionTable
qosActionTable

qosTargetTable
qosMeterTable

(3) Interface type and role specific IEEE 802 policy information in
qos802AceTable
qos802AclDefinitionTable

Instances of the qosTargetTable define how the Traffic Conditioning Elements are combined into Traffic Conditioning Blocks, as described in [MODEL]. Each instance of the qosTargetTable applies to an interface type defined by its roles and direction (ingress or egress). This is pictured in the following diagram where the InterfaceRoles X, and Y would be used by the network device to associate the traffic conditioning block with the interfaces needing each of thess policies.



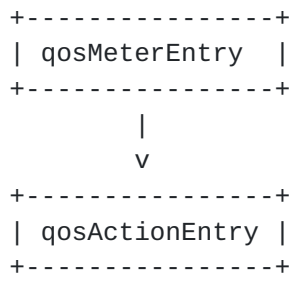


Figure 7.1 Diffserv PIB Table Relationships

Notice in the above diagram, IEEE 802 type classifiers are intermixed with the IP type classifiers, sharing the same pool of Traffic Conditioning Elements. The qosTargetTable allows use of heterogeneous classifiers with same instance of qosMeterTable. Using IP and IEEE 802 classifiers together is just one example. Other types of classifiers may be used heterogeneously.

After receiving the PIB, the PEP will associate the Classifier, Meter and Action with the corresponding interfaces supporting the specific interface type and roles.

7. PIB Definitions

NOTE

In these PIB definitions, we use the term "access control entry" (ACE) synonymous with filter, "access control list" (ACL) synonymous with filter group, and sets of ACLs synonymous with classifier.

7.1. The Policy Framework PIB Module

```
POLICY-FRAMEWORK-PIB PIB-DEFINITIONS ::= BEGIN

IMPORTS
    Unsigned32, MODULE-IDENTITY, OBJECT-TYPE
            FROM SNMPv2-SMI
    TEXTUAL-CONVENTION
            FROM SNMPv2-TC
    SnmpAdminString
            FROM SNMP-FRAMEWORK-MIB;

policyFrameworkPib  MODULE-IDENTITY
    LAST-UPDATED "9906241800Z"
    ORGANIZATION "IETF RAP WG"
    CONTACT-INFO "
                  Michael Fine
                  Cisco Systems, Inc.
                  170 West Tasman Drive
                  San Jose, CA  95134-1706 USA
                  Phone: +1 408 527 8218
                  Email: mfine@cisco.com

                  Keith McCloghrie
                  Cisco Systems, Inc.
                  170 West Tasman Drive,
                  San Jose, CA 95134-1706 USA
                  Phone: +1 408 526 5260
                  Email: kzm@cisco.com

                  John Seligson
                  Nortel Networks, Inc.
                  4401 Great America Parkway
                  Santa Clara, CA 95054 USA
                  Phone: +1 408 495 2992
                  Email: jseligso@nortelnetworks.com"
```

```
    DESCRIPTION
            "A PIB module containing the base set of policy
             rule classes that are required for support of
             all policies."

    ::= { tbd }

policyBasePibClasses
             OBJECT IDENTIFIER ::= { policyFrameworkPib 1 }

--
-- Textual Conventions
--

--
-- Interface Role
--

Role ::= TEXTUAL-CONVENTION
    STATUS       current
    DESCRIPTION
        "A display string but where the characters '+', ' ' (space),
        NULL, LF, CR, BELL, BS, HT (tab) VT and FF are illegal."

    SYNTAX SnmpAdminString (SIZE (0..31))

--
-- Interface Role Combination
--

RoleCombination ::= TEXTUAL-CONVENTION
    STATUS       current
    DESCRIPTION
        "A Display string consisting of a set of roles concatenated
        with a '+' character where the roles are in lexicographic
        order from minimum to maximum."

    SYNTAX SnmpAdminString  (SIZE (0..255))

--
-- Policy Instance Index
--

PolicyInstanceId ::= TEXTUAL-CONVENTION
    STATUS       current
```

```
    DESCRIPTION
        "A textual convention for an attribute that is an integer
        index of a class.  It is used for attributes that exist
        for the purpose of providing a policy rule instance with
        a unique instance identifier.

        For any instance identifier that refers to another policy
        rule instance, that other policy instance must exist.
        Furthermore, it is an error to try to delete a policy rule
        instance that is referred to by another instance without
        first deleting the referencing instance.

        Class instances of this type need not be contiguous."

    SYNTAX Unsigned32

--
-- Device Configuration Group
--

-- This group contains device configuration information.  This
-- configuration is either set by management or reflects the physical
-- configuration of the device.  This configuration is generally
-- reported to the PDP (i.e., the policy server) when configuration
-- is performed by the policy server so that the PDP can determine
-- what policies to download to the PEP (i.e., the device). Class
-- instances may also be downloaded by a network manager prior to
-- static configuration.
--

policyDeviceConfig OBJECT IDENTIFIER ::= { policyBasePibClasses 1 }

--
-- PRC Support Table
--

policyPrcSupportTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF PolicyPrcSupportEntry
    POLICY-ACCESS  notify
    STATUS         current
    DESCRIPTION
        "Each instance of this class specifies a PRC that the device
        supports and a bit string to indicate the attributes of the
        class that are supported.  These PRIs are sent to the PDP to
        indicate to the PDP which PRCs, and which attributes of these
```

```
        PRCs, the device supports. This table can also be downloaded
        by a network manager when static configuration is used.

        All install and install-notify PRCs supported by the device
        must be represented in this table."

    ::= { policyDeviceConfig 1 }

policyPrcSupportEntry OBJECT-TYPE
    SYNTAX         PolicyPrcSupportEntry
    STATUS         current
    DESCRIPTION
        "An instance of the policyPrcSupport class that identifies a
        specific policy class and associated attributes as supported
        by the device."

    INDEX { policyPrcSupportId }
    ::= { policyPrcSupportTable 1 }

PolicyPrcSupportEntry ::= SEQUENCE {
        policyPrcSupportId             PolicyInstanceId,
        policyPrcSupportSupportedPrc   OBJECT IDENTIFIER,
        policyPrcSupportSupportedAttrs OCTET STRING
}

policyPrcSupportId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies an
        instance of the policyPrcSupport class."

    ::= { policyPrcSupportEntry 1 }

policyPrcSupportSupportedPrc OBJECT-TYPE
    SYNTAX         OBJECT IDENTIFIER
    STATUS         current
    DESCRIPTION
        "The object identifier of a supported PRC. There may not
        be more than one instance of the policyPrcSupport class with
        the same value of policyPrcSupportSupportedPrc."

    ::= { policyPrcSupportEntry 2 }

policyPrcSupportSupportedAttrs OBJECT-TYPE
```

```
    SYNTAX         OCTET STRING
    STATUS         current
    DESCRIPTION
        "A bit string representing the supported attributes of the
        class that is identified by the policyPrcSupportSupportedPrc
        object.

        Each bit of this bit mask corresponds to a class attribute,
        with the most significant bit of the i-th octet of this octet
        string corresponding to the (8*i - 7)-th attribute, and the
        least significant bit of the i-th octet corresponding to the
        (8*i)-th class attribute. Each bit of this bit mask specifies
        whether or not the corresponding class attribute is currently
        supported, with a '1' indicating support and a '0' indicating
        no support. If the value of this bit mask is N bits long and
        there are more than N class attributes then the bit mask is
        logically extended with 0's to the required length."

    ::= { policyPrcSupportEntry 3 }

--
-- PIB Incarnation Table
--

policyDevicePibIncarnationTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF PolicyDevicePibIncarnationEntry
    POLICY-ACCESS  install-notify
    STATUS         current
    DESCRIPTION
        "This class contains a single policy rule instance that
        identifies the current incarnation of the PIB and the PDP
        or network manager that installed this incarnation.  The
        instance of this class is reported to the PDP at client
        connect time so that the PDP can (attempt to) ascertain the
        current state of the PIB. A network manager may use the
        instance to determine the state of the device with regard
        to existing NMS interactions."

    ::= { policyDeviceConfig 2 }

policyDevicePibIncarnationEntry OBJECT-TYPE
    SYNTAX         PolicyDevicePibIncarnationEntry
    STATUS         current
    DESCRIPTION
        "An instance of the policyDevicePibIncarnation class. Only
```

```
        one instance of this policy class is ever instantiated."

    INDEX { policyDevicePibIncarnationId }
    ::= { policyDevicePibIncarnationTable 1 }

PolicyDevicePibIncarnationEntry ::= SEQUENCE {
        policyDevicePibIncarnationId      PolicyInstanceId,
        policyDevicePibIncarnationName    SnmpAdminString,
        policyDevicePibIncarnationId      OCTET STRING,
        policyDevicePibIncarnationTtl     Unsigned32
}

policyDevicePibIncarnationId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An index to uniquely identify an instance of this
        policy class."

    ::= { policyDevicePibIncarnationEntry 1 }

policyDevicePibIncarnationName OBJECT-TYPE
    SYNTAX         SnmpAdminString
    STATUS         current
    DESCRIPTION
        "The name of the entity that installed the current
        incarnation of the PIB into the device. The name may
        reference a PDP when dynamic configuration is being
        used or a network manager when static configuration
        is being used. By default, it is the zero length
        string."

    ::= { policyDevicePibIncarnationEntry 2 }

policyDevicePibIncarnationId OBJECT-TYPE
    SYNTAX         OCTET STRING
    STATUS         current
    DESCRIPTION
        "An ID to identify the current incarnation.  It has meaning
        to the PDP/manager that installed the PIB and perhaps its
        standby PDPs/managers. By default, it is the zero-length
        string."

    ::= { policyDevicePibIncarnationEntry 3 }
```

```
policyDevicePibIncarnationTtl OBJECT-TYPE
    SYNTAX         Unsigned32
    STATUS         current
    DESCRIPTION
        "The number of seconds after a client close or TCP timeout
        for which the PEP continues to enforce the policy in the PIB.
        After this interval, the PIB is considered expired and the
        device no longer enforces the policy installed in the PIB.
        Policy enforcement timing only applies to policies that have
        been installed dynamically (e.g., by a PDP via COPS)."

    ::= { policyDevicePibIncarnationEntry 4 }

END
```

7.2. The QoS IP PIB

```
QOS-POLICY-IP-PIB PIB-DEFINITIONS ::= BEGIN

IMPORTS
    Unsigned32, IpAddress, Integer32,
    MODULE-IDENTITY, OBJECT-TYPE
            FROM SNMPv2-SMI
    TruthValue, TEXTUAL-CONVENTION
            FROM SNMPv2-TC
    RoleCombination, PolicyInstanceId
            FROM POLICY-FRAMEWORK-PIB;

qosPolicyIpPib  MODULE-IDENTITY
    LAST-UPDATED "9906241800Z"
    ORGANIZATION "IETF RAP WG"
    CONTACT-INFO "
                  Michael Fine
                  Cisco Systems, Inc.
                  170 West Tasman Drive
                  San Jose, CA  95134-1706 USA
                  Phone: +1 408 527 8218
                  Email: mfine@cisco.com

                  Keith McCloghrie
                  Cisco Systems, Inc.
                  170 West Tasman Drive,
                  San Jose, CA 95134-1706 USA
                  Phone: +1 408 526 5260
                  Email: kzm@cisco.com

                  John Seligson
                  Nortel Networks, Inc.
                  4401 Great America Parkway
                  Santa Clara, CA 95054 USA
                  Phone: +1 408 495 2992
                  Email: jseligso@nortelnetworks.com"
    DESCRIPTION
            "The PIB module containing an initial set of policy
             rule classes that describe the quality of service
             (QoS) policies. It includes general classes that may
             be extended by other PIB specifications as well as
             an initial set of PIB classes related to IP processing."

    ::= { tbd }
```

```
qosPolicyGenPibClasses  OBJECT IDENTIFIER ::= { qosPolicyIpPib 1 }
qosPolicyIpPibClasses   OBJECT IDENTIFIER ::= { qosPolicyIpPib 2 }

--
-- Textual Conventions
--

--
-- Diffserv Codepoint
--

Dscp ::= TEXTUAL-CONVENTION
    STATUS       current
    DESCRIPTION
        "An integer that is in the range of the diffserv codepoint
        values."

    SYNTAX INTEGER (0..63)

--
-- Interface types
--

QosInterfaceQueueCount ::= TEXTUAL-CONVENTION
    STATUS       current
    DESCRIPTION
        "An integer that describes the number of queues an interface
        supports.  It is limited to the number of DSCP values."

    SYNTAX INTEGER (1..64)

--
-- QoS Interface Group
--
--
-- This group specifies the configuration of the various interface
-- types including the setting of queueing parameters and the
-- mapping of DSCPs and 802.1 CoS to queues.
--

qosIfParameters OBJECT IDENTIFIER ::= { qosPolicyGenPibClasses 1 }

--
-- Interface Type Table
```

```
--

qosInterfaceTypeTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF QosInterfaceTypeEntry
    POLICY-ACCESS  notify
    STATUS         current
    DESCRIPTION
        "Interface type definitions. This class describes the types
        of interfaces that exist on the device. An interface type
        is denoted by its designated role identifier as well as
        by the queue set and queue capabilities it supports."

    ::= { qosIfParameters 1 }

qosInterfaceTypeEntry OBJECT-TYPE
    SYNTAX         QosInterfaceTypeEntry
    STATUS         current
    DESCRIPTION
        "An instance of this class describes the characteristics
        of a type of an interface. Interface type characteristics
        include a role combination identifier, a queue set
        identifier and a queue capabilities attribute.  An
        instance is required for each different unique role
        combination identifier which represents the different
        interface types that are operational in the device at
        any given time.  The PEP does not report which specific
        interfaces have which characteristics."

    INDEX { qosInterfaceTypeId }
    ::= { qosInterfaceTypeTable 1 }

QosInterfaceTypeEntry ::= SEQUENCE {
        qosInterfaceTypeId           PolicyInstanceId,
        qosInterfaceTypeRoles        RoleCombination,
        qosInterfaceTypeQueueSet     PolicyInstanceId,
        qosInterfaceTypeCapabilities BITS
}

qosInterfaceTypeId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies a
        instance of the qosInterfaceType class. Class instances
        may not be contiguous."
```

```
    ::= { qosInterfaceTypeEntry 1 }

qosInterfaceTypeRoles OBJECT-TYPE
    SYNTAX         RoleCombination
    STATUS         current
    DESCRIPTION
        "The role combination that is used to identify interfaces
        with the characteristics specified by the attributes
        of this class instance. Interface role combination
        identifiers are used within a number of classes to
        logically identify a physical set of interfaces to which
        policy rules and actions are applied. Role combination
        identifiers must exist in this table prior to being
        referenced in other class instances."

    ::= { qosInterfaceTypeEntry 2 }

qosInterfaceTypeQueueSet OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "The index of the queue set that is associated with
        interfaces that are identified with the role combination
        identifier that is associated with this class instance."

    ::= { qosInterfaceTypeEntry 3 }

qosInterfaceTypeCapabilities OBJECT-TYPE
    SYNTAX         BITS {
                        other(0),

                        -- Classification support
                        inputIpClassification(1),
                        outputIpClassification(2),
                        input802Classification(3),
                        output802Classification(4),

                        -- Queuing discipline support
                        singleQueuingDiscipline(5),
                        hybridQueuingDiscipline(6)
                   }
    STATUS         current
    DESCRIPTION
        "An enumeration of interface capabilities.  Used by the
        PDP or network manager to select which policies and
```

```
        configuration it should push to the PEP."

    ::= { qosInterfaceTypeEntry 4 }

--
-- Interface Queue Table
--
-- The Interface Queue Table enumerates the individual queues that
-- comprise a given queue set. Information specific to each queue
-- is exported by this table.
--

qosIfQueueTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF QosIfQueueEntry
    POLICY-ACCESS  notify
    STATUS         current
    DESCRIPTION
        "Contains information about the individual queues that
        comprise a queue set implemented on the device."

    ::= { qosIfParameters 2 }

qosIfQueueEntry OBJECT-TYPE
    SYNTAX         QosIfQueueEntry
    STATUS         current
    DESCRIPTION
        "A conceptual row in the qosIfQueueTable.

        Each row identifies a specific queue within a given queue
        set and contains detailed information about the queue. Queues
        are associated with a given set through this table and
        a queue set is associated with an interface set through
        the qosInterfaceTypeTable."

    INDEX { qosIfQueueId }
    ::= { qosIfQueueTable 1 }

QosIfQueueEntry ::= SEQUENCE {
        qosIfQueueId                    PolicyInstanceId
        qosIfQueueSetId                 INTEGER,
        qosIfQueueIndex                 QosInterfaceQueueCount,
        qosIfQueueGenDiscipline         INTEGER,
        qosIfQueueExtDiscipline         OBJECT IDENTIFIER,
        qosIfQueueDrainSize             Unsigned32,
        qosIfQueueAbsBandwidth          Unsigned32,
```

```
        qosIfQueueBandwidthAllocation   INTEGER,
        qosIfQueueServiceOrder          QosInterfaceQueueCount,
        qosIfQueueSize                  Unsigned32
}

qosIfQueueId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "The index that uniquely identifies this row in the table,
        i.e., this PRI."

    ::= { qosIfQueueEntry 1 }

qosIfQueueSetId OBJECT-TYPE
    SYNTAX         INTEGER
    STATUS         current
    DESCRIPTION
        "An index that uniquely identifies a specific queue set. The
        queue set that is identified with this value is associated
        with an interface set through the qosInterfaceTypeQueueSet
        object in the qosInterfaceTypeTable. The individual queues
        that are members of this set all have the same value for
        this attribute (i.e., they have the same set ID)."

    ::= { qosIfQueueEntry 2 }

qosIfQueueIndex OBJECT-TYPE
    SYNTAX         QosInterfaceQueueCount
    STATUS         current
    DESCRIPTION
        "An arbitrary index that uniquely identifies a specific
        queue within a set of queues that is identified by the
        qosIfQueueSetId value."

    ::= { qosIfQueueEntry 3 }

qosIfQueueGenDiscipline OBJECT-TYPE
    SYNTAX         INTEGER {
                       other(1),  -- Use qosIfQueueExtDiscipline
                       fifo(2),   -- First In First Out queuing
                       pq(3),     -- Priority Queuing
                       fq(4),     -- Fair Queuing
                       wfq(5)     -- Weighted Fair Queuing
                   }
```

```
    STATUS        current
    DESCRIPTION
        "This object identifies the queuing discipline that is
        associated with the specified queue. Several general
        purpose and well-known queuing disciplines are supported
        by this attribute. Queuing disciplines that differ from
        those that are supported by this object are specified
        by setting this attribute to other(1) and providing
        the object identifier that represents the different
        queuing paradigm in the qosIfQueueExtDiscipline object.

        A value of fifo(2) indicates that the queue is serviced
        on a first-in-first-out (FIFO) basis. This discipline is
        generally employed when only a single queue is available
        for a given interface.

        A value of pq(3) indicates that the queue is serviced
        using a priority queuing discipline. This technique is
        used when several queues are available for a given
        interface. Each queue is assigned a priority and queues
        are serviced in order of priority. Higher priority queues
        are completely drained before lower priority queues are
        serviced.

        A value of fq(4) indicates that the queue is serviced
        using a fair queuing discipline. This technique is used
        when several queues are available for a given interface.
        Each queue is treated equally and is serviced in a
        round-robin fashion.

        A value of wfq(5) indicates that the queue is serviced
        using a weighted fair queuing discipline. This technique is
        used when several queues are available for a given interface.
        Each queue is serviced based on queue weights which determine
        the scheduling and frequency of queue servicing. Queues that
        are assigned a greater weight are implicitly provided with
        more bandwidth.

        Note that the processing disciplines for all of the queues
        in a given set must be considered when trying to establish
        a processing profile for a given interface."

    ::= { qosIfQueueEntry 4 }

qosIfQueueExtDiscipline OBJECT-TYPE
```

```
    SYNTAX         OBJECT IDENTIFIER
    STATUS         current
    DESCRIPTION
        "This object identifies the queuing discipline that is
        associated with the specified queue.  This attribute
        provides a means through which additional queuing mechanisms
        can be identified should the general queuing disciplines
        be inadequate for a given device. As such. this attribute is
        consulted only when the value of the qosIfQueueGenDiscipline
        object is other(1). It contains an object identifier that
        uniquely identifies a queuing paradigm.

        Note that the processing disciplines for all of the queues
        in a given set must be considered when trying to establish
        a processing profile for a given interface."

    ::= { qosIfQueueEntry 5 }

qosIfQueueDrainSize OBJECT-TYPE
    SYNTAX         Unsigned32
    STATUS         current
    DESCRIPTION
        "The maximum number of bytes that may be drained from the
        queue in one cycle.  The percentage of the interface
        bandwidth allocated to this queue can be calculated from
        this attribute and the sum of the drain sizes of all the
        queues in a specific queue cluster in a queue set.

        This attribute represents the relative bandwidth that is
        available to a given queue with respect to other queues with
        which it is associated. The absolute bandwidth that is
        available to a given queue is specified by the attribute
        qosIfQueueAbsBandwidth.  Which of these two applies is
        specified by the attribute qosIfQueueBandwidthAllocation."

    ::= { qosIfQueueEntry 6 }

qosIfQueueAbsBandwidth OBJECT-TYPE
    SYNTAX         Unsigned32
    STATUS         current
    DESCRIPTION
        "The maximum interface bandwidth that is available for
        consumption when servicing this queue. This bandwidth is
        specified in terms of kilobits per second.
```

```
        This attribute represents the absolute bandwidth that is
        available to a given queue. The relative bandwidth that is
        available to a given queue, with respect to other queues with
        which it is associated, is specified by the attribute
        qosIfQueueDrainSize.  Which of these two applies is specified
        by the attribute qosIfQueueBandwidthAllocation."

    ::= { qosIfQueueEntry 7 }

qosIfQueueBandwidthAllocation OBJECT-TYPE
    SYNTAX         INTEGER {
                        absolute(1),  --use qosIfQueueAbsBandwidth
                        relative(2)   --use qosIfQueueDrainSize
                   }
    STATUS         current
    DESCRIPTION
        "This attribute specifies whether to configure the queue for
        an absolute bandwidth limit or one that is relative to other
        queues of the interface. i.e., whether to configure the queue
        using qosIfQueueAbsBandwidth or qosIfQueueDrainSize."

     ::= { qosIfQueueEntry 8 }

qosIfQueueServiceOrder OBJECT-TYPE
    SYNTAX         QosInterfaceQueueCount
    STATUS         current
    DESCRIPTION
        "This object is used to provide an additional level of
        priority that is required for certain queuing disciplines
        and when the different queues that comprise a queue set
        are serviced using a mix of queuing disciplines. This
        object can be used to specify, for example, the order in
        which queues will be serviced when priority queuing is
        used. It also supports the ability to describe the
        servicing hierarchy when a hybrid queuing scheme, such
        as priority queuing coupled with weighted fair queuing,
        is used.

        Queue service priority is assigned such that a lower
        service order value indicates a higher priority. For
        example, a priority queue with a value of 1 will be
        serviced (i.e., drained) before another priority queue
        with a service order value of 2.

        Note that multiple queues that are logically associated,
```

based on the queuing discipline that is being employed, will be assigned the same service order value. Under this scenario, other parameters that are related to the queuing discipline determine the order of queue servicing (e.g., queue drain size is used for 'wfq').

For example, an interface that is associated with a queue set supporting two priority queues and three queues that are serviced using WFQ would be modeled as follows:

Q Index	Q Discipline	Q Drain Size	Q Service Order
22	pq(1)	-	1
23	pq(1)	-	2
24	wfq(3)	500	3
25	wfq(3)	350	3
26	wfq(3)	150	3

The queue set presented in this example would service all queued traffic in queue 22 first, followed by all of the queued traffic in queue 23. Next the queued traffic in queues 24 through 26 would be serviced in a round robin fashion with queue 24 receiving 50% of the available bandwidth, queue 25 receiving 35% of the available bandwidth and queue 26 receiving 15% of the available bandwidth. This example is presented for expository purposes and has been simplified accordingly.

Note that, in this example, queues 24, 25 and 26 form a queue cluster. Members of a queue cluster are all assigned the same qosIfQueueServiceOrder as there are tightly coupled. The qosIfQueueDrainSize attribute is used to determine the additional processing characteristics of the individual queues in a cluster."

::= { qosIfQueueEntry 9 }

```
qosIfQueueSize OBJECT-TYPE
    SYNTAX         Unsigned32
    STATUS         current
    DESCRIPTION
       "The size of the queue in bytes.  Some devices set queue size
        in terms of packets.  These devices must calculate the queue
        size in packets by assuming an average packet size suitable
        for the particular interface.
```

```
        Some devices have a fixed size buffer to be shared among all
        queues.  These devices must allocate a fraction of the
        total buffer space to this queue calculated as the the ratio
        of the queue size to the sum of the queue sizes for the
        interface."

    ::= { qosIfQueueEntry 10 }

--
-- DSCP Assignment Table
--
-- Supports the assignment of DSCPs to queues for each
-- interface type.
--

qosIfDscpAssignmentTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF QosIfDscpAssignmentEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "Supports the assignment of DSCP values to a queue for
        each interface with a specific queue count. There will be
        64 instances of this class for each supported combination
        of queue count and role combination."

    ::= { qosIfParameters 3 }

qosIfDscpAssignmentEntry OBJECT-TYPE
    SYNTAX         QosIfDscpAssignmentEntry
    STATUS         current
    DESCRIPTION
        "An instance of the qosIfDscpAssignment class."

    INDEX { qosIfDscpAssignmentId }
    ::= { qosIfDscpAssignmentTable 1 }

QosIfDscpAssignmentEntry ::= SEQUENCE {
        qosIfDscpAssignmentId         PolicyInstanceId,
        qosIfDscpAssignmentRoles      RoleCombination,
        qosIfDscpAssignmentDscp       Dscp,
        qosIfDscpAssignmentQueue      QosInterfaceQueueCount
}

qosIfDscpAssignmentId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
```

```
    STATUS         current
    DESCRIPTION
        "An index that is used to uniquely identify the
        instance of the qosIfDscpAssignment class."

    ::= { qosIfDscpAssignmentEntry 1 }

qosIfDscpAssignmentRoles OBJECT-TYPE
    SYNTAX         RoleCombination
    STATUS         current
    DESCRIPTION
        "The role combination with which an interface must be
        configured to support the DSCP-to-queue assignment
        described by this instance. The specified role
        combination must be defined in the qosInterfaceType
        table prior to being referenced by this entry.
        Otherwise a 'priAssociationUnknown(3)' error code
        will be returned."

    ::= { qosIfDscpAssignmentEntry 2 }

qosIfDscpAssignmentDscp OBJECT-TYPE
    SYNTAX         Dscp
    STATUS         current
    DESCRIPTION
        "The DSCP to which this class instance applies."

    ::= { qosIfDscpAssignmentEntry 3 }

qosIfDscpAssignmentQueue OBJECT-TYPE
    SYNTAX         QosInterfaceQueueCount
    STATUS         current
    DESCRIPTION
        "The specific queue, within the queue set that is
        associated with the interface set identified by the
        qosIfDscpAssignmentRoles tag, on which traffic with
        the specified DSCP, dictated by the
        qosIfDscpAssignmentDscp value, is placed. Failure to
        specify an appropriate queue results in a
        'priAssociationConflict(4)' error indication being
        returned."

    ::= { qosIfDscpAssignmentEntry 4 }
```

```
--
-- QoS Meter Table
--
-- The QoS Meter Table contains metering specifications that
-- can be used to provide an acceptable flow bandwidth
-- dimension to the Target table.
--

qosMeter OBJECT IDENTIFIER ::= { qosPolicyGenPibClasses 2 }

qosMeterTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF QosMeterEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "Contains the current set of configured meters. The
        meters are associated with a classifier during
        operation through the QoS Target Table."

    ::= { qosMeter 1 }

qosMeterEntry OBJECT-TYPE
    SYNTAX         QosMeterEntry
    STATUS         current
    DESCRIPTION
        "General metering definitions. Each entry specifies
        an instance of the qosMeter class which specifies
        metering information in terms of traffic stream
        bandwidth parameters. An entry can thus be used to
        support traffic metering based on the specified
        service level specification."

    INDEX { qosMeterId }
    ::= { qosMeterTable 1 }

QosMeterEntry ::= SEQUENCE {
        qosMeterId               PolicyInstanceId,
        qosMeterDataSpecification INTEGER,
        qosMeterCommittedRate    Unsigned32,
        qosMeterCommittedBurst   Unsigned32,
        qosMeterPeakRate         Unsigned32,
        qosMeterPeakBurst        Unsigned32,
        qosMeterHighConfAction   PolicyInstanceId,
        qosMeterMedConfAction    PolicyInstanceId,
```

```
        qosMeterLowConfAction     PolicyInstanceId
}

qosMeterId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies
        the instance of the qosMeter class. Meters are
        associated with specific flows using this attribute
        through the qosTargetMeter attribute in the QoS
        Target class."

    ::= { qosMeterEntry 1 }

qosMeterDataSpecification OBJECT-TYPE
    SYNTAX         INTEGER {
                        noMeterData(1),   -- no metering reqd
                        committedData(2), -- committed rate only
                        peakData(3)       -- committed and peak
                   }
    STATUS         current
    DESCRIPTION
        "Specifies the metering data, and thus the actions, that
        are defined in a given entry.

        A value of noMeterData(1) indicates that no flow metering
        is necessary. All flows associated with this meter entry
        are considered to be at a high level of conformance.

        A value of committedData(2) indicates that committed rate
        and committed burst information has been specified and will
        be applied to associated flows. No peak rate and burst
        information has been specified meaning that two levels
        of conformance (high, medium) are supported.

        A value of peakData(3) indicates that peak rate and peak
        burst information has been provided in addition to the
        committed rate and committed burst information. All provided
        information will be applied to associated flows meaning that
        three levels of conformance (high, medium, low) are
        supported."

     ::= { qosMeterEntry 2 }
```

```
qosMeterCommittedRate OBJECT-TYPE
    SYNTAX         Unsigned32 (0..'ffffffff'h)
    STATUS         current
    DESCRIPTION
        "This object represents the committed information rate
        (CIR) against which associated traffic streams will be
        metered. The CIR specifies the rate at which incoming
        traffic can arrive to be considered to be at a high
        level of conformance. Typically, this value specifies
        the rate at which tokens are added to a token bucket
        used to meter received flows.

        This object specifies a rate in bytes per second units
        such that, for example, a value of 100 equates to a
        committed information rate of 100 bytes per second.

        Committed rate (and burst) information must be present
        if the qosMeterDataSpecification object has the value
        committedData(2) or peakRate(3). This, in turn, requires
        that at least both high and medium conformance actions
        be specified."

    ::= { qosMeterEntry 3 }

qosMeterCommittedBurst OBJECT-TYPE
    SYNTAX         Unsigned32 (0..'ffffffff'h)
    STATUS         current
    DESCRIPTION
        "This object represents the committed burst size
        (CBS) against which associated traffic streams will
        be metered. The CBS specifies the maximum burst size
        that is supported for flows to be considered to be at
        a high level of conformance. Typically, this value
        represents the maximum number of tokens in a token
        bucket.

        This object specifies flow data in bytes per second
        units such that, for example, a value of 100 equates
        to a committed information rate of 100 bytes per
        second.

        Committed burst (and rate) information must be present
        if the qosMeterDataSpecification object has the value
        committedData(2) or peakRate(3). This, in turn, requires
        that at least both high and medium conformance actions
```

```
        be specified."

    ::= { qosMeterEntry 4 }

qosMeterPeakRate OBJECT-TYPE
    SYNTAX         Unsigned32 (0..'ffffffff'h)
    STATUS         current
    DESCRIPTION
        "This object represents the peak information rate (PIR)
        against which associated traffic streams will be
        metered. The PIR specifies the rate at which incoming
        traffic can arrive to be considered to be at a medium
        level of conformance. Typically, this value specifies
        the rate at which tokens are added to a token bucket
        used to meter received flows.

        This object specifies a rate in bytes per second units
        such that, for example, a value of 100 equates to a
        committed information rate of 100 bytes per second.

        Peak rate (and burst) information must be present
        if the qosMeterDataSpecification object has the value
        peakData(3). This, in turn, requires that high, medium
        and low conformance actions be specified."

    ::= { qosMeterEntry 5 }

qosMeterPeakBurst OBJECT-TYPE
    SYNTAX         Unsigned32 (0..'ffffffff'h)
    STATUS         current
    DESCRIPTION
        "This object represents the peak burst size (PBS)
        against which associated traffic streams will
        be metered. The CBS specifies the maximum burst size
        that is supported for flows to be considered to be at
        a medium level of conformance. Typically, this value
        represents the maximum number of tokens in a token
        bucket.

        This object specifies flow data in bytes per second
        units such that, for example, a value of 100 equates
        to a committed information rate of 100 bytes per
        second.

        Peak burst (and rate) information must be present
```

```
        if the qosMeterDataSpecification object has the value
        peakData(3). This, in turn, requires that high, medium
        and low conformance actions be specified."

    ::= { qosMeterEntry 6 }

qosMeterHighConfAction OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "This attribute identifies the action that is to be
        initiated for flows that are determined to have a high
        level of conformance with regard to metering criteria
        being applied to the flow.

        Actions must be defined in the qosActionTable prior to
        being referenced by this attribute. A valid value for
        this attribute must always be provided."

    ::= { qosMeterEntry 7 }

qosMeterMedConfAction OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "This attribute identifies the action that is to be
        initiated for flows that are determined to have a medium
        level of conformance with regard to metering criteria
        being applied to the flow.

        Actions must be defined in the qosActionTable prior to
        being referenced by this attribute. A valid value for
        this attribute must be provided if the value of the
        associated qosMeterDataSpecification object is
        committedRate(2) or peakRate(3)."

    ::= { qosMeterEntry 8 }

qosMeterLowConfAction OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "This attribute identifies the action that is to be
        initiated for flows that are determined to have a low
        level of conformance with regard to metering criteria
```

```
        being applied to the flow.

        Actions must be defined in the qosActionTable prior to
        being referenced by this attribute. A valid value for
        this attribute must be provided if the value of the
        associated qosMeterDataSpecification object is
        peakRate(3)."

    ::= { qosMeterEntry 9 }

--
-- The Generic QoS ACL Action Group
--

qosAction OBJECT IDENTIFIER ::= { qosPolicyGenPibClasses 3 }

--
-- The QoS Action Table
--
-- The QoS Action Table describes actions that are associated with
-- specific IP, IEEE 802 and other ACLs through the QoS Target
-- Table.  An action specification may be simple (i.e., a single
-- action) or complex (i.e., multiple actions that are performed
-- in "parallel").
--

qosActionTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF QosActionEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "Contains the current set of configured actions. The actions
        are associated with IP, IEEE 802 and other ACLs and
        interfaces during operation."

    ::= { qosAction 1 }

qosActionEntry OBJECT-TYPE
    SYNTAX         QosActionEntry
    STATUS         current
    DESCRIPTION
        "General action definitions. Each entry specifies an instance
        of the qosAction class which describes (potentially)
```

```
        several distinct action attributes. Each action is taken
        individually regarding the data in question. Several actions
        can be taken for a single frame.

        An instance of this class can not be deleted while it is being
        referenced in a target instance in another class. This
        class may be extended with actions that apply to specific QoS
        policies (e.g., IP, IEEE 802, security) using augmentation."

    INDEX { qosActionId }
    ::= { qosActionTable 1 }

QosActionEntry ::= SEQUENCE {
        qosActionId         PolicyInstanceId,
        qosActionDrop       TruthValue,
        qosActionUpdateDSCP Integer32,
        qosActionMeter      PolicyInstanceId
}

qosActionId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies
        the instance of the QoS Action class. Class instances
        may not be contiguous. Actions are associated with
        Target instances in other classes (e.g., the QoS
        Target class) using this attribute."

    ::= { qosActionEntry 1 }

qosActionDrop OBJECT-TYPE
    SYNTAX         TruthValue
    STATUS         current
    DESCRIPTION
        "This action attribute, when specified, will cause the
        frame being evaluated to be dropped if the value is
        'true(1)'. A value of 'false(2)' indicates that this
        action will not be initiated (i.e., the frame will not
        be dropped) based on this attribute.

        Prior to discarding a packet, other actions that have
        been specified should be performed if they make protocol
        sense. For example, requests for traffic mirroring (if
        such an action is supported by a device) should be
```

```
        honored. However, updating protocol header values will
        typically not be necessary."

    ::= { qosActionEntry 2 }

qosActionUpdateDSCP OBJECT-TYPE
    SYNTAX         Integer32 (-1 | 0..63)
    STATUS         current
    DESCRIPTION
        "This action component, when specified, will cause the
        value contained in the Differentiated Services (DS)
        field of an associated IP datagram to be updated with
        the value of this object.

        A value of -1 indicates that this action component has not
        been set to an appropriate value and should not be used for
        action initiation. The DSCP should remain unchanged."

    ::= { qosActionEntry 3 }

qosActionMeter OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "This action component, when specified, will identify
        another level of metering that should be applied to
        the given flow. This action is only taken if it is
        not in conflict with other specified actions, i.e.,
        qosActionDrop.

        A value of 0 indicates that an additional metering
        component has not been specified. No additional metering
        is thus required."

    ::= { qosActionEntry 4 }

--
-- The QoS Target Table
--
-- The QoS Target Table supports the association of ACLs,
-- interfaces and actions. It allows ACL class instances, as
-- defined in various ACL Defintion classes, to be associated
-- with specific interfaces/flow direction (based on interface
-- role combination and traffic direction) and actions to be
```

```
-- performed based on traffic classification. Furthermore, it
-- allows heterogeneous ACL Definition class instances (e.g.,
-- IP, IEEE 802, security) to be applied to the same interface
-- group in a prescribed order of precedence.
--

qosTargetTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF QosTargetEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "A class that applies a set of ACLs to interfaces specifying,
        for each interface, the precedence order of the ACL with
        respect to other ACLs applied to the same interface and, for
        each ACL, the action to take for a packet that matches a
        permit ACE in that ACL.  Interfaces are specified abstractly
        in terms of interface roles.

        This class may contain ACLs that specify different types
        of traffic classification (e.g., IP ACLs and IEEE 802 ACLs
        defined in their respective definition tables). An ACL is
        identified by its class and instance within that class. An
        ACL association is formed when ACLs apply to the same
        interfaces, as determined by the specified interface role
        and direction. ACL evaluation precedence within an
        association is determined by the precedence attribute."

    INSTALL-ERRORS {
        priPrecedenceConflict(1) -- precedence conflict detected
        }

    ::= { qosAction 2 }

qosTargetEntry OBJECT-TYPE
    SYNTAX         QosTargetEntry
    STATUS         current
    DESCRIPTION
        "An instance of the qosTarget class. Instance creation
        may be prohibited based on the status of certain class
        attributes which must exist prior to class instantiation."

    INDEX { qosTargetId }
    ::= { qosTargetTable 1 }

QosTargetEntry ::= SEQUENCE {
```

```
        qosTargetId                 PolicyInstanceId,
        qosTargetAclId              PolicyInstanceId,
        qosTargetAclType            OBJECT IDENTIFIER,
        qosTargetInterfaceRoles     RoleCombination,
        qosTargetInterfaceDirection INTEGER,
        qosTargetOrder              Unsigned32,
        qosTargetMeter              PolicyInstanceId
}

qosTargetId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies
        the instance of the QoS Target class."

    ::= { qosTargetEntry 1 }

qosTargetAclId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "This attribute identifies the ACL that is associated
        with this target. It identifies (potentially many) ACL
        class instances in a specific ACL Definition table
        where ACLs, and their associated ACEs, are defined.
        For example, instances in the qosIpAclDefinitionTable
        are identified by setting the value of this object
        equal to the qosIpAclDefinitionAclId of the instances
        being targeted. This value, together with the value of
        the corresponding qosTargetAclType attribute,
        uniquely identifies one or more instances of a specific
        ACL Definition class.

        Attempting to specify an unknown ACL class instance will
        result in an appropriate error indication being returned
        to the entity that is attempting to install the conflicting
        entry. For example, a 'priUnknown(2)' error indication is
        returned to the policy server in this situation."

    ::= { qosTargetEntry 2 }

qosTargetAclType OBJECT-TYPE
    SYNTAX         OBJECT IDENTIFIER
    STATUS         current
```

```
    DESCRIPTION
        "The ACL Definition class that is being referenced by
        this instance of the ACL Target class. This policy
        class identifier, together with the corresponding
        qosTargetAclId attribute, uniquely identifies
        instances of a specific ACL Definition class.

        The object identifier value of this attribute must
        exist in the policyPrcSupportTable."

    ::= { qosTargetEntry 3 }

qosTargetInterfaceRoles OBJECT-TYPE
    SYNTAX         RoleCombination
    STATUS         current
    DESCRIPTION
        "The interfaces to which this ACL applies specified
        in terms of a set of roles. The role combination
        specified by this attribute must exist in the
        qosInterfaceTypeTable prior to being association
        with an instance of this class."

    ::= { qosTargetEntry 4 }

qosTargetInterfaceDirection OBJECT-TYPE
    SYNTAX         INTEGER {
                       in(1),
                       out(2)
                   }
    STATUS         current
    DESCRIPTION
        "The direction of packet flow at the interface in
        question to which this ACL applies."

    ::= { qosTargetEntry 5 }

qosTargetOrder OBJECT-TYPE
    SYNTAX         Unsigned32
    STATUS         current
    DESCRIPTION
        "An integer that determines the precedence order of
        this ACL in the list of ACLs applied to interfaces of
        the specified role combination. An ACL with a given
        precedence order is positioned in the list before one
        with a higher-valued precedence order.
```

As an example, consider the following ACL Target association:

Index	IfRoleCombo	IfDirection	AclId	AclType	Order
14	'eth1000+L2+L3'	'in'	8	'802'	1
15	'eth1000+L2+L3'	'in'	3	'802'	2
16	'eth1000+L2+L3'	'in'	12	'IP'	3
17	'eth1000+L2+L3'	'in'	6	'IP'	4
18	'eth1000+L2+L3'	'in'	21	'IP'	5

Five distinct ACL specifications, 3 from an IP ACL Definition class and 2 from an IEEE 802 ACL Definition class, form an Acl Target association (e.g., based on the specified interface role combination and direction attributes) with a prescribed order of evaluation. The AclType and AclId attributes identify the ACL Definition instances in their respective classes.

Precedence values within an association must be unique otherwise instance installation will be prohibited and an error value will be returned."

::= { qosTargetEntry 6 }

```
qosTargetMeter OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "This attribute identifies the meter that is associated
        with this QoS Target instance. Meters are defined
        in the qosMeterTable. The corresponding instance in
        the qosMeter class (i.e., the class instance where
        the qosMeterId is equal to the value of this object)
        must exist prior to being associated with a Target
        entry."

    ::= { qosTargetEntry 7 }

--
-- The IP Classification and Policing Group
--

qosIpQos OBJECT IDENTIFIER ::= { qosPolicyIpPibClasses 1 }

-- The IP ACE Table
```

```
qosIpAceTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF QosIpAceEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "ACE definitions.  A packet has to match all fields in an
        ACE.  Wildcards may be specified for those fields that are
        not relevant."

    ::= { qosIpQos 1 }

qosIpAceEntry OBJECT-TYPE
    SYNTAX         QosIpAceEntry
    STATUS         current
    DESCRIPTION
        "An instance of the qosIpAce class."

    INDEX { qosIpAceId }
    ::= { qosIpAceTable 1 }

QosIpAceEntry ::= SEQUENCE {
        qosIpAceId           PolicyInstanceId,
        qosIpAceDstAddr      IpAddress,
        qosIpAceDstAddrMask  IpAddress,
        qosIpAceSrcAddr      IpAddress,
        qosIpAceSrcAddrMask  IpAddress,
        qosIpAceDscp         Integer32,
        qosIpAceProtocol     INTEGER,
        qosIpAceDstL4PortMin INTEGER,
        qosIpAceDstL4PortMax INTEGER,
        qosIpAceSrcL4PortMin INTEGER,
        qosIpAceSrcL4PortMax INTEGER,
        qosIpAcePermit       TruthValue
}

qosIpAceId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An integer index to uniquely identify this ACE among all the
        ACEs."

    ::= { qosIpAceEntry 1 }

qosIpAceDstAddr OBJECT-TYPE
```

```
    SYNTAX         IpAddress
    STATUS         current
    DESCRIPTION
        "The IP address to match against the packet's destination IP
        address."

    ::= { qosIpAceEntry 2 }

qosIpAceDstAddrMask OBJECT-TYPE
    SYNTAX         IpAddress
    STATUS         current
    DESCRIPTION
        "A mask for the matching of the destination IP address.
        A zero bit in the mask means that the corresponding bit in
        the address always matches."

    ::= { qosIpAceEntry 3 }

qosIpAceSrcAddr OBJECT-TYPE
    SYNTAX         IpAddress
    STATUS         current
    DESCRIPTION
        "The IP address to match against the packet's source IP
        address."

    ::= { qosIpAceEntry 4 }

qosIpAceSrcAddrMask OBJECT-TYPE
    SYNTAX         IpAddress
    STATUS         current
    DESCRIPTION
        "A mask for the matching of the source IP address."

    ::= { qosIpAceEntry 5 }

qosIpAceDscp OBJECT-TYPE
    SYNTAX         Integer32 (-1 | 0..63)
    STATUS         current
    DESCRIPTION
        "The value that the DSCP in the packet can have and
        match this ACE. A value of -1 indicates that a specific
        DSCP value has not been defined and thus all DSCP values
        are considered a match."

    ::= { qosIpAceEntry 6 }
```

```
qosIpAceProtocol OBJECT-TYPE
    SYNTAX         INTEGER (0..255)
    STATUS         current
    DESCRIPTION
        "The IP protocol to match against the packet's protocol.
        A value of zero means match all."

    ::= { qosIpAceEntry 7 }

qosIpAceDstL4PortMin OBJECT-TYPE
    SYNTAX         INTEGER (0..65535)
    STATUS         current
    DESCRIPTION
        "The minimum value that the packet's layer 4 destination
        port number can have and match this ACE."

    ::= { qosIpAceEntry 8 }

qosIpAceDstL4PortMax OBJECT-TYPE
    SYNTAX         INTEGER (0..65535)
    STATUS         current
    DESCRIPTION
        "The maximum value that the packet's layer 4 destination
        port number can have and match this ACE. This value must be
        equal to or greater that the value specified for this ACE in
        qosIpAceDstL4PortMin."

    ::= { qosIpAceEntry 9 }

qosIpAceSrcL4PortMin OBJECT-TYPE
    SYNTAX         INTEGER (0..65535)
    STATUS         current
    DESCRIPTION
        "The minimum value that the packet's layer 4 source port
        number can have and match this ACE."

    ::= { qosIpAceEntry 10 }

qosIpAceSrcL4PortMax OBJECT-TYPE
    SYNTAX         INTEGER (0..65535)
    STATUS         current
    DESCRIPTION
        "The maximum value that the packet's layer 4 source port
        number can have and match this ACE.  This value must be equal
        to or greater that the value specified for this ACE in
```

```
        qosIpAceSrcL4PortMin."

    ::= { qosIpAceEntry 11 }

qosIpAcePermit OBJECT-TYPE
    SYNTAX         TruthValue
    STATUS         current
    DESCRIPTION
        "If the packet matches this ACE and the value of this
        attribute is true, then the matching process terminates
        and the QoS associated with this ACE (indirectly through
        the ACL) is applied to the packet.  If the value of this
        attribute is false, then no more ACEs in this ACL are
        compared to this packet and matching continues with the
        first ACE of the next ACL."

    ::= { qosIpAceEntry 12 }

--
-- The IP ACL Definition Table
--

qosIpAclDefinitionTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF QosIpAclDefinitionEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "A class that defines a set of ACLs each being an ordered list
        of ACEs.  Each instance of this class identifies one ACE of
        an ACL and the precedence order of that ACE with respect to
        other ACEs in the same ACL."

    INSTALL-ERRORS {
        priPrecedenceConflict(1) -- precedence conflict detected
        }

    ::= { qosIpQos 2 }

qosIpAclDefinitionEntry OBJECT-TYPE
    SYNTAX         QosIpAclDefinitionEntry
    STATUS         current
    DESCRIPTION
        "An instance of the qosIpAclDefinition class."

    INDEX { qosIpAclDefinitionId }
```

```
    ::= { qosIpAclDefinitionTable 1 }

QosIpAclDefinitionEntry ::= SEQUENCE {
        qosIpAclDefinitionId       PolicyInstanceId,
        qosIpAclDefinitionAclId    PolicyInstanceId,
        qosIpAclDefinitionAceId    PolicyInstanceId,
        qosIpAclDefinitionAceOrder Unsigned32
}

qosIpAclDefinitionId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "Unique index of this policy rule instance."

    ::= { qosIpAclDefinitionEntry 1 }

qosIpAclDefinitionAclId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An ID for this ACL.  There will be one instance of
        the class qosIpAclDefinition with this ID for each ACE in
        the ACL per role combination."

    ::= { qosIpAclDefinitionEntry 2 }

qosIpAclDefinitionAceId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "This attribute specifies the ACE in the qosIpAceTable that
        is in the ACL specified by qosIpAclDefinitionAclId at the
        position specified by qosIpAceOrder.

        Attempting to specify an unknown class instance will result
        in an appropriate error indication being returned to the
        entity that is attempting to install the conflicting entry.
        For example, a 'priUnknown(2)' error indication is returned
        to the policy server in this situation."

    ::= { qosIpAclDefinitionEntry 3 }

qosIpAclDefinitionAceOrder OBJECT-TYPE
    SYNTAX         Unsigned32
```

```
    STATUS         current
    DESCRIPTION
        "The precedence order of this ACE.  The precedence order
        determines the position of this ACE in the ACL.  An ACE with
        a given precedence order is positioned in the access control
        list before one with a higher-valued precedence order.

        Precedence values within a group must be unique otherwise
        instance installation will be prohibited and an error
        value will be returned."

    ::= { qosIpAclDefinitionEntry 4 }

END
```

7.3. The QoS IEEE 802 PIB

```
QOS-POLICY-802-PIB PIB-DEFINITIONS ::= BEGIN

IMPORTS
    Unsigned32, Integer32,
    MODULE-IDENTITY, OBJECT-TYPE
            FROM SNMPv2-SMI
    TruthValue, PhysAddress,
    TEXTUAL-CONVENTION
            FROM SNMPv2-TC
    RoleCombination, PolicyInstanceId
            FROM POLICY-FRAMEWORK-PIB
    Dscp
            FROM QOS-POLICY-IP-PIB;

qosPolicy802Pib  MODULE-IDENTITY
    LAST-UPDATED "9906241800Z"
    ORGANIZATION "IETF RAP WG"
    CONTACT-INFO "
                  Michael Fine
                  Cisco Systems, Inc.
                  170 West Tasman Drive
                  San Jose, CA  95134-1706 USA
                  Phone: +1 408 527 8218
                  Email: mfine@cisco.com

                  Keith McCloghrie
                  Cisco Systems, Inc.
                  170 West Tasman Drive,
                  San Jose, CA 95134-1706 USA
                  Phone: +1 408 526 5260
                  Email: kzm@cisco.com

                  John Seligson
                  Nortel Networks, Inc.
                  4401 Great America Parkway
                  Santa Clara, CA 95054 USA
                  Phone: +1 408 495 2992
                  Email: jseligso@nortelnetworks.com"
    DESCRIPTION
            "The PIB module containing an initial set of policy
             rule classes that describe the quality of service
             (QoS) policies supported by devices for IEEE 802-
             based traffic."
```

```
    ::= { tbd }

qosPolicy802PibClasses OBJECT IDENTIFIER ::= { qosPolicy802Pib 1 }

--
-- Textual Conventions
--

--
-- IEEE 802 CoS
--

QosIeee802Cos ::= TEXTUAL-CONVENTION
    STATUS       current
    DESCRIPTION
        "An integer that is in the range of the IEEE 802 CoS
        values. This corresponds to the 802.1p priority values."

    SYNTAX INTEGER (0..7)

--
-- General configuration information for the entire domain
--

qos802DomainConfig OBJECT IDENTIFIER ::= { qosPolicy802PibClasses 1 }

--
-- Differentiated Services Code Point Mapping Table
--
-- Supports the mapping of DSCP values to IEEE CoS values.
--

qos802DscpMappingTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Qos802DscpMappingEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "Maps each DSCP to an QosIeee802Cos.  When configured
        for the first time, all 64 entries of the table must
        be specified. Thereafter, instances may be modified but
        not deleted unless all instances are deleted."

    INSTALL-ERRORS {
        priInstNotComplete(1)    -- required instances not created
        }
```

```
    ::= { qos802DomainConfig 1 }

qos802DscpMappingEntry OBJECT-TYPE
    SYNTAX         Qos802DscpMappingEntry
    STATUS         current
    DESCRIPTION
        "An instance of the qos802DscpMapping class. A total of 64
        class instances are constantly maintained after initial device
        configuration."

    INDEX { qos802DscpMappingId }
    ::= { qos802DscpMappingTable 1 }

Qos802DscpMappingEntry ::= SEQUENCE {
        qos802DscpMappingId           PolicyInstanceId,
        qos802DscpMappingDscp         Dscp,
        qos802DscpMapping802Cos       QosIeee802Cos
}

qos802DscpMappingId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "A unique ID for this policy rule instance."

    ::= { qos802DscpMappingEntry 1 }

qos802DscpMappingDscp OBJECT-TYPE
    SYNTAX         Dscp
    STATUS         current
    DESCRIPTION
        "The DSCP class instance attribute that is used to
        determine the appropriate layer 2 CoS mappings. DSCP
        values 0 through 63 (inclusive) are maintained in
        the table."

    ::= { qos802DscpMappingEntry 2 }

qos802DscpMapping802Cos OBJECT-TYPE
    SYNTAX         QosIeee802Cos
    STATUS         current
    DESCRIPTION
        "The IEEE 802 CoS value to use when mapping the DSCP
        value specified by the qos802DscpMappingDscp attribute
        to a IEEE 802 CoS."
```

```
    ::= { qos802DscpMappingEntry 3 }

--
-- Layer 2 CoS-to-DSCP Mapping Table
--
-- Supports the mapping of IEEE CoS values to DSCP values
-- for generic QoS traffic classification
--

qos802CosToDscpTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Qos802CosToDscpEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "Maps each of eight layer 2 CoS values to a DSCP.  When
        configured for the first time, all 8 entries of the table
        must be specified. Thereafter, instances may be modified
        but not deleted unless all instances are deleted."

    INSTALL-ERRORS {
        priInstNotComplete(1)    -- required instances not created
        }

    ::= { qos802DomainConfig 2 }

qos802CosToDscpEntry OBJECT-TYPE
    SYNTAX         Qos802CosToDscpEntry
    STATUS         current
    DESCRIPTION
        "An instance of the qosCosToDscp class. A total of 8
        class instances are constantly maintained after initial
        device configuration."

    INDEX { qos802CosToDscpId }
    ::= { qos802CosToDscpTable 1 }

Qos802CosToDscpEntry ::= SEQUENCE {
        qos802CosToDscpId   PolicyInstanceId,
        qos802CosToDscpCos  QosIeee802Cos,
        qos802CosToDscpDscp Dscp
}

qos802CosToDscpId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
```

```
    DESCRIPTION
        "A unique index for this policy rule instance."

    ::= { qos802CosToDscpEntry 1 }

qos802CosToDscpCos OBJECT-TYPE
    SYNTAX         QosIeee802Cos
    STATUS         current
    DESCRIPTION
        "The layer 2 CoS class instance attribute that is used to
        determine the appropriate DSCP mappings. CoS values 0
        through 7 (inclusive) are maintained in the table."

    ::= { qos802CosToDscpEntry 2 }

qos802CosToDscpDscp OBJECT-TYPE
    SYNTAX         Dscp
    STATUS         current
    DESCRIPTION
        "The DSCP value to use when mapping the layer 2 CoS value
        specified by the qosCosToDscp attribute to a DSCP."

    ::= { qos802CosToDscpEntry 3 }

--
-- The IEEE 802 Classification and Policing Group
--

qos802Qos OBJECT IDENTIFIER ::= { qosPolicy802PibClasses 2 }

--
-- The IEEE 802 ACE Table
--
-- The IEEE 802 ACE Table supports the specification of IEEE
-- 802-based (e.g., 802.3) information that is used to perform
-- traffic classification.
--

qos802AceTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Qos802AceEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "IEEE 802-based ACE definitions. A class that contains
        attributes of IEEE 802 (e.g., 802.3) traffic that form
```

```
        an association that is used to perform traffic
        classification."

    ::= { qos802Qos 1 }

qos802AceEntry OBJECT-TYPE
    SYNTAX         Qos802AceEntry
    STATUS         current
    DESCRIPTION
        "IEEE 802-based ACE definitions.  An entry specifies
        (potentially) several distinct matching components. Each
        component is tested against the data in a frame
        individually. An overall match occurs when all of the
        individual components match the data they are compared
        against in the frame being processed. A failure of any
        one test causes the overall match to fail.

        Wildcards may be specified for those fields that are not
        relevant."

    INDEX { qos802AceId }
    ::= { qos802AceTable 1 }

Qos802AceEntry ::= SEQUENCE {
        qos802AceId              PolicyInstanceId,
        qos802AceDstAddr         PhysAddress,
        qos802AceDstAddrMask     PhysAddress,
        qos802AceSrcAddr         PhysAddress,
        qos802AceSrcAddrMask     PhysAddress,
        qos802AceVlanId          Integer32,
        qos802AceVlanTagRequired INTEGER,
        qos802AceEtherType       Integer32,
        qos802AceUserPriority    BITS,
        qos802AcePermit          TruthValue
}

qos802AceId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies this
        802 ACE among all of the 802 ACEs. Note that this identifier
        is used in instances of the qos802Acl class to associate a
        802 ACE with a 802 ACL. An active ACE/ACL association
        prohibits the deletion of the 802 ACE until the ACE/ACL
```

```
        association is terminated. Class instances may not be
        contiguous."

    ::= { qos802AceEntry 1 }

qos802AceDstAddr OBJECT-TYPE
    SYNTAX         PhysAddress
    STATUS         current
    DESCRIPTION
        "The 802 address against which the 802 DA of incoming traffic
        streams will be compared. Frames whose 802 DA matches the
        physical address specified by this object, taking into account
        address wildcarding as specified by the qos802AceDstAddrMask
        object, are potentially subject to the processing guidelines
        that are associated with this entry through the related
        action class."

    ::= { qos802AceEntry 2 }

qos802AceDstAddrMask OBJECT-TYPE
    SYNTAX         PhysAddress
    STATUS         current
    DESCRIPTION
        "This object specifies the bits in a 802 destination address
        that should be considered when performing a 802 DA comparison
        against the address specified in the qos802AceDstAddr object.

        The value of this object represents a mask that is logically
        and'ed with the 802 DA in received frames to derive the value
        to be compared against the qos802AceDstAddr address. A zero
        bit in the mask thus means that the corresponding bit in the
        address always matches. The qos802AceDstAddr value must also
        be masked using this value prior to any comparisons.

        The length of this object in octets must equal the length in
        octets of the qos802AceDstAddr. Note that a mask with no bits
        set (i.e., all zeroes) effectively wildcards the
        qos802AceDstAddr object."

    ::= { qos802AceEntry 3 }

qos802AceSrcAddr OBJECT-TYPE
    SYNTAX         PhysAddress
    STATUS         current
    DESCRIPTION
```

```
        "The 802 MAC address against which the 802 MAC SA of incoming
        traffic streams will be compared. Frames whose 802 MAC SA
        matches the physical address specified by this object,
        taking into account address wildcarding as specified by the
        qos802AceSrcAddrMask object, are potentially subject to the
        processing guidelines that are associated with this entry
        through the related action class."

    ::= { qos802AceEntry 4 }

qos802AceSrcAddrMask OBJECT-TYPE
    SYNTAX         PhysAddress
    STATUS         current
    DESCRIPTION
        "This object specifies the bits in a 802 MAC source address
        that should be considered when performing a 802 MAC SA
        comparison against the address specified in the
        qos802AceSrcAddr object.

        The value of this object represents a mask that is logically
        and'ed with the 802 MAC SA in received frames to derive the
        value to be compared against the qos802AceSrcAddr address. A
        zero bit in the mask thus means that the corresponding bit
        in the address always matches. The qos802AceSrcAddr value
        must also be masked using this value prior to any
        comparisons.

        The length of this object in octets must equal the length in
        octets of the qos802AceSrcAddr. Note that a mask with no bits
        set (i.e., all zeroes) effectively wildcards the
        qos802AceSrcAddr object."

    ::= { qos802AceEntry 5 }

qos802AceVlanId OBJECT-TYPE
    SYNTAX         Integer32 (-1 | 1..4094)
    STATUS         current
    DESCRIPTION
        "The VLAN ID (VID) that uniquely identifies a VLAN
        within the device. This VLAN may be known or unknown
        (i.e., traffic associated with this VID has not yet
        been seen by the device) at the time this entry
        is instantiated.

        Setting the qos802AceVlanId object to -1 indicates that
```

```
        VLAN data should not be considered during traffic
        classification."

    ::= { qos802AceEntry 6 }

qos802AceVlanTagRequired OBJECT-TYPE
    SYNTAX         INTEGER {
                       taggedOnly(1),
                       priorityTaggedPlus(2),
                       untaggedOnly(3),
                       ignoreTag(4)
                   }
    STATUS         current
    DESCRIPTION
        "This object indicates whether the presence of an
        IEEE 802.1Q VLAN tag in data link layer frames must
        be considered when determining if a given frame
        matches this 802 ACE entry.

        A value of 'taggedOnly(1)' means that only frames
        containing a VLAN tag with a non-Null VID (i.e., a
        VID in the range 1..4094) will be considered a match.

        A value of 'priorityTaggedPlus(2)' means that only
        frames containing a VLAN tag, regardless of the value
        of the VID, will be considered a match.

        A value of 'untaggedOnly(3)' indicates that only
        untagged frames will match this filter component.

        The presence of a VLAN tag is not taken into
        consideration in terms of a match if the value is
        'ignoreTag(4)'."

    ::= { qos802AceEntry 7 }

qos802AceEtherType OBJECT-TYPE
    SYNTAX         Integer32 (-1 | 0..'ffff'h)
    STATUS         current
    DESCRIPTION
        "This object specifies the value that will be compared
        against the value contained in the EtherType field of an
        IEEE 802 frame. Example settings would include 'IP'
        (0x0800), 'ARP' (0x0806) and 'IPX' (0x8137).
```

```
        Setting the qos802AceEtherTypeMin object to -1 indicates
        that EtherType data should not be considered during traffic
        classification.

        Note that the position of the EtherType field depends on
        the underlying frame format. For Ethernet-II encapsulation,
        the EtherType field follows the 802 MAC source address. For
        802.2 LLC/SNAP encapsulation, the EtherType value follows the
        Organization Code field in the 802.2 SNAP header. The value
        that is tested with regard to this filter component therefore
        depends on the data link layer frame format being used. If
        this 802 ACE component is active when there is no EtherType
        field in a frame (e.g., 802.2 LLC), a match is implied."

    ::= { qos802AceEntry 8 }

qos802AceUserPriority OBJECT-TYPE
    SYNTAX         BITS {
                        matchPriority0(0),
                        matchPriority1(1),
                        matchPriority2(2),
                        matchPriority3(3),
                        matchPriority4(4),
                        matchPriority5(5),
                        matchPriority6(6),
                        matchPriority7(7)
                   }
    STATUS         current
    DESCRIPTION
        "The set of values, representing the potential range
        of user priority values, against which the value contained
        in the user priority field of a tagged 802.1 frame is
        compared. A test for equality is performed when determining
        if a match exists between the data in a data link layer
        frame and the value of this 802 ACE component. Multiple
        values may be set at one time such that potentially several
        different user priority values may match this 802 ACE
        component.

        Setting all of the bits that are associated with this
        object causes all user priority values to match this
        attribute. This essentially makes any comparisons
        with regard to user priority values unnecessary. Untagged
        frames are treated as an implicit match."
```

```
    ::= { qos802AceEntry 9 }

qos802AcePermit OBJECT-TYPE
    SYNTAX         TruthValue
    STATUS         current
    DESCRIPTION
        "If the frame matches this ACE and the value of this
        attribute is true, then the matching process terminates
        and the QoS associated with this 802-based ACE (indirectly
        through the 802 ACL) is applied to the packet.  If the
        value of this attribute is false, then no more 802 ACEs in
        this 802 ACL are compared to this packet and matching
        continues with the first 802-based ACE of the next 802 ACL."

    ::= { qos802AceEntry 10 }

--
-- The IEEE 802 ACL Definition Table
--
-- The IEEE 802 ACL Definition Table supports the association of
-- distinct IEEE 802-based (e.g., 802.3) traffic classification
-- specifications into an ordered list.
--

qos802AclDefinitionTable OBJECT-TYPE
    SYNTAX         SEQUENCE OF Qos802AclDefinitionEntry
    POLICY-ACCESS  install
    STATUS         current
    DESCRIPTION
        "IEEE 802-based ACL definitions. A class that defines a
        set of 802 ACLs, each of which is comprised of an ordered
        list of 802 ACEs."

    INSTALL-ERRORS {
        priPrecedenceConflict(1) -- precedence conflict detected
        }

    ::= { qos802Qos 2 }

qos802AclDefinitionEntry OBJECT-TYPE
    SYNTAX         Qos802AclDefinitionEntry
    STATUS         current
    DESCRIPTION
        "IEEE 802-based ACL definitions. An entry specifies an
        instance of this class that associates an 802 ACE with
```

```
        a given 802 ACL. The evaluation order of distinct 802
        ACEs that are associated with a specific 802 ACL is
        specified as well."

    INDEX { qos802AclDefinitionId }
    ::= { qos802AclDefinitionTable 1 }

Qos802AclDefinitionEntry ::= SEQUENCE {
        qos802AclDefinitionId       PolicyInstanceId,
        qos802AclDefinitionAclId    PolicyInstanceId,
        qos802AclDefinitionAceId    PolicyInstanceId,
        qos802AclDefinitionAceOrder Unsigned32
}

qos802AclDefinitionId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies this
        802 ACE / 802 ACL association."

    ::= { qos802AclDefinitionEntry 1 }

qos802AclDefinitionAclId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "An index for this 802 ACL. Each 802 ACL in the device is
        assigned a unique integer index. There will (potentially) be
        multiple instances of the qos802AclDefinition class with this
        identifier, one for each 802 ACE that is associated with the
        specified 802 ACL.

        For example, assume that 2 802 ACLs, each comprised of 4 802
        ACEs, have been installed. The instances of this class may
        appear as follows:

                 Index   AclId   AceId   AceOrder
                   10      6       4        1
                   11      6       5        2
                   12      6       9        23
                   13      6       11       24
                   65      18      5        8
                   66      18      9        12
                   67      18      13       15
```

```
                    70      18      14       16

        Note that this identifier is used in instances of the
        qosAclTarget class to associate an 802 ACL with an interface
        set and action. An active ACL Target association prohibits
        the deletion of all of the qos802AclDefinition instances
        with a given qos802AclDefinitionAclId (i.e., at least one
        entry for the specific qos802AclDefinitionAclId must be
        present in this table) until the ACL Target association is
        terminated."

    ::= { qos802AclDefinitionEntry 2 }

qos802AclDefinitionAceId OBJECT-TYPE
    SYNTAX         PolicyInstanceId
    STATUS         current
    DESCRIPTION
        "This attribute identifies the 802 ACE in the qos802AceTable
        that is associated with the 802 ACL specified by
        qos802AclDefinitionAclId object. The corresponding instance
        in the qos802Ace class must exist prior to being associated
        with a 802 ACL.

        Attempting to specify an unknown class instance will result
        in an appropriate error indication being returned to the
        entity that is attempting to install the conflicting entry.
        For example, a 'priUnknown(2)' error indication is returned
        to the policy server in this situation."

    ::= { qos802AclDefinitionEntry 3 }

qos802AclDefinitionAceOrder OBJECT-TYPE
    SYNTAX         Unsigned32
    STATUS         current
    DESCRIPTION
        "The precedence of the 802 ACE, identified via the
        qos802AclDefinitionAceId object, with regard to evaluation
        order. The precedence determines the order of evaluation of
        this ACE in relation to related 802 ACEs that are associated
        with an ACL. An ACE with a given precedence order in the
        access control list is evaluated before one with a higher-
        valued precedence order.

        Precedence values within a group must be unique otherwise
        instance installation will be prohibited and an error
```

```
        value will be returned.

        Note that qos802AclDefinitionAceOrder values within a given
        ACL need not be contiguous."

    ::= { qos802AclDefinitionEntry 4 }

END
```

8. Security Considerations

The information contained in a PIB when transported by the COPS protocol [COPS-PR] may be sensitive, and its function of provisioning a PEP requires that only authorized communication take place. The use of IPSEC between PDP and PEP, as described in [COPS], provides the necessary protection against these threats.

9. Intellectual Property Considerations

The IETF is being notified of intellectual property rights claimed in regard to some or all of the specification contained in this document. For more information consult the online list of claimed rights.

10. Authors' Addresses

Michael Fine
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706 USA
Phone: +1 408 527 8218
Email: mfine@cisco.com

Keith McCloghrie
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706 USA
Phone: +1 408 526 5260
Email: kzm@cisco.com

John Seligson
Nortel Networks, Inc.
4401 Great America Parkway
Santa Clara, CA 95054 USA
Phone: +1 408 495 2992
Email: jseligso@nortelnetworks.com

Kwok Ho Chan
Nortel Networks, Inc.
600 Technology Park Drive
Billerica, MA 01821 USA
Phone: +1 978 288 8175
Email: khchan@nortelnetworks.com

Scott Hahn
Intel
2111 NE 25th Avenue
Hillsboro, OR 97124 USA
Phone: +1 503 264 8231
Email: scott.hahn@intel.com

Andrew Smith
Extreme Networks
10460 Bandley Drive
Cupertino CA 95014 USA
Phone: +1 408 342 0999
Email: andrew@extremenetworks.com

11. References

[COPS] J. Boyle, R. Cohen, D. Durham, S. Herzog, R. Rajan, A. Sastry, "The COPS (Common Open Policy Service) Protocol" Internet-Draft, draft-ietf-rap-cops-07.txt, August 1999.

[COPS-PR] F. Reichmeyer, S. Herzog, K. Chan, D. Durham, R. Yavatkar, S. Gai, K. McCloghrie, A. Smith, "COPS Usage for Policy Provisioning," draft-ietf-rap-cops-pr-01.txt, June 1999.

[POLICY] M. Stevens, W. Weiss H. Mahon, B. Moore, J. Strassner, G. Waters, A. Westerinen, J. Wheeler, "Policy Framework", draft-ietf-policy-framework-00.txt, September 1999.

[RAP-FRAMEWORK] R. Yavatkar, D. Pendarakis, "A Framework for Policy-based Admission Control", draft-ietf-rap-framework-03.txt, April 1999.

[SNMP-SMI] K. McCloghrie, D. Perkins, J. Schoenwaelder, J. Case, M. Rose and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999.

[MODEL] Y. Bernet, A. Smith, S. Blake, "A Conceptual Model for Diffserv Routers", draft-ietf-diffserv-model-00.txt, June 1999.

Table of Contents

1 Glossary 2
2 Introduction 2
3 General PIB Concepts 2
3.1 Roles 2
3.2 Reporting of Device Capabilities 3
4 DiffServ PIB Concepts 4
4.1 Filters, Filter Groups and Classifiers 4
4.2 Applying QoS Policy Using Targets 4
4.3 Queue Modeling with Queue Sets 5
4.4 IP Mapping to and from Layer 2 6
5 Summary of the PIB Modules 7
6 PIB Operational Overview 8
7 PIB Definitions 11
7.1 The Policy Framework PIB Module 11
7.2 The QoS IP PIB 18
7.3 The QoS IEEE 802 PIB 48
8 Security Considerations 62
9 Intellectual Property Considerations 62
10 Authors' Addresses 62
11 References 63