

Network Working Group
Request for Comments: 3318
Category: Informational

R. Sahita, Ed.
S. Hahn
Intel Labs
K. Chan
Nortel Networks
K. McCloghrie
Cisco Systems
March 2003

Framework Policy Information Base

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2003). All Rights Reserved.

Abstract

This document defines a set of PProvisioning Classes (PRCs) and textual conventions that are common to all clients that provision policy using Common Open Policy Service (COPS) protocol for Provisioning.

Structure of Policy Provisioning Information (SPPI) describes a structure for specifying policy information that can then be transmitted to a network device for the purpose of configuring policy at that device. The model underlying this structure is one of well-defined (PRCs) and instances of these classes (PRIs) residing in a virtual information store called the Policy Information Base (PIB).

One way to provision policy is by means of the (COPS) protocol with the extensions for provisioning. This protocol supports multiple clients, each of which may provision policy for a specific policy domain such as QoS, virtual private networks, or security.

As described in COPS usage for Policy Provisioning (COPS-PR), each client supports a non-overlapping and independent set of PIB modules. However, some PProvisioning Classes are common to all subject-categories (client-types) and need to be present in each.

Table of Contents

Conventions used in this document.....	2
1. Glossary.....	2
2. General PIB Concepts.....	3
2.1. Roles.....	3
2.1.1. An Example.....	5
2.2. Management of Role-Combinations from the PDP.....	6
2.3. Updating a Request State.....	7
2.3.1 Full Request State.....	8
2.3.2 Installing PRIs in a Request.....	8
2.3.3 Updating PRIs in a Request.....	8
2.3.4 Removing PRIs from a Request.....	9
2.3.5 Removing EXTENDED, AUGMENTED PRIs.....	9
2.3.6 Error Handling in Request updates.....	9
2.4. Multiple PIB Instances.....	10
2.5. Reporting and Configuring of Device Capabilities.....	11
2.6. Reporting of Device Limitations.....	12
3. The Framework TC PIB module.....	12
4. Summary of the Framework PIB.....	17
4.1. Base PIB classes Group.....	17
4.2. Device Capabilities group.....	19
4.3. Classifier group.....	20
4.4. Marker group.....	20
5. The Framework PIB Module.....	21
6. Security Considerations.....	66
7. IANA Considerations.....	67
8. References.....	67
8.1 Normative References.....	67
8.2 Informative References.....	68
9. Acknowledgments.....	68
10. Authors' Addresses.....	69
11. Full Copyright Statement.....	70

Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1. Glossary

PRC	PRovisioning Class. A type of policy data. See [POLTERM].
PRI	PRovisioning Instance. An instance of a PRC. See [POLTERM].
PIB	Policy Information Base. The database of policy information. See [POLTERM]
PDP	Policy Decision Point. See [RAP-FRAMEWORK].
PEP	Policy Enforcement Point. See [RAP-FRAMEWORK].

2. General PIB Concepts

2.1. 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 of all devices in the network, 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. Provisioning classes have an attribute called a "RoleCombination" which is a lexicographically ordered set of roles. Instances of a given PProvisioning Class are applied to an interface if and only if the set of roles in the role combination matches 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. That is, roles provide a level of indirection to the application of a set of policies to specific interfaces. This separates the policy definition from device implementation specific interface identification. Furthermore, if the same policy is being applied to several interfaces, that policy needs to 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 a unique policy for each interface, the administrator can configure each interface with a unique role.

The PEP sends all its Capability Set Names, Role Combinations, Policy Controlled Interfaces, and their relationships to the PDP in the first COPS request (REQ) message for a handle, and whenever any updates or deletes occur. The PDP can install new instances or change existing instances of these PRIs. This operation can also occur in subsequent request messages generated in response to COPS state synchronization (SSQ) requests and local configuration changes.

The comparing of roles (or role combinations) is case sensitive.

By convention, when formatting the role-combination for exchange within a protocol message, within a PIB object's value, or as a printed value, the set is formatted in lexicographical order of the role's ASCII values; that is, the role that is first is formatted first. For example, "a+b" and "b+a" are NOT different role-combinations; rather, they are different formatting of the same role-combination, and hence for this example:

- "a+b" is the valid formatting of that role-combination,
- "b+a" is an invalid formatting of that role-combination.

The role-combination of interfaces to which no roles have been assigned is known as the "null" role-combination. (Note the deliberate use of lower-case letters for "null" so that it avoids confusion with the ASCII NULL character that has a value of zero but a length of one.)

In an "install" or an "install-notify" class, the wildcard role-combination "*" can be used. In addition to providing for interface-specific roles, it also allows for other optimizations in reducing the number of role-combinations for which a policy has to be specified. For example:

Suppose we have three interfaces:

```
Roles A, B and R1 are assigned to interface I1
Roles A, B and R2 are assigned to interface I2
Roles A, B and R3 are assigned to interface I3
```

Then, a PRI of a fictional IfDscpAssignTable that has the following values for its attributes:

```
ifDscpAssignPrid      = 1
ifDscpAssignRoles     = "+A+B"
ifDscpAssignName      = "4queues"
ifDscpAssignDscpMap   = 1
```

will apply to all three interfaces, because "*" matches with R1, R2 and R3. The policies can be assigned to an interface due to more than one wild-carded role combo matching a given interface's role combo string. The PDP should attempt to resolve conflicts between policies before sending policies to the PEP. In the situation where the PDP sends multiple policies to a PEP and they do conflict, either because of an error by the PDP or because of a device specific conflict, the PEP MUST reject the installation of the conflicting policies and return an error.

Formally,

- The wildcard Role is denoted by "*",
- The "*" Role is not allowed to be defined as part of the role-combination of an interface as notified by the PEP to the PDP; it is only allowed in policies installed/deleted via COPS-PR from the PDP to the PEP.
- For a policy to apply to an interface when the policy's role-combination is "+a+b", the interface's role-combination:
 - Must include "a" and "b", and
 - Can include zero or more other roles.
- The wildcard character "*" is listed before the other roles as "*" is lexicographically before "a"; however, the wildcard matches any zero or more roles, irrespective of lexicographical order. For example: "+b+e+g" would match "a+b+c+e+f+g".

Note that the characters "+" and "*" MUST not be used in an interface Role. The Framework Role PIB module in section 4 of this document contains the Role and RoleCombination Textual Conventions.

2.1.1. An Example

The functioning of roles might be best understood by an example. Suppose I have a device with three interfaces, with roles as follows:

```
IF1: "finance"
IF2: "finance"
IF3: "manager"
```

Suppose, I also have a PDP with two policies:

```
P1: Packets from finance department (role "finance") get DSCP 5
P2: Packets from managers (role "manager") get DSCP 6
```

To obtain policy, the PEP reports to the PDP that it has some interfaces with role combination "finance" and some with role combination "manager". In response, the PDP downloads policy P1 associated with role combination "finance" and downloads a second policy P2 associated with role combination "manager".

Now suppose the finance person attached to IF2 is promoted to manager and so the system administrator adds the role "manager" to IF2. The PEP now reports to the PDP that it has three role combinations: some interfaces with role combination "finance", some with role combination "manager" and some with role combination "finance+manager". In response, the PDP downloads an additional third policy associated with the new role combination "finance+manager".

How the PDP determines the policy for this new role combination is entirely the responsibility of the PDP. It could do so algorithmically or by rule. For example, there might be a rule that specifies that manager policy takes preference over department policy. Or there might be a third policy installed in the PDP as follows:

```
P3: Packets from finance managers (role "finance" and role
    "manager") get DSCP 7
```

The point here is that the PDP is required to determine what policy applies to this new role combination and to download a third policy to the PEP for the role combination "finance+manager", even if that policy is the same as one already downloaded. The PEP is not required (or allowed) to construct policy for new role combinations from existing policy.

2.2. Management of Role-Combinations from the PDP

The PEP notifies the PDP of the Role-Combination assigned to each interface and capability set name in a COPS configuration request (instances of the `frwkIfRoleComboTable`). The first request sent to the PDP must be a 'full state' request. A 'full state' request for a PEP includes notify and install-notify table PRIs for the PEP which must be interpreted as the complete state of the PEP and must not be interpreted as updates to any previous set of PRIs sent in a previous message. Any previous PRIs from the PEP should be discarded when a 'full state' request is received for the particular request handle. A request is specified as a 'full state' request by setting the `frwkPibIncarnationFullState` attribute in the `frwkPibIncarnation` PRI sent in the request.

All existing `frwkIfRoleCombo` instances must be sent to the PDP in the first configuration request for a request handle. If the Role-Combinations are not assigned specific values, default ('null') Role-Combinations must be sent to the PDP for all `ifIndices` active on the PEP and updates must be sent every time the `IfIndices` are updated. The PEP may notify the PDP of the Capability sets (if any) via the `frwkCapabilitySetTable`. If the PEP does not need to notify the PDP of capability sets, it must set the capability set name in the `frwkIfRoleComboTable` instances to a zero length string.

In response to this configuration request, if applicable, the PDP may send policies for the PEP in a solicited decision or must send a null decision. The PEP must then send a solicited report message for the decision.

At any later time, the PDP can update the Role-Combinations assigned to a specific interface, identified by IfIndex, or for an aggregate, identified by the capability set name, via an unsolicited decision to the PEP on any open request handle. The PDP does this by sending updated PRIs for the frwkIfRoleComboTable.

When the Interface Role Combination associations are updated by the PDP, the PEP SHOULD send updated 'full state' requests for all open contexts. A context is an instantiation of the PIB module(s) namespace identified by a unique COPS handle for a particular COPS client type. This is true even if the PEP's request state changes due to an internal event or if the state is changed by the PDP. If the role-combination updates were sent by the PDP, the PEP SHOULD send these updated requests only if it can process the unsolicited decision containing the frwkIfRoleCombo PRIs successfully, and it MUST do so after sending the success report for the unsolicited decision. If the PEP failed to process the decision (i.e., the frwkIfRoleCombo PRIs), it MUST only send a failure report to the PDP.

On the other hand, the PDP must not expect to receive the updated requests with the revised role-combination information until after it receives a success report for these updates from the PEP. If the PDP does not receive updated requests on some request handles, the PEP must not be sent decision updates for that frwkIfRoleCombo updates, i.e., the PDP must have the previous request state that it maintained for that request handle.

Note that, any unsolicited decisions received by the PEP in the time period after it receives updates to its Role-Combination associations and before receiving solicited decisions for the updated requests it sent for all context handles, could possibly contain outdated policies corresponding to the old Role-Combination associations as notified by this PEP in a previous request state.

The PDP must respond to the updated requests by solicited decisions, sending policies if applicable or null decisions. The PEP must respond to these solicited decisions with solicited reports to complete the transaction.

2.3. Updating a Request State

This section describes the messages exchanged between the PEP and PDP when the PEP is updating a previously sent request for a particular COPS handle. Note that a PEP can incrementally update a request only if the frwkPibIncarnationFullState attribute is shown to be supported via the supported PRC table. If this attribute is not supported, the PDP must treat all PEP requests as the full request state.

2.3.1 Full Request State

When the PEP wants to send the entire request state to the PDP (for example, in response to a Synchronize State Request from the PDP), the PEP MUST send the incarnation instance with the `frwkPibIncarnationFullState` attribute set to 'true'.

A PDP that receives an incarnation instance in the request message with this attribute set to 'true', must clear the request information it maintains for this request handle and re-install the information received.

If this attribute is set to 'false' or if the incarnation instance is missing in the request message, the request must be interpreted as an incremental update to the previous request message.

2.3.2 Installing PRIs in a Request

If the PEP wants to install additional PRIs for a request handle, the PEP MUST ensure that the `frwkPibIncarnationFullState` attribute is set to 'false', and the PEP MUST use new (unused in this context) `InstanceIds` [SPPI] for these PRIs.

When a PDP receives instances with new `InstanceIds` for a request with the `frwkPibIncarnationFullState` in the incarnation instance set to 'false', or if the request has no incarnation information, it must interpret these PRIs as an incremental update to the request state and add them to the request state it maintains for this handle.

2.3.3 Updating PRIs in a Request

If the PEP wants to update previously installed PRIs for a request handle, the PEP MUST ensure that the `frwkPibIncarnationFullState` attribute is set to 'false' for these PRIs. Note that the PEP must send the same `InstanceIds` for the PRIs being updated. If the PEP uses new `InstanceIds`, the PDP must interpret them as Install's for this request state.

When a PDP receives a request with instances having `InstanceIds` that exist in its state for that handle with the `frwkPibIncarnationFullState` in the incarnation instance set to 'false' or if the request has no incarnation information, it must interpret these PRIs as an update to the PRIs in the request state it maintains for this handle.

2.3.4 Removing PRIs from a Request

If the PEP wants to remove previously installed PRIs for a request handle, the PEP MUST ensure that the `frwkPibIncarnationFullState` attribute is set to 'false', and MUST send the PRI bindings with the `PRID` set to the `InstanceId` of the PRI to be removed, and the `length` field in the EPD object header set to the header length only, effectively setting the data length to zero.

Note that the PEP must send the same `InstanceIds` for the PRIs being removed. If the PEP sends new `InstanceIds` and the `length` field in the EPD object header is set to the header length only (implying the data length is zero), the PEP is attempting to remove an unknown/non-existent PRI. This SHOULD result in the PDP sending error PRIs in the solicited decision (see section 2.3.6 for a description of the `frwkErrorTable`).

If the PEP sends new `InstanceIds`, and the `length` field in the EPD object header is greater than the header length only (implying the EPD object has some attributes encoded in it), the PDP will interpret this as an install of the PRI if it can decode the EPD successfully.

When a PDP receives a request with instances having `InstanceIds` that exist in its state for that handle with the `frwkPibIncarnationFullState` in the incarnation instance set to 'false', or if the request has no incarnation information, and the `length` field in the EPD object header is set to the header length only (implying the data length is zero), it must remove these PRIs from the request state it maintains for this handle.

2.3.5 Removing EXTENDED, AUGMENTED PRIs

The PEP should remove the extended/augmented PRIs when it removes the base PRIs in the same COPS message. See [SPPI] for a description of EXTENDED/AUGMENTED PRCs. A PDP that receives removes for a base PRI must implicitly remove the extensions.

2.3.6 Error Handling in Request updates

If the PDP cannot process all the request installs/updates/removes in the COPS request message successfully, it MUST rollback to its previous request state and it MUST send a solicited decision to the PEP that contains `frwkErrorTable` instances. These instances contain an error code and a sub-code as defined in the [COPS-PR] CPERR object. For example, if the PEP tries to remove an instance that does not exist, the 'priInstanceInvalid' error code must be sent to the PEP in a `frwkError` PRI. The `frwkError` PRIs also contain the PRC and the `InstanceId` of the error-causing PRI. The PEP may then

examine these error PRIs and resend the modified request. Note that, until the PEP resends the request updates/removes, it will have configuration information for the last successful request state it sent to the PDP.

2.4. Multiple PIB Instances

[COPS-PR] supports multiple, disjoint, independent instances of the PIB to represent multiple instances of configured policy. The intent is to allow for the pre-provisioning of policy that can then be made active by a single, short decision from the PDP.

A COPS context can be defined as an independent COPS request state for a particular subject category (client-type). A context may be an outsourcing context or a configuration context. A configuration context is an instance of the PIB triggered and controlled by the PDP, which contains device setup information. This device configuration information dictates the device behavior as specified by the PDP. An outsourcing context on the other hand, is a PIB instance that is triggered from the PEP side and is a request to the PDP for action. The action requested will be interpreted in the domain of the client-type. Configuration contexts belong to a set of configuration contexts for a specific client type - out of which one configuration context may be active. However, multiple outsourcing contexts can be active simultaneously.

With the [COPS-PR] protocol, each of these states is identified by a unique client handle. The creation and deletion of these PIB instances can be controlled by the PDP as described in [COPS-PR] or can be triggered by an event by the PEP. A PEP must open at least one "request-state" for configuration for a given subject-category (client type). Additional "request-states" at the PEP may be initiated by the PDP or asynchronously generated by the PEP for outsourcing due to local events, which will be fully specified by the PRID/EPD data carried in the request.

The `frwkPibIncarnationInCtxtSet` flag defines a set of contexts out of which only one context can be active at any given time. This set is called the 'configuration contexts' set. At most, one context may be active from this 'configuration context' set at any given time. Contexts that have the `frwkPibIncarnationInCtxtSet` attribute set to 'true' belong to this set. Contexts that do not belong to this set have the `frwkPibIncarnationInCtxtSet` set to 'false' and belong to the set of 'outsourcing contexts'. Note that a PEP can have these two sets of contexts only if the `frwkPibIncarnationInCtxtSet` attribute is shown to be supported via the supported PRC table. If the

frwkPibIncarnationInCtxtSet is not supported, a PEP must treat all contexts as belonging to the set of 'configuration contexts' i.e., at the most one context can be active at any given time.

Note that in the event that a PEP has a capability change such as a card hot swap or any other change in its notify information that may warrant a policy refresh, a subsequent complete or incremental request must be issued to the PDP containing the new/updated capabilities for all the configuration contexts. A request for re-configuration is issued for all request state configuration contexts, both for the active configuration context as well as any inactive configuration contexts. This is to ensure that when an inactive configuration context is activated, it has been pre-configured with policies compatible with the PEP's current capabilities.

Although many PIB instances may be configured on a device (the maximum number of these instances being determined by the device itself), only one of the contexts from the 'configuration contexts' set can be active at any given time; the active one being selected by the PDP. The Framework PIB supports the attribute frwkPibIncarnationActive in the frwkPibIncarnationTable to allow the PDP to denote the PIB instance as being active in a COPS decision message, and similarly, to report the active state (active or not) of the PIB instance to the PDP in a COPS request message.

When the PEP installs an attribute frwkPibIncarnationActive that is 'true' in one PIB instance which belongs to the 'configuration contexts' set, the PEP must ensure, re-setting the attribute if necessary, that the frwkPibIncarnationActive attribute is 'false' in all other installed contexts that belong to this set. To switch contexts, the PDP should set the frwkPibIncarnationActive attribute to 'true' in the context it wants to make the active context. The PDP should set this attribute in a context to 'false' only if it wants to send an inactive context to the PEP or deactivate the active context on the PEP. If an active context is made inactive without activating another context, the PEP must not have any policies enforced from any configuration contexts installed.

2.5. Reporting and Configuring 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 organized into Capability Sets, with each Capability Set given a unique name (frwkCapabilitySetName) and associated with a set of Role Combinations. In that way, each Role Combination may be associated with a set of interfaces. These capabilities are

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

Specific capability PRCs may be defined in other PIBs. These capability instances are grouped via the `frwkCapabilitySetTable`. If the PEP wishes to send capability information to the PDP, the PIB must indicate which capabilities the PEP may send to the PDP by means of the 'notify' PIB-ACCESS clause as described in [SPPI]. If a PIB does not have any capabilities to communicate to the PDP, it must not send any instances for the `frwkCapabilitySetTable`. If in this case the `frwkIfRoleCombo` table is used to communicate role combinations assigned to interfaces (via `IfIndex`), the `frwkRoleComboCapSetName` attribute in the `frwkIfRoleComboTable` instances must be set to a zero length string.

2.6. Reporting of Device Limitations

To facilitate efficient policy installation, it is important to understand a device's limitations in relation to the advertised device capabilities. Limitations may be class-based, e.g., an "install" class is supported as a "notify" or only a limited number of class instances may be created, or attribute-based. Attribute limitations, such as supporting a restricted set of enumerations or requiring related attributes to have certain values, detail implementation limitations at a fine level of granularity.

A PDP can avoid certain installation issues in a proactive fashion by taking into account a device's limitations prior to policy installation rather than in a reactive mode during installation. As with device capabilities, device limitations are communicated to the PDP when policy is requested.

Reported device limitations may be accompanied by guidance values that can be used by a PDP to determine acceptable values for the identified attributes.

3. The Framework TC PIB module

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

IMPORTS  MODULE-IDENTITY, TEXTUAL-CONVENTION,
         Unsigned32, pib FROM COPS-PR-SPPI;

frwkTcPib  MODULE-IDENTITY
    SUBJECT-CATEGORIES    { all }
    LAST-UPDATED "200302130000Z"  -- 13 Feb 2003
    ORGANIZATION "IETF RAP WG"
```

CONTACT-INFO "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

Ravi Sahita
Intel Labs.
2111 NE 25th Ave.
Hillsboro, OR 97124 USA
Phone: +1 503 712 1554
Email: ravi.sahita@intel.com

RAP WG Mailing list: rap@ops.ietf.org "

DESCRIPTION

"The PIB module containing the Role and RoleCombination
Textual Conventions and other generic TCs.

Copyright (C) The Internet Society (2003). This version of
this PIB module is part of RFC 3318; see the RFC itself for
full legal notices."

REVISION "200302130000Z" -- 13 Feb 2003

DESCRIPTION "Initial version, published in RFC 3318."
 ::= { pib 3 }

Role ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"A role represents a functionality characteristic or
capability of a resource to which policies are applied.
Examples of roles include Backbone_interface,
Frame_Relay_interface, BGP-capable-router, web-server,
firewall, etc.

The only valid character set is US-ASCII. Valid characters
are a-z, A-Z, 0-9, period, hyphen and underscore. A role
must always start with a letter (a-z or A-Z). A role must
not contain the US-ASCII characters '*' or '+' since they
have special meaning associated with them, explained in the
RoleCombination TEXTUAL CONVENTION."

SYNTAX OCTET STRING (SIZE (1..31))

RoleCombination ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"An octet string containing concatenated Roles. For the format specification of roles, refer to the 'Role' TEXTUAL-CONVENTION. A valid Role Combination must be formed by a set of valid Roles, concatenated by the US-ASCII character '+', where the roles are in lexicographic order from minimum to maximum. For example, 'a+b' and 'b+a' are NOT different role-combinations; rather, they are different formatting of the same (one) role-combination.

Notice the roles within a role-combination are in Lexicographic order from minimum to maximum, hence, we declare:

'a+b' is the valid formatting of the role-combination,
'b+a' is an invalid formatting of the role-combination.

Notice the need of zero-length role-combination as the role-combination of interfaces to which no roles have been assigned. This role-combination is also known as the 'null' role-combination. (Note the deliberate use of lower case letters to avoid confusion with the US-ASCII NULL character which has a value of zero but length of one.)

The US-ASCII character '*' is used to specify a wild carded Role Combination. '*' must not be used to wildcard Roles. Hence, we declare:

'*+a+b' is a valid wild carded Role Combination.

'eth*+a+b' is not a valid wild carded Role Combination.

Note that since Roles are lexicographically listed in a Role Combination, the following is an invalid role combination, since '*' is lexicographically before 'a': 'a+b*'. "

SYNTAX OCTET STRING (SIZE (0..255))

PrcIdentifierOid ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"An OID that identifies a PRC. The value MUST be an OID assigned to a PRC's entry definition. The Entry definition of a PRC has an OID value XxxTable.1 where XxxTable is the OID assigned to the PRC table object.

An attribute with this syntax MUST specify a PRC, which is defined in the PIB module(s) registered in the context of the client-type used.

An attribute with this syntax cannot have the value 0.0 (zeroDotZero). If the attribute using this syntax can be set to 0.0 use the PrcIdentifierOidOrZero TEXTUAL-CONVENTION which makes such use explicit."

SYNTAX OBJECT IDENTIFIER

PrcIdentifierOidOrZero ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"An OID that identifies a PRC or zeroDotZero (0.0). The value MUST be an OID assigned to a PRC's entry definition or 0.0 (zeroDotZero). The Entry definition of a PRC has an OID value XxxTable.1 where XxxTable is the OID assigned to the PRC table object.

An attribute with this syntax can have the value 0.0 (zeroDotZero) to indicate that it currently does not identify a PRC."

SYNTAX OBJECT IDENTIFIER

AttrIdentifier ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"A Unsigned32 value that identifies an attribute in a PRC by its sub-id. The sub-id is the OID assigned to this attribute in the PRC definition.

A AttrIdentifier value is always interpreted within the context of an attribute of type PrcIdentifierOid or PrcIdentifierOidOrZero. The PrcIdentifierOid (or

PrcIdentifierOidOrZero) object which defines the context must be registered immediately before the object which uses the AttrIdentifier textual convention. If the context defining attribute is of type PrcIdentifierOidOrZero and has the value 0.0, then in that case this attribute value has no meaning.

An attribute with this syntax MUST specify a sub-id which MUST be defined in the PRC identified (if any) in the PrcIdentifierOid (or PrcIdentifierOidOrZero) attribute. The PrcIdentifierOid (orZero) and the AttrIdentifier attributes together identify a particular attribute in a particular PRC.

An attribute with this syntax cannot have the value 0 (zero). If the attribute using this syntax can be set to 0 use the AttrIdentifierOrZero TEXTUAL-CONVENTION which makes that explicit."

SYNTAX Unsigned32 (1..4294967295)

AttrIdentifierOrZero ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"A Unsigned32 value that identifies an attribute in a PRC by its sub-id or has the value 0 (zero). The sub-id if non-zero, is the OID assigned to this attribute in the PRC definition.

An AttrIdentifierOrZero value is always interpreted within the context of an attribute of type PrcIdentifierOid or PrcIdentifierOidOrZero. The PrcIdentifierOid (or PrcIdentifierOidOrZero) object that defines the context must be registered immediately before the object which uses the AttrIdentifierOrZero textual convention. If the context defining attribute is of type PrcIdentifierOidOrZero and has the value 0.0, then in that case this attribute value has no meaning.

An attribute with this syntax can have the value 0 (zero) to indicate that it currently does not identify a PRC attribute. If it has a non-zero value, the PrcIdentifierOid (orZero) and the AttrIdentifierOrZero attributes together identify a particular attribute in a particular PRC."

SYNTAX Unsigned32

AttrIdentifierOid ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"An OID that identifies an attribute in a PRC. The value MUST be an OID assigned to a PRC's attribute definition. The last sub-id is the sub-id of the attribute as it is defined in the PRC entry definition. The prefix OID (after dropping the last sub-id) is the OID assigned to the Entry object of a defined PRC. The Entry definition of a PRC has an OID value XxxTable.1 where XxxTable is the OID assigned to the PRC Table object.

An attribute with this syntax MUST not have the value 0.0 (zeroDotZero). If 0.0 is a valid value, the TEXTUAL CONVENTION AttrIdentifierOidOrZero must be used which makes such use explicit."

SYNTAX OBJECT IDENTIFIER

AttrIdentifierOidOrZero ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"An OID that identifies an attribute in a PRC or has a value 0.0 (zeroDotZero). The value MUST be an OID assigned to a PRC's attribute definition or the value 0.0.

If not 0.0, the last sub-id MUST be the sub-id of the attribute as it is defined in the PRC Entry object definition. The prefix OID (after dropping the last sub-id) is the OID assigned to the Entry object of a defined PRC. The Entry definition of a PRC has an OID value XxxTable.1 Where, XxxTable is the OID assigned to the PRC Table object.

An attribute with this syntax can have the value 0.0 (zeroDotZero) to indicate that it currently does not identify a PRC's attribute."

SYNTAX OBJECT IDENTIFIER

ClientType ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"An Unsigned32 value that identifies a COPS Client-type. An attribute with this syntax must be set to zero if it does not specify a COPS client-type for the PRI."

REFERENCE

"The COPS (Common Open Policy Service) Protocol, RFC 2748."

SYNTAX Unsigned32 (0..65535)

ClientHandle ::= TEXTUAL-CONVENTION

STATUS current

DESCRIPTION

"An octet string that identifies a COPS Client handle. A zero length value implies the attribute does not specify a valid client handle."

REFERENCE

"The COPS (Common Open Policy Service) Protocol, RFC 2748."

SYNTAX OCTET STRING (SIZE(0..65535))

END

4. Summary of the Framework PIB

The Framework PIB defines four groups of PRCs:

4.1. Base PIB classes Group

This contains PRCs intended to describe the PRCs supported by the PEP, PRC and/or attribute limitations and its current configuration.

PRC Support Table

As the technology evolves, we expect devices to be enhanced with new PIBs, existing PIBs to add new PRCs and existing PRCs to be augmented or extended with new attributes. Also, it is likely that some existing PRCs or individual attributes of PRCs will be deprecated. The PRC Support Table describes the PRCs that the device supports as well as the individual attributes of each PRC. Using this information the PDP can potentially tailor the policy to more closely match the capabilities of the device. The PRC Support Table instances are specific to the particular Subject Category (Client-Type). That is, the PRC Support Table for Subject Category 'A' will not include instances for classes supported by the Subject Category 'B'. Note that the COPS client-type [COPS] used for Framework PIB PRIs sent/received over COPS-PR MUST be the unique SUBJECT-CATEGORY number assigned for the area of policy being managed (e.g., QoS, Security etc). The PEP MUST ignore the attributes that it reports as not Supported in the decision from the PDP. The PEP SHOULD not send duplicate PRC support instances in a COPS Request and the PDP MUST ignore duplicate instances and MUST use the first instance received for a supported PRC in a COPS Request.

PIB Incarnation Table

This PRC contains exactly one row (corresponding to one PRI) per context. It identifies the PDP that was the last to download policy into the device and also contains an identifier to identify the version of the policy currently downloaded. This identifier, both its syntax and value, is meaningful only to the PDPs. It is intended to be a mechanism whereby a PDP, when accepting a connection from a PEP, can easily identify a known incarnation of policy. This PRC defines a flag via which the installed contexts are divided into a set of contexts ('configuration contexts') out of which only one context is active and the remaining contexts form a set of 'outsourcing contexts' which are all active. The incarnation PRC also defines an attribute to indicate which configuration context is

the active one at the present time in the 'configuration contexts' set. The incarnation instance is specific to the particular Subject Category (Client-Type).

Component Limitations Table

Some devices may not be able to implement the full range of values for all attributes. In principle, each PRC supports a set of errors that the PEP can report to the PDP in the event that the specified policy is not implementable. It may be preferable for the PDP to be informed of the device limitations before actually attempting to install policy, and while the error can indicate that a particular attribute value is unacceptable to the PEP, this does not help the PDP ascertain which values would be acceptable. To alleviate these limitations, the PEP can report some limitations of attribute values and/or classes and possibly guidance values for the attribute in the Component Limitations Table

Device Identification Table

This PRC contains a single PRI that contains device-specific information that is used to facilitate efficient policy installation by a PDP. The instance of this PRC is reported to the PDP in a COPS request message so that the PDP can take into account certain device characteristics during policy installation.

4.2. Device Capabilities group

This group contains the PRCs that describe the characteristics of interfaces of the device and the Role Combinations assigned to them.

Capabilities Set Table

The capabilities the PEP supports are described by rows in this PRC (frwkCapabilitySetTable). Each row, or instance of this class, associates a unique capability name with a set of capabilities that an entity on the PEP may support. The unique name is used to form a set of capabilities that the name represents. The capability references can specify instances in relevant capability tables in any PIB. The PEP notifies the PDP of these capability sets and then the PDP configures the interfaces, per role combination. The unique name (frwkCapabilitySetName) is not to be confused with the IfType object in the Interfaces Group MIB [RFC2863].

Interface and Role Combination Table

The Capabilities Set Table (explained above) describes the entities on the PEP (for example, interfaces) by their capabilities, by assigning the capability sets a unique name (`frwkCapabilitySetName`). It is possible to tailor the behavior of interfaces by assigning specific role-combinations to the capability sets. This allows interfaces with the same capability sets to be assigned different policies, based on the current roles assigned to them. At the PDP, configuration is done in terms of these interface capability set names and the role-combinations assigned to them. Thus, each row of this class is a `<Interface Index, interface capability set name, Role Combo>` tuple, that indicates the roles that have been assigned to a particular capability set (as identified by `frwkRoleComboCapSetName`) and to a particular interface. Note that the uniqueness criteria for this PRC has all the attributes, thus a `frwkRoleComboCapSetName` may have multiple role-combinations that it is associated with. Via the `IfIndex`, this PRC answers the questions of 'which interfaces have a specific role combination?' and 'what role combination a specific interface is a part of?'.

4.3. Classifier group

This group contains the IP, IEEE 802 and Internal Label Classifier elements. The set of tables consist of a Base Filter table that contains the Index InstanceId and the Negation flag for the filter. This `frwkBaseFilterTable` is extended to form the IP Filter table, the 802 Filter table [802] and the Internal Label table. Filters may also be defined outside this document and used to extend the Base Filter table.

The Extended classes do not have a separate Index value. Instances of the extended classes have the same indices as their base class instance. Inheritance is achieved using the EXTENDS keyword as defined in [SPPI].

4.4. Marker group

This group contains the 802 marker and internal label marker PRCs. The 802 marker may be applied to mark 802 packets with the required VLAN Id and/or priority value. The Internal Label marker is applied to traffic in order to label it with a network device specific label. Such a label is used to assist the differentiation of an input flow after it has been aggregated with other flows. The label is

implementation specific and may be used for other policy related functions like flow accounting purposes and/or other data path treatments.

5. The Framework PIB Module

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

IMPORTS
    Unsigned32, Integer32, MODULE-IDENTITY,
    MODULE-COMPLIANCE, OBJECT-TYPE, OBJECT-GROUP, pib
        FROM COPS-PR-SPPI
    InstanceId, Prid
        FROM COPS-PR-SPPI-TC
    RoleCombination, PrcIdentifierOid, AttrIdentifierOrZero,
    ClientType, ClientHandle
        FROM FRAMEWORK-TC-PIB
    InetAddress, InetAddressType,
    InetAddressPrefixLength, InetPortNumber
        FROM INET-ADDRESS-MIB
    InterfaceIndex
        FROM IF-MIB
    DscpOrAny
        FROM DIFFSERV-DSCP-TC
    TruthValue, PhysAddress
        FROM SNMPv2-TC
    SnmpAdminString
        FROM SNMP-FRAMEWORK-MIB;

frameworkPib MODULE-IDENTITY
    SUBJECT-CATEGORIES { all }
    LAST-UPDATED "200302130000Z" -- 13 Feb 2003
    ORGANIZATION "IETF RAP WG"
    CONTACT-INFO "
        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
```

Ravi Sahita
Intel Labs.
2111 NE 25th Ave.

Hillsboro, OR 97124 USA
Phone: +1 503 712 1554
Email: ravi.sahita@intel.com

RAP WG Mailing list: rap@ops.ietf.org"

DESCRIPTION

"A PIB module containing the base set of PRCs that provide support for management of multiple PIB contexts, association of roles to device capabilities and other reusable PRCs. PEPs are required for to implement this PIB if the above features are desired. This PIB defines PRCs applicable to 'all' subject-categories.

Copyright (C) The Internet Society (2003). This version of this PIB module is part of RFC 3318; see the RFC itself for full legal notices."

REVISION "200302130000Z" -- 13 Feb 2003

DESCRIPTION

"Initial version, published in RFC 3318."

::= { pib 2 }

--

-- The root OID for PRCs in the Framework PIB

--

frwkBasePibClasses

OBJECT IDENTIFIER ::= { frameworkPib 1 }

--

-- PRC Support Table

--

frwkPrcSupportTable OBJECT-TYPE

SYNTAX SEQUENCE OF FrwkPrcSupportEntry
 PIB-ACCESS notify
 STATUS current

DESCRIPTION

"Each instance of this PRC 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.

All install and install-notify PRCs supported by the device must be represented in this PRC. Notify PRCs may be represented for informational purposes."

::= { frwkBasePibClasses 1 }

frwkPrcSupportEntry OBJECT-TYPE

SYNTAX FrwkPrcSupportEntry
 STATUS current

DESCRIPTION

"An instance of the frwkPrcSupport class that identifies a specific PRC and associated attributes as supported by the device."

PIB-INDEX { frwkPrcSupportPrid }
 UNIQUENESS { frwkPrcSupportSupportedPrc }

::= { frwkPrcSupportTable 1 }

FrwkPrcSupportEntry ::= SEQUENCE {

frwkPrcSupportPrid InstanceId,
 frwkPrcSupportSupportedPrc PrcIdentifierOid,
 frwkPrcSupportSupportedAttrs OCTET STRING

}

frwkPrcSupportPrid OBJECT-TYPE

SYNTAX InstanceId
 STATUS current

DESCRIPTION

"An arbitrary integer index that uniquely identifies an instance of the frwkPrcSupport class."

::= { frwkPrcSupportEntry 1 }

frwkPrcSupportSupportedPrc OBJECT-TYPE

SYNTAX PrcIdentifierOid

STATUS current

DESCRIPTION

"The object identifier of a supported PRC. The value is the OID of the Entry object of the PRC definition. The Entry Object definition of a PRC has an OID with value XxxTable.1 Where, XxxTable is the OID assigned to the PRC Table Object definition. There may not be more than one instance of the frwkPrcSupport class with the same value of frwkPrcSupportSupportedPrc."

::= { frwkPrcSupportEntry 2 }

frwkPrcSupportSupportedAttrs OBJECT-TYPE

SYNTAX OCTET STRING

STATUS current

DESCRIPTION

"A bit string representing the supported attributes of the class that is identified by the frwkPrcSupportSupportedPrc object.

Each bit of this bit string 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 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 string is N bits long and there are more than N class attributes then the bit string is logically extended with 0's to the required length. On the other hand, If the PDP receives a bit string of length N and there are less than N class attributes then the PDP should ignore the extra bits in the bit string, i.e., assume those attributes are unsupported."

REFERENCE

"COPS Usage for Policy Provisioning. RFC 3084, section 2.2.1."

::= { frwkPrcSupportEntry 3 }

--

-- PIB Incarnation Table

--

frwkPibIncarnationTable OBJECT-TYPE

SYNTAX SEQUENCE OF FrwkPibIncarnationEntry
 PIB-ACCESS install-notify
 STATUS current

DESCRIPTION

"This PRC contains a single PROvisioning Instance per installed context that identifies the current incarnation of the PIB and the PDP or network manager that installed this incarnation. The instance of this PRC is reported to the PDP in the REQ message 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."

::= { frwkBasePibClasses 2 }

frwkPibIncarnationEntry OBJECT-TYPE

SYNTAX FrwkPibIncarnationEntry
 STATUS current

DESCRIPTION

"An instance of the frwkPibIncarnation class. Only one instance of this PRC is ever instantiated per context"

PIB-INDEX { frwkPibIncarnationPrid }

::= { frwkPibIncarnationTable 1 }

FrwkPibIncarnationEntry ::= SEQUENCE {

frwkPibIncarnationPrid	InstanceId,
frwkPibIncarnationName	SnmpAdminString,
frwkPibIncarnationId	OCTET STRING,
frwkPibIncarnationLongevity	INTEGER,
frwkPibIncarnationTtl	Unsigned32,
frwkPibIncarnationInCtxtSet	TruthValue,
frwkPibIncarnationActive	TruthValue,
frwkPibIncarnationFullState	TruthValue

}

frwkPibIncarnationPrid OBJECT-TYPE

SYNTAX InstanceId
 STATUS current

DESCRIPTION

"An index to uniquely identify an instance of this PRC."

::= { frwkPibIncarnationEntry 1 }

frwkPibIncarnationName OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..255))

STATUS current

DESCRIPTION

"The name of the PDP that installed the current incarnation of the PIB into the device. A zero-length string value for this type implies the PDP has not assigned this type any value. By default, it is the zero length string."

```
::= { frwkPibIncarnationEntry 2 }
```

frwkPibIncarnationId OBJECT-TYPE

SYNTAX OCTET STRING (SIZE (0..255))

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. A zero-length string value for this type implies the PDP has not assigned this type any value. By default, it is the zero-length string."

```
::= { frwkPibIncarnationEntry 3 }
```

frwkPibIncarnationLongevity OBJECT-TYPE

```
SYNTAX      INTEGER {
                expireNever(1),
                expireImmediate(2),
                expireOnTimeout(3)
            }
```

STATUS current

DESCRIPTION

"This attribute controls what the PEP does with the downloaded policy on a Client Close message or a loss of connection to the PDP.

If set to expireNever, the PEP continues to operate with the installed policy indefinitely. If set to expireImmediate, the PEP immediately expires the policy obtained from the PDP and installs policy from local configuration. If set to expireOnTimeout, the PEP continues to operate with the policy installed by the PDP for a period of time specified by frwkPibIncarnationTtl. After this time (and it has not reconnected to the original or new PDP) the PEP expires this policy and reverts to local configuration.

For all cases, it is the responsibility of the PDP to check the incarnation and download new policy, if necessary, on a reconnect. On receiving a Remove-State for the active

context, this attribute value MUST be ignored and the PEP should expire the policy in that active context immediately. Policy enforcement timing only applies to policies that have been installed dynamically (e.g., by a PDP via COPS)."

REFERENCE

"COPS Usage for Policy Provisioning. RFC 3084."

::= { frwkPibIncarnationEntry 4 }

frwkPibIncarnationTtl OBJECT-TYPE

SYNTAX Unsigned32

UNITS "seconds"

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.

This attribute is only meaningful if
frwkPibIncarnationLongevity is set to expireOnTimeout."

::= { frwkPibIncarnationEntry 5 }

frwkPibIncarnationInCtxtSet OBJECT-TYPE

SYNTAX TruthValue

STATUS current

DESCRIPTION

"When the PDP installs a PRI with this flag set to 'true' it implies this context belongs to the set of contexts out of which at the most one context can be active at a given time. If this attribute is set to 'false' this context is one of the outsourcing (simultaneous active) contexts on the PEP.

This attribute is 'true' for all contexts belong to the set of configuration contexts. Within the configuration context set, one context can be active identified by the
frwkPibIncarnationActive attribute."

REFERENCE

"TruthValue Textual Convention, defined in RFC 2579."

::= { frwkPibIncarnationEntry 6 }

frwkPibIncarnationActive OBJECT-TYPE

SYNTAX TruthValue

STATUS current

DESCRIPTION

"When the PDP installs a PRI on the PEP with this attribute set to 'true' and if this context belongs to the 'configuration contexts' set, i.e., the frwkPibIncarnationInCtxtSet is set to 'true', then the PIB instance to which this PRI belongs must become the active PIB instance. In this case, the previous active instance from this set MUST become inactive and the frwkPibIncarnationActive attribute in that PIB instance MUST be set to 'false'.

When the PDP installs an attribute frwkPibIncarnationActive on the PEP that is 'true' in one PIB instance and if the context belongs to the 'configuration contexts' set, the PEP must ensure, re-setting the attribute if necessary, that the frwkPibIncarnationActive attribute is 'false' in all other contexts which belong to the 'configuration contexts' set."

::= { frwkPibIncarnationEntry 7 }

frwkPibIncarnationFullState OBJECT-TYPE

SYNTAX TruthValue

STATUS current

DESCRIPTION

"This attribute is interpreted only when sent in a COPS request message from the PEP to the PDP. It does not have any meaning when sent from the PDP to the PEP.

If this attribute is set to 'true' by the PEP, then the request that the PEP sends to the PDP must be interpreted as the complete configuration request for the PEP. The PDP must in this case refresh the request information for the handle that the request containing this PRI was received on. If this attribute is set to 'false', then the request PRIs sent in the request must be interpreted as updates to the previous request PRIs sent using that handle. See section 3.3 for details on updating request state information."

REFERENCE

"RFC 3318 Section 2.3"

::= { frwkPibIncarnationEntry 8 }

--

-- Device Identification Table

--

frwkDeviceIdTable OBJECT-TYPE

SYNTAX SEQUENCE OF FrwkDeviceIdEntry
PIB-ACCESS notify
STATUS current
DESCRIPTION

"This PRC contains a single PRC Provisioning Instance that contains general purpose device-specific information that is used to facilitate efficient policy communication by a PDP. The instance of this PRC is reported to the PDP in a COPS request message so that the PDP can take into account certain device characteristics during policy installation."

::= { frwkBasePibClasses 3 }

frwkDeviceIdEntry OBJECT-TYPE

SYNTAX FrwkDeviceIdEntry
STATUS current
DESCRIPTION

"An instance of the frwkDeviceId class. Only one instance of this PRC is ever instantiated."

PIB-INDEX { frwkDeviceIdPrid }

::= { frwkDeviceIdTable 1 }

FrwkDeviceIdEntry ::= SEQUENCE {
 frwkDeviceIdPrid InstanceId,
 frwkDeviceIdDescr SnmpAdminString,
 frwkDeviceIdMaxMsg Unsigned32,
 frwkDeviceIdMaxContexts Unsigned32
}

frwkDeviceIdPrid OBJECT-TYPE

SYNTAX InstanceId
STATUS current
DESCRIPTION

"An index to uniquely identify an instance of this PRC."

::= { frwkDeviceIdEntry 1 }

frwkDeviceIdDescr OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (1..255))

STATUS current

DESCRIPTION

"A textual description of the PEP. This value should include the name and version identification of the PEP's hardware and software."

::= { frwkDeviceIdEntry 2 }

frwkDeviceIdMaxMsg OBJECT-TYPE

SYNTAX Unsigned32 (64..4294967295)

UNITS "octets"

STATUS current

DESCRIPTION

"The maximum COPS-PR message size, in octets, that the device is capable of processing. Received messages with a size in excess of this value must cause the PEP to return an error to the PDP containing the global error code 'maxMsgSizeExceeded'. This is an additional error-avoidance mechanism to allow the administrator to know the maximum message size supported so that they have the ability to control the message size of messages sent to the device. This attribute must have a non-zero value. The device should send the MAX value for Unsigned32 for this attribute if it not defined."

DEFVAL { 4294967295 }

::= { frwkDeviceIdEntry 3 }

frwkDeviceIdMaxContexts OBJECT-TYPE

SYNTAX Unsigned32 (1..4294967295)

UNITS "contexts"

STATUS current

DESCRIPTION

"The maximum number of unique contexts supported by the device. This is an additional error-avoidance mechanism to allow the administrators to have the ability to know the maximum number of contexts supported so that they can control the number of configuration contexts they install on the device. This attribute must have a non-zero value. The device should send the MAX value for Unsigned32 for this attribute if it not defined."

DEFVAL { 4294967295 }

::= { frwkDeviceIdEntry 4 }

--

-- Component Limitations Table
--

frwkCompLimitsTable OBJECT-TYPE

SYNTAX SEQUENCE OF FrwkCompLimitsEntry

PIB-ACCESS notify

STATUS current

DESCRIPTION

"This PRC supports the ability to export information detailing PRC/attribute implementation limitations to the policy management system. Instances of this PRC apply only for PRCs with access type 'install' or 'install-notify'.

Each instance of this PRC identifies a PProvisioning Class or attribute and a limitation related to the implementation of the class/attribute in the device. Additional information providing guidance related to the limitation may also be present. These PRIs are sent to the PDP to indicate which PRCs or PRC attributes the device supports in a restricted manner."

::= { frwkBasePibClasses 4 }

frwkCompLimitsEntry OBJECT-TYPE

SYNTAX FrwkCompLimitsEntry

STATUS current

DESCRIPTION

"An instance of the frwkCompLimits class that identifies a PRC or PRC attribute and a limitation related to the PRC or PRC attribute implementation supported by the device. COPS-PR lists the error codes that MUST be returned (if applicable) for policy installation that don't abide by the restrictions indicated by the limitations exported. [SPPI] defines an INSTALL-ERRORS clause that allows PIB designers to define PRC specific error codes that can be returned for policy installation. This allows efficient debugging of PIB implementations."

REFERENCE

"COPS Usage for Policy Provisioning. RFC 3084."

PIB-INDEX { frwkCompLimitsPrid }

UNIQUENESS { frwkCompLimitsComponent,
frwkCompLimitsAttrPos,
frwkCompLimitsNegation,
frwkCompLimitsType,
frwkCompLimitsSubType,
frwkCompLimitsGuidance }

```
::= { frwkCompLimitsTable 1 }
```

```
FrwkCompLimitsEntry ::= SEQUENCE {
    frwkCompLimitsPrid          InstanceId,
    frwkCompLimitsComponent    PrcIdentifierOid,
    frwkCompLimitsAttrPos      AttrIdentifierOrZero,
    frwkCompLimitsNegation     TruthValue,
    frwkCompLimitsType         INTEGER,
    frwkCompLimitsSubType      INTEGER,
    frwkCompLimitsGuidance     OCTET STRING
}
```

```
frwkCompLimitsPrid OBJECT-TYPE
    SYNTAX      InstanceId
    STATUS      current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies an
        instance of the frwkCompLimits class."
```

```
::= { frwkCompLimitsEntry 1 }
```

```
frwkCompLimitsComponent OBJECT-TYPE
    SYNTAX      PrcIdentifierOid
    STATUS      current
    DESCRIPTION
        "The value is the OID of a PRC (the table entry) which is
        supported in some limited fashion or contains an attribute
        that is supported in some limited fashion with regard to
        it's definition in the associated PIB module. The same OID
        may appear in the table several times, once for each
        implementation limitation acknowledged by the device."
```

```
::= { frwkCompLimitsEntry 2 }
```

```
frwkCompLimitsAttrPos OBJECT-TYPE
    SYNTAX      AttrIdentifierOrZero
    STATUS      current
    DESCRIPTION
        "The relative position of the attribute within the PRC
        specified by the frwkCompLimitsComponent. A value of 1 would
        represent the first columnar object in the PRC and a value
        of N would represent the Nth columnar object in the PRC. A
        value of zero (0) indicates that the limit applies to the
        PRC itself and not to a specific attribute."
```

```
::= { frwkCompLimitsEntry 3 }
```


frwkCompLimitsNegation OBJECT-TYPE

SYNTAX TruthValue

STATUS current

DESCRIPTION

"A boolean value ,if 'true', negates the component limit exported."

::= { frwkCompLimitsEntry 4 }

frwkCompLimitsType OBJECT-TYPE

SYNTAX INTEGER {
 priSpaceLimited(1),
 attrValueSupLimited(2),
 attrEnumSupLimited(3),
 attrLengthLimited(4),
 prcLimitedNotify(5)
}

STATUS current

DESCRIPTION

"A value describing an implementation limitation for the device related to the PRC or PRC attribute identified by the frwkCompLimitsComponent and the frwkCompLimitsAttrPos attributes."

Values for this object are one of the following:

priSpaceLimited(1) - No more instances than that specified by the guidance value may be installed in the given class. The component identified MUST be a valid PRC. The SubType used MUST be valueOnly(9).

attrValueSupLimited(2) - Limited values are acceptable for the identified component. The component identified MUST be a valid PRC attribute. The guidance OCTET STRING will be decoded according to the attribute type.

attrEnumSupLimited(3) - Limited enumeration values are legal for the identified component. The attribute identified MUST be a valid enum type.

attrLengthLimited(4) - The length of the specified value for the identified component is limited. The component identified MUST be a valid PRC attribute of base-type OCTET STRING.

prcLimitedNotify (5) - The component is currently limited for use by request or report messages prohibiting decision installation. The component identified must be a valid PRC."

```
::= { frwkCompLimitsEntry 5 }
```

```
frwkCompLimitsSubType OBJECT-TYPE
```

```
SYNTAX          INTEGER {  
                                none(1),  
                                lengthMin(2),  
                                lengthMax(3),  
                                rangeMin(4),  
                                rangeMax(5),  
                                enumMin(6),  
                                enumMax(7),  
                                enumOnly(8),  
                                valueOnly(9),  
                                bitMask(10)  
                                }
```

```
STATUS          current
```

```
DESCRIPTION
```

"This object indicates the type of guidance related to the noted limitation (as indicated by the `frwkCompLimitsType` attribute) that is provided in the `frwkCompLimitsGuidance` attribute.

A value of 'none(1)' means that no additional guidance is provided for the noted limitation type.

A value of 'lengthMin(2)' means that the guidance attribute provides data related to the minimum acceptable length for the value of the identified component. A corresponding class instance specifying the 'lengthMax(3)' value is required in conjunction with this sub-type.

A value of 'lengthMax(3)' means that the guidance attribute provides data related to the maximum acceptable length for the value of the identified component. A corresponding class instance specifying the 'lengthMin(2)' value is required in conjunction with this sub-type.

A value of 'rangeMin(4)' means that the guidance attribute provides data related to the lower bound of the range for the value of the identified component. A corresponding class instance specifying the 'rangeMax(5)' value is required in conjunction with this sub-type.

A value of 'rangeMax(5)' means that the guidance attribute provides data related to the upper bound

of the range for the value of the identified component. A corresponding class instance specifying the 'rangeMin(4)' value is required in conjunction with this sub-type.

A value of 'enumMin(6)' means that the guidance attribute provides data related to the lowest enumeration acceptable for the value of the identified component. A corresponding class instance specifying the 'enumMax(7)' value is required in conjunction with this sub-type.

A value of 'enumMax(7)' means that the guidance attribute provides data related to the largest enumeration acceptable for the value of the identified component. A corresponding class instance specifying the 'enumMin(6)' value is required in conjunction with this sub-type.

A value of 'enumOnly(8)' means that the guidance attribute provides data related to a single enumeration acceptable for the value of the identified component.

A value of 'valueOnly(9)' means that the guidance attribute provides data related to a single value that is acceptable for the identified component.

A value of 'bitMask(10)' means that the guidance attribute is a bit mask such that all the combinations of bits set in the bitmask are acceptable values for the identified component which should be an attribute of type

'BITS'.

For example, an implementation of the frwkIpFilter class may be limited in several ways, such as address mask, protocol and Layer 4 port options. These limitations could be exported using this PRC with the following instances:

Component	Type	Sub-Type	Guidance
DstPrefixLength	attrValueSupLimited	valueOnly	24
SrcPrefixLength	attrValueSupLimited	valueOnly	24
Protocol	attrValueSupLimited	rangeMin	10
Protocol	attrValueSupLimited	rangeMax	20

The above entries describe a number of limitations that may be in effect for the `frwkIpFilter` class on a given device. The limitations include restrictions on acceptable values for certain attributes.

Also, an implementation of a PRC may be limited in the ways it can be accessed. For instance, for a fictitious PRC `dscpMapEntry`, which has a PIB-ACCESS of 'install-notify':

Component	Type	SubType	Guidance
dscpMapEntry	prcLimitedNotify	none	zero-length string."

```
::= { frwkCompLimitsEntry 6 }
```

`frwkCompLimitsGuidance` OBJECT-TYPE

SYNTAX OCTET STRING

STATUS current

DESCRIPTION

"A value used to convey additional information related to the implementation limitation. Note that a guidance value will not necessarily be provided for all exported limitations. If a guidance value is not provided, the value must be a zero-length string.

The format of the guidance value, if one is present as indicated by the `frwkCompLimitsSubType` attribute, is described by the following table. Note that the format of guidance value is dictated by the base-type of the component whose limitation is being exported, interpreted in the context of the `frwkCompLimitsType` and `frwkCompLimitsSubType` values. Any other restrictions (such as size/range/enumerated value) on the guidance value MUST be complied with according to the definition of the component for which guidance is being specified.

Note that numbers are encoded in network byte order.

Base Type	Value
Unsigned32/Integer32/INTEGER	32-bit value.
Unsigned64/Integer64	64-bit Value.
OCTET STRING	octets of data.
OID	32-bit OID components.
BITS	Binary octets of length same as Component specified."

```
::= { frwkCompLimitsEntry 7 }
```

```
--
-- Complete Reference specification table
--

frwkReferenceTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF FrwkReferenceEntry
    PIB-ACCESS      install-notify
    STATUS          current
    DESCRIPTION
        "Each instance of this PRC specifies a reference to a PRI
        in a specific PIB context (handle) for a specific client-
        type. This table gives the PDP the ability to set up
        policies that span installed contexts and the PEP the
        ability to reference instances in another, perhaps
        configured context. The PEP must send a
        'attrReferenceUnknown' COPS-PR error to the PDP if it
        encounters an invalid reference. "
    REFERENCE
        "COPS Usage for Policy Provisioning. RFC 3084, error
        codes section 4.5."

    ::= { frwkBasePibClasses 5 }

frwkReferenceEntry OBJECT-TYPE
    SYNTAX          FrwkReferenceEntry
    STATUS          current
    DESCRIPTION
        "Entry specification for the frwkReferenceTable."

    PIB-INDEX { frwkReferencePrid }
    UNIQUENESS { }

    ::= { frwkReferenceTable 1 }

FrwkReferenceEntry ::= SEQUENCE {
    frwkReferencePrid      InstanceId,
    frwkReferenceClientType ClientType,
    frwkReferenceClientHandle ClientHandle,
    frwkReferenceInstance  Prid
}

frwkReferencePrid OBJECT-TYPE
    SYNTAX          InstanceId
    STATUS          current
    DESCRIPTION
        "An arbitrary integer index that uniquely identifies an
        instance of the frwkReference class."
```

```
::= { frwkReferenceEntry 1 }
```

```
frwkReferenceClientType OBJECT-TYPE
```

```
SYNTAX          ClientType
```

```
STATUS          current
```

```
DESCRIPTION
```

```
    "Is unused if set to zero else specifies a client-type for
    which the reference is to be interpreted. This non-zero
    client-type must be activated explicitly via a separate
    COPS client-open else this attribute is not valid."
```

```
::= { frwkReferenceEntry 2 }
```

```
frwkReferenceClientHandle OBJECT-TYPE
```

```
SYNTAX          ClientHandle
```

```
STATUS          current
```

```
DESCRIPTION
```

```
    "Must be set to specify a valid client-handle in the scope
    of the client-type specified."
```

```
::= { frwkReferenceEntry 3 }
```

```
frwkReferenceInstance OBJECT-TYPE
```

```
SYNTAX          Prid
```

```
STATUS          current
```

```
DESCRIPTION
```

```
    "References a PRI in the context identified by
    frwkReferenceClientHandle for client-type identified by
    frwkReferenceClientType."
```

```
::= { frwkReferenceEntry 4 }
```

```
--
```

```
-- Error specification table
```

```
--
```

```
frwkErrorTable OBJECT-TYPE
```

```
SYNTAX          SEQUENCE OF FrwkErrorEntry
```

```
PIB-ACCESS      install
```

```
STATUS          current
```

```
DESCRIPTION
```

```
    "Each instance of this PRC specifies a class specific
    error object. Instances of this PRC are transient, i.e.,
    instances received in a COPS decision message must not be
    maintained by the PEP in its copy of the PIB instances. This
    PRC allows a PDP to send error information to the PEP if the
    PDP cannot process updates to a Request successfully."
```

```
::= { frwkBasePibClasses 6 }
```

```
frwkErrorEntry OBJECT-TYPE
```

```
SYNTAX          FrwkErrorEntry
```

```
STATUS          current
```

```
DESCRIPTION
```

```
"Entry specification for the frwkErrorTable."
```

```
PIB-INDEX { frwkErrorPrid }
```

```
UNIQUENESS {
```

```
    frwkErrorCode,
    frwkErrorSubCode,
    frwkErrorPrc,
    frwkErrorInstance
}
```

```
::= { frwkErrorTable 1 }
```

```
FrwkErrorEntry ::= SEQUENCE {
```

```
    frwkErrorPrid      InstanceId,
```

```
    frwkErrorCode      Unsigned32,
```

```
    frwkErrorSubCode    Unsigned32,
```

```
    frwkErrorPrc        PrcIdentifierOid,
```

```
    frwkErrorInstance    InstanceId
```

```
}
```

```
frwkErrorPrid OBJECT-TYPE
```

```
SYNTAX          InstanceId
```

```
STATUS          current
```

```
DESCRIPTION
```

```
"An arbitrary integer index that uniquely identifies an
instance of the frwkError class."
```

```
::= { frwkErrorEntry 1 }
```

```
frwkErrorCode OBJECT-TYPE
```

```
SYNTAX          Unsigned32 (0..65535)
```

```
STATUS          current
```

```
DESCRIPTION
```

```
"Error code defined in COPS-PR CPERR object."
```

```
REFERENCE
```

```
"COPS Usage for Policy Provisioning. RFC 3084."
```

```
::= { frwkErrorEntry 2 }
```

```
frwkErrorSubCode OBJECT-TYPE
```

```
SYNTAX          Unsigned32 (0..65535)
```

```
STATUS          current
```

DESCRIPTION

"The class-specific error object is used to communicate errors relating to specific PRCs."

```
::= { frwkErrorEntry 3 }
```

frwkErrorPrc OBJECT-TYPE

SYNTAX PrcIdentifierOid

STATUS current

DESCRIPTION

"The PRC due to which the error specified by codes (frwkErrorCode , frwkErrorSubCode) occurred."

```
::= { frwkErrorEntry 4 }
```

frwkErrorInstance OBJECT-TYPE

SYNTAX InstanceId

STATUS current

DESCRIPTION

"The PRI of the identified PRC (frwkErrorPrc) due to which the error specified by codes (frwkErrorCode , frwkErrorSubCode) occurred. Must be set to zero if unused."

```
::= { frwkErrorEntry 5 }
```

```
--
```

```
-- The device capabilities and role combo classes group
```

```
--
```

frwkDeviceCapClasses

OBJECT IDENTIFIER ::= { frameworkPib 2 }

```
--
```

```
-- Capability Set Table
```

```
--
```

frwkCapabilitySetTable OBJECT-TYPE

SYNTAX SEQUENCE OF FrwkCapabilitySetEntry

PIB-ACCESS notify

STATUS current

DESCRIPTION

"This PRC describes the capability sets that exist on the interfaces on the device. The capability set is given a unique name that identifies a set. These capability set names are used by the PDP to determine policy information to be associated with interfaces that possess similar sets of capabilities."


```
::= { frwkDeviceCapClasses 1 }
```

```
frwkCapabilitySetEntry OBJECT-TYPE
```

```
SYNTAX      FrwkCapabilitySetEntry
```

```
STATUS      current
```

```
DESCRIPTION
```

"An instance of this PRC describes a particular set of capabilities and associates a unique name with the set."

```
PIB-INDEX { frwkCapabilitySetPrid }
```

```
UNIQUENESS { frwkCapabilitySetName,
              frwkCapabilitySetCapability }
```

```
::= { frwkCapabilitySetTable 1 }
```

```
FrwkCapabilitySetEntry ::= SEQUENCE {
```

```
    frwkCapabilitySetPrid      InstanceId,
```

```
    frwkCapabilitySetName     SnmpAdminString,
```

```
    frwkCapabilitySetCapability Prid
```

```
}
```

```
frwkCapabilitySetPrid OBJECT-TYPE
```

```
SYNTAX      InstanceId
```

```
STATUS      current
```

```
DESCRIPTION
```

"An arbitrary integer index that uniquely identifies a instance of the class."

```
::= { frwkCapabilitySetEntry 1 }
```

```
frwkCapabilitySetName OBJECT-TYPE
```

```
SYNTAX      SnmpAdminString (SIZE (1..255))
```

```
STATUS      current
```

```
DESCRIPTION
```

"The name for the capability set. This name is the unique identifier of a set of capabilities. This attribute must not be assigned a zero-length string."

```
::= { frwkCapabilitySetEntry 2 }
```

```
frwkCapabilitySetCapability OBJECT-TYPE
```

```
SYNTAX      Prid
```

```
STATUS      current
```

```
DESCRIPTION
```

"The complete PRC OID and instance identifier specifying the capability PRC instance for the interface. This attribute references a specific instance of a capability table. The

capability table whose instance is referenced must be defined in the client type specific PIB that this PIB is used with. The referenced capability instance becomes a part of the set of capabilities associated with the specified frwkCapabilitySetName."

```
::= { frwkCapabilitySetEntry 3 }
```

```
--
```

```
-- Interface and Role Combination Tables
```

```
--
```

```
frwkRoleComboTable OBJECT-TYPE
```

```
SYNTAX SEQUENCE OF FrwkRoleComboEntry
```

```
PIB-ACCESS install-notify
```

```
STATUS current
```

```
DESCRIPTION
```

"This is an abstract PRC that may be extended or referenced to enumerate the role combinations, capability set names assigned to any interface on a PEP. The identification of the interface is to be defined by its extensions or referencing PRCs."

```
::= { frwkDeviceCapClasses 2 }
```

```
frwkRoleComboEntry OBJECT-TYPE
```

```
SYNTAX FrwkRoleComboEntry
```

```
STATUS current
```

```
DESCRIPTION
```

"An instance of this PRC describes one association of an interface to a role-combination and capability set name. Note that an interface can have multiple associations. This constraint is controlled by the extending or referencing PRC's uniqueness clause."

```
PIB-INDEX { frwkRoleComboPrid }
```

```
UNIQUENESS { }
```

```
::= { frwkRoleComboTable 1 }
```

```
FrwkRoleComboEntry ::= SEQUENCE {
```

```
frwkRoleComboPrid InstanceId,
```

```
frwkRoleComboRoles RoleCombination,
```

```
frwkRoleComboCapSetName SnmpAdminString
```

```
}
```

```
frwkRoleComboPrid OBJECT-TYPE
```

```
SYNTAX InstanceId
```

STATUS current

DESCRIPTION

"An arbitrary integer index that uniquely identifies an instance of the class."

::= { frwkRoleComboEntry 1 }

frwkRoleComboRoles OBJECT-TYPE

SYNTAX RoleCombination

STATUS current

DESCRIPTION

"The role combination assigned to a specific interface."

::= { frwkRoleComboEntry 2 }

frwkRoleComboCapSetName OBJECT-TYPE

SYNTAX SnmpAdminString (SIZE (0..255))

STATUS current

DESCRIPTION

"The name of the capability set associated with the Role Combination specified in frwkRoleComboRoles. If this is a zero length string it implies the PEP is not exporting any capability set information for this RoleCombination. The PDP must then use the RoleCombinations provided as the only means of assigning policies. If a non-zero length string is specified, the name must exist in frwkCapabilitySetTable."

::= { frwkRoleComboEntry 3 }

--

-- Interface, Role Combination association via IfIndex

--

frwkIfRoleComboTable OBJECT-TYPE

SYNTAX SEQUENCE OF FrwkIfRoleComboEntry

PIB-ACCESS install-notify

STATUS current

DESCRIPTION

"This PRC enumerates the interface to role combination and frwkRoleComboCapSetName mapping for all policy managed interfaces of a device. Policy for an interface depends not only on the capability set of an interface but also on its roles. This table specifies all the <interface index, interface capability set name, role combination> tuples currently on the device"

::= { frwkDeviceCapClasses 3 }

```

frwkIfRoleComboEntry OBJECT-TYPE
    SYNTAX          FrwkIfRoleComboEntry
    STATUS          current
    DESCRIPTION
        "An instance of this PRC describes the association of
        a interface to an capability set name and a role
        combination.
        Note that a capability set name can have multiple role
        combinations assigned to it, but an IfIndex can have only
        one role combination associated."

    EXTENDS { frwkRoleComboEntry }
    UNIQUENESS { frwkIfRoleComboIfIndex,
                  frwkRoleComboCapSetName }

    ::= { frwkIfRoleComboTable 1 }

FrwkIfRoleComboEntry ::= SEQUENCE {
    frwkIfRoleComboIfIndex      InterfaceIndex
}

frwkIfRoleComboIfIndex OBJECT-TYPE
    SYNTAX          InterfaceIndex
    STATUS          current
    DESCRIPTION
        "The value of this attribute is the ifIndex which is
        associated with the specified RoleCombination and interface
        capability set name."

    ::= { frwkIfRoleComboEntry 1 }

--
-- The Classification classes group
--

frwkClassifierClasses
    OBJECT IDENTIFIER ::= { frameworkPib 3 }

--
-- The Base Filter Table
--

frwkBaseFilterTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF FrwkBaseFilterEntry
    PIB-ACCESS      install
    STATUS          current

```

DESCRIPTION

"The Base Filter class. A packet has to match all fields in an Filter. Wildcards may be specified for those fields that are not relevant."

::= { frwkClassifierClasses 1 }

frwkBaseFilterEntry OBJECT-TYPE

SYNTAX FrwkBaseFilterEntry

STATUS current

DESCRIPTION

"An instance of the frwkBaseFilter class."

PIB-INDEX { frwkBaseFilterPrid }

::= { frwkBaseFilterTable 1 }

FrwkBaseFilterEntry ::= SEQUENCE {

frwkBaseFilterPrid InstanceId,

frwkBaseFilterNegation TruthValue

}

frwkBaseFilterPrid OBJECT-TYPE

SYNTAX InstanceId

STATUS current

DESCRIPTION

"An integer index to uniquely identify this Filter among all the Filters."

::= { frwkBaseFilterEntry 1 }

frwkBaseFilterNegation OBJECT-TYPE

SYNTAX TruthValue

STATUS current

DESCRIPTION

"This attribute behaves like a logical NOT for the filter. If the packet matches this filter and the value of this attribute is 'true', the action associated with this filter is not applied to the packet. If the value of this attribute is 'false', then the action is applied to the packet."

::= { frwkBaseFilterEntry 2 }

--

-- The IP Filter Table

--

```

frwkIpFilterTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF FrwkIpFilterEntry
    PIB-ACCESS      install
    STATUS          current
    DESCRIPTION
        "Filter definitions.  A packet has to match all fields in a
        filter.  Wildcards may be specified for those fields that
        are not relevant."

    INSTALL-ERRORS {
        invalidDstL4PortData(1),
        invalidSrcL4PortData(2)
    }
    ::= { frwkClassifierClasses 2 }

frwkIpFilterEntry OBJECT-TYPE
    SYNTAX          FrwkIpFilterEntry
    STATUS          current
    DESCRIPTION
        "An instance of the frwkIpFilter class."

    EXTENDS { frwkBaseFilterEntry }
    UNIQUENESS { frwkBaseFilterNegation,
        frwkIpFilterAddrType,
        frwkIpFilterDstAddr,
        frwkIpFilterDstPrefixLength,
        frwkIpFilterSrcAddr,
        frwkIpFilterSrcPrefixLength,
        frwkIpFilterDscp,
        frwkIpFilterFlowId,
        frwkIpFilterProtocol,
        frwkIpFilterDstL4PortMin,
        frwkIpFilterDstL4PortMax,
        frwkIpFilterSrcL4PortMin,
        frwkIpFilterSrcL4PortMax }

    ::= { frwkIpFilterTable 1 }

FrwkIpFilterEntry ::= SEQUENCE {
    frwkIpFilterAddrType      InetAddressType,
    frwkIpFilterDstAddr       InetAddress,
    frwkIpFilterDstPrefixLength  InetAddressPrefixLength,
    frwkIpFilterSrcAddr       InetAddress,
    frwkIpFilterSrcPrefixLength  InetAddressPrefixLength,
    frwkIpFilterDscp          DscpOrAny,
    frwkIpFilterFlowId        Integer32,
    frwkIpFilterProtocol      Unsigned32,
    frwkIpFilterDstL4PortMin   InetPortNumber,

```

```
frwkIpFilterDstL4PortMax      InetPortNumber,
frwkIpFilterSrcL4PortMin      InetPortNumber,
frwkIpFilterSrcL4PortMax      InetPortNumber
}
```

frwkIpFilterAddrType OBJECT-TYPE

```
SYNTAX      InetAddressType
STATUS      current
```

DESCRIPTION

"The address type enumeration value to specify the type of the packet's IP address.

While other types of addresses are defined in the InetAddressType textual convention, an IP filter can only use IPv4 and IPv6 addresses directly to classify traffic. All other InetAddressTypes require mapping to the corresponding Ipv4 or IPv6 address before being used to classify traffic. Therefore, this object as such is not limited to IPv4 and IPv6 addresses, i.e., it can be assigned any of the valid values defined in the InetAddressType TC, but the mapping of the address values to IPv4 or IPv6 addresses for the address attributes (frwkIpFilterDstAddr and frwkIpFilterSrcAddr) must be done by the PEP. For example when dns (16) is used, the PEP must resolve the address to IPv4 or IPv6 at install time."

REFERENCE

"Textual Conventions for Internet Network Addresses.
RFC 3291."

```
::= { frwkIpFilterEntry 1 }
```

frwkIpFilterDstAddr OBJECT-TYPE

```
SYNTAX      InetAddress
STATUS      current
```

DESCRIPTION

"The IP address to match against the packet's destination IP address. If the address type is 'ipv4', 'ipv6', 'ipv4z' or 'ipv6z' then, the attribute frwkIpFilterDstPrefixLength indicates the number of bits that are relevant. "

REFERENCE

"Textual Conventions for Internet Network Addresses.
RFC 3291."

```
::= { frwkIpFilterEntry 2 }
```

frwkIpFilterDstPrefixLength OBJECT-TYPE

SYNTAX InetAddressPrefixLength

STATUS current

DESCRIPTION

"The length of a mask for the matching of the destination IP address. This attribute is interpreted only if the InetAddressType is 'ipv4', 'ipv4z', 'ipv6' or 'ipv6z'. Masks are constructed by setting bits in sequence from the most-significant bit downwards for frwkIpFilterDstPrefixLength bits length. All other bits in the mask, up to the number needed to fill the length of the address frwkIpFilterDstAddr are cleared to zero. A zero bit in the mask then means that the corresponding bit in the address always matches.

In IPv4 addresses, a length of 0 indicates a match of any address; a length of 32 indicates a match of a single host address, and a length between 0 and 32 indicates the use of a CIDR Prefix. IPv6 is similar, except that prefix lengths range from 0..128."

REFERENCE

"Textual Conventions for Internet Network Addresses.
RFC 3291."

DEFVAL { 0 }

::= { frwkIpFilterEntry 3 }

frwkIpFilterSrcAddr OBJECT-TYPE

SYNTAX InetAddress

STATUS current

DESCRIPTION

"The IP address to match against the packet's source IP address. If the address type is 'ipv4', 'ipv6', 'ipv4z' or 'ipv6z' then, the attribute frwkIpFilterSrcPrefixLength indicates the number of bits that are relevant."

REFERENCE

"Textual Conventions for Internet Network Addresses.
RFC 3291."

::= { frwkIpFilterEntry 4 }

frwkIpFilterSrcPrefixLength OBJECT-TYPE

SYNTAX InetAddressPrefixLength

UNITS "bits"

STATUS current

DESCRIPTION

"The length of a mask for the matching of the source IP address. This attribute is interpreted only if the

InetAddressType is 'ipv4', 'ipv4z', 'ipv6' or 'ipv6z'. Masks are constructed by setting bits in sequence from the most-significant bit downwards for frwkIpFilterSrcPrefixLength bits length. All other bits in the mask, up to the number needed to fill the length of the address frwkIpFilterSrcAddr are cleared to zero. A zero bit in the mask then means that the corresponding bit in the address always matches.

In IPv4 addresses, a length of 0 indicates a match of any address; a length of 32 indicates a match of a single host address, and a length between 0 and 32 indicates the use of a CIDR Prefix. IPv6 is similar, except that prefix lengths range from 0..128."

REFERENCE

"Textual Conventions for Internet Network Addresses.
RFC 3291."

DEFVAL { 0 }

::= { frwkIpFilterEntry 5 }

frwkIpFilterDscp OBJECT-TYPE

SYNTAX DscpOrAny

STATUS current

DESCRIPTION

"The value that the DSCP in the packet can have and match this filter. A value of -1 indicates that a specific DSCP value has not been defined and thus all DSCP values are considered a match."

REFERENCE

"Management Information Base for the Differentiated Services Architecture. RFC 3289."

DEFVAL { -1 }

::= { frwkIpFilterEntry 6 }

frwkIpFilterFlowId OBJECT-TYPE

SYNTAX Integer32 (-1 | 0..1048575)

STATUS current

DESCRIPTION

"The flow label or flow identifier in an IPv6 header that may be used to discriminate traffic flows. The value of -1 for this attribute MUST imply that any flow label value in the IPv6 header will match, resulting in the flow label field of the IPv6 header being ignored for matching this filter entry."

::= { frwkIpFilterEntry 7 }

frwkIpFilterProtocol OBJECT-TYPE

SYNTAX Unsigned32 (0..255)

STATUS current

DESCRIPTION

"The layer-4 protocol Id to match against the IPv4 protocol number or the IPv6 Next-Header number in the packet. A value of 255 means match all. Note the protocol number of 255 is reserved by IANA, and Next-Header number of 0 is used in IPv6."

DEFVAL { 255 }

::= { frwkIpFilterEntry 8 }

frwkIpFilterDstL4PortMin OBJECT-TYPE

SYNTAX InetPortNumber

STATUS current

DESCRIPTION

"The minimum value that the packet's layer 4 destination port number can have and match this filter. This value must be equal to or lesser than the value specified for this filter in frwkIpFilterDstL4PortMax."

COPS-PR error code 'attrValueInvalid' must be returned if the frwkIpFilterSrcL4PortMin is greater than frwkIpFilterSrcL4PortMax"

REFERENCE

"COPS Usage for Policy Provisioning. RFC 3084, error codes section 4.5."

DEFVAL { 0 }

::= { frwkIpFilterEntry 9 }

frwkIpFilterDstL4PortMax OBJECT-TYPE

SYNTAX InetPortNumber

STATUS current

DESCRIPTION

"The maximum value that the packet's layer 4 destination port number can have and match this filter. This value must be equal to or greater than the value specified for this filter in frwkIpFilterDstL4PortMin."

COPS-PR error code 'attrValueInvalid' must be returned if the frwkIpFilterDstL4PortMax is less than frwkIpFilterDstL4PortMin"

REFERENCE

"COPS Usage for Policy Provisioning. RFC 3084, error codes section 4.5."

```
DEFVAL { 65535 }
```

```
::= { frwkIpFilterEntry 10 }
```

```
frwkIpFilterSrcL4PortMin OBJECT-TYPE
```

```
SYNTAX      InetPortNumber
```

```
STATUS      current
```

```
DESCRIPTION
```

"The minimum value that the packet's layer 4 source port number can have and match this filter. This value must be equal to or lesser that the value specified for this filter in frwkIpFilterSrcL4PortMax.

COPS-PR error code 'attrValueInvalid' must be returned if the frwkIpFilterSrcL4PortMin is greated than frwkIpFilterSrcL4PortMax"

```
REFERENCE
```

"COPS Usage for Policy Provisioning. RFC 3084, error codes section 4.5."

```
DEFVAL { 0 }
```

```
::= { frwkIpFilterEntry 11 }
```

```
frwkIpFilterSrcL4PortMax OBJECT-TYPE
```

```
SYNTAX      InetPortNumber
```

```
STATUS      current
```

```
DESCRIPTION
```

"The maximum value that the packet's layer 4 source port number can have and match this filter. This value must be equal to or greater that the value specified for this filter in frwkIpFilterSrcL4PortMin.

COPS-PR error code 'attrValueInvalid' must be returned if the frwkIpFilterSrcL4PortMax is less than frwkIpFilterSrcL4PortMin"

```
REFERENCE
```

"COPS Usage for Policy Provisioning. RFC error codes section 4.5."

```
DEFVAL { 65535 }
```

```
::= { frwkIpFilterEntry 12 }
```

```
--
```

```
-- The IEEE 802 Filter Table
```

```
--
```

```
frwk802FilterTable OBJECT-TYPE
```

```
SYNTAX      SEQUENCE OF Frwk802FilterEntry
```

```

PIB-ACCESS      install
STATUS          current
DESCRIPTION
    "IEEE 802-based filter definitions. A class that contains
    attributes of IEEE 802 (e.g., 802.3) traffic that form
    filters that are used to perform traffic classification."
REFERENCE
    "IEEE Standards for Local and Metropolitan Area Networks.
    Overview and Architecture, ANSI/IEEE Std 802, 1990."
::= { frwkClassifierClasses 3 }

```

```

frwk802FilterEntry OBJECT-TYPE
SYNTAX          Frwk802FilterEntry
STATUS          current
DESCRIPTION
    "IEEE 802-based filter 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."

```

```

EXTENDS { frwkBaseFilterEntry }
UNIQUENESS { frwkBaseFilterNegation,
              frwk802FilterDstAddr,
              frwk802FilterDstAddrMask,
              frwk802FilterSrcAddr,
              frwk802FilterSrcAddrMask,
              frwk802FilterVlanId,
              frwk802FilterVlanTagRequired,
              frwk802FilterEtherType,
              frwk802FilterUserPriority }

```

```

::= { frwk802FilterTable 1 }

```

```

Frwk802FilterEntry ::= SEQUENCE {
    frwk802FilterDstAddr      PhysAddress,
    frwk802FilterDstAddrMask  PhysAddress,
    frwk802FilterSrcAddr      PhysAddress,
    frwk802FilterSrcAddrMask  PhysAddress,
    frwk802FilterVlanId       Integer32,
    frwk802FilterVlanTagRequired INTEGER,
    frwk802FilterEtherType     Integer32,
    frwk802FilterUserPriority  BITS
}

```

}

frwk802FilterDstAddr 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 frwk802FilterDstAddrMask object, are potentially subject to the processing guidelines that are associated with this entry through the related action class."

REFERENCE

"Textual Conventions for SMIV2, RFC 2579."

::= { frwk802FilterEntry 1 }

frwk802FilterDstAddrMask 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 frwk802FilterDstAddr 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 frwk802FilterDstAddr address. A zero bit in the mask thus means that the corresponding bit in the address always matches. The frwk802FilterDstAddr 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 frwk802FilterDstAddr. Note that a mask with no bits set (i.e., all zeroes) effectively wildcards the frwk802FilterDstAddr object."

::= { frwk802FilterEntry 2 }

frwk802FilterSrcAddr 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 frwk802FilterSrcAddrMask object, are potentially subject to the processing guidelines that are associated with this entry through the related action class."

```
::= { frwk802FilterEntry 3 }
```

frwk802FilterSrcAddrMask 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 frwk802FilterSrcAddr 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 frwk802FilterSrcAddr address. A zero bit in the mask thus means that the corresponding bit in the address always matches. The frwk802FilterSrcAddr 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 frwk802FilterSrcAddr. Note that a mask with no bits set (i.e., all zeroes) effectively wildcards the frwk802FilterSrcAddr object."

```
::= { frwk802FilterEntry 4 }
```

frwk802FilterVlanId 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 frwk802FilterVlanId object to -1 indicates that VLAN data should not be considered during traffic classification."

```
::= { frwk802FilterEntry 5 }
```

frwk802FilterVlanTagRequired 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 filter 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)'."

```
 ::= { frwk802FilterEntry 6 }
```

frwk802FilterEtherType 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 frwk802FilterEtherTypeMin 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 filter component is active when there is no EtherType field in a frame (e.g., 802.2 LLC), a match is implied."

```
::= { frwk802FilterEntry 7 }
```

frwk802FilterUserPriority 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 filter component. Multiple values may be set at one time such that potentially several different user priority values may match this 802 filter 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."

```
::= { frwk802FilterEntry 8 }
```

```
--
-- The Internal label filter extension
--
```

frwkILabelFilterTable OBJECT-TYPE

```
SYNTAX          SEQUENCE OF FrwkILabelFilterEntry
PIB-ACCESS      install
STATUS          current
```


DESCRIPTION

"Internal label filter Table. This PRC is used to achieve classification based on the internal flow label set by the PEP possibly after ingress classification to avoid re-classification at the egress interface on the same PEP."

```
::= { frwkClassifierClasses 4 }
```

frwkILabelFilterEntry OBJECT-TYPE

```
SYNTAX          FrwkILabelFilterEntry
```

```
STATUS          current
```

DESCRIPTION

"Internal label filter entry definition."

```
EXTENDS { frwkBaseFilterEntry }
```

```
UNIQUENESS { frwkBaseFilterNegation,
              frwkILabelFilterILabel }
```

```
::= { frwkILabelFilterTable 1 }
```

FrwkILabelFilterEntry ::= SEQUENCE {

```
    frwkILabelFilterILabel      OCTET STRING
```

```
}
```

frwkILabelFilterILabel OBJECT-TYPE

```
SYNTAX          OCTET STRING
```

```
STATUS          current
```

DESCRIPTION

"The Label that this flow uses for differentiating traffic flows. The flow labeling is meant for network device internal usage. A value of zero length string matches all internal labels."

```
::= { frwkILabelFilterEntry 1 }
```

```
--
```

```
-- The Marker classes group
```

```
--
```

frwkMarkerClasses

```
OBJECT IDENTIFIER ::= { frameworkPib 4 }
```

```
--
```

```
-- The 802 Marker Table
```

```
--
```

frwk802MarkerTable OBJECT-TYPE

```
SYNTAX          SEQUENCE OF Frwk802MarkerEntry
```

```
PIB-ACCESS      install
```

```
STATUS          current
```

DESCRIPTION

"The 802 Marker class. An 802 packet can be marked with the specified VLAN id, priority level."

::= { frwkMarkerClasses 1 }

frwk802MarkerEntry OBJECT-TYPE

SYNTAX Frwk802MarkerEntry

STATUS current

DESCRIPTION

"frwk802Marker entry definition."

PIB-INDEX { frwk802MarkerPrid }

UNIQUENESS { frwk802MarkerVlanId,
frwk802MarkerPriority }

::= { frwk802MarkerTable 1 }

Frwk802MarkerEntry ::= SEQUENCE {

frwk802MarkerPrid InstanceId,

frwk802MarkerVlanId Unsigned32,

frwk802MarkerPriority Unsigned32

}

frwk802MarkerPrid OBJECT-TYPE

SYNTAX InstanceId

STATUS current

DESCRIPTION

"An integer index to uniquely identify this 802 Marker."

::= { frwk802MarkerEntry 1 }

frwk802MarkerVlanId OBJECT-TYPE

SYNTAX Unsigned32 (1..4094)

STATUS current

DESCRIPTION

"The VLAN ID (VID) that uniquely identifies a VLAN within the device."

::= { frwk802MarkerEntry 2 }

frwk802MarkerPriority OBJECT-TYPE

SYNTAX Unsigned32 (0..7)

STATUS current

DESCRIPTION

"The user priority field of a tagged 802.1 frame."

::= { frwk802MarkerEntry 3 }

```
--
-- The Internal Label Marker Table
--

frwkILabelMarkerTable OBJECT-TYPE
    SYNTAX          SEQUENCE OF FrwkILabelMarkerEntry
    PIB-ACCESS      install
    STATUS          current
    DESCRIPTION
        "The Internal Label Marker class. A flow in a PEP can be
        marked with an internal label using this PRC."

    ::= { frwkMarkerClasses 2 }

frwkILabelMarkerEntry OBJECT-TYPE
    SYNTAX          FrwkILabelMarkerEntry
    STATUS          current
    DESCRIPTION
        "frwkILabelkMarker entry definition."

    PIB-INDEX { frwkILabelMarkerPrid }
    UNIQUENESS { frwkILabelMarkerILabel }

    ::= { frwkILabelMarkerTable 1 }

FrwkILabelMarkerEntry ::= SEQUENCE {
    frwkILabelMarkerPrid      InstanceId,
    frwkILabelMarkerILabel    OCTET STRING
}

frwkILabelMarkerPrid OBJECT-TYPE
    SYNTAX          InstanceId
    STATUS          current
    DESCRIPTION
        "An integer index to uniquely identify this Label Marker."

    ::= { frwkILabelMarkerEntry 1 }

frwkILabelMarkerILabel OBJECT-TYPE
    SYNTAX          OCTET STRING
    STATUS          current
    DESCRIPTION
        "This internal label is implementation specific and may be
        used for other policy related functions like flow
        accounting purposes and/or other data path treatments."

    ::= { frwkILabelMarkerEntry 2 }
```

```
--
-- Conformance Section
--

frwkBasePibConformance
    OBJECT IDENTIFIER ::= { frameworkPib 5 }

frwkBasePibCompliances
    OBJECT IDENTIFIER ::= { frwkBasePibConformance 1 }

frwkBasePibGroups
    OBJECT IDENTIFIER ::= { frwkBasePibConformance 2 }

frwkBasePibCompliance MODULE-COMPLIANCE
    STATUS current
    DESCRIPTION
        "Describes the requirements for conformance to the
        Framework PIB."

    MODULE -- this module
        MANDATORY-GROUPS { frwkPrcSupportGroup,
                           frwkPibIncarnationGroup,
                           frwkDeviceIdGroup,
                           frwkCompLimitsGroup,
                           frwkCapabilitySetGroup,
                           frwkRoleComboGroup,
                           frwkIfRoleComboGroup }

    OBJECT          frwkPibIncarnationLongevity
    PIB-MIN-ACCESS  notify
    DESCRIPTION
        "Install support is required if policy expiration is to
        be supported."

    OBJECT          frwkPibIncarnationTtl
    PIB-MIN-ACCESS  notify
    DESCRIPTION
        "Install support is required if policy expiration is to
        be supported."

    OBJECT          frwkPibIncarnationInCtxtSet
    PIB-MIN-ACCESS  notify
    DESCRIPTION
        "Install support is required if configuration contexts
        and outsourcing contexts are both to be supported."

    OBJECT          frwkPibIncarnationFullState
```

```
PIB-MIN-ACCESS  notify
DESCRIPTION
    "Install support is required if incremental updates to
    request states is to be supported."

GROUP    frwkReferenceGroup
DESCRIPTION
    "The frwkReferenceGroup is mandatory if referencing
    across PIB contexts for specific client-types is to be
    supported."

GROUP    frwkErrorGroup
DESCRIPTION
    "The frwkErrorGroup is mandatory sending errors in
    decisions is to be supported."

GROUP    frwkBaseFilterGroup
DESCRIPTION
    "The frwkBaseFilterGroup is mandatory if filtering
    based on traffic components is to be supported."

GROUP    frwkIpFilterGroup
DESCRIPTION
    "The frwkIpFilterGroup is mandatory if filtering
    based on IP traffic components is to be supported."

GROUP    frwk802FilterGroup
DESCRIPTION
    "The frwk802FilterGroup is mandatory if filtering
    based on 802 traffic criteria is to be supported."

GROUP    frwkILabelFilterGroup
DESCRIPTION
    "The frwkILabelFilterGroup is mandatory if filtering
    based on PEP internal label is to be supported."

GROUP    frwk802MarkerGroup
DESCRIPTION
    "The frwk802MarkerGroup is mandatory if marking a packet
    with 802 traffic criteria is to be supported."

GROUP    frwkILabelMarkerGroup
DESCRIPTION
    "The frwkILabelMarkerGroup is mandatory if marking a
    flow with internal labels is to be supported."

 ::= { frwkBasePibCompliances 1 }
```

```
frwkPrcSupportGroup OBJECT-GROUP
  OBJECTS {
    frwkPrcSupportPrid,
    frwkPrcSupportSupportedPrc,
    frwkPrcSupportSupportedAttrs }
  STATUS current
  DESCRIPTION
    "Objects from the frwkPrcSupportTable."

  ::= { frwkBasePibGroups 1 }

frwkPibIncarnationGroup OBJECT-GROUP
  OBJECTS {
    frwkPibIncarnationPrid,
    frwkPibIncarnationName,
    frwkPibIncarnationId,
    frwkPibIncarnationLongevity,
    frwkPibIncarnationTtl,
    frwkPibIncarnationInCtxtSet,
    frwkPibIncarnationActive,
    frwkPibIncarnationFullState
  }
  STATUS current
  DESCRIPTION
    "Objects from the frwkDevicePibIncarnationTable."

  ::= { frwkBasePibGroups 2 }

frwkDeviceIdGroup OBJECT-GROUP
  OBJECTS {
    frwkDeviceIdPrid,
    frwkDeviceIdDescr,
    frwkDeviceIdMaxMsg,
    frwkDeviceIdMaxContexts }
  STATUS current
  DESCRIPTION
    "Objects from the frwkDeviceIdTable."

  ::= { frwkBasePibGroups 3 }

frwkCompLimitsGroup OBJECT-GROUP
  OBJECTS {
    frwkCompLimitsPrid,
    frwkCompLimitsComponent,
    frwkCompLimitsAttrPos,
    frwkCompLimitsNegation,
    frwkCompLimitsType,
    frwkCompLimitsSubType,
```

```
        frwkCompLimitsGuidance }
STATUS   current
DESCRIPTION
    "Objects from the frwkCompLimitsTable."

 ::= { frwkBasePibGroups 4 }

frwkReferenceGroup OBJECT-GROUP
OBJECTS {
    frwkReferencePrid,
    frwkReferenceClientType,
    frwkReferenceClientHandle,
    frwkReferenceInstance }
STATUS   current
DESCRIPTION
    "Objects from the frwkReferenceTable."

 ::= { frwkBasePibGroups 5 }

frwkErrorGroup OBJECT-GROUP
OBJECTS {
    frwkErrorPrid,
    frwkErrorCode,
    frwkErrorSubCode,
    frwkErrorPrc,
    frwkErrorInstance }
STATUS   current
DESCRIPTION
    "Objects from the frwkErrorTable."

 ::= { frwkBasePibGroups 6 }

frwkCapabilitySetGroup OBJECT-GROUP
OBJECTS {
    frwkCapabilitySetPrid,
    frwkCapabilitySetName,
    frwkCapabilitySetCapability }
STATUS   current
DESCRIPTION
    "Objects from the frwkCapabilitySetTable."

 ::= { frwkBasePibGroups 7 }

frwkRoleComboGroup OBJECT-GROUP
OBJECTS {
    frwkRoleComboPrid,
    frwkRoleComboRoles,
    frwkRoleComboCapSetName }
```

```
STATUS    current
DESCRIPTION
    "Objects from the frwkRoleComboTable."
```

```
::= { frwkBasePibGroups 8 }
```

```
frwkIfRoleComboGroup OBJECT-GROUP
    OBJECTS { frwkIfRoleComboIfIndex }
    STATUS    current
    DESCRIPTION
        "Objects from the frwkIfRoleComboTable."
```

```
::= { frwkBasePibGroups 9 }
```

```
frwkBaseFilterGroup OBJECT-GROUP
    OBJECTS {
        frwkBaseFilterPrid,
        frwkBaseFilterNegation }
    STATUS    current
    DESCRIPTION
        "Objects from the frwkBaseFilterTable."
```

```
::= { frwkBasePibGroups 10 }
```

```
frwkIpFilterGroup OBJECT-GROUP
    OBJECTS {
        frwkIpFilterAddrType,
        frwkIpFilterDstAddr,
        frwkIpFilterDstPrefixLength,
        frwkIpFilterSrcAddr,
        frwkIpFilterSrcPrefixLength,
        frwkIpFilterDscp,
        frwkIpFilterFlowId,
        frwkIpFilterProtocol,
        frwkIpFilterDstL4PortMin,
        frwkIpFilterDstL4PortMax,
        frwkIpFilterSrcL4PortMin,
        frwkIpFilterSrcL4PortMax }
    STATUS    current
    DESCRIPTION
        "Objects from the frwkIpFilterTable."
```

```
::= { frwkBasePibGroups 11 }
```

```
frwk802FilterGroup OBJECT-GROUP
    OBJECTS {
        frwk802FilterDstAddr,
        frwk802FilterDstAddrMask,
```



```
        frwk802FilterSrcAddr,
        frwk802FilterSrcAddrMask,
        frwk802FilterVlanId,
        frwk802FilterVlanTagRequired,
        frwk802FilterEtherType,
        frwk802FilterUserPriority }
STATUS    current
DESCRIPTION
    "Objects from the frwk802FilterTable."

 ::= { frwkBasePibGroups 12 }

frwkILabelFilterGroup OBJECT-GROUP
  OBJECTS { frwkILabelFilterILabel }
  STATUS    current
  DESCRIPTION
    "Objects from the frwkILabelFilterTable."

 ::= { frwkBasePibGroups 13 }

frwk802MarkerGroup OBJECT-GROUP
  OBJECTS {
    frwk802MarkerPrid,
    frwk802MarkerVlanId,
    frwk802MarkerPriority }
  STATUS    current
  DESCRIPTION
    "Objects from the frwk802MarkerTable."

 ::= { frwkBasePibGroups 14 }

frwkILabelMarkerGroup OBJECT-GROUP
  OBJECTS {
    frwkILabelMarkerPrid,
    frwkILabelMarkerILabel }
  STATUS    current
  DESCRIPTION
    "Objects from the frwkILabelMarkerTable."

 ::= { frwkBasePibGroups 15 }

END
```

6. Security Considerations

It is clear that this PIB is used for configuration using [COPS-PR], and anything that can be configured can be misconfigured, with a potentially disastrous effect. At this writing, no security holes have been identified beyond those that the COPS base protocol security is itself intended to address. These relate primarily to controlled access to sensitive information and the ability to configure a device - or which might result from operator error, which is beyond the scope of any security architecture.

There are a number of PROvisioning Classes defined in this PIB that have a PIB-ACCESS clause of install and install-notify (read-create). These are:

frwkPibIncarnationTable: Malicious access of this PRC can cause the PEP to use an incorrect context of policies.

frwkReferenceTable: Malicious access of this PRC can cause the PEP to interpret the installed policy in an incorrect manner.

frwkErrorTable: Malicious access of this PRC can cause the PEP to incorrectly assume that the PDP could not process its messages.

FrwkCapabilitySetTable, frwkRoleComboTable and frwkIfRoleComboTable: Malicious access of these PRCs can cause the PEP to apply policies to the wrong interfaces.

FrwkBaseFilterTable, frwkIpFilterTable, frwk802FilterTable and frwkILabelFilterTable: Malicious access of these PRCs can cause unintended classification of traffic on the PEP potentially leading to incorrect policies being applied.

frwk802MarkerTable, frwkILabelMarkerTable: Malicious access of these PRCs can cause unintended marking of traffic on the PEP potentially leading to incorrect policies being applied.

Such objects may be considered sensitive or vulnerable in some network environments. The support for "Install" or "Install-Notify" decisions sent over [COPS-PR] in a non-secure environment without proper protection can have a negative effect on network operations. There are a number of PROvisioning Classes in this PIB that may contain information that may be sensitive from a business perspective, in that they may represent a customer's service contract or the filters that the service provider chooses to apply to a customer's ingress or egress traffic. There are no PRCs that are sensitive in their own right, such as passwords or monetary amounts. It may be important to control even "Notify"(read-only) access to

these PRCs and possibly to even encrypt the values of these PRIs when sending them over the network via COPS-PR. The use of IPSEC between the PDP and the PEP, as described in [COPS], provides the necessary protection against security threats. However, even if the network itself is secure, there is no control as to who on the secure network is allowed to "Install/Notify" (read/change/create/delete) the PRIs in this PIB.

It is then a customer/user responsibility to ensure that the PEP/PDP giving access to an instance of this PIB, is properly configured to give access to only the PRIs and principals (users) that have legitimate rights to indeed "Install" or "Notify" (change/create/delete) them.

7. IANA Considerations

This document describes the frameworkPib and frwkTcPib Policy Information Base (PIB) modules for registration under the "pib" branch registered with IANA. The IANA has assigned PIB numbers 2 and 3, respectively.

Both these PIBs use "all" in the SUBJECT-CATEGORIES clause, i.e., they apply to all COPS client types. No new COPS client type is to be registered for these two PIB modules.

8. References

8.1 Normative References

- | | |
|------------|--|
| [COPS] | Boyle, J., Cohen, R., Durham, D., Herzog, S., Rajan, R. and A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748, January 2000. |
| [COPS-PR] | Chan, K., Durham, D., Gai, S., Herzog, S., McCloghrie, K., Reichmeyer, Seligson, J., Smith, A. and R. Yavatkar, "COPS Usage for Policy Provisioning", RFC 3084, March 2001. |
| [SPPI] | McCloghrie, K., Fine, M., Seligson, J., Chan, K., Hahn, S., Sahita, R., Smith, A. and F. Reichmeyer, "Structure of Policy Provisioning Information", RFC 3159, August 2001. |
| [SNMP-SMI] | McCloghrie, K., Perkins, D., Schoenwaelder, J., Case, J., Rose, M. and S. Waldbusser, "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, April 1999. |

- [INETADDR] Daniele, M., Haberman, B., Routhier, S. and J. Schoenwaelder, "Textual Conventions for Internet Network Addresses", RFC 3291, May 2002.
- [802] IEEE Standards for Local and Metropolitan Area Networks: Overview and Architecture, ANSI/IEEE Std 802, 1990.
- [SNMPFRWK] Harrington, D., Presuhn, R. and B. Wijnen, "An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks", STD 62, RFC 3411, December 2002.
- [RFC2863] McCloghrie, K. and F. Kastenholz, "The Interfaces Group MIB", RFC 2863, June 2000.
- [DS-MIB] Baker, F., Chan, K. and A. Smith, "Management Information Base for the Differentiated Services Architecture", RFC 3289, May 2002.
- [SNMPv2TC] McCloghrie, K., Perkins, D., and J. Schoenwaelder, "Textual Conventions for SMIV2", STD 58, RFC 2579, April 1999.
- [RFC2279] Yergeau, F. "UTF-8, a transformation format of ISO 10646", RFC 2279, January 1998.
- [RFC2119] Bradner, S., "Key words to use in the RFCs", BCP 14, RFC 2119, March 1997.

8.2 Informative References

- [RAP-FRAMEWORK] Yavatkar, R and D. Pendarakis, "A Framework for Policy-based Admission Control", RFC 2753, January 2000.
- [POLTERM] Westerinen, A., Schnizlein, J., Strassner, J., Scherling, M., Quinn, B., Herzog, S., Huynh, A., Carlson, M., Perry, J. and S. Waldbusser, "Terminology for Policy-Based Management", RFC 3198, November 2001.

9. Acknowledgments

Early versions of this specification were also co-authored by Michael Fine, Francis Reichmeyer, John Seligson and Andrew Smith.

Special thanks to Carol Bell, David Durham and Bert Wijnen for their many significant comments.

Additional useful comments have been made by Diana Rawlins, Martin Bokaemper, Tina Iliff, Pedro Da Silva, Juergen Schoenwaelder, Noisette Yoann and Man Li.

10. Authors' Addresses

Ravi Sahita
Intel Labs.
2111 NE 25th Avenue
Hillsboro, OR 97124 USA

Phone: +1 503 712 1554
EMail: ravi.sahita@intel.com

Scott Hahn
Intel Corp.
2111 NE 25th Avenue
Hillsboro, OR 97124 USA

Phone: +1 503 264 8231
EMail: scott.hahn@intel.com

Kwok Ho Chan
Nortel Networks, Inc.
600 Technology Park Drive
Billerica, MA 01821 USA

Phone: +1 978 288 8175
EMail: khchan@nortelnetworks.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

11. Full Copyright Statement

Copyright (C) The Internet Society (2003). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.

