Network Working Group
Request for Comments: 5394
Category: Informational

I. Bryskin
Adva Optical
D. Papadimitriou
Alcatel
L. Berger
LabN Consulting
J. Ash
AT&T

December 2008

Policy-Enabled Path Computation Framework

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

Abstract

The Path Computation Element (PCE) architecture introduces the concept of policy in the context of path computation. This document provides additional details on policy within the PCE architecture and also provides context for the support of PCE Policy. This document introduces the use of the Policy Core Information Model (PCIM) as a framework for supporting path computation policy. This document also provides representative scenarios for the support of PCE Policy.

Table of Contents

1. Introduction

The Path Computation Element (PCE) Architecture is introduced in [RFC4655]. This document describes the impact of policy-based decision making when incorporated into the PCE architecture and provides additional details on, and context for, applying policy within the PCE architecture.

Policy-based Management (PBM), see [RFC3198], is a network management approach that enables a network to automatically perform actions in response to network events or conditions based on pre-established rules, also denoted as policies, from a network administrator. PBM enables network administrators to operate in a high-level manner through rule-based strategy (policies can be defined as a set of rules and actions); the latter are translated automatically (i.e., dynamically, without human interference) into individual device configuration directives, aimed at controlling a network as a whole. Two IETF Working Groups have considered policy networking in the past: The Resource Allocation Protocol (RAP) working group and the Policy Framework working group.

A framework for policy-based admission control [RFC2753] was defined and a protocol for use between Policy Enforcement Points (PEP) and Policy Decision Points (PDP) was specified: Common Open Policy Service (COPS) [RFC2748]. This document uses the terms PEP and PDP to refer to the functions defined in the COPS context. This document makes no assumptions nor does it require that the actual COPS protocol be used. Any suitable policy exchange protocol (for example, Simple Object Access Protocol (SOAP) [W3CSOAP]) may be substituted.

The IETF has also produced a general framework for representing, managing, sharing, and reusing policies in a vendor-independent, interoperable, and scalable manner. It has also defined an extensible information model for representing policies, called the Policy Core Information Model (PCIM) [RFC3060], and an extension to this model to address the need for QoS management, called the Quality of Service (QoS) Policy Information Model (QPIM) [RFC3644]. However, additional mechanisms are needed in order to specify policies related to the path computation logic as well as its control.

In Section 2, this document presents policy-related background and scenarios to provide a context for this work. Section 3 provides requirements that must be addressed by mechanisms and protocols that enable policy-based control over path computation requests and decisions. Section 4 introduces PCIM as a core component in a framework for providing policy-enabled path computation. Section 5 introduces a set of components that may be used to support policy-enabled path computation. Sections 6, 7, and 8 provide details on possible component configurations, communication, and events. Section 10 discusses the ability to introduce new constraints with minimal impact. It should be noted that this document, in Section 4, only introduces PCIM; specific PCIM definitions to support path computation will be discussed in a separate document.

1.1. Terminology

The reader is assumed to be familiar with the following terms:

BEEP: Blocks Extensible Exchange Protocol, see [RFC3080].

CIM: Common Information Model, see [DMTF].

COPS: Common Open Policy Service, see [RFC2748].

CSPF: Constraint-based Shortest Path First, see [RFC3630].

LSP:	Label Switched Path, see [RFC3031].
LSR:	Label Switching Router, see [RFC3031].
PBM:	Policy-Based Management, see [RFC3198].
PC:	Path Computation.
PCC:	Path Computation Client, see [RFC4655].
PCCIM:	Path Computation Core Information Model.
PCE:	Path Computation Element, see [RFC4655].
PCEP:	Path Computation Element Communication Protocol, see [PCEP].
PCIM:	Policy Core Information Model, see [RFC3060].
PDP:	Policy Decision Point, see [RFC2753].
PEP:	Policy Enforcement Point, see [RFC2753].
QPIM:	QoS Policy Information Model, see [RFC3644].
SLA:	Service Level Agreement.
SOAP:	Simple Object Access Protocol, see [W3CSOAP].
TE:	Traffic Engineering, see [RFC3209] and [RFC3473].
TED:	Traffic Engineering Database, see [RFC3209] and [RFC3473].
TE LSP:	Traffic Engineering MPLS Label Switched Path, see [RFC3209] and [RFC3473].
WDM:	Wavelength Division Multiplexing

2. Background

This section provides some general background on the use of policies within the PCE architecture. It presents the rationale behind the use of policies in the TE path computation process, as well as representative policies usage scenarios. This information is intended to provide context for the presented PCE policy framework. This section does not attempt to present an exhaustive list of rationales or scenarios.

2.1. Motivation

The PCE architecture as introduced in [RFC4655] includes policy as an integral part of the PCE architecture. This section presents some of the rationale for this inclusion.

Network operators require a certain level of flexibility to shape the TE path computation process, so that the process can be aligned with their business and operational needs. Many aspects of the path computation may be governed by policies. For example, a PCC may use policies configured by the operator to decide which optimization criteria, constraints, diversities and their relaxation strategies to request while computing path(s) for a particular service. Depending on SLAs, TE and cost/performance ratio goals, path computation requests may be issued differently for different services. A given Service A, for instance, may require two Shared Risk Link Group (SRLG)-disjoint paths for building end-to-end recovery scheme, while for a Service B link-disjoint paths may be sufficient. Service A may need paths with minimal end-to-end delay, while Service B may be looking for shortest (minimal-cost) paths. Different constraint relaxation strategies may be applied while computing paths for Service A and for Service B, and so forth. So, based on distinct service requirements, distinct or similar policies may be adopted when issuing/handling path computation requests.

Likewise, a PCE may apply policies to decide which algorithm(s) to use while performing path computations requested from a particular PCC or for a particular domain, see [RFC4927]; whether to seek the cooperation of other PCEs to satisfy a particular request or to handle a request on its own (possibly responding with non-explicit paths), or how the request should be modified before being sent to other member(s) of a group of cooperating PCEs, etc.

Additional motivation for supporting policies within the PCE architecture can be described as follows. Historically, a path computation entity was an intrinsic part of an LSR's control plane and always co-located with the LSR's signaling and routing subsystems. This approach allowed for unlimited flexibility in providing various path computation enhancements, such as: adding new types of constraints, diversities and their relaxation strategies, adopting new objective functions and optimization criteria, etc. All that had to be done to support an enhancement was to upgrade the control plane software of a particular LSR (and no other LSRs or any other network elements).

With the introduction of the PCE architecture, the introduction of new PCE capabilities becomes more complicated: it isn't enough for a PCE to upgrade its own software. In order to take advantage of a PCE's new capabilities, new advertising and signaling objects may need to be standardized, all PCCs may need to be upgraded with new software, and new interoperability problems may need to be resolved, etc.

Within the context of the PCE architecture, it is therefore highly desirable to find a way to introduce new path computation capabilities without requiring modifying either the discovery/communication protocols or the PCC software. One way to achieve this objective is to consider path selection constraints, their relaxations, and objective functions, as path computation request-specific policies. Furthermore, such policies may be configured and managed by a network operator as any other policies and may be interpreted in real time by PCCs and PCEs.

There are a number of advantages and useful by-products of such an approach:

- New path computation capabilities may be introduced without changing PCE-PCC communication and discovery protocols or PCC software. Only the PCE module providing the path computation capabilities (referred to in this document as a path computation engine) needs to be updated.

- Existing constraints, objective functions and their relaxations may be aggregated and otherwise associated, thus producing new, more complex objective functions that do not require a change of code even on the PCEs supporting the functions.

- Different elements such as conditions, actions, variables, etc., may be reused by multiple constraints, diversities, and optimizations.

- PCCs and PCEs need to handle other (that is, not request-specific) policies. Path computation-related policies of all types can be placed within the same policy repositories, managed by the same policy management tools, and interpreted using the same mechanisms. Also, policies need to be supported by PCCs and PCEs independent of the peculiarities of a specific PCC-PCE communication protocol, see [PCEP]. Thus, introducing a new (request-specific) type of policy describing constraints and other elements of a path computation request will be a natural and relatively inexpensive addition to the policy-enabled path computation architecture.

2.2. Policy Attributes

This section provides a summary listing of the policy attributes that may be included in the policy exchanges described in the scenarios that follow. This list is provided for guidance and is not intended to be exclusive. Implementation of this framework might include additional policy attributes not listed here.

Identities

- LSP head-end
- LSP destination
- PCC
- PCE

LSP identifiers

- LSP head-end
- LSP destination
- Tunnel identifier
- Extended tunnel identifier
- LSP ID
- Tunnel name

Requested LSP qualities

- bandwidth
- traffic parameters
- LSP attributes
- explicit path inclusions
- explicit path exclusions
- link protection level
- setup priority
- holding priority
- preexisting LSP route

Requested path computation behavior

- objective function
- other LSPs to be considered

Additional policy information

- Transparent policy information as received in Resource Reservation Protocol (RSVP)-TE

2.3. Representative Policy Scenarios

This section provides example scenarios of how policies may be applied using the PCE policy framework within the PCE architecture context. Actual networks may deploy one of the scenarios discussed, some combination of the presented scenarios, or other scenarios (not discussed). This section should not be viewed as limiting other applications of policies within the PCE architecture.

2.3.1. Scenario: Policy Configured Paths

A very simple usage scenario for PCE policy would be to use PCE to centrally administer configured paths. Configured paths are composed of strict and loose hops in the form of Explicit Route Objects (EROs), see [RFC3209], and are used by one or more LSPs. Typically, such paths are configured at the LSP ingress. In the context of policy-enabled path computation, an alternate approach is possible.

In particular, service-specific policies can be installed that will provide configured path(s) for a specific service request. The request may be identified based on service parameters such as endpoints, requested QoS, or even a token that identifies the initiator of a service request. The configured path(s) would then be used as input to the path computation process, which would return explicit routes by expanding of all specified loose hops.

Example of policy:

```
 if(service_destination matches 10.132.12.0/24)
    Use path: 10.125.13.1 => 10.125.15.1 => 10.132.12.1.
 else
    Compute path dynamically.
```

```
      ----------------------
     |              -----   |
     |             | TED |<-+------------>
     |              -----   |  TED synchronization
     |                |     |  mechanism (e.g., routing protocol)
     |                |     |
     |                v     |
     |  ------      -----   |  Inter-PCE Request/Response
     | |Policy|<-->| PCE |<.+...........>  (when present)
     |  ------      -----   |
      ----------------------
                      ^
                      | Request/
                      | Response
                      v
        Service  -------------  Signaling
        Request |[PCC][Policy]| Protocol
        <------>|    Node     |<------->
   or Signaling  -------------
      Protocol
```

Figure 1: Policy Enabled PCC and PCE

Path computation policies may be applied at either a PCC or a PCE, see Figure 1. In the PCC case, the configured path would be processed at the PCC and then passed to the PCE along with the PCE request, probably in the form of (inclusion) constraints. When applied at the PCE, the configured path would be used locally. Both cases require some method to configure and manage policies. In the PCC case, the real benefit would come when there is an automated policy distribution mechanism.

```
       ------------------       -------------------
      |                  |     |                   |
      |        PCE       |     |        PCE        |
      |                  |     |                   |
      |  ------   -----  |     |   -----   ------  |
      | |Policy| | TED | |     |  | TED | |Policy| |
      |  ------   -----  |     |   -----   ------  |
       ------------------       -------------------
               ^                       ^
               | Request/              | Request/
               | Response              | Response
               v                       v
   Service --------  Signaling  ------------  Signaling  ------------
   Request|Head-End| Protocol  |Intermediate| Protocol  |Intermediate|
     ---->|  Node  |<--------->|    Node    |<--------->|    Node    |
           --------             ------------             ------------
```

Figure 2: Multiple PCE Path Computation

```
    ------------------                              ------------------
   |                  | Inter-PCE Request/Response |                  |
   |       PCE        |<-------------------------->|       PCE        |
   |                  |                            |                  |
   |  ------   -----  |                            |  ------   -----  |
   | |Policy| | TED | |                            | |Policy| | TED | |
   |  ------   -----  |                            |  ------   -----  |
    ------------------                              ------------------
               ^
               | Request/
               | Response
               v
   Service ----------  Signaling   ----------  Signaling   ----------
   Request| Head-End | Protocol   | Adjacent | Protocol   | Adjacent |
     ---->|  Node    |<---------->|   Node   |<---------->|   Node   |
           ----------              ----------              ----------
```

Figure 3: Multiple PCE Path Computation with Inter-PCE Communication

Policy-configured paths may also be used in environments with multiple (more than one) cooperating PCEs (see Figures 2 and 3). For example, consider the case when there is limited TE visibility and independent PCEs are used to determine path(s) within each area of the TE visibility. In such a case, it may not be possible (or desirable) to configure entire explicit path(s) on a single PCE. However, it is possible to configure explicit path(s) for each area of the TE visibility and each responsible PCE. One by one, the PCEs would then map an incoming signaling request to appropriate configured path(s). Note that to make such a scenario work, it would likely be necessary to start and finish the configured paths on TE domain boundary nodes. Clearly, consistent PCE Policy Repositories are also critical in this example.

2.3.2. Scenario: Provider Selection Policy

A potentially more interesting scenario is applying PC policies in multi-provider topologies. There are numerous interesting policy applications in such topologies. A rudimentary example is simple access control, that is, deciding which PCCs are permitted to request inter-domain path computation.

A more complicated example is applying policy to determine which domain or network provider will be used to support a particular PCE request. Consider the topology presented in Figure 4. In this example, there are multiple transit domains available to provide a path from a source domain to a destination domain. Furthermore, each transit domain may have one or more options for reaching a particular domain. Each domain will need to select which of the multiple available paths will be used to satisfy a particular PCE request.

In today’s typical path computation process, TE reachability, availability, and metric are the basic criteria for path selection. However, policies can provide an important added consideration in the decision process. For example, transit domain A may be more expensive and provide lower delay or loss than transit domain B. Likewise, a transit domain may wish to treat PCE requests from its own customers differently than requests from other providers. In both cases, computation based on traffic engineering databases will result in multiple transit domains that provide reachability, and policies can be used to govern which PCE requests get better service.

```
                              +-------+
                   +----------+Transit+----------+
               +---+---+      | Domain|      +---+---+
               |Transit|      |   C   |      |Transit|
      +--------+ Domain|      +---+---+      | Domain+--------+
      |        |   A   +--+       |       +--+   F   |        |
   +--+---+    +---+---+  |       |       |  +---+---+     +--+---+
   |Source|        |      |   +---+---+   |      |         |Target|
   |Domain|        |      +---+Transit+---+      |         |Domain|
   +--+---+        |      +---+ Domain|---+      |         +--+---+
      |        +---+---+  |   |   D   |   |  +---+---+        |
      |        |Transit|  |   +---+---+   |  |Transit|        |
      +--------+ Domain+--+       |       +--+ Domain+--------+
               |   B   |          |          |   G   |
               +---+---+      +---+---+      +---+---+
                   |          |Transit|          |
                   +----------+ Domain+----------+
                              |   E   |
                              +-------+
```

Figure 4: Multi-Domain Network with Multiple Transit Options

There are multiple options for differentiating which PCE requests use a particular transit domain and get a particular (better or worse) level of service. For example, a PCE in the source domain may use user- and request-specific policies to determine the level of service to provide. A PCE in the source domain may also use domain-specific policies to choose which transit domains are acceptable. A PCE in a transit domain may use request-specific policies to determine if a request is from a direct customer or another provider, and then use domain-specific policies to identify how the request should be processed.

Example of policy:

```
 if(path computation request issued by a PCC within Source Domain)
    Route the path through Transit Domain A.
 else
    Route the path through Transit Domain B.
```

2.3.3. Scenario: Policy Based Constraints

Another usage scenario is the use of policy to provide constraints in a PCE request. Consider an LSR with a policy enabled PCC, as shown in Figure 1, which receives a service request via signaling, including over a Network-Network Interface (NNI) or User Network Interface (UNI) reference point, or receives a configuration request over a management interface to establish a service. In either case, the path(s) needed to support the service are not explicitly specified in the message/request, and hence path computation is needed.

In this case, the PCC may apply user- or service-specific policies to decide how the path selection process should be constrained, that is, which constraints, diversities, optimization criterion, and constraint relaxation strategies should be applied in order for the service LSP(s) to have a likelihood to be successfully established and provide necessary QoS and resilience against network failures. When deciding on the set of constraints, the PCC uses as an input all information it knows about the user and service, such as the contents of the received message, port ID over which message was received, associated VPN ID, signaling/reference point type, request time, etc. Once the constraints and other parameters of the required path computation are determined, the PCC generates a path computation request that includes the request-specific policies that describe the determined set of constraints, optimizations, and other parameters that indicate how the request is to be considered in the path computation process.

Example of policy:

```
 if(LSP belongs to a WDM layer network)
    Compute the path with wavelength continuity constraint with the
    maximum Optical Signal Noise Ratio (OSNR) at the path end
    optimization.
 else if(LSP belongs to a connection oriented Ethernet layer network)
    Compute the path with minimum end-to-end delay.
 else
    Compute the shortest path.
```

The PCC may also apply server-specific policies in order to select which PCE to use from the set of known (i.e., discovered or configured) PCEs. The PCC may also use server-specific policies to form the request to match the PCE’s capabilities so that the request will not be rejected and has a higher likelihood of being satisfied in an efficient way. An example of a request modification as the result of a server-specific policy is removing a constraint not supported by the PCE. Once the policy processing is completed at the

PCC, and the path computation request resulting from the original service request is updated by the policy processing, the request is sent to the PCE.

Example of policy:

```
 if(LSP belongs to a WDM layer network)
    Identify a PCE supporting wavelength continuity and optical
    impairment constraints;
    Send a request to such PCE, requesting path computation with the
    following constraints:
       a) wavelength continuity;
       b) maximum Polarization Mode Dispersion (PMD) at the path end.
    if(the path computation fails) remove the maximum PMD constraint
       and try the computation again.
```

The PCE that receives the request validates and otherwise processes the request, applying the policies found in the request as well as any policies that are available at the PCE, e.g., client- and domain-specific policies. As a result of the policy processing, the PCE may decide to reject the request.

Example of policy:

```
 Authenticate the PCC requesting the path computation using the
 PCC ID found in the path computation request;
 Reject the request if the authentication fails.
```

The PCE also may decide to respond with one or several pre-computed paths if user- or client-specific policies instruct the PCE to do so. If the PCE decides to satisfy the request by performing a path computation, it determines if it needs the cooperation of other PCEs and defines parameters for path computations to be performed locally and remotely. After that, the PCE instructs a co-located path computation engine to perform the local path computation(s) and, if necessary, sends path computation requests to one or more other PCEs. It then waits for the responses from the local path computation engine and, when used, the remote PCE. It then combines the resulting paths and sends the result back to the requesting PCC. The response may indicate policies describing the resulting paths, their characteristics (summary cost, expected end-to-end delay, etc.), as well as additional information related to the request, e.g., which constraints were honored, which were dismissed, and which were relaxed and in what way.

Example of policy:

```
 if(the path destination belongs to domain A)
    Instruct local path computation engine to perform the path
    computation;
 else
    Identify the PCE supporting the destination domain;
    Send path computation request to such PCE;
    Wait for and process the response.
 Send the path computation response to the requesting PCC.
```

The PCC processes the response and instructs the LSR to encode the received path(s) into the outgoing signaling message(s).

2.3.4. Scenario: Advanced Load Balancing (ALB) Example

Figure 5 illustrates a problem that stems from the coupling between BGP and IGP in the BGP decision process. If a significant portion of the traffic destined for the data center (or customer network) enters a PCE-enabled network from AS 1 and all IGP links' weights are the same, then both PE3 and PE4 will prefer to reach the data center using the routes advertised by PE2. PE5 will use the router-IDs of PE1 and PE2 to break the tie and might therefore also select to use the path through PE2 (if the router ID of PE2 is smaller than that of PE1). Either way, the net result is that the link between PE2 and CE will carry most of the traffic while the link between PE1 and the Customer Edge (CE) will be mostly idle.

```
                           ..............................
                           .          AS 1              .
                           .                            .
                           .   +---+   +---+   +----+   .
                           ....|PE8|...|PE9|...|PE10|....
                               +---+   +---+   +----+
                                 |       |       |
                               +---+   +---+   +---+
                         ......|PE3|...|PE4|...|PE5|......
                         .     +---+   +---+   +---+     .
    ..............     +---+     \      /    ___/      +---+
    .            .    _|PE2|_____+--+__/    /         _|PE6|
    .           +--+ / +---+     |P1|_____+--+_______/ +---+
    . Customer  |CE|=    .       +--+     |P2|           .
    . Network   +--+ \_+---+        \     +--+           .
    .            .     |PE1|________+--+___/|     x===x  .  PCE used
    ..............     +---+        |P3|    |     |PCE|  .  by all
                         .          +--+    |     x===x  .  AS0 nodes
                         .    AS 0         +---+         .
                         ..................|PE7|..........
                                           +---+
```

Figure 5: Advanced Load Balancing

This is a common problem for providers and customers alike. Analysis of Netflow records, see [IRSCP], for a large ISP network on a typical day has shown that for 71.8% of multi-homed customers, there is a complete imbalance, where the most loaded link carries all the traffic and the least loaded link carries none.

PCE policies can address this problem by basing the routing decision at the ingress routers on the offered load towards the multi-homed customer. For example, in Figure 5, PCE policies could be configured such that traffic load is monitored (e.g., based on Netflow data) at ingress routers PE3 to PE7 towards the data center prefixes served by egress routers PE1 and PE2. Using this offered load information, the path computations returned by PCE, based on the enabled PCE policies, can direct traffic to the appropriate egress router, on a per-ingress router basis. For example, the PCE path computation might direct traffic from both PE4 and PE5 to egress PE1, thus overriding the default IGP based selection. Alternatively, traffic from each ingress router to each egress link could be split 50-50.

This scenario is a good example of how a policy-governed PCE can account for some information that was not or cannot be advertised as TE link/node attributes, and, therefore, cannot be subject for explicit path computation constraints. More generally, such information can be pretty much anything. For example, traffic demand forecasts, flow monitoring feedback, any administrative policies, etc. Further examples are described in [IRSCP] of how PCE policies might address certain network routing problems, such as selective distributed denial-of-service (DDoS) blackholing, planned maintenance, and VPN gateway selection.

Example of policy:

```
 for(all traffic flows destined to Customer Network)
    if(flow ingresses on PE3, PE4, or PE5)
       Route the flow over PE1.
    else
       Route the flow over PE2.
```

3. Requirements

The following requirements must be addressed by mechanisms and protocols that enable policy-based control over path computation requests and decisions:

- (G)MPLS path computation-specific
  The mechanisms must meet the policy-based control requirements specific to the problem of path computation using RSVP-TE as the signaling protocol on MPLS and GMPLS LSRs.

- Support for non-(G)MPLS PCCs
  The mechanisms must be sufficiently generic to support non-(G)MPLS (LSR) clients such as a Network Management System (NMS), or network planner, etc.

- Support for many policies
  The mechanisms must include support for many policies and policy configurations. In general, the determination and configuration of viable policies are the responsibility of the service provider.

- Provision for monitoring and accounting information
  The mechanisms must include support for monitoring policy state and provide access information. In particular, mechanisms must provide usage and access information that may be used for accounting purposes.

- Fault tolerance and recovery
  The mechanisms must include provisions for fault tolerance and recovery from failure cases such as failure of PCC/PCE PDPs, disruption in communication that separate a PCC/PCE PDP from its associated PCC/PCE PEPs.

- Support for policy-ignorant nodes
  The mechanisms should not be mandatory for every node in a network. Policy-based path computation control may be enforced at a subset of nodes, for example, on boundary nodes within an administrative domain. These policy-capable nodes will function as trusted nodes from the point of view of the policy-ignorant nodes in that administrative domain. Alternatively, policy may be applied solely on PCEs with all PCCs being policy-ignorant nodes.

- Scalability
  One of the important requirements for the mechanisms is scalability. The mechanisms must scale at least to the same extent that RSVP-TE signaling scales in terms of accommodating multiple LSPs and network nodes in the path of an LSP. There are several sensitive areas in terms of scalability of policy-based path computation control. First, not every policy-aware node in an infrastructure should be expected to contact a remote PDP. This would cause potentially long delays in verifying requests. Additionally, the policy control architecture must scale at least as well as RSVP-TE protocol based on factors such as the size of RSVP-TE messages, the time required for the network to service an RSVP-TE request, local processing time required per node, and local memory consumed per node. These scaling considerations are of particular importance during re-routing of a set of LSPs.

- Security and denial-of-service considerations
  The policy control architecture, protocols, and mechanisms must be secure as far as the following aspects are concerned:

  - First, the mechanisms proposed must minimize theft and denial-of-service threats.

  - Second, it must be ensured that the entities (such as PEPs and PDPs) involved in policy control can verify each other's identity and establish necessary trust before communicating.

- Inter-AS and inter-area requirements
  There are several inter-AS policy-related requirements discussed in [RFC4216] and [RFC5376], and inter-area policy-related requirements discussed in [RFC4927]. These requirements must be addressed by policy-enabled PCE mechanisms and protocols.

It should be noted that this document only outlines the communication elements and mechanisms needed to allow a wide variety of possible policies to be applied for path computation control. It does not include any discussion of any specific policy behavior, nor does it define or require use of specific policies.

4. Path Computation Policy Information Model (PCPIM)

The Policy Core Information Model (PCIM) introduced in [RFC3060] and expanded in [RFC3460] presents the object-oriented information model for representing general policy information.

This model defines two hierarchies of object classes:

- Structural classes representing policy information and control of policies.

- Association classes that indicate how instances of the structural classes are related to each other.

These classes can be mapped to various concrete implementations, for example, to a directory that uses Lightweight Directory Access Protocol version 3 (LDAPv3) as its access protocol.

Figure 6 shows an abstract from the class inheritance hierarchy for PCIM.

```
   ManagedElement (abstract)
      |
      +--Policy (abstract)
      |  |
      |  +---PolicySet (abstract)
      |  |   |
      |  |   +---PolicyGroup
      |  |   |
      |  |   +---PolicyRule
      |  |
      |  +---PolicyCondition (abstract)
      |  |   |
      |  |   +---PolicyTimePeriodCondition
      |  |   |
      |  |   +---VendorPolicyCondition
      |  |   |
      |  |   +---SimplePolicyCondition
      |  |   |
      |  |   +---CompoundPolicyCondition
      |  |       |
      |  |       +---CompoundFilterCondition
      |  |
      |  +---PolicyAction (abstract)
      |  |   |
      |  |   +---VendorPolicyAction
      |  |   |
      |  |   +---SimplePolicyAction
      |  |   |
      |  |   +---CompoundPolicyAction
      |  |
      |  +---PolicyVariable (abstract)
      |  |   |
      |  |   +---PolicyExplicitVariable
      |  |   |
      |  |   +---PolicyImplicitVariable
      |  |       |
      |  |       +---(subtree of more specific classes)
      |  |
      |  +---PolicyValue (abstract)
      |      |
      |      +---(subtree of more specific classes)
```

Figure 6: PCIM Class Inheritance

The policy classes and associations defined in PCIM are sufficiently generic to allow them to represent policies related to anything.

Policy models for application-specific areas such as the Path Computation Service may extend the PCIM in several ways. The preferred way is to use the PolicyGroup, PolicyRule, and PolicyTimePeriodCondition classes directly as a foundation for representing and communicating policy information. Then, specific subclasses derived from PolicyCondition and PolicyAction can capture application-specific definitions of conditions and actions of policies.

The Policy Quality of Service Information Model [RFC3644] further extends the PCIM to represent QoS policy information for large-scale policy domains. New classes introduced in this document describing QoS- and RSVP-related variables, conditions, and actions can be used as a foundation for the PCPIM.

Detailed description of the PCPIM will be provided in a separate document.

5. Policy-Enabled Path Computation Framework Components

The following components are defined as part of the framework to support policy-enabled path computation:

- PCE Policy Repository
  A database from which PCE policies are available in the form of instances of PCPIM classes. PCE Policies are configured and managed by PCE Policy Management Tools;

- PCE Policy Decision Point (PCE-PDP)
  A logical entity capable of retrieving relevant path computation policies from one or more Policy Repositories and delivering the information to associated PCE-PEP(s);

- PCE Policy Enforcement Point (PCE-PEP)
  A logical entity capable of issuing device-specific Path Computation Engine configuration requests for the purpose of enforcing the policies;

- PCC Policy Decision Point (PCC-PDP)
  A logical entity capable of retrieving relevant path computation policies from one or more Policy Repositories and delivering the information to associated PCC-PEP(s);

- PCC Policy Enforcement Point (PCC-PEP)
  A logical entity capable of issuing device-specific Path Computation Service User configuration requests for the purpose of enforcing the policies.

From the policy perspective a PCC is logically decomposed into two parts: PCC-PDP and PCC-PEP. When present, a PCC-PEP is co-located with a Path Computation Service User entity that requires remote path computation (for example, the GMPLS control plane of an LSR). The PCC-PEP and PCC-PDP may be physically co-located (as per [RFC2748]) or separated. In the latter case, they talk to each other via such protocols as SOAP [W3CSOAP] or BEEP [RFC3080].

Likewise, a PCE is logically decomposed into two parts: PCE-PEP and PCE-PDP. When present, PCE-PEP is co-located with a Path Computation Engine entity that actually provides the Path Computation Service (that is, runs path computation algorithms). PCE-PEP and PCE-PDP may be physically co-located or separated. In the later case, they communicate using such protocols as SOAP and/or BEEP.

PCC-PDP/PCE-PDP may be co-located with, or separated from, an associated PCE Policy Repository. In the latter case, the PDPs use some access protocol (for example, LDAPv3 or SNMP). The task of PDPs is to retrieve policies from the repository (or repositories) and convey them to respective PEPs either in an unsolicited way or upon the PEP’s requests.

A PCC-PEP may receive policy information not only from PCC-PDP(s) but also from PCE-PEP(s) via PCC-PCE communication and/or PCE discovery protocols. Likewise, a PCE-PEP may receive policy information not only from PCE-PDP(s) but also from PCC-PEP(s), via the PCC-PCE communication protocol [PCEP].

Any given policy can be interpreted (that is, translated into a sequence of concrete device specific configuration requests) either on a PDP or on the associated PEP or partly on the PDP and partly on the PEP.

Generally speaking, the task of the PCC-PEP is to select the PCE and build path computation requests applying service-specific policies provided by the PCC-PDP. The task of the PCE-PEP is to control path computations by applying request-specific policies found in the requests as well as client-specific and domain-specific policies supplied by the PCE-PDP.

6. Policy Component Configurations

6.1. PCC-PCE Configurations

The PCE policy architecture supports policy being applied at a PCC and at a PCE. While the architecture supports policy being applied at both, there is no requirement for policy to always be applied at both, or even at either. The use of policy in a network, on PCCs, and on PCEs, is a specific network design choice. Some networks may choose to apply policy only at PCCs (Figure 7), some at PCEs (Figure 8), and others at both PCCs and PCEs (Figure 9). Regardless of where policy is applied, it must be applied in a consistent fashion in order to achieve the intended results.

```
                         .........................
                         .                       .
                         . PCE Policy Management .
                         .                       .
                         .........................
                                     .
                                     .
    ---------  Policy     -----------------------
   | PCC-PDP |<--------- | PCE Policy Repository |
    ---------             -----------------------
        ^
        | e.g., SOAP
        v
    ---------                     PCEP                     ---------
   | PCC-PEP |<------------------------------------------->|   PCE   |
    ---------         PCC-PCE Communication Protocol        ---------
```

Figure 7: Policies Applied on PCC Only

Along with supporting flexibility in where policy may be applied, the PCE architecture is also flexible in terms of where specific types of policies may be applied. Also, the PCE architecture allows for the application of only a subset of policy types. [RFC4655] defines several PC policy types. Each of these may be applied at either a PCC or a PCE or both. Clearly, when policy is only applied at PCCs or at PCEs, all PCE policy types used in the network must be applied at those locations.

```
                         .........................
                         .                       .
                         . PCE Policy Management .
                         .                       .
                         .........................
                                     .
                                     .
                          -----------------------  Policy    ---------
                         | PCE Policy Repository | -------->| PCE-PDP |
                          -----------------------            ---------
                                                                ^
                                                     e.g., SOAP |
                                                                v
    ---------                     PCEP                      ---------
   |   PCC   |<------------------------------------------->| PCE-PEP |
    ---------         PCC-PCE Communication Protocol        ---------
```

Figure 8: Policies Applied on Only

In the case where policy is only applied at a PCE, it is expected that the PCC will pass to the PCE all information about the service that it can gather in the path computation request (most likely in the form of PCPIM policy variables). The PCE is expected to understand this information and apply appropriate policies while defining the actual parameters of the path computation to be performed. Note that in this scenario, the PCC cannot apply server-specific or any other policies, and PCE selection is static.

When applying policy at both the PCC and PCE, it is necessary to select which types of policies are applied at each. In such configurations, it is likely that the application of policy types will be distributed across the PCC and PCE rather than applying all of them at both. For example, user-specific and server-specific policies may be applied at a PCC, request- and client-specific policies may be applied at a PCE, while domain-specific policies may be applied at both the PCC and PCE.

In the case when policy is only applied at a PCC, the PCC must apply all the types of required policies, for example user-, service-, server-, and domain-specific policies. The PCC uses the policies to construct a path computation request that appropriately represents the applied policies. The request will necessarily be limited to the set of "basic" (that is, non-policy capable) constraints explicitly defined by the PCC-PCE communication protocol.

6.2. Policy Repositories

Within the policy-enabled path computation framework policy repositories may be used in a single or multiple PCE policy repository configuration:

o) Single PCE Policy Repository

In this configuration, there is a single PCE Policy Repository shared between PCCs and PCEs.

```
                         .........................
                         .                       .
                         . PCE Policy Management .
                         .                       .
                         .........................
                                     .
                                     .
    ---------  Policy a   -----------------------  Policy b  ---------
   | PCC-PDP |<--------- | PCE Policy Repository | -------->| PCE-PDP |
    ---------             -----------------------            ---------
        ^                                                       ^
        | e.g., SOAP                                 e.g., SOAP |
        v                                                       v
    ---------                     PCEP                      ---------
   | PCC-PEP |<------------------------------------------->| PCE-PEP |
    ---------         PCC-PCE Communication Protocol        ---------
```

Figure 9: Single PCC/PCE Policy Repository

o) Multiple PCE Policy Repositories

The repositories in this case may be fully or partially synchronized by some discovery/synchronization management protocol or may be completely independent. Note that the situation when PCE Policy Repository A exactly matches PC Policy Repository B, results in the single PCE Policy Repository configuration case.

```
             --------------                   --------------
            |  PCE Policy  |                 |  PCE Policy  |
         ---| Repository A |                 | Repository B |---
        |    --------------                   --------------    |
        |                                                       |
        | Policy a                                     Policy b |
        |                                                       |
        v                                                       v
    ---------                                               ---------
   | PCC-PDP |                                             | PCE-PDP |
    ---------                                               ---------
        ^                                                       ^
        | e.g., SOAP                                 e.g., SOAP |
        v                                                       v
    ---------                     PCEP                      ---------
   | PCC-PEP |<------------------------------------------->| PCE-PEP |
    ---------         PCC-PCE Communication Protocol        ---------
```

Figure 10: Multiple PCE/PCC Policy Repositories

6.3. Cooperating PCE Configurations

The previous section shows the relationship between PCCs and PCEs. A parallel relationship exists between cooperating PCEs, and, in fact, this relationship can be viewed as the same as the relationship between PCCs and PCEs. The one notable difference is that there will be cases where having a shared PCE Policy Repository will not be desirable, for example, when the PCEs are managed by different entities. Note that in this case, it still remains necessary for the policies to be consistent across the domains in order to identify usable paths. The other notable difference is that a PCE, while processing a path computation request, may need to apply requester-specific (that is, client-specific) policies in order to modify the request before sending it to other cooperating PCE(s). This relationship is particularly important as the PCE architecture allows for configuration where all PCCs are not policy-enabled.

The following are example configurations. These examples do not represent an exhaustive list and other configurations are expected.

o) Single Policy Repository

In this configuration, there is a single PCE Policy Repository shared between PCEs. This configuration is likely to be useful within a single administrative domain where multiple PCEs are provided for redundancy or load distribution purposes.

```
                         .........................
                         .                       .
                         . PCE Policy Management .
                         .                       .
                         .........................
                                     .
                                     .
    ---------  Policy a   -----------------------  Policy b  ---------
   | PCE-PDP |<--------- | PCE Policy Repository | -------->| PCE-PDP |
    ---------             -----------------------            ---------
        ^                                                       ^
        | e.g., SOAP                                 e.g., SOAP |
        v                                                       v
    ---------                                               ---------
   | PCE-PEP |<------------------------------------------->| PCE-PEP |
    ---------         PCE-PCE Communication Protocol        ---------
```

Figure 11: Single PCC Policy Repository

o) Multiple Policy Repositories

The repositories in this case may be fully or partially synchronized by some discovery/synchronization management protocol(s) or may be completely independent. In the multi-domain case, it is expected that the repositories will be distinct, providing, however, consistent policies.

```
             --------------                   --------------
            |  PCE Policy  |                 |  PCE Policy  |
         ---| Repository A |                 | Repository B |---
        |    --------------                   --------------    |
        |                                                       |
        | Policy a                                     Policy b |
        |                                                       |
        v                                                       v
    ---------                                               ---------
   | PCE-PDP |                                             | PCE-PDP |
    ---------                                               ---------
        ^                                                       ^
        | e.g., SOAP                                 e.g., SOAP |
        v                                                       v
    ---------                     PCEP                      ---------
   | PCE-PEP |<------------------------------------------->| PCE-PEP |
    ---------         PCC-PCE Communication Protocol        ---------
```

Figure 12: Multiple PCC Policy Repositories

6.4. Policy Configuration Management

The management of path computation policy information used by PCCs and PCEs is largely out of scope of the described framework. The framework assumes that such information is installed, removed, and otherwise managed using typical policy management techniques. Policy Repositories may be populated and managed via static configuration, standard and proprietary policy management tools, or even dynamically via policy management/discovery protocols and applications.

7. Inter-Component Communication

7.1. Policy Communication

Flexibility in the application of policy types is imperative from the architecture perspective. However, this commodity implies added complexity on the part of the PCE-related communication protocols.

One added complexity is that PCE communication protocols must carry certain information to support various policy types that may be applied. For example, in the case where policy is only applied at a PCE, a PCC-PCE request must carry sufficient information for the PCE to apply service- or user-specific policies. This does imply that a PCC must have sufficient understanding of what policies can be applied at the PCE. Such information may be obtained via local configuration, static coding, or even via a PCE discovery mechanism. The PCC must also have sufficient understanding to properly encode the required information for each policy type.

Another added complexity is that PCE communication protocols must also be able to carry information that may result from a policy decision. For example, user- or service-specific policy applied at a PCC may result in policy-related information that must be carried along with the request for use by a PCE. This complexity is particularly important as it may be used to introduce new path computation parameters (e.g., constraints, objection functions, etc.) without modification of the core PCC and PCE. This communication will likely simply require the PCE communication protocols to support opaque policy-related information elements.

A final added complexity is that PCE communication protocols must also be able to support updated or unsolicited responses from a PCE. For example, changes in PCE policy may force a change to a previously provided path. Such updated or unsolicited responses may contain information that the PCC must act on, and may contain policy information that must be provided to a PCC.

PCC-PEP and PCE-PEP or a pair of PCE-PEPs communicate via a request-response type PCC-PCE Communication Protocol, i.e., [PCEP]. This document makes no assumptions as to what exact protocol is used to support this communication. This document does assume that the semantics of a path computation request are sufficiently abstract and general, and support both PCE-PCC and PCE-PCE communication.

From a policy perspective, a path computation request should include at a minimum:

```
o One or more source addresses;
o One or more destination addresses;
o Computation type (P2P (point to point), P2MP (point to multipoint),
  MP2P (multipoint to point), etc.);
o Number of required paths;
o Zero or more policy descriptors in the following format:
  <policy name>,
  <policy variable1 name>, <param11>, <param12>,...,<param1N>
  <policy variable2 name>, <param21>, <param12>,...,<param2N>
  ...
  <policy variableM name>, <paramM1>, <paramM2>,...,<paramMN>
```

A successful path computation response, at minimum, should include the list of computed paths and may include policies (in the form of policy descriptors as in path computation request, see above) for use in evaluating and otherwise applying the computed paths.

PCC-PCE Communication Protocol provides transport for policy information and should not understand nor make any assumptions about the semantics of policies specified in path computation requests and responses.

Note: This document explicitly allows for (but does not require) the PCC to decide that all necessary constraints, objective functions, etc. pertinent to the computation of paths for the service in question are to be determined by the PCE performing the computation. In this case, the PCC will use a set of policies (more precisely, PCPIM policy variables) describing the service-specific information. These policies may be placed within the path computation request and delivered to the PCE via a PCC-PCE communication protocol such as [PCEP]. The PCE (more precisely, PCE-PEP) is expected to understand this information and use it to determine the constraints and optimization functions applying local policies (that is, policies locally configured or provided by the associated PCE-PDP(s)).

7.2. PCE Discovery Policy Considerations

Dynamic PCE discovery allows for PCCs and PCEs to automatically discover a set of PCEs (including information required for the PCE selection). It also allows for PCCs and PCEs to dynamically detect new PCEs or any modification of PCEs status. Policy can be applied in two ways in this context:

1. Restricting the scope of information distribution for the mandatory set of information (in particular the PCE presence and location).

2. Restricting the type and nature of the optional information distributed by the discovery protocol. The latter is also subject to policy since the PCE architecture allows for distributing this information using either PCE discovery protocol(s) or PCC-PCE communication protocol(s). One important policy decision in this context is the nature of the information to be distributed, especially, when this information is not strictly speaking "discovery" information, rather, the PCE state changes. Client-specific and domain-specific policies may be applied when deciding whether this information should be distributed and to which clients of the path computation service (that is, which PCCs and/or PCEs).

Another place where policy applies is at the administrative boundaries. In multi-domain networks, multiple PCEs will communicate with each other and across administrative boundaries. In such cases, domain-specific policies would be applied to 1) filter the information exchanged between peering PCEs during the discovery process (to the bare minimum in most cases if at all allowed by the security policy) and 2) limit the content of information being passed in path computation request and responses.

8. Path Computation Sequence of Events

This section presents a non-exhaustive list of representative scenarios.

8.1. Policy-Enabled PCC, Policy-Enabled PCE

When a GMPLS LSR receives a Setup (RSVP Path) message from an upstream LSR, the LSR may decide to use a remote Path Computation Entity. The following sequence of events occurs in this case:

- A PCC-PEP co-located with the LSR applies the service-specific policies to select a PCE for the service path computation as well as to build the path computation request (that is, to select a list of policies, their variables, conditions and actions expressing constraints, diversities, objective functions and relaxation strategies appropriate for the service path computation). The policies may be:

a) Statically configured on the PCC-PEP;

b) Communicated to the PCC-PEP by a remote or local PCC-PDP via protocol such as SOAP either proactively (most of the cases) or upon an explicit request by the PCC-PEP in cases when some specifics of the new service have not been covered yet by the policies so far known to the PCC-PEP).

The input for the decision process on the PCC-PEP is the information found in the signaling message as well as any other service-specific information such as port ID over which the message was received, associated VPN ID, the reference point type (UNI, E-NNI, etc.) and so forth. After the path computation request is built, it is sent directly to the PCE-PEP using the PCC-PCE Communication Protocol, e.g., [PCEP].

- PCE-PEP validates and otherwise processes the request applying the policies found in the request- as well as client- and domain-specific policies. The latter, again, may be either statically configured on the PCE-PEP or provided by the associated local or remote PCE-PDP via a protocol such as SOAP. The outcome of the decision process is the following information:

a) Whether the request should be satisfied, rejected, or dismissed.

b) The sets of sources and destinations for which paths should be locally computed.

c) The set of constraints, diversities, optimization functions, and relaxations to be considered in each of locally performed path computation.

d) The address of the next-in-chain PCE.

e) The path computation request to be sent to the next-in-chain PCE.

The PCE-PEP instructs a co-located path computation engine to perform the local path computation(s) and, if necessary, sends the path computation request to the next-in-chain PCE using a PCC-PCE Communication Protocol. Then, it waits for the responses from the local path computation engine and the remote PCE, combines the resulting paths, and sends them back to the PCC-PEP using the PCC-

PCE Communication Protocol. The response contains the resulting paths as well as policies describing some additional information (for example, which of constraints were honored, which were dismissed, and which were relaxed and in what way).

- PCC-PEP instructs the signaling subsystem of the GMPLS LSR to encode the received path(s) into the outgoing Setup message(s).

8.2. Policy-Ignorant PCC, Policy-Enabled PCE

This case parallels the previous example, but the user- and service-specific policies should be applied at the PCE as the PCC is policy ignorant. Again, when a GMPLS LSR has received a Setup (RSVP Path) message from an upstream LSR, the LSR may decide to use a non-co-located Path Computation Entity. The following sequence of events occurs in this case:

- The PCC constructs a PCE request using information found in the signaling/provisioning message as well as any other service-specific information such as port ID over which the message was received, associated VPN ID, the reference point type (UNI, E-NNI, etc.) and so forth. This information is encoded in the request in the form of policy variables. After the request is built, it is sent directly to the PCE-PEP using a PCC-PCE Communication Protocol.

- PCE-PEP validates and otherwise processes the request interpreting the policy variables found in the request and applying user-, service-, client-, and domain-specific policies to build the actual path computation request. The policies, again, may be either statically configured on the PCE-PEP or provided by the associated local or remote PCE-PDP via a protocol such as SOAP. The outcome of the decision process is the following information:

  a) Whether the request should be satisfied, rejected, or dismissed.

  b) The sets of sources and destinations for which paths should be locally computed.

  c) The set of constraints, diversities, optimization functions, and relaxations to be considered in each of locally performed path computation.

  d) The address of the next-in-chain PCE.

  e) The path computation request to be sent to the next-in-chain PCE.

The PCE-PEP instructs a co-located path computation engine to perform the local path computation(s) and, if necessary, sends the path computation request to the next-in-chain PCE using the PCC-PCE Communication Protocol. Then, it waits for the responses from the local path computation engine and the remote PCE, combines the resulting paths, and sends them back to the PCC-PEP using the PCC-PCE Communication Protocol. The response contains the resulting paths as well as policies describing some additional information (for example, which of constraints were honored, which were dismissed, and which were relaxed and in what way)

- PCC-PEP instructs the signaling sub-system of the GMPLS LSR to encode the received path(s) into the outgoing Setup message(s).

9. Introduction of New Constraints

An important aspect of the policy-enabled path computation framework discussed above is the ability to introduce new constraints with minimal impact. In particular, only those components and mechanisms that will use a new constraint need to be updated in order to support the new constraint. Importantly, those components and mechanisms that will not use the new constraint must not require any change in order for the new constraint to be utilized. For example, the PCE communication protocols must not require any changes to support new constraints. Likewise, PCC and PCEs that will not process new constraints must not require any modification.

Consider the case where a PCE has been upgraded with software supporting optical physical impairment constraint, such as Polarization Mode Dispersion (PMD), that previously was not supported in the domain. In this case, one or more new policies will be installed in the PCE Policy Repository (associated with the PCE) defining the constraint (rules that determine application criteria, set of policy variables, conditions, actions, etc.) and its relaxation strategy (or strategies). The new policies will be also propagated into other PCE Policy Repositories within the domain via discovery and synchronization protocols or via local configuration. PCE-PDPs and PCC-PDPs will then retrieve the corresponding policies from the repository (or repositories). From then on, PCC-PDPs will instruct associated PCC-PEPs to add the new policy information into path computation requests for services with certain parameters (for example, for services provisioned in the optical channel (OCh) layer).

It is important to note that policy-enabled path computation model naturally solves the PCE capability discovery issues. Suppose a PCE working in a single PCE Policy Repository configuration starts to support a new constraint. Once a corresponding policy installed in the repository, it automatically becomes available for all repository users, that is, PCCs. In the multi-repository case some policy synchronization must be provided; however, this problem is one of the management plane which is solved already.

10. Security Considerations

This document adds to the policy security considerations mentioned in [RFC4655]. In particular, it is now necessary to consider the security issues related to policy information maintained in PCE Policy Repositories and policy-related transactions. The most notable issues, some of which are also listed in [RFC4655], are:

- Unauthorized access to the PCE Policy Repositories;

- Interception of policy information when it is retrieved from the repositories and/or transported from PDPs to PEPs;

- Interception of policy-related information in path computation requests and responses;

  - o Impersonation of user and client identities;

  - o Falsification of policy information and/or PCE capabilities;

  - o Denial-of-service attacks on policy-related communication mechanisms.

As with [RFC4655], it is expected that PCE solutions will address the PCE aspects of these issues in detail.

11. Acknowledgments

Adrian Farrel contributed significantly to this document. We would like to thank Bela Berde for fruitful discussions on PBM and policy-driven path computation. We would also like to thank Kobus Van der Merwe for providing insights and examples regarding PCE policy applications.

12. References

12.1. Normative References

[RFC2753] Yavatkar, R., Pendarakis, D., and R. Guerin, "A Framework for Policy-based Admission Control", RFC 2753, January 2000.

[RFC3060] Moore, B., Ellesson, E., Strassner, J., and A. Westerinen, "Policy Core Information Model -- Version 1 Specification", RFC 3060, February 2001.

[RFC3209] Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

[RFC3460] Moore, B., Ed., "Policy Core Information Model (PCIM) Extensions", RFC 3460, January 2003.

[RFC3473] Berger, L., Ed., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

[RFC3644] Snir, Y., Ramberg, Y., Strassner, J., Cohen, R., and B. Moore, "Policy Quality of Service (QoS) Information Model", RFC 3644, November 2003.

[RFC4216] Zhang, R., Ed., and J.-P. Vasseur, Ed., "MPLS Inter-Autonomous System (AS) Traffic Engineering (TE) Requirements", RFC 4216, November 2005.

[RFC4655] Farrel, A., Vasseur, J.-P., and J. Ash, "A Path Computation Element (PCE)-Based Architecture", RFC 4655, August 2006.

[RFC4927] Le Roux, J.-L., Ed., "Path Computation Element Communication Protocol (PCECP) Specific Requirements for Inter-Area MPLS and GMPLS Traffic Engineering", RFC 4927, June 2007.

12.2. Informative References

[DMTF] Common Information Model (CIM) Schema, version 2.x. Distributed Management Task Force, Inc. The components of the CIM v2.x schema are available via links on the following DMTF web page: http://www.dmtf.org/standards/standard_cim.php.

[IRSCP] Van der Merwe, J., et al., "Dynamic Connectivity Management with an Intelligent Route Service Control Point," ACM SIGCOMM Workshop on Internet Network Management (INM), Pisa, Italy, September 11, 2006.

[PCEP] Vasseur, JP., Ed. and JL. Le Roux, Ed., "Path Computation Element (PCE) Communication Protocol (PCEP)", Work in Progress, November 2008.

[RFC2748] Durham, D., Ed., Boyle, J., Cohen, R., Herzog, S., Rajan, R., and A. Sastry, "The COPS (Common Open Policy Service) Protocol", RFC 2748, January 2000.

[RFC3031] Rosen, E., Viswanathan, A., and R. Callon, "Multiprotocol Label Switching Architecture", RFC 3031, January 2001.

[RFC3080] Rose, M., "The Blocks Extensible Exchange Protocol Core", RFC 3080, March 2001.

[RFC3198] 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.

[RFC3630] Katz, D., Kompella, K., and D. Yeung, "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC 3630, September 2003.

[RFC5376] Bitar, N., Zhang, R., and K. Kumaki, "Inter-AS Requirements for the Path Computation Element Communication Protocol (PCECP)", RFC 5376, November 2008.

[W3CSOAP] Hadley, M., Mendelsohn, N., Moreau, J., Nielsen, H., and Gudgin, M., "SOAP Version 1.2 Part 1: Messaging Framework", W3C REC REC-soap12-part1-20030624, June 2003.

Authors' Addresses

Igor Bryskin
ADVA Optical
7926 Jones Branch Drive
Suite 615
McLean, VA 22102
EMail: ibryskin@advaoptical.com

Dimitri Papadimitriou
Alcatel
Fr. Wellesplein 1,
B-2018 Antwerpen, Belgium
Phone: +32 3 240-8491
EMail: dimitri.papadimitriou@alcatel.be

Lou Berger
LabN Consulting, LLC
Phone: +1 301 468 9228
EMail: lberger@labn.net

Jerry Ash
AT&T
EMail: gash5107@yahoo.com