PCP working group	D. Wing, Ed.
Internet-Draft	Cisco
Intended status: Standards Track	S. Cheshire
Expires: November 21, 2011	Apple
	M. Boucadair
	France Telecom
	R. Penno
	Juniper Networks
	P. Selkirk
	Internet Systems Consortium
	May 20, 2011

Port Control Protocol (PCP)
draft-ietf-pcp-base-12

Abstract

Port Control Protocol allows a host to control how incoming IPv6 or
IPv4 packets are translated and forwarded by a network address
translator (NAT) or simple firewall to an IPv6 or IPv4 host, and also
allows a host to optimize its NAT keepalive messages.

Table of Contents

1. Introduction

The Port Control Protocol (PCP) provides a mechanism to control how incoming packets are forwarded by upstream devices such as NAT64, NAT44, and firewall devices, and a mechanism to reduce application keepalive traffic. PCP is primarily designed to be implemented in the context of both Carrier-Grade NATs (CGN) and small NATs (e.g., residential NATs). PCP allows hosts to operate server for a long time (e.g., a webcam) or a short time (e.g., while playing a game or on a phone call) when behind a NAT device, including when behind a CGN operated by their Internet service provider.
PCP allows applications to create mappings from an external IP address and port to an internal IP address and port. These mappings are required for successful inbound communications destined to machines located behind a NAT or a firewall.
After creating a mapping for incoming connections, it is necessary to inform remote computers about the IP address and port for the incoming connection. This is usually done in an application-specific manner. For example, a computer game would use a rendezvous server specific to that game (or specific to that game developer), and a SIP phone would use a SIP proxy. PCP does not provide this rendezvous function. The rendezvous function will support IPv4, IPv6, or both. Depending on that support and the application's support of IPv4 or IPv6, the PCP client will need an IPv4 mapping, an IPv6 mapping, or both.
Many NAT-friendly applications send frequent application-level messages to ensure their session will not be timed out by a NAT. These are commonly called "NAT keepalive" messages, even though they are not sent to the NAT itself (rather, they are sent 'through' the NAT). These applications can reduce the frequency of those NAT keepalive messages by using PCP to learn (and influence) the NAT mapping lifetime. This helps reduce bandwidth on the subscriber's access network, traffic to the server, and battery consumption on mobile devices.
Many NATs and firewalls have included application layer gateways (ALGs) to create mappings for applications that establish additional streams or accept incoming connections. ALGs incorporated into NATs additionally modify the application payload. Industry experience has shown that these ALGs are detrimental to protocol evolution. PCP allows an application to create its own mappings in NATs and firewalls, reducing the incentive to deploy ALGs in NATs and firewalls.

2. Scope

2.1. Deployment Scenarios

PCP can be used in various deployment scenarios, including:

* *Dual-Stack Lite (DS-Lite)* [I-D.ietf-softwire-dual-stack-lite], and;

* NAT64, both Stateful [RFC6146] and Stateless [RFC6145], and;

* Carrier-Grade NAT [I-D.ietf-behave-lsn-requirements], and;

* Basic NAT [RFC3022], and;

* Network Address and Port Translation (NAPT) [RFC3022], such as commonly deployed in residential NAT devices, and;

* Layer-2 aware NAT [I-D.miles-behave-l2nat] and Dual-Stack Extra Lite [I-D.arkko-dual-stack-extra-lite], and;

* IPv4 and IPv6 simple firewall control [RFC6092].

2.2. Supported Protocols

The PCP OpCodes defined in this document are designed to support transport-layer protocols that use a 16-bit port number (e.g., TCP, UDP, SCTP, DCCP). Protocols that do not use a port number (e.g., IPsec ESP), and the ability to use PCP to forward all traffic to a single default host (often nicknamed a "DMZ"), are beyond the scope of this document.

2.3. Single-homed Customer Premises Network

The PCP machinery assumes a single-homed host model. That is, for a given IP version, only one default route exists to reach the Internet. This is important because after a PCP mapping is created and an inbound packet (e.g., TCP SYN) arrives at the host, the outbound response (e.g., TCP SYNACK) has to go through the same path so it is seen by the firewall or rewritten by the NAT. This restriction exists because otherwise there would need to be one PCP server for each egress, because the host could not reliably determine which egress path packets would take, so the client would need to be able to reliably make the same internal/external mapping in every NAT gateway, which in general is not possible (because the other NATs would likely have the necessary port mapped to another host).

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 "Key words for use in RFCs to Indicate Requirement Levels" *[RFC2119]*.

Internal Host:
A host served by a NAT gateway, or protected by a firewall. This is the host that receives the incoming traffic created by a PCP MAP request, or the host that initiated an implicit dynamic mapping (e.g., by sending a TCP SYN) across a firewall or a NAT.

Remote Host:
A host with which an Internal Host is communicating.

Internal Address:
The address of an Internal Host served by a NAT gateway (typically a private address [RFC1918]) or protected by a firewall.

External Address:
The address of an Internal Host as seen by other Remote Hosts on the Internet with which the Internal Host is communicating, after translation by any NAT gateways on the path. An External Address is generally a public routable (i.e., non-private) address. In the case of an Internal Host protected by a pure firewall, with no address translation on the path, its External Address is the same as its Internal Address.

Remote Peer Address:
The address of a Remote Host, as seen by the Internal Host. A Remote Address is generally a public routable address. In the case of a Remote Host that is itself served by a NAT gateway, the Remote Address may in fact be the Remote Host's External Address, but since this remote translation is generally invisible to software running on the Internal Host, the distinction can safely be ignored for the purposes of this document.

Third Party:
In the common case, an Internal Host manages its own Mappings using PCP requests, and the Internal Address of those Mappings is the same as the source IP address of the PCP request packet.

In the case where one device is managing Mappings on behalf of some other device, the presence of the THIRD_PARTY option in the MAP request signifies that the specified address, not the source IP address of the PCP request packet, should be used as the Internal

Address for the Mapping. For example, this can occur if the internal
host does not implement PCP.

Mapping, Port Mapping, Port Forwarding:
A NAT mapping creates a
relationship between an internal IP address, protocol, and port and
an external IP address, protocol, and port. More specifically, it
creates a translation rule where packets destined to the external IP
and port are translated to the internal IP and port, and vice versa.
In the case of a pure firewall, the "Mapping" is the identity
function, translating an internal port number to the same external
port number, and this "Mapping" indicates to the firewall that
traffic to and from this internal port number is permitted to pass.

Mapping Types:
There are three different ways to create mappings:
implicit dynamic mappings, explicit dynamic mappings, and static
mappings. Implicit dynamic mappings are created as a result of a TCP
SYN or outgoing UDP packet, and allow Internal Hosts to receive
replies to their outbound packets. Explicit dynamic mappings are
created as a result of PCP MAP requests. Static mappings are created
by manual configuration (e.g., command-line interface or web page).
Explicit and static mappings allow Internal Hosts to receive inbound
traffic that is not in direct response to any immediately preceding
outbound communication (i.e., allow Internal Hosts to operate a
"server" that is accessible to other hosts on the Internet). Both
implicit and explicit dynamic mappings are dynamic in the sense that
they are created on demand, as requested (implicitly or explicitly)
by the Internal Host, and have a lifetime. After the lifetime, the
mapping is deleted unless the lifetime is extended by action by the
Internal Host (e.g., sending more traffic or sending a new PCP MAP
request). Static mappings differ from dynamic mappings in that their
lifetime is typically infinite (they exist until manually removed)
but otherwise they behave exactly the same as an explicit dynamic
mapping with an infinite lifetime. For example, a PCP MAP request to
create a mapping that already exists as a static mapping will return
a successful result, confirming that the requested mapping exists.

PCP Client:
A PCP software instance responsible for issuing PCP
requests to a PCP server. One or several PCP Clients can be embedded
in the same host. Several PCP Clients can be located in the same
local network. A PCP Client can issue PCP request on behalf of a
third party device for which it is authorized to do so. An
interworking function from Universal Plug and Play Internet Gateway
Device (UPnP IGD, [IGD]) to PCP is another example of a PCP Client.
A PCP server in a NAT gateway that is itself a client of another NAT
gateway (nested NAT) may itself act as a PCP client to the upstream
NAT.

PCP Server:

> A network element which receives and processes PCP requests from a
> PCP client. Generally this is a PCP-capable NAT gateway or firewall.
> A NAT gateway creates mappings determining how it translates packets
> it forwards, and PCP enables clients to communicate with the NAT
> gateway about those mappings. In principle it is also possible for
> the PCP server to be some other device, which in turn communicates
> with the NAT gateway using some other network protocol, but this
> introduces additional complexity and fragility into the system, and
> is a deployment detail which should be implemented in a way that is
> invisible to the PCP client. See also Section 4.

Interworking Function:

> a functional element responsible for
> interworking another protocol with PCP. For example interworking
> between UPnP IGD *[IGD]* with PCP.

subscriber:

> an entity provided access to the network. In the case of a
> commercial ISP, this is typically a single home.

5-tuple The 5 pieces of information that fully identify a flow, from
the perspective of a subscriber: source IP address, destination IP
address, protocol, source port number, destination port number. From
the perspective of a NAPT device, in certain deployments an
additional piece of information is necessary to distinguish
subscribers with overlapping IP addresses. This additional
information depends on the deployment scenario, but examples of the
information include the subscriber's IPv6 address (for the
subscriber's Dual-Stack Lite tunnel) or the subscriber's Virtual LAN
number ([I-D.miles-behave-l2nat]), or other similar identifier.

4. Relationship between PCP Server and its NAT/firewall

The PCP server receives PCP requests. The PCP server might be
integrated within the NAT or firewall device (as shown in Figure 1)
which is expected to be a common deployment.

```
                        +-----------------+
+------------+          | NAT or firewall |
| PCP client |-<network>-+     with        +---<Internet>
+------------+          |    PCP server   |
                        +-----------------+
```
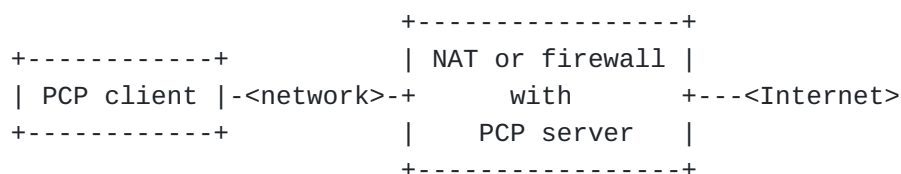
It is also possible to operate the PCP server in a separate device from
the NAT, so long as such operation is indistinguishable from the PCP
client's perspective.

5. Common Request and Response Header Format

All PCP messages contain a request (or response) header containing an opcode, any relevant opcode-specific information, and zero or more options. The packet layout for the common header, and operation of the PCP client and PCP server are described in the following sections. The information in this section applies to all OpCodes. Behavior of the OpCodes defined in this document is described in Section 8 and Section 9.

5.1. Request Header

All requests have the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Version = 1  |R|   OpCode    |        Reserved (16 bits)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Requested Lifetime                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Reserved                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|             PCP Client's IP address (always 128 bits)         |
|                                                               |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:             (optional) opcode-specific information            :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:             (optional) PCP Options                            :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
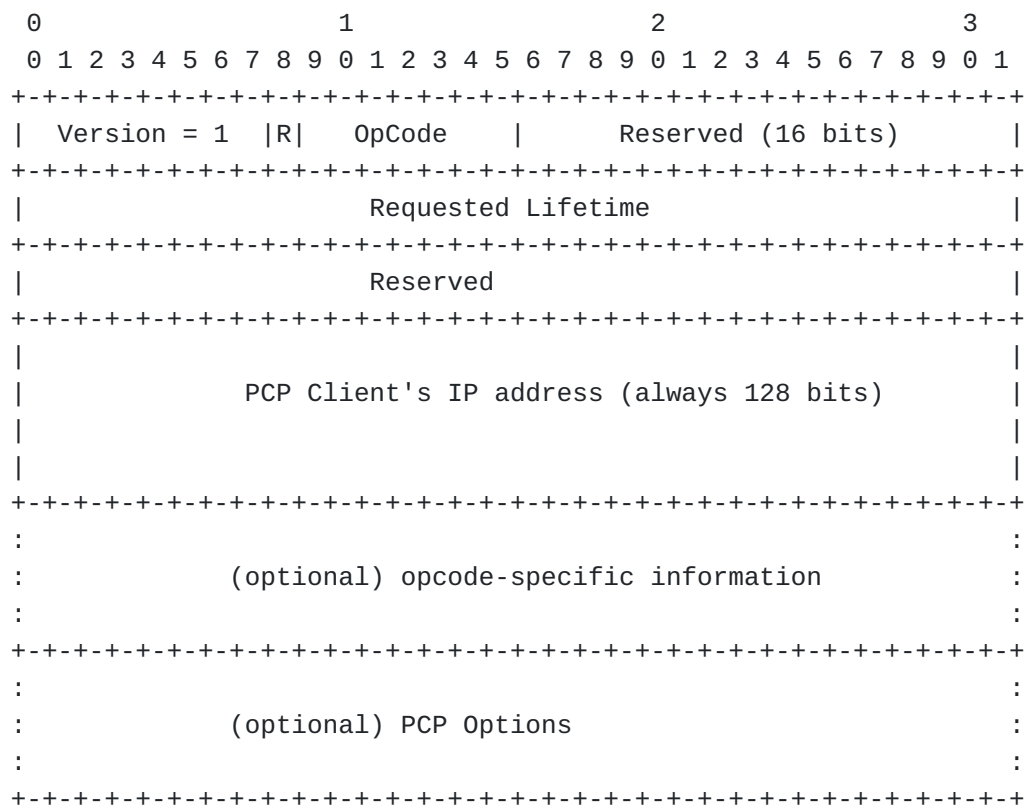
These fields are described below:

Version: This document specifies protocol version 1. NAT-PMP, a precursor to PCP, specified protocol version 0. Should later updates to this document specify different message formats with a version number greater than 1, the first two bytes of those new message formats will continue to contain the version number and opcode as shown here, so that a PCP server receiving a message format newer or older than the version(s) it understands can still parse enough of the message to correctly identify the version number, and determine whether the problem is that this server is too old and needs to be

updated to work with the PCP client, or whether the PCP client is
too old and needs to be updated to work with this server.

R: Indicates Request (0) or Response (1). All Requests MUST use 0.

OpCode: Opcodes are defined in Section 8 and Section 9.

Reserved: 16 reserved bits, MUST be sent as 0 and MUST be ignored when
   received.

Requested Lifetime: The Requested Lifetime field is an unsigned 32-bit
   integer, in seconds, ranging from 0 to 4,294,967,295 seconds. This
   is used by the MAP and PEER OpCodes defined in this document for
   their requested lifetime. Future OpCodes which don't need this field
   MUST set the field to zero on transmission and ignore on reception.

Reserved: 32 reserved bits, MUST be sent as 0 and MUST be ignored when
   received.

PCP Client's IP Address: The IP address of the PCP client, from the
   PCP client's perspective. If IPv4, only the first 32 bits are used,
   the other bits MUST be set to 0.

5.2. Response Header

All responses have the following format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Version = 1  |R|   OpCode    |   Reserved   |  Result Code  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Lifetime                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Epoch                            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                             |
|             PCP Client's IP address (always 128 bits)       |
|                                                             |
|                                                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                             :
:             (optional) OpCode-specific response data        :
:                                                             :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:             (optional) Options                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

These fields are described below:

Version:

Responses MUST use version 1.

R: Indicates Request (0) or Response (1). All Responses MUST use 1.

OpCode: The OpCode value, copied from the request.

Reserved: 8 reserved bits, MUST be sent as 0, MUST be ignored when received. This is set by the server.

Result Code: The result code for this response. See [Section 5.4](#) for values. This is set by the server.

Lifetime: The Lifetime field is an unsigned 32-bit integer, in seconds, ranging from 0 to 4,294,967,295 seconds. On an error response, this indicates how long clients should assume they'll get the same error response from that PCP server if they repeat the same request. On a success response for the currently-defined PCP OpCodes -- MAP and PEER -- this indicates the lifetime for this mapping. If future OpCodes are defined that do not have a lifetime associated with them, then in success responses for those OpCodes the Lifetime MUST be set to zero on transmission and MUST be ignored on reception.

Epoch: The server's Epoch value. See [Section 6.5](#) for discussion. This value is set in both success and error responses.

PCP Client's IP Address: The IP address of the PCP client, from the PCP server's perspective. If IPv4, only the first 32 bits are used, the other bits MUST be set to 0.

[5.3.](#) [Options](#)

A PCP OpCode can be extended with an Option. Options can be used in requests and responses. The decision about whether to include a given piece of information in the base opcode format or in an option is an engineering trade-off between packet size and code complexity. For information that is usually (or always) required, placing it in the fixed opcode data results in simpler code to generate and parse the packet, because the information is a fixed location in the opcode data, but wastes space in the packet in the event that that field is all-zeroes because the information is not needed or not relevant. For information that is required less often, placing it in an option results in slightly more complicated code to generate and parse packets containing that option, but saves space in the packet when that information is not needed. Placing information in an option also means that an implementation that never uses that information doesn't even need to implement code to generate and parse it. For example, a client that never requests mappings on behalf of some other device doesn't need to implement code to generate the THIRD_PARTY option, and a PCP server that doesn't implement the necessary security measures to create

third-party mappings safely doesn't need to implement code to parse the
THIRD_PARTY option.
Options use the following Type-Length-Value format:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Option Code  |    Reserved   |         Option-Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                       (optional) data                         :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
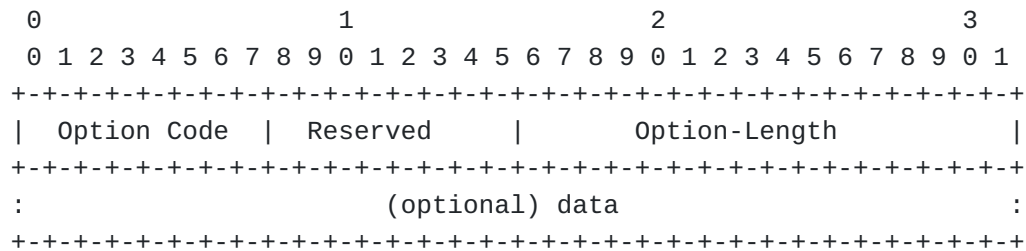
The description of the fields is as follows:

Option Code: 8 bits. Its highest bit is the "O" bit and indicates if
   this Option is mandatory (0) or optional (1) to process.

Reserved: 8 bits. MUST be set to 0 on transmission and MUST be ignored
   on reception.

Option-Length: 16 bits. Indicates the length of the enclosed data in
   octets. Options with length of 0 are allowed.

data: Option data. The option data MUST end on a 32-bit boundary,
   padded with 0's when necessary.

A given Option MAY be included in a request containing a specific
OpCode. The handling of an Option by the PCP client and PCP server MUST
be specified in an appropriate document and MUST include whether the
PCP Option can appear (one or more times) in a request and/or response,
and indicate the contents of the Option in the request and in the
response. If several Options are included in a PCP request or response,
they MAY be encoded in any order by the PCP client and are processed in
the order received.
If, while processing an option, an error is encountered that causes a
PCP error response to be generated, the PCP request MUST cause no state
change in the PCP server or the PCP-controlled device (i.e., it rolls
back any changes it might have made while processing the request). The
response MUST encode the Options in the same order, but may omit some
PCP Options in the response, as is necessary to indicate the PCP server
does not understand that Option or that Option is not permitted to be
included in responses by the definition of the Option itself.
Additional Options included in the response (if any) MUST be included
at the end. A certain Option MAY appear more than once in a request or
in a response, if permitted by the definition of the Option itself. If
the Option's definition allows the Option to appear only once but it
appears more than once in a request, the PCP server MUST respond with
the MALFORMED_OPTION result code; if this occurs in a response, the PCP

client processes the first occurrence and ignores the other occurrences
as if they were not present.
If the "O" bit (high bit) in the OpCode is clear,

     *the PCP server MUST only generate a positive PCP response if it
      can successfully process the PCP request and this Option.

     *if the PCP server does not implement this Option, or cannot
      perform the function indicated by this Option (e.g., due to a
      parsing error with the option), it MUST generate a failure
      response with code UNSUPP_OPTION or MALFORMED_OPTION (as
      appropriate) and include the UNPROCESSED option in the response
      (Section 6.7.1).

If the "O" bit is set, the PCP server MAY process or ignore this
Option, entirely at its discretion.
Option definitions MUST include the information below:

     *This Option:

        -name: <mnemonic>

        -number: <value>

        -purpose: <textual description>

        -is valid for OpCodes: <list of OpCodes>

        -length: <rules for length>

        -may appear in: <requests/responses/both>

        -maximum occurrences: <count>

5.4. Result Codes

The following result codes may be returned as a result of any OpCode
received by the PCP server. The only success result code is 0, other
values indicate an error. If a PCP server has encountered multiple
errors during processing of a request, it SHOULD use the most specific
error message.

0   SUCCESS, success

1   UNSUPP_VERSION, unsupported version.

2   MALFORMED_REQUEST, indicating the request could not be successfully
     parsed.

3   UNSUPP_OPCODE, unsupported OpCode.

UNSUPP_OPTION, unsupported Option. This error only occurs if the Option is in the mandatory-to-process range.

5 MALFORMED_OPTION, malformed Option (e.g., exists too many times, invalid length).

6 PROCESSING_ERROR, server encountered an error after parsing while attempting to process a request.

7 SERVER_OVERLOADED, server is processing too many PCP requests from this client or from other clients, and requests this client delay sending any other requests for the time indicated in Lifetime.

Additional result codes, specific to the OpCodes and Options defined in this document, are listed in Section 8.2 and Section 10.1.

6. **General PCP Operation**

PCP messages MUST be sent over UDP *[RFC0768]*. Every PCP request generates a response, so PCP does not need to run over a reliable transport protocol.
PCP is idempotent, so if the PCP client sends the same request multiple times and the PCP server processes those requests, the same result occurs. The order of operation is that a PCP client generates and sends a request to the PCP server, which processes the request and generates a response back to the PCP client.

6.1. **General PCP Client: Generating a Request**

This section details operation specific to a PCP client, for any OpCode. Procedures specific to the MAP OpCodes are described in Section 8, and procedures specific to the PEER OpCodes are described in Section 9.
Prior to sending its first PCP message, the PCP client determines which servers to use. The PCP client performs the following steps to determine its PCP server(s):

1. if a PCP server is configured (e.g., in a configuration file or DHCP), that single configuration source is used as the list of PCP server(s), else;

2. the address of the default router is used as the PCP server.

With that list of PCP servers, the PCP client formulates its PCP request. The PCP request contains a PCP common header, PCP OpCode and payload, and (possibly) Options. As with all UDP or TCP clients on any operating system, when several PCP clients are embedded in the same host, each uses a distinct source port number to disambiguate their

requests and replies. The PCP client's source port SHOULD be randomly generated [RFC6056].

When attempting to contact a PCP server, the PCP client initializes a timer to 2 seconds. The PCP client sends a PCP message the first server in its list of PCP servers. If no response is received before the timer expires, the timer is doubled (to 4 seconds) and the request is re-transmitted. If no response is received before the timer expires, the timer is doubled again (to 8 seconds) and the request is re-transmitted. This procedure is repeated in parallel or in series to each PCP server in the list, on each interface, until a response is received from a PCP server. If the requests are sent in parallel and responses from multiple PCP servers are received, only the PCP server closest to the top of the list, on that interface, is used for subsequent requests; PCP requests which received a positive response and create state (e.g., MAP) SHOULD have their state cleared (e.g., lifetime set to 0).

Once a PCP client has successfully received a response from a PCP server on that interface, it sends subsequent PCP requests to that same server, with a retransmission timer of 2 seconds. If, after 2 seconds, a response is not received from that PCP server, the same back-off algorithm described above is performed.

Upon receiving a response (success or error), the PCP client does not change to a different PCP server. That is, it does not "shop around" trying to find a PCP server to service its (same) request.

6.2. General PCP Server: Processing a Request

This section details operation specific to a PCP server. Processing SHOULD be performed in the order of the following paragraphs.

A PCP server processes incoming requests on the PCP port from clients or an operator-configured interface (e.g., from the ISP's network operations center). The PCP server MUST drop (ignore) requests that arrive from elsewhere (e.g., the Internet).

Upon receiving a message, the PCP server parses and validates it. A valid request contains a valid PCP common header, one valid PCP Opcode, and zero or more Options (which the server might or might not comprehend). If an error is encountered during processing, the server generates an error response which is sent back to the PCP client. Processing an OpCode and the Options are specific to each OpCode.

If the received message is shorter than 4 octets or has the R bit set the message is simply dropped. If the length of the request exceeds 1024 octets or is not a multiple of 4 octets, it is invalid. Invalid requests are handled by copying up to 1024 octets of the request into the response, setting the result code to MALFORMED_REQUEST, and zero-padding the response to a multiple of 4 octets if necessary. If the version number is not supported, a response is generated with the UNSUPP_VERSION result code and the other steps detailed in Section 6.6. If the OpCode is not supported, a response is generated with the UNSUPP_OPCODE result code.

If the source IP address of the received packet does not match the
contents of the PCP Client IP Address field, a response is generated
with the ADDRESS_MISMATCH result code. This is done to detect and
prevent accidental use of PCP where a non-PCP-aware NAT or NAPT exists
between the PCP client and PCP server.

Error responses have the same packet layout as success responses, with
fields from the request copied into the response, and fields assigned
by the PCP server are set as indicated in Figure 3

6.3. General PCP Client: Processing a Response

The PCP client receives the response and verifies the source IP address
and port belong to the PCP server of an outstanding PCP request. It
validates the OpCode matches an outstanding PCP request. Responses
shorter than 12 octets, longer than 1024 octets, or not a multiple of 4
octets are invalid and ignored, likely causing the request to be re-
transmitted. The response is further matched by comparing fields in the
response OpCode-specific data to fields in the request OpCode-specific
data, as described by the processing for that OpCode. After these
matches are successful, the PCP client checks the Epoch field to
determine if it needs to restore its state to the PCP server (see
Section 6.5).

If the result code is 0, the PCP client knows the request was
successful.

If the result code is not 0, the request failed. If the result code is
UNSUPP_VERSION, processing continues as described in Section 6.6. If
the result code is SERVER_OVERLOADED, clients SHOULD NOT send *any*
further requests to that PCP server for the indicated error lifetime.
For other error result codes, The PCP client SHOULD NOT resend the same
request for the indicated error lifetime. If a PCP server indicates an
error lifetime in excess of 30 minutes, A PCP client MAY choose to set
its retry timer to 30 minutes.

If the PCP client has discovered a new PCP server (e.g., connected to a
new network), the PCP client MAY immediately begin communicating with
this PCP server, without regard to hold times from communicating with a
previous PCP server.

6.4. Multi-Interface Issues

Hosts which desire a PCP mapping might be multi-interfaced (i.e., own
several logical/physical interfaces). Indeed, a host can be configured
with several IPv4 addresses (e.g., WiFi and Ethernet) or dual-stacked.
These IP addresses may have distinct reachability scopes (e.g., if IPv6
they might have global reachability scope as for Global Unicast Address
(GUA, [RFC3587]) or limited scope as for Unique Local Address (ULA)
[RFC4193]).

IPv6 addresses with global reachability (e.g., GUA) SHOULD be used as
the source address when generating a PCP request. IPv6 addresses
without global reachability (e.g., ULA [RFC4193]), SHOULD NOT be used

as the source interface when generating a PCP request. If IPv6 privacy addresses [RFC4941] are used for PCP mappings, a new PCP request will need to be issued whenever the IPv6 privacy address is changed. This PCP request SHOULD be sent from the IPv6 privacy address itself. It is RECOMMENDED that mappings to the previous privacy address be deleted. Due to the ubiquity of IPv4 NAT, IPv4 addresses with limited scope (e.g., private addresses [RFC1918]) MAY be used as the source interface when generating a PCP request.
As mentioned in Section 2.3, only single-homed CP routers are in scope. Therefore, there is no viable scenario where a host located behind a CP router is assigned with two Global Unicast Addresses belonging to different global IPv6 prefixes.

6.5. Epoch

Every PCP response sent by the PCP server includes an Epoch field. This field increments by 1 every second, and is used by the PCP client to determine if PCP state needs to be restored. If the PCP server resets or loses the state of its explicit dynamic Mappings (that is, those mappings created by PCP MAP requests), due to reboot, power failure, or any other reason, it MUST reset its Epoch time to 0. Similarly, if the public IP address(es) of the NAT (controlled by the PCP server) changes, the Epoch MUST be reset to 0. A PCP server MAY maintain one Epoch value for all PCP clients, or MAY maintain distinct Epoch values for each PCP client; this choice is implementation-dependent.
Whenever a client receives a PCP response, the client computes its own conservative estimate of the expected Epoch value by taking the Epoch value in the last packet it received from the gateway and adding 7/8 (87.5%) of the time elapsed since that packet was received. If the Epoch value in the newly received packet is less than the client's conservative estimate by more than one second, then the client concludes that the PCP server lost state, and the client MUST immediately renew all its active port mapping leases as described in Section 11.3.1.
When a client notices that the PCP server reduced its Epoch value, the PCP clients will send PCP requests to refresh their mappings. The PCP server needs to be scaled appropriately to accommodate this traffic. Because PCP lacks a mechanism to simultaneously inform all PCP clients of the Epoch value, the PCP clients will only flood the PCP server simultaneously when a power outage and restoration event causes state loss in both the PCP clients and PCP server.

6.6. Version Negotiation

A PCP client sends its requests using PCP version number 1. Should later updates to this document specify different message formats with a version number greater than 1 it is expected that PCP servers will still support version 1 in addition to the newer version(s). However, in the event that a server returns a response with error code

UNSUPP_VERSION, the client MAY log an error message to inform the user that it is too old to work with this server.
When sending a response containing the UNSUPP_VERSION result code, the PCP message MUST be 12 octets long.
If future PCP versions greater than 1 are specified, version negotiation is expected to proceed as follows:

1. If a client or server supports more than one version it SHOULD support a contiguous range of versions -- i.e., a lowest version and a highest version and all versions in between.

2. Client sends first request using highest (i.e., presumably 'best') version number it supports.

3. If server supports that version it responds normally.

4. If server does not support that version it replies giving a result containing the error code UNSUPP_VERSION, and the closest version number it does support (if the server supports a range of versions higher than the client's requested version, the server returns the lowest of that supported range; if the server supports a range of versions lower than the client's requested version, the server returns the highest of that supported range).

5. If the client receives an UNSUPP_VERSION result containing a version it does support, it records this fact and proceeds to use this message version for subsequent communication with this PCP server (until a possible future UNSUPP_VERSION response if the server is later updated, at which point the version negotiation process repeats).

6. If the client receives an UNSUPP_VERSION result containing a version it does not support then the client MAY log an error message to inform the user that it is too old to work with this server, and the client SHOULD set a timer to retry its request in 30 minutes or the returned Lifetime value, whichever is smaller.

6.7. General PCP Option

The following option can appear in certain PCP responses, without regard to the OpCode.

6.7.1. UNPROCESSED Option

If the PCP server cannot process a mandatory-to-process option, for whatever reason, it includes the UNPROCESSED Option in the response, shown in Figure 5. This helps with debugging interactions between the PCP client and PCP server. This option MUST NOT appear more than once

in a PCP response. The unprocessed options are listed once, and the
option data is zero-filled to the necessary 32 bit boundary. If a
certain Option appeared more than once in the PCP request, that Option
value can appear once or as many times as it occurred in the request.
The order of the Options in the PCP request has no relationship with
the order of the Option values in this UNPROCESSED Option. This Option
MUST NOT appear in a response unless the associated request contained
at least one mandatory-to-process Option.
The UNPROCESSED option is formatted as follows, showing an example of
two option codes that were unprocessed:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| option-code-1 | option-code-2 |          padding              |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Padding: 0, 1, 2, or 3 octets. If the number of option-codes is not a
multiple of 4, padding is used to make it 32-bit aligned. The padding
MUST be on on sending, and MUST be ignored by the receiver.

    *This Option:

        -name: UNPROCESSED

        -number: 0

        -purpose: indicates which PCP options in the request are not
         supported by the PCP server

        -is valid for OpCodes: all

        -length: 1 or more

        -may appear in: responses, and only if the result code is non-
         zero.

        -maximum occurrences: 1

7. Introduction to MAP and PEER OpCodes

There are four uses for the MAP and PEER OpCodes defined in this
document: a host operating a server and wanting an incoming connection;
a host operating a client and wanting to optimize the application
keepalive traffic or restore lost state in its NAT; and a host
operating a client and server on the same port. These are discussed in
the following sections.
When operating a server (Section 7.1 and Section 7.4) the PCP client
knows if it wants an IPv4 listener, IPv6 listener, or both on the
Internet. The PCP client also knows if it has an IPv4 address on itself

or an IPv6 interface on itself. It takes the union of this knowledge to decide to send a one or two MAP requests for each of its interfaces. Applications that embed IP addresses in payloads (e.g., FTP, SIP) will find it beneficial to avoid address family translation, if possible. It is REQUIRED that the PCP-controlled device assign the same external IP address to PCP-created explicit dynamic mappings and to implicit dynamic mappings. It is RECOMMENDED that static mappings (e.g., those created by a command-line interface on the PCP server or PCP-controlled device) also be assigned to the same IP address. Once all internal addresses belonging to a given subscriber have no implicit dynamic mappings and have no explicit dynamic mappings in the PCP-controlled device, a subsequent PCP request for that internal address MAY be assigned to a different external IP address. Generally, this re-assignment would occur when a CGN device is load balancing newly-seen hosts to its public IPv4 address pool.

7.1. For Operating a Server

A host operating a server (e.g., a web server) listens for traffic on a port, but the server never initiates traffic from that port. For this to work across a NAT or a firewall, the host needs to (a) create a mapping from a public IP address and port to itself as described in Section 8 and (b) publish that public IP address and port via some sort of rendezvous server (e.g., DNS, a SIP message, a proprietary protocol). Publishing the public IP address and port is out of scope of this specification. To accomplish (a), the host follows the procedures described in this section.
As normal, the application needs to begin listening on a port. Then, the application constructs a PCP message with the appropriate MAP OpCode depending on if it is listening on an IPv4 or IPv6 address and if it wants a public IPv4 or IPv6 address.
The following pseudo-code shows how PCP can be reliably used to operate a server:

```
/* start listening on the local server port */
int s = socket(...);
bind(s, ...);
listen(s, ...);

getsockname(s, &internal_sockaddr, ...);
external_sockaddr = 0;

while (1)
    {
    /* Note: the "time_to_send_pcp_request" check below includes:
     * 1. Sending the first request
     * 2. Retransmitting requests due to packet loss
     * 3. Resending a request due to impending lease expiration
     * The PCP packet sent is identical in all cases, apart from the
     * Suggested External Address and Port which may change over time
     */
    if (time_to_send_pcp_request)
        pcp_send_map_request(internal_sockaddr.sin_port,
            internal_sockaddr.sin_addr,
            &external_sockaddr, /* will be zero the first time */
            requested_lifetime, &assigned_lifetime);

    if (pcp_response_received)
        update_rendezvous_server("Client Ident", external_sockaddr);

    if (received_incoming_connection_or_packet)
        process_it(s);

    if (other_work_to_do)
        do_it();

    /* ... */

    block_until_we_need_to_do_something_else();
    }
```

7.2. For Reducing NAT Keepalive Messages

A host operating a client (e.g., XMPP client, SIP client) sends from a
port but never accepts incoming connections on this port. It wants to
ensure the flow to its server is not terminated (due to inactivity) by
an on-path NAT or firewall. To accomplish this, the application uses
the procedure described in this section.
Middleboxes such as NATs or firewalls need to see occasional traffic or
will terminate their session state, causing application failures. To
avoid this, many applications routinely generate keepalive traffic for
the primary (or sole) purpose of maintaining state with such

middleboxes. Applications can reduce such application keepalive traffic
by using PCP.

>   *Note: For reasons beyond NAT, an application may find it useful
>   to perform application-level keepalives, such as to detect a
>   broken path between the client and server, detect a crashed
>   server, or detect a powered-down client. These keepalives are not
>   related to maintaining middlebox state, and PCP cannot do
>   anything useful to reduce those keepalives.

To use PCP for this function, the application first connects to its
server, as normal. Afterwards, it issues a PCP request with the PEER4
or PEER6 OpCode as described in [Section 9](#). The PEER4 OpCode is used if
the host is using IPv4 for its communication to its peer; PEER6 if
using IPv6. The same 5-tuple as used for the connection to the server
is placed into the PEER4 or PEER6 payload.
The following pseudo-code shows how PCP can be reliably used with a
dynamic socket, for the purposes of reducing application keepalive
messages:

```
int s = socket(...);
connect(s, &remote_peer, ...);

getsockname(s, &internal_sockaddr, ...);
external_sockaddr = 0;

while (1)
    {
    /* Note: the "time_to_send_pcp_request" check below includes:
     * 1. Sending the first request
     * 2. Retransmitting requests due to packet loss
     * 3. Resending a request due to impending lease expiration
     * The PCP packet sent is identical in all cases, apart from the
     * Suggested External Address and Port which may change over time
     */
    if (time_to_send_pcp_request)
        pcp_send_peer_request(internal_sockaddr.sin_port,
            internal_sockaddr.sin_addr,
            &external_sockaddr, /* will be zero the first time */
            remote_peer, requested_lifetime, &assigned_lifetime);

    if (data_to_send)
        send_it(s);

    if (other_work_to_do)
        do_it();

    /* ... */

    block_until_we_need_to_do_something_else();
    }
```

7.3. For Restoring Lost Implicit TCP Dynamic Mapping State

After a NAPT loses state (e.g., because of a crash or power failure),
it is useful for clients to re-establish TCP mappings on the NAPT. This
allows servers on the Internet to see traffic from the same IP address
and port, so that sessions can be resumed exactly where they were left
off. This can be useful for long-lived connections (e.g., instant
messaging) or for connections transferring a lot of data (e.g., FTP).
This can be accomplished by establishing a TCP connection normally and
then sending a PEER request/response and remember the External Address
and External Port. Later, when the NAPT has lost state, the client can
send a PEER request with the Suggested External Port and Suggested
External Address remembered from the previous session, which will
create a mapping in the NAPT that functions exactly as an implicit

dynamic mapping. The client then resumes sending TCP data to the
server.

   *Note: This procedure works well for TCP, provided the NAPT only
    creates a new implicit dynamic mapping for TCP segments with the
    SYN bit set (i.e., the newly-booted NAPT drops the re-transmitted
    data segments from the client because the NAPT does not have an
    active mapping for those segments), and if the server is not
    sending data that elicits a RST from the NAPT. This is not the
    case for UDP.

7.4. For Operating a Symmetric Client/Server

A host operating a client and server on the same port (e.g., Symmetric
RTP *[RFC4961]* or SIP Symmetric Response Routing (rport) *[RFC3581]*)
first establishes a local listener, (usually) sends the local and
public IP addresses and ports to a rendezvous service (which is out of
scope of this document), and initiates an outbound connection from that
same source address and same port. To accomplish this, the application
uses the procedure described in this section.
An application that is using the same port for outgoing connections as
well as incoming connections MUST first signal its operation of a
server using the PCP MAP OpCode, as described in Section 8, and receive
a positive PCP response before it sends any packets from that port.

   *Discussion: Although reversing those steps is tempting (to
    eliminate the PCP round trip before a packet can be sent from
    that port) and will work if the NAT has endpoint-independent
    mapping (EIM) behavior, reversing the steps will fail if the NAT
    has non-EIM behavior. With a non-EIM NAT, the implicit mapping
    created by an outgoing TCP SYN and the explicit mapping created
    using the MAP OpCode will cause different ports to be assigned
    (which is not desirable; after all, the application is using the
    same port for outgoing and incoming traffic on purpose) and they
    will generally also have different lifetimes. PCP does not
    attempt to change or dictate how a NAT creates its mappings
    (endpoint independent mapping, or otherwise) so there is no
    assurance that an implicit mapping will be EIM or non-EIM. Thus,
    it is necessary for an application to first signal its operation
    of a server using the PCP MAP OpCode. See also Section 11.1.

The following pseudo-code shows how PCP can be used to operate a
symmetric client and server:

```
/* start listening on the local server port */
int s = socket(...);
bind(s, ...);
listen(s, ...);

getsockname(s, &internal_sockaddr, ...);
external_sockaddr = 0;

while (1)
    {
    /* Note: the "time_to_send_pcp_request" check below includes:
     * 1. Sending the first request
     * 2. Retransmitting requests due to packet loss
     * 3. Resending a request due to impending lease expiration
     * The PCP packet sent is identical in all cases, apart from the
     * Suggested External Address and Port which may change over time
     */
    if (time_to_send_pcp_request)
        pcp_send_map_request(internal_sockaddr.sin_port,
            internal_sockaddr.sin_addr,
            &external_sockaddr, /* will be zero the first time */
            requested_lifetime, &assigned_lifetime);

    if (pcp_response_received)
        update_rendezvous_server("Client Ident", external_sockaddr);

    if (received_incoming_connection_or_packet)
        process_it(s);

    if (need_to_make_outgoing_connection)
        make_outgoing_connection(s, ...);

    if (data_to_send)
        send_it(s);

    if (other_work_to_do)
        do_it();

    /* ... */

    block_until_we_need_to_do_something_else();
    }
```

8. MAP OpCodes

This section defines two OpCodes which control forwarding from a NAT
(or firewall) to an internal host. They are:

MAP4=1: create a mapping between an internal address and external IPv4
   address (e.g., NAT44, NAT64, or firewall)

MAP6=2:

>   create a mapping between an internal target address and
>   external IPv6 address (e.g., NAT46, or firewall)

The internal address is the source IP address of the PCP request
message itself, unless the THIRD_PARTY option is used.
Note that all mappings created by PCP MAP requests are, by definition,
Endpoint Independent Mappings (even on a NAT that usually creates
Endpoint Dependent Mappings for outgoing connections) since the purpose
of a MAP mapping is to receive inbound traffic from any remote
endpoint, not from only one specific remote endpoint.
Note also that all NAT mappings (created by PCP or otherwise) are by
necessity bidirectional and symmetrical. For any packet going in one
direction (in or out) that is translated by the NAT, a reply going in
the opposite direction needs to have the corresponding opposite
translation done so that the reply arrives at the right endpoint. This
means that if a client creates a MAP mapping, and then later sends an
outgoing packet using the mapping's internal source port, the NAT
should translate that packet's Internal Address and Port to the
mapping's External Address and Port, so that replies addressed to the
External Address and Port are correctly translated to the mapping's
Internal Address and Port.
The operation of the MAP OpCodes is described in this section.

8.1. OpCode Packet Formats

The two MAP OpCodes (MAP4, MAP6) share a similar packet layout for both
requests and responses. Because of this similarity, they are shown
together. For both of the MAP OpCodes, if the assigned external IP
address and assigned external port match the request's Internal IP
address and port, the functionality is purely a firewall; otherwise it
pertains to a network address translator which might also perform
firewall-like functions.
The following diagram shows the OpCode-specific information format in a
request for the MAP4 and MAP6 OpCodes.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Protocol     |           Reserved (24 bits)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Internal Port           |    Suggested External Port    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
: Suggested External IP Address (32 or 128, depending on OpCode):
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

These fields are described below:
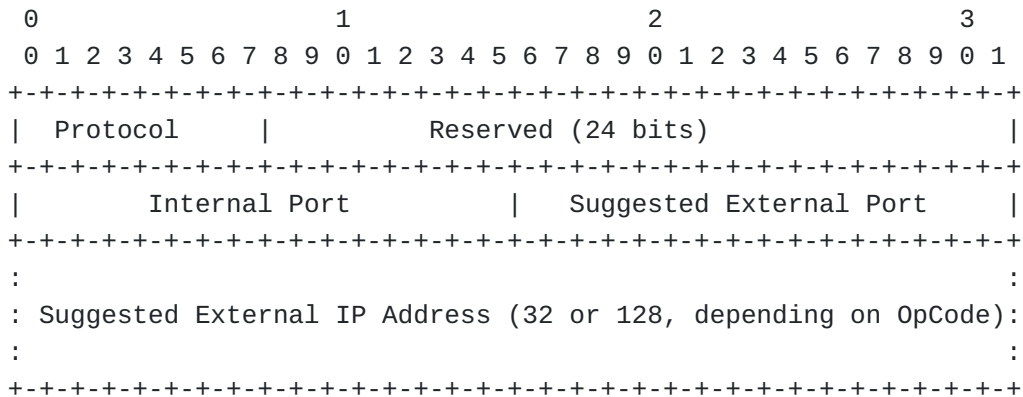
Requested lifetime (in common header):
                                     Requested lifetime of this
   mapping, in seconds. The value 0 indicates "delete".

Protocol: indicates upper-layer protocol associated with this OpCode.
   Values are taken from the [IANA protocol registry](#) *[proto_numbers]*.
   For example, this field contains 6 (TCP) if the opcode is intended
   to create a TCP mapping. The value 0 has a special meaning for 'all
   protocols', and is used only for delete requests. This means that
   HOPOPT (which is assigned by IANA as protocol 0) cannot have a
   mapping deleted by PCP.

Reserved: 24 reserved bits, MUST be sent as 0 and MUST be ignored when
   received.

Internal Port: Internal port for the mapping. The value 0 indicates
   "all ports", and is only legal in a request if lifetime=0.

Suggested External Port: suggested external port for the mapping. This
   is useful for refreshing a mapping, especially after the PCP server
   loses state. If the PCP client does not know the external port, or
   does not have a preference, it uses 0.

Suggested External IP Address: Suggested external IP address. This is
   useful for refreshing a mapping, especially after the PCP server
   loses state. If the PCP server can fulfill the request, it will do
   so. If the PCP client does not know the external address, or does
   not have a preference, it MUST use 0.

The following diagram shows the OpCode-specific information format in a
response packet for the MAP4 and MAP6 OpCodes:
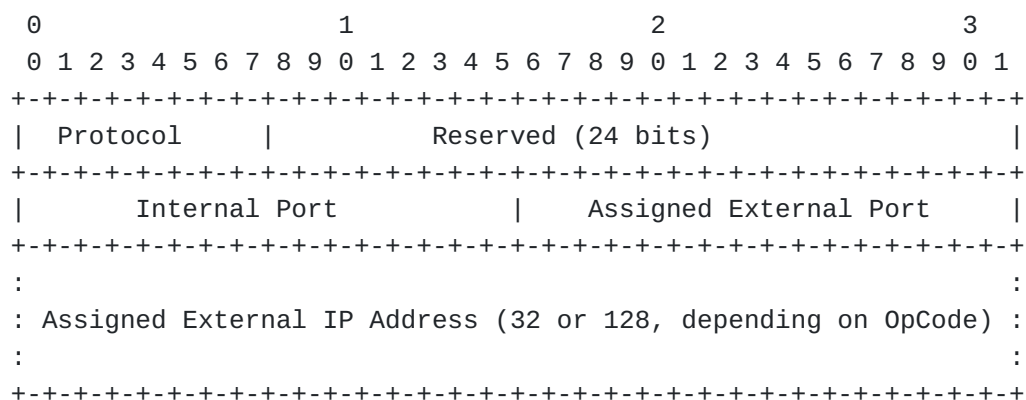
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Protocol     |            Reserved (24 bits)                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Internal Port          |      Assigned External Port    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
: Assigned External IP Address (32 or 128, depending on OpCode) :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

These fields are described below:

Lifetime (in common header): On a success response, this indicates the
   lifetime for this mapping, in seconds. On an error response, this
   indicates how long clients should assume they'll get the same error
   response from the that PCP server if they repeat the same request.

Protocol:

        Copied from the request

Reserved: 24 reserved bits, MUST be sent as 0 and MUST be ignored when
   received.

Internal Port: Internal port for the mapping, copied from request.

Assigned External Port: On success responses, this is the assigned
   external port for the mapping. On error responses, the value from
   Suggested External Port is used.

Assigned External IP Address: On success responses, this is the
   assigned external IPv4 or IPv6 address for the mapping; IPv4 or IPv6
   address is indicated by the OpCode. On error responses, the value
   from Suggested External IP Address is used.

8.2. [OpCode-Specific Result Codes](#)

In addition to the general PCP result codes ([Section 5.4](#)), the
following additional result codes may be returned as a result of the
four MAP OpCodes received by the PCP server. These errors are
considered 'long lifetime' or 'short lifetime', which provides guidance
to PCP server developers for the value of the Lifetime field for these
errors. It is RECOMMENDED that short lifetime errors use 30 second
lifetime and long lifetime errors use 30 minute lifetime.

20 NETWORK_FAILURE, PCP server or the device it controls are
   experiencing a network failure of some sort (e.g., has not obtained
   an IP address). This is a short lifetime error.

21 NO_RESOURCES, e.g., NAT device cannot create more mappings at this
   time. This is a system-wide error, and different from USER_EX_QUOTA.
   This is a short lifetime error.

22 UNSUPP_PROTOCOL, unsupported Protocol. This is a long lifetime
   error.

23 NOT_AUTHORIZED, e.g., PCP server supports mapping, but the feature
   is disabled for this PCP client, or the PCP client requested a
   mapping that cannot be fulfilled by the PCP server's security
   policy. This is a long lifetime error.

24 USER_EX_QUOTA, mapping would exceed user's port quota. This is a
   short lifetime error.

25 CANNOT_PROVIDE_EXTERNAL_PORT is returned only if the request
   included the PREFER_FAILURE option, because otherwise a new external
   port could have been allocated. See [Section 10.2](#) for processing
   details. The error lifetime depends on the reason for the failure.

EXCESSIVE_REMOTE_PEERS, indicates the PCP server was not able to
create the filters in this request. This result code MUST only be
returned if the MAP request contained the REMOTE_FILTER Option. See
Section 10.3 for processing information. This is a long lifetime
error.

Additional result codes may be returned if the THIRD_PARTY option is
used, see Section 10.1.

8.3. OpCode-Specific Client: Generating a Request

This section describes the operation of a PCP client when sending
requests with OpCodes MAP4 and MAP6.
The request MAY contain values in the suggested-external-ip-address and
suggested-external-port fields. This allows the PCP client to attempt
to rebuild the PCP server's state, so that the PCP client could avoid
having to change information maintained at the rendezvous server. Of
course, due to other activity on the network (e.g., by other users or
network renumbering), the PCP server may not be able to fulfill the
request.
An existing mapping can have its lifetime extended by the PCP client.
To do this, the PCP client sends a new MAP request indicating the
internal port. The PCP MAP request SHOULD also include the currently
allocated external IP address and port as the suggested external IP
address and port, so that if the NAT gateway has lost state it can
recreate the lost mapping with the same parameters.
The PCP client SHOULD renew the mapping before its expiry time,
otherwise it will be removed by the PCP server (see Section 8.6). In
order to prevent excessive PCP chatter, it is RECOMMENDED to send a
single renewal request packet when a mapping is halfway to expiration
time, then, if no SUCCESS result is received, another single renewal
request 3/4 of the way to expiration time, and then another at 7/8 of
the way to expiration time, and so on, subject to the constraint that
renewal requests MUST NOT be sent less than four seconds apart (a PCP
client MUST NOT send ever-closer-together requests in the last few
seconds before a mapping expires).

8.4. OpCode-Specific Server: Processing a Request

This section describes the operation of a PCP server when processing a
request with the OpCodes MAP4 or MAP6. Processing SHOULD be performed
in the order of the following paragraphs.
If the server is overloaded by requests (from a particular client or
from all clients), it MAY simply discard requests, as the requests will
be retried by PCP clients, or MAY generate the SERVER_OVERLOADED error
response, or both.
If the request contains internal-port=0 and the lifetime is non-zero,
the server MUST generate a MALFORMED_REQUEST error.

If the requested lifetime is not zero, it indicates a request to create a mapping or extend the lifetime of an existing mapping. Processing of the lifetime is described in Section 8.6.

If the PCP-controlled device is stateless (that is, it does not establish any per-flow state, and simply rewrites the address and/or port in a purely algorithmic fashion), the PCP server simply returns an answer indicating the external IP address and port yielded by this stateless algorithmic translation. This allows the PCP client to learn its external IP address and port as seen by remote peers. Examples of stateless translators include stateless NAT64 and 1:1 NAT44, both of which modify addresses but not port numbers.

If an Option with value less than 128 exists (i.e., mandatory to process) but that option does not make sense (e.g., the PREFER_FAILURE option is included in a request with lifetime=0), the request is invalid and generates a MALFORMED_OPTION error.

If the PCP server can allocate the suggested external port, and the request did not contain the PREFER_FAILURE Option, it SHOULD do so. This is beneficial for re-establishing state lost when the PCP server loses its state (e.g., due to a reboot). If the PCP server cannot allocate the suggested external port but can allocate some other port and the request did not contain the PREFER_FAILURE Option, the PCP server MUST do so and return the allocated port in the response. Cases where a NAT gateway cannot allocate the suggested external port include:

   *Where the suggested external port is already allocated to another
    existing explicit, implicit, or static mapping, or already
    forwarding traffic to some other internal address:port, or;

   *Where the suggested external port is already used by the NAT
    gateway for one of its own services (e.g., port 80 for the NAT
    gateway's own configuration pages), or;

   *When the suggested external port is otherwise prohibited by the
    PCP server's policy.

By default, a PCP-controlled device MUST NOT create mappings for a protocol not indicated in the request. For example, if the request was for a TCP mapping, a UDP mapping MUST NOT be created.

If the THIRD_PARTY option is not present in the request, the source IP address of the PCP packet is used when creating the mapping. If the THIRD_PARTY option is present, the PCP server validates that the client is authorized to make mappings on behalf of the indicated internal IP address. This validation depends on the PCP deployment scenario; see Section 13.3 for the validation procedure. If the internal IP address in the PCP request is not authorized to make mappings on behalf of the indicated internal IP address, an error response MUST be generated with result code NOT_AUTHORIZED.

Mappings typically consume state on the PCP-controlled device, and it is RECOMMENDED that a per-subscriber or per-host limit be enforced by the PCP server to prevent exhausting the mapping state. If this limit is exceeded, the result code USER_EX_QUOTA is returned.
If all of the proceeding operations were successful (did not generate an error response), then the requested mappings are created or refreshed as described in the request and a SUCCESS response is built. This SUCCESS response contains the same OpCode as the request, but with the "R" bit set.

8.5. OpCode-Specific Client: Processing a Response

This section describes the operation of the PCP client when it receives a PCP response for the OpCodes MAP4 or MAP6.
After performing common PCP response processing, the response is further matched with an outstanding request by comparing the protocol, internal IP address, internal port. On error responses, the assigned external address and assigned external port can also be used to match the responses (which is useful if several requests with the PREFER_FAILURE option are outstanding). Other fields are not compared, because the PCP server sets those fields.
If a successful response, the PCP client can use the external IP address and port(s) as desired. Typically the PCP client will communicate the external IP address and port(s) to another host on the Internet using an application-specific rendezvous mechanism such as DNS SRV records.
On an error response, clients SHOULD NOT repeat the same request to the same PCP server within the lifetime returned in the response.

8.6. Mapping Lifetime and Deletion

The PCP client requests a certain lifetime, and the PCP server responds with the assigned lifetime. The PCP server MAY grant a lifetime smaller or larger than the requested lifetime. The PCP server SHOULD be configurable for permitted minimum and maximum lifetime, and the RECOMMENDED values are 120 seconds for the minimum value and 24 hours for the maximum. It is RECOMMENDED that the server be configured to restrict lifetimes to less than 24 hours, because they will consume ports even if the internal host is no longer interested in receiving the traffic or no longer connected to the network. These recommendations are not strict, and deployments should evaluate the tradeoffs to determine their own minimum and maximum lifetime values.
Once a PCP server has responded positively to a mapping request for a certain lifetime, the port forwarding is active for the duration of the lifetime unless the lifetime is reduced by the PCP client (to a shorter lifetime or to zero) or until the PCP server loses its state (e.g., crashes). Mappings created by PCP MAP requests are not special or different to mappings created other ways. In particular, it is implementation-dependent if outgoing traffic extends the lifetime of

such mappings. PCP clients MUST NOT depend on this behavior to keep
mappings active, and MUST explicitly renew their mappings as required
by the Lifetime field in PCP response messages.
If the requested lifetime is zero (lifetime==0) then:

    *If the internal port is non-zero (port!=0) and protocol is non-
     zero (protocol!=0), it indicates a request to delete the
     indicated mapping immediately.

    *If the internal port is zero (port==0) and the protocol is non-
     zero (protocol!=0), it indicates a request to delete all mappings
     for this Internal Address for the given internal port for all
     transport protocols.

    *If the internal port is zero (port==0) and protocol is zero
     (protocol==0), it indicates a request to delete all mappings for
     this Internal Address for all transport protocols. This is useful
     when a host reboots or joins a new network, to clear out prior
     stale state from the NAT gateway before beginning to install new
     mappings.

The suggested external address and port fields are ignored in requests
where the requested lifetime is 0.
If the PCP client attempts to delete a single static mapping (i.e., a
mapping created outside of PCP itself), the error NOT_AUTHORIZED is
returned. If the PCP client attempts to delete an implicit dynamic
mapping (e.g., created by a TCP SYN), the PCP server deletes the
mapping and responds with the SUCCESS result code. If the PCP client
attempts to delete a mapping that does not exist, the SUCCESS result
code is returned (this is necessary for PCP to be idempotent). If the
PCP MAP request was for port=0 (indicating 'all ports'), the PCP server
deletes all of the explicit dynamic mappings it can (but not any
implicit mappings), and returns a SUCCESS response. If the deletion
request was properly formatted and successfully processed, a SUCCESS
response is generated with lifetime of 0 and the server copies the
protocol and internal port number from the request into the response.
An explicit dynamic mapping MUST NOT have its lifetime reduced by
transport protocol messages (e.g., TCP RST, TCP FIN).
An application that forgets its PCP-assigned mappings (e.g., the
application or OS crashes) will request new PCP mappings. This may
consume port mappings, if the application binds to a different Internal
Port every time it runs. The application will also likely initiate new
implicit dynamic mappings (e.g., TCP connections) without using PCP,
which will also consume port mappings. If there is a port mapping quota
for the internal host, frequent restarts such as this may exhaust the
quota. PCP provides some protections against such port consumption:
When a PCP client first acquires a new IP address (e.g., reboots or
joins a new network), it SHOULD remove mappings that may already be
instantiated for that new Internal Address. To do this, the PCP client

sends a MAP request with protocol, internal port, and lifetime set to
0. Some port mapping APIs (e.g., the "DNSServiceNATPortMappingCreate"
API provided by Apple's Bonjour on Mac OS X, iOS, Windows, Linux)
automatically monitor for process exit (including application crashes)
and automatically send port mapping deletion requests if the process
that requested them goes away without explicitly relinquishing them.
To reduce unwanted traffic and data corruption, UDP and TCP ports
should not be immediately re-used for an interval (TIME_WAIT interval
as discussed in [RFC0793]). However, the PCP server MUST allow the same
subscriber and same internal address to re-acquire the same port during
that interval.
As a side-effect of creating a mapping, ICMP messages associated with
the mapping MUST be forwarded (and also translated, if appropriate) for
the duration of the mapping's lifetime. This is done to ensure that
ICMP messages can still be used by hosts, without application
programmers or PCP client implementations needing to signal PCP
separately to create ICMP mappings for those flows.
The following list summarizes the sentinel values when deleting a
mapping using lifetime=0:

* **all ports, all protocols, all Internal Addresses for which the client
  is authorized:**
    internal address=0, via the THIRD_PARTY option

* **all ports, all protocols:** internal port=0, protocol=0

* **all ports, specific protocol:** internal port=0, protocol={protocol
    value} (e.g., protocol=6 for TCP)

* **one port, specific protocol:** internal port={port number},
    protocol={protocol value} (e.g., port=12345, protocol=6 for TCP)

8.7. Subscriber Renumbering and Address Change Events

The customer premises router might obtain a new IP address. This can
occur because of a variety of reasons including a reboot, power outage,
DHCP lease expiry, or other action by the ISP. If this occurs, traffic
forwarded to the subscriber might be delivered to another customer who
now has that address. This affects both implicit dynamic mappings and
explicit dynamic mappings. However, this same problem occurs today when
a subscriber's IP address is re-assigned, without PCP and without an
ISP-operated CGN. The solution is the same as today: the problems
associated with subscriber renumbering are caused by subscriber
renumbering and are eliminated if subscriber renumbering is avoided.
PCP defined in this document does not provide machinery to reduce the
subscriber renumbering problem.
When a new Internal Address is assigned to a host embedding a PCP
client, the NAT (or firewall) controlled by the PCP server will
continue to send traffic to the old IP address. Typically, the PCP

client will no longer receive traffic sent to that old IP address.
Assuming the PCP client wants to continue receiving traffic, it needs
to install new mappings for its new IP address. The suggested external
port field will not be fulfilled by the PCP server, in all likelihood,
because it is still being forwarded to the old IP address. Thus, a
mapping is likely to be assigned a new external port number and/or
public IP address. Note that this scenario is not expected to happen
routinely on a regular basis for most hosts, since most hosts renew
their DHCP leases before they expire (or re-request the same address
after reboot) and most DHCP servers honor such requests and grant the
host the same address it was previously using before the reboot.
A host might gain or lose interfaces while existing mappings are active
(e.g., Ethernet cable plugged in or removed, joining/leaving a WiFi
network). Because of this, if the PCP client is sending a PCP request
to maintain state in the PCP server, it SHOULD ensure those PCP
requests continue to use the same interface (e.g., when refreshing
mappings). If the PCP client is sending a PCP request to create new
state in the PCP server, it MAY use a different source interface or
different source address.

9. PEER OpCodes

This section defines two OpCodes for controlling dynamic connections.
They are:

PEER4=3: Create a mapping, or set or query an implicit dynamic mapping
   to a remote peer's IPv4 address.

PEER6=4: Create a mapping, or set or query an implicit dynamic mapping
   to a remote peer's IPv6 address.

The operation of these OpCodes is described in this section.

9.1. OpCode Packet Formats

The PEER OpCodes provide a single function: the ability for the PCP
client to query and (possibly) extend the lifetime of an existing
mapping.
The two PEER OpCodes (PEER4 and PEER6) share a similar packet layout
for both requests and responses. Because of this similarity, they are
shown together.
The following diagram shows the request packet format for PEER4 and
PEER6. This packet format is aligned with the response packet format:
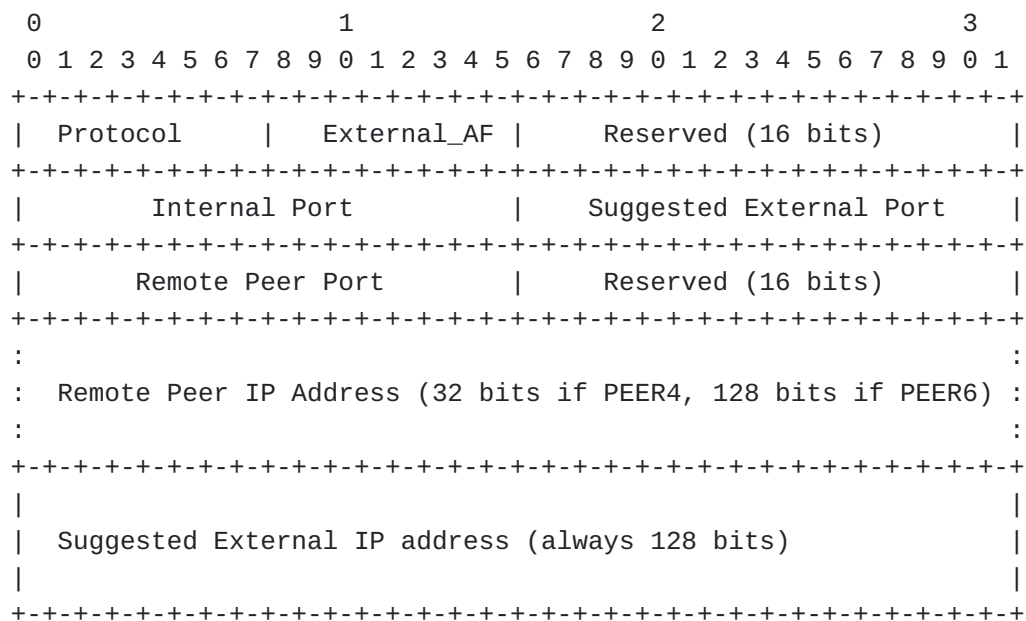
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Protocol    |  External_AF  |        Reserved (16 bits)     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Internal Port          |    Suggested External Port    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        Remote Peer Port        |       Reserved (16 bits)      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:  Remote Peer IP Address (32 bits if PEER4, 128 bits if PEER6) :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|   Suggested External IP address (always 128 bits)            |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

These fields are described below:

Requested Lifetime (in common header): Requested lifetime of this
   mapping, in seconds. Note that, depending on the implementation of
   the PCP-controlled device, it may not be possible to reduce the
   lifetime of a mapping (or delete it, with requested lifetime=0)
   using PEER.

Protocol: indicates upper-level protocol associated with this OpCode.
   Values are taken from the IANA protocol registry *[proto_numbers]*.
   For example, this field contains 6 (TCP) if the OpCode is describing
   a TCP peer.

External_AF: indicates the address family in the Suggested External IP
   Address field. Values are from IANA's address family numbers (IPv4
   is 1, IPv6 is 2), with the value 0 indicating the client is not
   attempting to re-create an existing mapping, and does not have a
   preference.

Reserved: 16 reserved bits, MUST be 0 on transmission and MUST be
   ignored on reception.

Internal Port: Internal port of the 5-tuple.

Suggested External Port: suggested external port for the mapping. This
   is useful for refreshing a mapping, especially after the PCP server
   loses state. If the PCP server can fulfill the request, it will do
   so. If the PCP client does not know the external port, or does not
   have a preference, it uses 0.

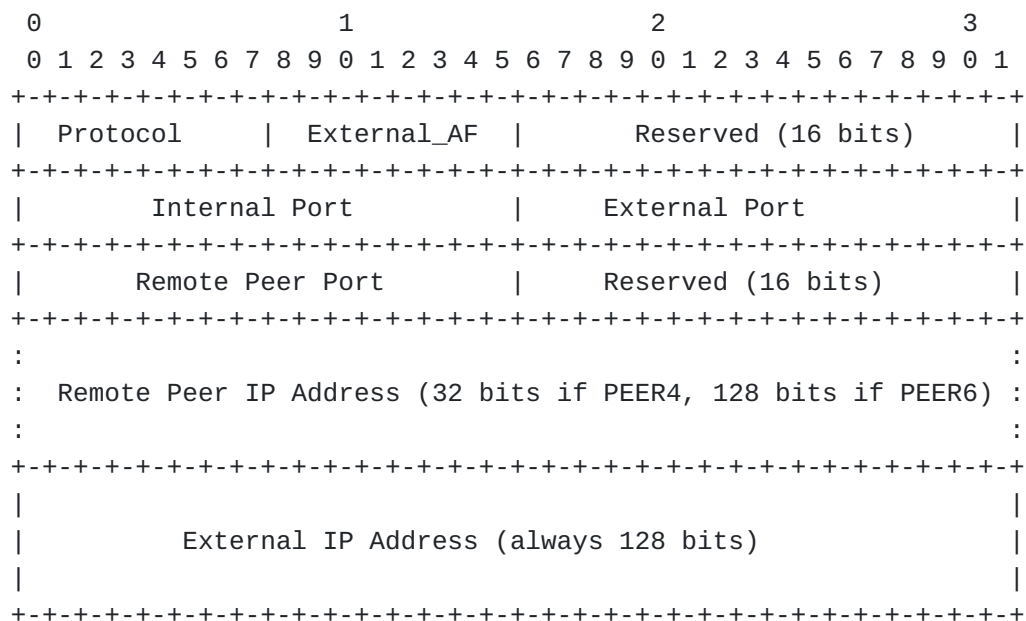**Remote Peer Port:** Remote peer's port of the 5-tuple.

Reserved:
> 16 reserved bits, MUST be 0 on transmission and MUST be ignored on reception.

Remote Peer IP Address: This is the Remote peer's IP address from the perspective of the PCP client so that the PCP client does not need to concern itself with NAT64 or NAT46 (which both cause the client's idea of the remote peer's IP address to differ from the remote peer's actual IP address). This field allows the PCP client and PCP server to disambiguate multiple connections from the same port on the internal host to different servers. Note this field has no bearing whatsoever on any filtering associated with the mapping.

Suggested External IP Address: always 128 reserved bits. If an IPv4 address, it is placed into the first 32 bits and the other 96 bits MUST be 0. The External_AF field indicates if the address is IPv4 or IPv6. If the client is not attempting to re-create a mapping, it MUST use the value 0.

The following diagram shows the response packet format for PEER4 and PEER6:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Protocol      | External_AF   |       Reserved (16 bits)      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Internal Port           |       External Port           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       Remote Peer Port        |       Reserved (16 bits)      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:  Remote Peer IP Address (32 bits if PEER4, 128 bits if PEER6) :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|         External IP Address (always 128 bits)                 |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Lifetime (in common header): On a success response, this indicates the lifetime for this mapping, in seconds. On an error response, this indicates how long clients should assume they'll get the same error response from the PCP server if they repeat the same request.

Protocol: Copied from the request.

External_AF: For success responses, this contains the address family of the external IP address associated with this peer connection, to properly decode the External IP Address. This field is necessary

because the Remote Peer's IP Address is from the PCP client's
perspective, whereas the External_AF and External IP Address are
from the PCP-controlled device's perspective. As an example, if the
PCP-controlled device is a NAT64, the PCP client only knows the
remote peer's IPv6 address, whereas the NAT64 knows the remote
peer's IPv4 address. Values are from IANA's address family numbers
(IPv4 is 1, IPv6 is 2). For error responses, the value is copied
from the request.

Reserved: 16 reserved bits, MUST be 0 on transmission, MUST be ignored
on reception.

Internal Port: copied from request.

External Port: For success responses, this is the external port
number, assigned by the NAT (or firewall) to this mapping. If
firewall or 1:1 NAT, this will match the internal port. For error
responses, this MUST be 0.

Remote Peer port: Copied from request.

Reserved: 16 reserved bits, MUST be 0 on transmission, MUST be ignored
on reception.

Remote Peer IP Address: Copied from the request.

External IP Address: For success responses, this contains the external
IP address, assigned by the NAT (or firewall) to this mapping. This
field allows the PCP client and its remote peer to determine if
there is another NAT between the PCP-controlled NAT and remote peer.
If the PCP-controlled device is a firewall, this will match the
internal IP address. If an IPv4 address, it is placed in the first
32 bits and the remaining 96 bits MUST be 0. For error responses,
the value is copied from the request.

9.2. OpCode-Specific Client: Generating a Request

This section describes the operation of a client when generating the
OpCodes PEER4 or PEER6.
The PEER4 or PEER6 OpCodes MAY be sent before or after establishing bi-
directional communication with the remote peer. If sent before, PEER4
or PEER6 OpCodes will create a mapping in the PCP-controlled device
(for the purpose described in Section 7.3), and the client SHOULD set
the External_AF and Suggested External IP Address to the values of the
previous mapping. If sent after, the PEER4 or PEER6 OpCodes query (and
control) the implicit dynamic mapping (for the purpose described in
Section 7.2).
The PEER4 and PEER6 OpCodes contain a description of the remote peer
address, from the perspective of the PCP client. This is necessary when
the PCP-controlled device is performing address family translation

(NAT46 or NAT64), because the destination address from the perspective of the PCP client is different from the destination address on the other side of the address family translation device. For this reason, the PEER4 and PEER6 responses contain an External_AF field.

9.3. OpCode-Specific Server: Processing a Request

This section describes the operation of a server when receiving a request with the OpCode PEER4 or PEER6. Processing SHOULD be performed in the order of the following paragraphs.
On receiving the PEER4 or PEER6 OpCode, the PCP server examines the mapping table. If the described mapping does not yet exist yet, it is created, honoring and the Suggested External Port and Suggested External IP Address are honored (if possible; if not possible, a mapping on a different IP address or different port is created). By having PEER create such a mapping, we avoid a race condition between the PEER request or the initial outgoing packet arriving at the NAT gateway first, and allow PEER to be used to recreate an implicit dynamic mapping (see last paragraph of Section 11.3.1).
The PEER4 or PEER6 OpCode MAY reduce the lifetime of an existing mapping; this is implementation-dependent.
If the PCP-controlled device can extend the lifetime of a mapping, the PCP server uses the smaller of its configured maximum lifetime value and the requested lifetime from the PEER request, and sets the lifetime to that value.
If all of the proceeding operations were successful (did not generate an error response), then a SUCCESS response is generated, with the Lifetime field containing the lifetime of the mapping.
After a successful PEER response is sent, it is implementation-specific if the PCP-controlled device destroys the mapping when the lifetime expires, or if the PCP-controlled device's implementation allows traffic to keep the mapping alive. Thus, if the PCP client wants the mapping to persist beyond the lifetime, it MUST refresh the mapping (by sending another PEER message) prior to the expiration of the lifetime.
If the mapping is terminated by the TCP client or server (e.g., TCP FIN or TCP RST), the mapping will be destroyed normally; the mapping will not persist for the time indicated by Lifetime. This means the Lifetime in a PEER response indicates how long the mapping will persist in the absence of a transport termination message (e.g., TCP RST).

9.4. OpCode-Specific Client: Processing a Response

This section describes the operation of a client when processing a response with the OpCode PEER4 or PEER6.
After performing common PCP response processing, the response is further matched with a request by comparing the protocol, external AF, internal IP address, internal port, remote peer address and remote peer port. Other fields are not compared, because the PCP server changes

those fields to provide information about the mapping created by the OpCode.

If a successful response, the application can use the assigned lifetime value to reduce its frequency of application keepalives for that particular NAT mapping. Of course, there may be other reasons, specific to the application, to use more frequent application keepalives. For example, the PCP assigned lifetime could be one hour but the application may want to maintain state on its server (e.g., "busy" / "away") more frequently than once an hour.

If the PCP client wishes to keep this mapping alive beyond the indicated lifetime, it SHOULD issue a new PCP request prior to the expiration. That is, inside->outside traffic is not sufficient to ensure the mapping will continue to exist. It is RECOMMENDED to send a single renewal request packet when a mapping is halfway to expiration time, then, if no SUCCESS response is received, another single renewal request 3/4 of the way to expiration time, and then another at 7/8 of the way to expiration time, and so on, subject to the constraint that renewal requests MUST NOT be sent less than four seconds apart (a PCP client MUST NOT ever-closer-together requests in the last few seconds before a mapping expires).

   *Note: implementations need to expect the PEER response may
    contain a different External_AF value than the request (e.g., due
    to NAT64 or NAT46).

10. Options for MAP and PEER OpCodes

This section describes Options for the MAP4, MAP6, PEER OpCodes. These Options MUST NOT appear with other OpCodes, unless permitted by those OpCodes.

10.1. THIRD_PARTY Option for MAP and PEER OpCodes

This Option is used when a PCP client wants to control a mapping to an internal host other than itself. This is used with both MAP and PEER OpCodes.

A THIRD_PARTY Option MUST NOT contain the same address as the source address of the packet. A PCP server receiving a THIRD_PARTY Option specifying the same address as the source address of the packet MUST return a MALFORMED_REQUEST result code. This is because many PCP servers may not implement the THIRD_PARTY Option at all, and a client using the THIRD_PARTY Option to specify the same address as the source address of the packet will cause mapping requests to fail where they would otherwise have succeeded.

Where possible, it may beneficial if a client using the THIRD_PARTY option to create and maintain mappings on behalf of some other device can take steps to verify that the other device is still present and active on the network. Otherwise the client using the THIRD_PARTY option to maintain mappings on behalf of some other device risks

maintaining those mappings forever, long after the device that required
them has gone. This would defeat the purpose of PCP mappings having a
finite lifetime so that they can be automatically deleted after they
are no longer needed.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:    Internal IP Address (32 bits of 128 bits, depending       :
:                                          on Option length)    :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

The fields are described below:

Internal IP Address: IP address of this mapping. If the length of this
   Option is 4, this is a 32-bit IPv4 address. If the length of this
   Option is 16, this is a 128-bit IPv6 address. This can contain the
   special value "0" (all zeros), which indicates "all Internal
   Addresses for which this client is authorized" which is used to
   delete all pre-existing mappings with the MAP Opcode.

This Option:

     *name: THIRD_PARTY

     *number: 4

     *purpose: Indicates the MAP or PEER request is for a host other
      than the host sending the PCP option.

     *is valid for OpCodes: MAP4, MAP6, PEER4, PEER6

     *length: 4 if Internal IP Address is IPv4, 16 if Internal IP
      Address is IPv6.

     *may appear in: request. May appear in response only if it
      appeared in the associated request.

     *maximum occurrences: 1

The following additional result codes may be returned as a result of
using this Option.

51 UNAUTH_THIRD_PARTY_INTERNAL_ADDRESS, indicating the internal IP
   address specified is not permitted (e.g., client is not authorized
   to make mappings for this Internal Address, or is otherwise
   prohibited.). This error can be returned for both MAP and PEER
   requests. If this is a MAP request, this is a long-term error.

A PCP server MAY be configured to permit or to restrict the use of the THIRD_PARTY option. If this option is permitted, any host can create, modify, or destroy mappings for another host on the subscriber's network. If third party mappings are restricted, only authorized clients can perform these operations. If a PCP server is configured to restrict third party mappings, and receives a PCP MAP request with a THIRD_PARTY option, it MUST generate a UNAUTH_THIRD_PARTY_INTERNAL_ADDRESS response. Determining which PCP clients are authorized to use the THIRD_PARTY option depends on the deployment scenario. For Dual-Stack Lite deployments, the PCP server only supports this option if the source IPv6 address is the B4's source IP address. For home deployments (where the PCP server is embedded in the NAT device), this option MUST NOT be processed. For scenarios where the subscriber has only one IP address (e.g., typical residential ISP service) this Option serves no purpose (and will only generate error messages from the server). If a subscriber has more than one IP address the ISP MUST determine its own policy for how to identify the trusted device within the subscriber's home. This might be, for example, the lowest- or highest-numbered host address for that user's IPv4 prefix. A cryptographic authentication and authorization model is outside the scope of this specification.

It is RECOMMENDED that PCP servers embedded into customer premise equipment be configured to refuse third party mappings by default. With this default, if a user wants to create a third party mapping, the user needs to interact out-of-band with their customer premise router (e.g., using its HTTP administrative interface).

It is RECOMMENDED that PCP servers embedded into service provider NAT and firewall devices be configured to permit the THIRD_PARTY option, when sent by the customer premise router. With this configuration, if a user wants to create an explicit dynamic mapping or query an implicit dynamic mapping for another host within their network, the user needs to interact out-of-band with their customer premise router (e.g., using its HTTP administrative interface). To accomplish this, the PCP server in the ISP's network processes requests with the THIRD_PARTY option if they arrived from the IP address of the customer premise router. In deployments with only one IP address (e.g., which is common in residential networks), the PCP messages will -- by necessity -- arrive from the IP address of the customer premise router router. In networks where users have multiple IPv4 or multiple IPv6 addresses, the PCP server MUST only allow the THIRD_PARTY option if the PCP message was sent by the IP address of the subscriber's customer premise router. In Dual-Stack Lite, this would be the B4 element's IPv6 address. If the packet arrived from a different address, the PCP server MUST generate an UNAUTH_THIRD_PARTY_INTERNAL_ADDRESS error.

If authorized to do so, a PCP client can delete all the PCP-created explicit dynamic mappings (i.e., those created by PCP MAP requests) for all hosts belonging to the same subscriber. This is done by sending a PCP MAP request including the THIRD_PARTY option with its Internal Address field set to 0.

10.2. PREFER_FAILURE Option for MAP OpCodes

This option is only used with the MAP4 and MAP6 OpCodes.
This option indicates that if the PCP server is unable to map the
Suggested External Port, the PCP server should instead return an error.
The error returned would be a general MAP error (e.g., NOT_AUTHORIZED)
or the error code specific to this Option,
CANNOT_PROVIDE_EXTERNAL_PORT.
The error code CANNOT_PROVIDE_EXTERNAL_PORT is returned if the
requested external port cannot be mapped. This can occur because the
external port is already mapped to another host's implicit dynamic
mapping, an explicit dynamic mapping, a static mapping, or the same
internal host and port has an implicit dynamic mapping which is mapped
to a different external port than requested. The server MAY set the
Lifetime in the response to the remaining lifetime of the conflicting
mapping.
This option is intended solely for use by UPnP IGD interworking *[I-
D.bpw-pcp-upnp-igd-interworking]*, where the semantics of UPnP IGD
version 1 only allow the UPnP IGD client to dictate mapping a specific
port. A PCP server MAY support this option, if its designers wish to
support downstream devices that perform UPnP IGD interworking. PCP
servers MAY choose to rate-limit their handling of PREFER_FAILURE
requests, to protect themselves from a rapid flurry of 65535
consecutive PREFER_FAILURE requests from clients probing to discover
which external ports are available. PCP servers that are not intended
to support downstream devices that perform UPnP IGD interworking are
not required to support this option. PCP clients other than UPnP IGD
interworking clients SHOULD NOT use this option because it results in
inefficient operation, and they cannot safely assume that all PCP
servers will implement it. It is anticipated that this option will be
deprecated in the future as more clients adopt PCP natively and the
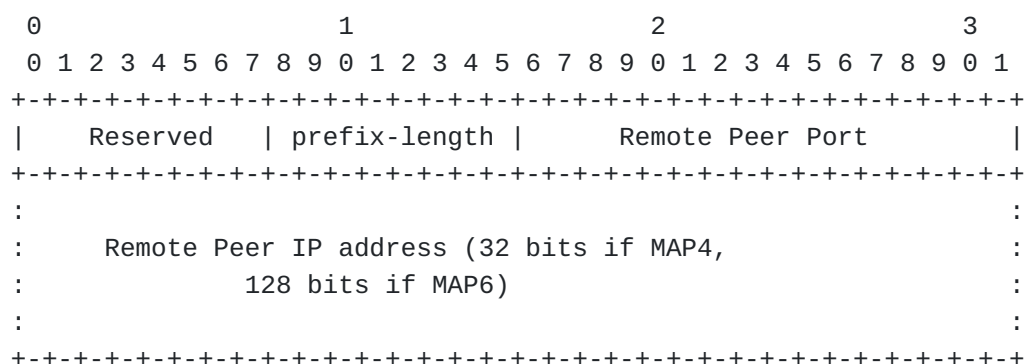need for UPnP IGD interworking declines.

   *This Option:

      -name: PREFER_FAILURE

      -number: 3

      -is valid for OpCodes: MAP4, MAP6

      -is included in responses: MUST

      -length: 0

      -may appear in: requests

      -maximum occurrences: 1

10.3. FILTER Option for MAP OpCodes

This Option indicates filtering incoming packets is desired. The remote peer port and remote peer IP Address indicate the permitted remote peer's source IP address and port for packets from the Internet. The remote peer prefix length indicates the length of the remote peer's IP address that is significant; this allows a single Option to permit an entire subnet. After processing this MAP request containing the FILTER option and generating a successful response, the PCP-controlled device will drop packets received on its public-facing interface that don't match the filter fields. After dropping the packet, if its security policy allows, the PCP-controlled device MAY also generate an ICMP error in response to the dropped packet.
The FILTER packet layout is described below:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Reserved   | prefix-length |      Remote Peer Port         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:      Remote Peer IP address (32 bits if MAP4,                 :
:                128 bits if MAP6)                              :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

These fields are described below:

Reserved: 8 reserved bits, MUST be sent as 0 and MUST be ignored when received.

prefix-length: indicates how many bits of the IPv4 or IPv6 address are relevant for this filter. The value 0 indicates "no filter", and will remove all previous filters. See below for detail.

Remote Peer Port: the port number of the remote peer. The value 0 indicates "all ports"

Remote Peer IP address: The IP address of the remote peer.

This Option:

> *name: FILTER

> *number: 2

> *is valid for OpCodes: MAP4, MAP6

> *is included in responses: MUST, if it appeared in the request

*length: 2 if used with MAP4, 5 if used with MAP6

   *may appear in: requests, and MUST appear in successfully-
    processed responses

   *maximum occurrences: as many as fit within maximum PCP message
    size

The prefix-length indicates how many bits of the IPv6 address or IPv4
address are used for the filter. For MAP4, a prefix-length of 32
indicates the entire IPv4 address is used. For MAP6, a prefix-length of
128 indicates the entire IPv6 address is used. For MAP4 the minimum
prefix-length value is 0 and the maximum value is 32. For MAP6 the
minimum prefix-length value is 0 and the maximum value is 128. Values
outside those range cause an MALFORMED_OPTION result code.
If multiple occurrences of the FILTER option exist in the same MAP
request, they are processed in the same order received, and they MUST
all be successfully processed or return an error (e.g.,
MALFORMED_OPTION if one of the options was malformed), and they MAY
overlap the filtering requested. As with other PCP errors, returning an
error causes no state to be changed in the PCP server or in the PCP-
controlled device. If an existing mapping exists (with or without a
filter) and the server receives a MAP request with FILTER, the filters
indicated in the new request are added to any existing filters. If a
MAP request has a lifetime of 0 and contains the FILTER option, the
error MALFORMED_OPTION is returned.
To remove all existing filters, the prefix-length 0 is used. There is
no mechanism to remove a specific filter.
To change an existing filter, the PCP client sends a MAP request
containing two FILTER options, the first option containing a prefix-
length of 0 (to delete all existing filters) and the second containing
the new remote peer's IP address and port. Other FILTER options in that
PCP request, if any, add more allowed remote hosts.
The PCP server or the PCP-controlled device is expected to have a limit
on the number of remote peers it can support. This limit might be as
small as one. If a MAP request would exceed this limit, the entire MAP
request is rejected with the result code EXCESSIVE_REMOTE_PEERS, and
the state on the PCP server is unchanged.

11. Implementation Considerations

This section provides non-normative guidance that may be useful to
implementors.

11.1. Implementing MAP with non-EIM port-mapping NAPT

For implicit dynamic mappings, some existing NAT devices have endpoint-
independent mapping (EIM) behavior while other NAPT devices have non-
endpoint-independent mapping (non-EIM) behavior. NAPTs which have EIM

behavior do not suffer from the problem described in this section. EIM behavior is strongly encouraged by both [RFC4787] and [RFC5382]. In such non-EIM NAPT devices, the same external port may be used by an implicit dynamic connection (from the same internal host or from a different internal host) and an explicit dynamic connection. This complicates the interaction with the MAP4 and MAP6 OpCodes. With such NAT devices, there are two ways envisioned to implement the MAP4 and MAP6 OpCodes:

1. have implicit dynamic mappings (e.g., TCP SYN) use a different set of public ports than explicit dynamic mappings (e.g., those created with MAP4 or MAP6), thus avoiding the interaction problem between them.

2. on arrival of a packet (from the Internet or from an internal host), first attempt to use an implicit dynamic mapping to process that packet. If none match, then the incoming packet should use the explicit dynamic mapping to process that packet. This effectively 'prioritizes' implicit dynamic mappings above explicit dynamic mappings.

11.2. Interaction of MAP with Implicit Dynamic Mappings

No matter if a NAPT is EIM or non-EIM, it is possible that one (or more) implicit dynamic mappings, using the same internal port on the internal host, might be created before or after a MAP request. When this occurs, it is important that the PCP server and NAPT honor the Lifetime returned in the MAP response. Specifically, if a mapping was created with the MAP OpCode, the implementation needs to ensure that termination of an implicit dynamic mapping (e.g., via a TCP FIN handshake) does not prematurely destroy the MAP-created mapping. On an EIM NAT, this could be implemented by extending the lifetime of the implicit dynamic mapping to the lifetime of the explicit dynamic mapping.

11.3. PCP Failure Scenarios

If an event occurs that causes the PCP server to lose explicit dynamic mapping state (such as a crash or power outage), the mappings created by PCP are lost. Such loss of state is rare in a service provider environment (due to redundant power, disk drives for storage, etc.), but more common in a residential NAT device which does not write information to its non-volatile memory. Of course, due to outright failure of service provider equipment (e.g., software malfunction), state may still be lost.
The Epoch allows a client to deduce when a PCP server may have lost its state. When the Epoch value is smaller than expected, the PCP client can attempt to recreate the mappings following the procedures described in this section.

11.3.1. Recreating Mappings

When the PCP server loses state and begins processing new PCP messages, its Epoch is reset to zero (per the procedure of Section 6.5). A mapping renewal packet is formatted identically to an original mapping request; from the point of view of the client it is a renewal of an existing mapping, but from the point of view of the PCP server it appears as a new mapping request. In the normal process of routinely renewing its mappings before they expire, a PCP client will automatically recreate all its lost mappings.
In addition, as the result of receiving a packet where the Epoch field indicates that a reboot or similar loss of state has occurred, the client can renew its port mappings sooner, without waiting for the normal routine renewal time.

11.3.2. Maintaining Mappings

A PCP client can refresh a mapping by sending a new PCP request containing information from the earlier PCP response. The PCP server will respond indicating the new lifetime. It is possible, due to failure of the PCP server, that the public IP address and/or public port, or the PCP server itself, has changed (due to a new route to a different PCP server). To detect such events more quickly, the PCP client may find it beneficial to use shorter lifetimes (so that it communicates with the PCP server more often). If the PCP client has several mappings, the Epoch value only needs to be retrieved for one of them to verify the PCP server has not lost explicit dynamic mapping state.
If the client wishes to check the PCP server's Epoch, it sends a PCP request for any one of the client's mappings. This will return the current Epoch value. In that request the PCP client could extend the mapping lifetime (by asking for more time) or maintain the current lifetime (by asking for the same number of seconds that it knows are remaining of the lifetime).
If an internal IP address is no longer valid (e.g., because the internal host has moved to a new network), and the PCP client wishes to still receive incoming traffic, it needs create a new mapping on that new network. A new mapping will also require an update to the application-specific rendezvous server (see Section 7.1 and Section 8.7).

12. Deployment Considerations

12.1. Ingress Filtering

To prevent spoofing of PCP requests, ingress filtering [RFC2827] MUST be performed by devices between the PCP clients and PCP server. For example, with a PCP server integrated into a customer premise router, the Ethernet switch needs to perform ingress filtering. As another

example, with a PCP server deployed by a service provider, the service provider's aggregation router (the first device connecting to subscribers) needs to do ingress filtering.

12.2. Per-Subscriber Explicit Dynamic Mapping Quota

On PCP-controlled devices that create state when a mapping is created (e.g., NAPT), the PCP server SHOULD maintain a per-subscriber quota for explicit dynamic mappings. It is implementation-specific if the PCP server has a separate or combined quota for both implicit dynamic mappings (e.g., created by TCP SYNs) and explicit dynamic mappings (created using PCP).

13. Security Considerations

This document defines Port Control Protocol and two types of OpCodes, PEER and MAP. The PEER OpCode allows querying and extending (if permitted) the lifetime of an existing implicit dynamic mapping, so a host can reduce its keepalive messages. The MAP OpCode allows creating a mapping so a host can receive incoming unsolicited connections from the Internet in order to run a server.
The PEER OpCode can create a mapping (which behaves exactly as if an implicit dynamic mapping were created (e.g., by a TCP SYN)). In that case, the security implications for PEER are similar to MAP, described below. When PEER is used to set (or query) an existing mapping, it does not introduce any new security considerations, unless the THIRD_PARTY Option is included. Discussion of the THIRD_PARTY Option is below.
With the exception of wireless providers (who are interested in protecting their radio access network), Internet service providers do not typically filter traffic from the Internet towards their subscribers. However, when an ISP introduces stateful address sharing with a NAPT device, such filtering will occur as a side effect of the NAPT device. Filtering will also occur with an IPv6 CPE [RFC6092]. The MAP OpCode allows a PCP client to create a mapping so that a host can receive inbound traffic and operate a server. Security considerations for the MAP OpCode are described in the following sections.

13.1. Denial of Service

Because of the state created in a NAPT or firewall, a per-subscriber quota will likely exist for both implicit dynamic mappings (e.g., outgoing TCP connections) and explicit dynamic mappings (PCP). A subscriber might make an excessive number of implicit or explicit dynamic mappings, consuming an inordinate number of ports, causing a denial of service to other subscribers. Thus, Section 12.2 recommends that subscribers be limited to a reasonable number of explicit dynamic mappings.

13.2. Ingress Filtering

It is important to prevent a subscriber from creating a mapping for another subscriber (or for another host), because this allows incoming packets from the Internet and consumes the other user's mapping quota. Both implicit dynamic mappings (e.g., outgoing TCP connections) and explicit dynamic mappings (PCP) need ingress filtering. Thus, PCP relies on the same ingress filtering as implicit dynamic mappings and does not create a new requirement for ingress filtering.

13.3. Validating THIRD_PARTY Internal Address

The THIRD_PARTY Option contains a Internal Address field, which allows a PCP client to create, extend, or delete an implicit or explicit dynamic mapping for another host.
In scenarios where the subscriber has one IP address (e.g., as commonly occurs with IPv4 residential deployments) or the subscriber has multiple IP addresses and a CP router enforces a PCP policy (by operating its own PCP server or performing filtering [RFC6092]), the PCP server in both the CP router and the ISP's equipment will both reject any message containing THIRD_PARTY. Thus, PCP cannot be used by a host to create, modify, or delete mappings of other hosts, except by using the administrative interface of the customer premise router (e.g., HTTP interface), as described in Section 10.1.
In other scenarios, where the subscriber has multiple IP addresses and the subscriber CP router is not filtering, but the ISP is providing filtering, the ISP should only accept PCP messages containing the THIRD_PARTY Option from the IP address of the customer's router, as described in Section 10.1.

13.4. Theft of mapping

In the time between when a PCP server loses state and the PCP client notices the lower-than-expected Epoch value, it is possible that the PCP client's mapping will be acquired by another host (via an explicit dynamic mapping or implicit dynamic mapping). This means incoming traffic will be sent to a different host ("theft"). A mechanism to immediately inform the PCP client of state loss would reduce this interval, but would not eliminate this threat. The PCP client can reduce this interval by using a relatively short lifetime; however, this increases the amount of PCP chatter. This threat is eliminated by using persistent storage of explicit dynamic mappings in the PCP server (so it does not lose explicit dynamic mapping state), or by ensuring the previous external IP address and port cannot be used by another host (e.g., by using a different IP address pool). This threat can be mitigated by authenticating the data connection between the hosts (e.g., using TLS).

14. IANA Considerations

IANA is requested to perform the following actions:

14.1. Port Number

PCP will use port 5351 (currently assigned by IANA to NAT-PMP). We request that IANA re-assign that same port number to PCP, and relinquish UDP port 44323.

14.2. OpCodes

IANA shall create a new protocol registry for PCP OpCodes, initially populated with the values in Section 8 and Section 9. The values 0 and 127 are reserved.
Additional OpCodes in the range 4-95 can be created via Standards Action *[RFC5226]*, and the range 96-126 is for Private Use *[RFC5226]*.

14.3. Result Codes

IANA shall create a new registry for PCP result codes, numbered 0-255, initially populated with the result codes from Section 5.4, Section 8.2, Section 10.3, and Section 10.1. The values 0 and 255 are reserved. Additional Result Codes can be defined via Specification Required *[RFC5226]*.

14.4. Options

IANA shall create a new registry for PCP Options, numbered 0-255 with an associated mnemonic. The values 0-127 are mandatory-to-process, and 128-255 are optional-to-process. The initial registry contains the options described in Section 10 and Section 10.1. The option values 127 and 255 are reserved.
Additional PCP option codes in the ranges 5-63 and 128-191 can be created via Standards Action *[RFC5226]*, and the ranges 64-126 and 192-254 are for Private Use *[RFC5226]*.

15. Acknowledgments

Thanks to Xiaohong Deng, Alain Durand, Christian Jacquenet, Jacni Qin, Simon Perreault, Paul Selkirk, and James Yu for their comments and review. Thanks to Simon Perreault for highlighting the interaction of dynamic connections with PCP-created mappings.
Thanks to Francis Dupont for his several thorough reviews of the specification, which improved the protocol significantly.

16. References

16.1. Normative References

[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
[RFC5226]	Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
[RFC0768]	Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.
[RFC2827]	Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, May 2000.
[RFC4193]	Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.
[RFC6056]	Larsen, M. and F. Gont, "Recommendations for Transport-Protocol Port Randomization", BCP 156, RFC 6056, January 2011.
[proto_numbers]	IANA, , "Protocol Numbers", 2010.

16.2. Informative References

[RFC0793]	Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.
[I-D.ietf-softwire-dual-stack-lite]	Durand, A, Droms, R, Woodyatt, J and Y Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", Internet-Draft draft-ietf-softwire-dual-stack-lite-11, May 2011.
[RFC6145]	Li, X., Bao, C. and F. Baker, "IP/ICMP Translation Algorithm", RFC 6145, April 2011.
[RFC6146]	Bagnulo, M., Matthews, P. and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, April 2011.
[RFC4941]	Narten, T., Draves, R. and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.
[RFC3587]	Hinden, R., Deering, S. and E. Nordmark, "IPv6 Global Unicast Address Format", RFC 3587, August 2003.
[RFC4787]	Audet, F. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, January 2007.
[RFC5382]	

	Guha, S., Biswas, K., Ford, B., Sivakumar, S. and P. Srisuresh, "NAT Behavioral Requirements for TCP", BCP 142, RFC 5382, October 2008.
[RFC1918]	Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G. and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.
[RFC4961]	Wing, D., "Symmetric RTP / RTP Control Protocol (RTCP)", BCP 131, RFC 4961, July 2007.
[RFC3581]	Rosenberg, J. and H. Schulzrinne, "An Extension to the Session Initiation Protocol (SIP) for Symmetric Response Routing", RFC 3581, August 2003.
[RFC3022]	Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.
[I-D.miles-behave-l2nat]	Miles, D and M Townsley, "Layer2-Aware NAT", Internet-Draft draft-miles-behave-l2nat-00, March 2009.
[RFC6092]	Woodyatt, J., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, January 2011.
[I-D.bpw-pcp-upnp-igd-interworking]	Boucadair, M, Penno, R, Wing, D and F Dupont, "Universal Plug and Play (UPnP) Internet Gateway Device (IGD)-Port Control Protocol (PCP) Interworking Function", Internet-Draft draft-bpw-pcp-upnp-igd-interworking-02, February 2011.
[I-D.ietf-behave-lsn-requirements]	Perreault, S, Yamagata, I, Miyakawa, S, Nakagawa, A and H Ashida, "Common requirements for Carrier Grade NAT (CGN)", Internet-Draft draft-ietf-behave-lsn-requirements-04, October 2011.
[I-D.cheshire-nat-pmp]	Cheshire, S, "NAT Port Mapping Protocol (NAT-PMP)", Internet-Draft draft-cheshire-nat-pmp-03, April 2008.
[I-D.arkko-dual-stack-extra-lite]	Arkko, J, Eggert, L and M Townsley, "Scalable Operation of Address Translators with Per-Interface Bindings", Internet-Draft draft-arkko-dual-stack-extra-lite-05, February 2011.
[IGD]	UPnP Gateway Committee, , "WANIPConnection:1", November 2001.

Appendix A. NAT-PMP Transition

The Port Control Protocol (PCP) is a successor to the NAT Port Mapping Protocol (NAT-PMP), and shares similar semantics, concepts, and packet formats. Because of this NAT-PMP and PCP both use the same port, and

use the NAT-PMP and PCP's version negotiation capabilities to determine which version to use. This section describes how an orderly transition may be achieved.

A client supporting both NAT-PMP and PCP SHOULD send its request using the PCP packet format. This will be received by a NAT-PMP server or a PCP server. If received by a NAT-PMP server, the response will be as indicated by [I-D.cheshire-nat-pmp], which will cause the client to downgrade to NAT-PMP and re-send its request in NAT-PMP format. If received by a PCP server, the response will be as described by this document and processing continues as expected.

A PCP server supporting both NAT-PMP and PCP can respond to requests in either format. The first byte of the packet indicates if it is NAT-PMP (first byte zero) or PCP (first byte non-zero).

A PCP-only gateway receiving a NAT-PMP request (identified by the first byte being zero) will interpret the request as a version mismatch. Normal PCP processing will emit a PCP response that is compatible with NAT-PMP, without any special handling by the PCP server.

Appendix B. **Change History**

[Note to RFC Editor: Please remove this section prior to publication.]

Appendix B.1. **Changes from draft-ietf-pcp-base-11 to -12**

> *added implementation note that MAP and implicit dynamic mappings have independent mapping lifetimes.

Appendix B.2. **Changes from draft-ietf-pcp-base-10 to -11**

> *clarified what can cause CANNOT_PROVIDE_EXTERNAL_PORT error to be generated.

Appendix B.3. **Changes from draft-ietf-pcp-base-09 to -10**

> *Added External_AF field to PEER requests. Made PEER's Suggested External IP Address and Assigned External IP Address always be 128 bits long.

Appendix B.4. **Changes from draft-ietf-pcp-base-08 to -09**

> *Clarified in PEER OpCode introduction (Section 9) that they can also create mappings (as well as query and set existing mappings).

> *More clearly explained how PEER can re-create an implicit dynamic mapping, for purposes of rebuilding state to maintain an existing session (e.g., long-lived TCP connection to a server).

*Added Suggested External IP Address to the PEER OpCodes, to allow
 more robust rebuilding of connections. Added related text to the
 PEER server processing section.

*Removed text encouraging PCP server to statefully remember its
 mappings from Section 11.3.1, as it didn't belong there. Text in
 Section 13.4 already encourages persistent storage.

*More clearly discussed how PEER is used to re-establish TCP
 mapping state. Moved it to a new section, as well (it is now
 Section 7.3).

*MAP errors now copy the Requested IP Address (and port) fields to
 Assigned IP Address (and port), to allow PCP client to
 distinguish among many outstanding requests when using
 PREFER_FAILURE.

*Mapping theft can also be mitigated by ensuring hosts can't re-
 use same IP address or port after state loss.

*the UNPROCESSED option is renumbered to 0 (zero), which ensures
 no other option will be given 0 and be unable to be expressed by
 the UNPROCESSED option (due to its 0 padding).

*created new Implementation Considerations section (Section 11)
 which discusses non-normative things that might be useful to
 implementors. Some new text is in here, and the Failure Scenarios
 text (Section 11.3) has been moved to here.

*Tweaked wording of non-EIM NATs in Section 11.1 to clarify the
 problem occurs both inside->outside and outside->inside.

*removed "Interference by Other Applications on Same Host" section
 from security considerations.

*fixed zero/non-zero text in Section 8.6.

*removed duplicate text saying MAP is allowed to delete an
 implicit dynamic mapping. It is still allowed to do that, but it
 didn't need to be said twice in the same paragraph.

*Renamed error from UNAUTH_TARGET_ADDRESS to
 UNAUTH_THIRD_PARTY_INTERNAL_ADDRESS.

*for FILTER option, removed unnecessary detail on how FILTER would
 be bad for PEER, as it is only allowed for MAP anyway.

*In Security Considerations, explain that PEER can create a
 mapping which makes its security considerations the same as MAP.

Appendix B.5. Changes from draft-ietf-pcp-base-07 to -08

*moved all MAP4-, MAP6-, and PEER-specific options into a single
  section.

*discussed NAPT port-overloading and its impact on MAP (new
  section Section 11.1), which allowed removing the
  IMPLICIT_MAPPING_EXISTS error.

*eliminated NONEXIST_PEER error (which was returned if a PEER
  request was received without an implicit dynamic mapping already
  being created), and adjusted PEER so that it creates an implicit
  dynamic mapping.

*Removed Deployment Scenarios section (which detailed NAT64,
  NAT44, Dual-Stack Lite, etc.).

*Added Client's IP Address to PCP common header. This allows
  server to refuse a PCP request if there is a mismatch with the
  source IP address, such as when a non-PCP-aware NAT was on the
  path. This should reduce failure situations where PCP is deployed
  in conjunction with a non-PCP-aware NAT. This addition was
  consensus at IETF80.

*Changed UNSPECIFIED_ERROR to PROCESSING_ERROR. Clarified that
  MALFORMED_REQUEST is for malformed requests (and not related to
  failed attempts to process the request).

*Removed MISORDERED_OPTIONS. Consensus of IETF80.

*SERVER_OVERLOADED is now a common PCP error (instead of specific
  to MAP).

*Tweaked PCP retransmit/retry algorithm again, to allow more
  aggressive PCP discovery if an implementation wants to do that.

*Version negotiation text tweaked to soften NAT-PMP reference, and
  more clearly explain exactly what UNSUPP_VERSION should return.

*PCP now uses NAT-PMP's UDP port, 5351. There are no normative
  changes to NAT-PMP or PCP to allow them both to use the same port
  number.

*New Appendix A to discuss NAT-PMP / PCP interworking.

*improved pseudocode to be non-blocking.

*clarified that PCP cannot delete a static mapping (i.e., a
  mapping created by CLI or other non-PCP means).

*moved theft of mapping discussion from Epoch section to Security Considerations, (Section 13.4).

Appendix B.6. Changes from draft-ietf-pcp-base-06 to -07

*tightened up THIRD_PARTY security discussion. Removed "highest numbered address", and left it as simply "the CPE's IP address".

*removed UNABLE_TO_DELETE_ALL error.

*renumbered Opcodes

*renumbered some error codes

*assigned value to IMPLICIT_MAPPING_EXISTS.

*UNPROCESSED can include arbitrary number of option codes.

*Moved lifetime fields into common request/response headers

*We've noticed we're having to repeatedly explain to people that the "requested port" is merely a hint, and the NAT gateway is free to ignore it. Changed name to "suggested port" to better convey this intention.

*Added NAT-PMP transition section

*Separated Internal Address, External Address, Remote Peer Address definition

*Unified Mapping, Port Mapping, Port Forwarding definition

*adjusted so DHCP configuration is non-normative.

*mentioned PCP refreshes need to be sent over the same interface.

*renamed the REMOTE_PEER_FILTER option to FILTER.

*Clarified FILTER option to allow sending an ICMP error if policy allows.

*for MAP, clarified that if the PCP client changed its IP address and still wants to receive traffic, it needs to send a new MAP request.

*clarified that PEER requests have to be sent from same interface as the connection itself.

*for MAP opcode, text now requires mapping be deleted when lifetime expires (per consensus on 8-Mar interim meeting)

*PEER OpCode: better description of remote peer's IP address,
 specifically that it does not control or establish any filtering,
 and explaining why it is 'from the PCP client's perspective'.

*Removed latent text allowing DMZ for 'all protocols'
 (protocol=0). Which wouldn't have been legal, anyway, as protocol
 0 is assigned by IANA to HOPOPT (thanks to James Yu for catching
 that one).

*clarified that PCP server only listens on its internal interface.

*abandoned 'target' term and reverted to simplier 'internal' term.

Appendix B.7. Changes from draft-ietf-pcp-base-05 to -06

*Dual-Stack Lite: consensus was encapsulation mode. Included a
 suggestion that the B4 will need to proxy PCP-to-PCP and UPnP-to-
 PCP.

*defined THIRD_PARTY option to work with the PEER OpCode, too.
 This meant moving it to its own section, and having both MAP and
 PEER OpCodes reference that common section.

*used "target" instead of "internal", in the hopes that clarifies
 internal address used by PCP itself (for sending its packets)
 versus the address for MAPpings.

*Options are now required to be ordered in requests, and ordering
 has to be validated by the server. Intent is to ease server
 processing of mandatory-to-implement options.

*Swapped Option values for the mandatory- and optional-to-process
 Options, so we can have a simple lowest..highest ordering.

*added MISORDERED_OPTIONS error.

*re-ordered some error messages to cause MALFORMED_REQUEST (which
 is PCP's most general error response) to be error 1, instead of
 buried in the middle of the error numbers.

*clarified that, after successfully using a PCP server, that PCP
 server is declared to be non-responsive after 5 failed
 retransmissions.

*tightened up text (which was inaccurate) about how long general
 PCP processing is to delay when receiving an error and if it
 should honor OpCode-specific error lifetime. Useful for MAP
 errors which have an error lifetime. (This all feels awkward to
 have only some errors with a lifetime.)

*Added better discussion of multiple interfaces, including
 highlighting WiFi+Ethernet. Added discussion of using IPv6
 Privacy Addresses and RFC1918 as source addresses for PCP
 requests. This should finish the section on multi-interface
 issues.

*added some text about why server might send SERVER_OVERLOADED, or
 might simply discard packets.

*Dis-allow internal-port=0, which means we dis-allow using PCP as
 a DMZ-like function. Instead, ports have to be mapped
 individually.

*Text describing server's processing of PEER is tightened up.

*Server's processing of PEER now says it is implementation-
 specific if a PCP server continues to allow the mapping to exist
 after a PEER message. Client's processing of PEER says that if
 client wants mapping to continue to exist, client has to continue
 to send recurring PEER messages.

Appendix B.8. Changes from draft-ietf-pcp-base-04 to -05

*tweaked PCP common header packet layout.

*Re-added port=0 (all ports).

*minimum size is 12 octets (missed that change in -04).

*removed Lifetime from PCP common header.

*for MAP error responses, the lifetime indicates how long the
 server wants the client to avoid retrying the request.

*More clearly indicated which fields are filled by the server on
 success responses and error responses.

*Removed UPnP interworking section from this document. It will
 appear in [I-D.bpw-pcp-upnp-igd-interworking].

Appendix B.9. Changes from draft-ietf-pcp-base-03 to -04

*"Pinhole" and "PIN" changed to "mapping" and "MAP".

*Reduced from four MAP OpCodes to two. This was done by implicitly
 using the address family of the PCP message itself.

*New option THIRD_PARTY, to more carefully split out the case
 where a mapping is created to a different host within the home.

*Integrated a lot of editorial changes from Stuart and Francis.

    *Removed nested NAT text into another document, including the
     IANA-registered IP addresses for the PCP server.

    *Removed suggestion (MAY) that PCP server reserve UDP when it maps
     TCP. Nobody seems to need that.

    *Clearly added NAT and NAPT, such as in residential NATs, as
     within scope for PCP.

    *HONOR_EXTERNAL_PORT renamed to PREFER_FAILURE

    *Added 'Lifetime' field to the common PCP header, which replaces
     the functions of the 'temporary' and 'permanent' error types of
     the previous version.

    *Allow arbitrary Options to be included in PCP response, so that
     PCP server can indicate un-supported PCP Options. Satisfies PCP
     Issue #19

    *Reduced scope to only deal with mapping protocols that have port
     numbers.

    *Reduced scope to not support DMZ-style forwarding.

    *Clarified version negotiation.

[Appendix B.10.](#) **Changes from draft-ietf-pcp-base-02 to -03**

    *Adjusted abstract and introduction to make it clear PCP is
     intended to forward ports and intended to reduce application
     keepalives.

    *First bit in PCP common header is set. This allows DTLS and non-
     DTLS to be multiplexed on same port, should a future update to
     this specification add DTLS support.

    Moved subscriber identity from common PCP section to MAP
     section.

    *made clearer that PCP client can reduce mapping lifetime if it
     wishes.

    *Added discussion of host running a server, client, or symmetric
     client+server.

    *Introduced PEER4 and PEER6 OpCodes.

*Removed REMOTE_PEER Option, as its function has been replaced by
    the new PEER OpCodes.

   *IANA assigned port 44323 to PCP.

   *Removed AMBIGUOUS error code, which is no longer needed.

Changes from draft-ietf-pcp-base-01 to -02

   *more error codes

   *PCP client source port number should be random

   *PCP message minimum 8 octets, maximum 1024 octets.

   *tweaked a lot of text in section 7.4, "Opcode-Specific Server
    Operation".

   *opening a mapping also allows ICMP messages associated with that
    mapping.

   *PREFER_FAILURE value changed to the mandatory-to-process range.

   *added text recommending applications that are crashing obtain
    short lifetimes, to avoid consuming subscriber's port quota.

Changes from draft-ietf-pcp-base-00 to -01

   *Significant document reorganization, primarily to split base PCP
    operation from OpCode operation.

   *packet format changed to move 'protocol' outside of PCP common
    header and into the MAP* opcodes

   *Renamed Informational Elements (IE) to Options.

   *Added REMOTE_PEER (for disambiguation with dynamic ports),
    REMOTE_PEER_FILTER (for simple packet filtering), and
    PREFER_FAILURE (to optimize UPnP IGD interworking) options.

   *Is NAT or router behind B4 in scope?

   *PCP option MAY be included in a request, in which case it MUST
    appear in a response. It MUST NOT appear in a response if it was
    not in the request.

   *Result code most significant bit now indicates permanent/
    temporary error

*PCP Options are split into mandatory-to-process ("P" bit), and
    into Specification Required and Private Use.

   *Epoch discussion simplified.

Authors' Addresses

   Dan Wing editor Wing Cisco Systems, Inc. 170 West Tasman Drive San
   Jose, California 95134 USA EMail: dwing@cisco.com

   Stuart Cheshire Cheshire Apple Inc. 1 Infinite Loop Cupertino,
   California 95014 USA Phone: +1 408 974 3207 EMail:
   cheshire@apple.com

   Mohamed Boucadair Boucadair France Telecom Rennes, 35000 France
   EMail: mohamed.boucadair@orange-ftgroup.com

   Reinaldo Penno Penno Juniper Networks 1194 N Mathilda Avenue
   Sunnyvale, California 94089 USA EMail: rpenno@juniper.net

   Paul Selkirk Selkirk Internet Systems Consortium EMail:
   pselkirk@isc.org