Port Control Protocol (PCP)
draft-ietf-pcp-base-03

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.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on July 23, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust’s Legal Provisions Relating to IETF Documents (http://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents

Wing                      Expires July 23, 2011  [Page 1]
carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction ................................... 4
2. Scope ......................................... 4
   2.1. Deployment Scenarios ....................... 4
   2.2. Supported Transport Protocols ............ 5
   2.3. Single-homed Customer Premise Network .... 5
3. Terminology ................................... 5
4. Relationship of PCP Server and its NAT .............. 7
5. Common Request and Response Header Format .......... 8
   5.1. Request Header ........................... 8
   5.2. Response Header ......................... 9
   5.3. Options ................................ 9
   5.4. Result Codes ............................ 11
6. General PCP Operation ................................ 12
   6.1. General PCP Client Operation .............. 12
   6.2. General PCP Server Operation .......... 15
7. Introduction to PIN and PEER OpCodes ............ 15
   7.1. For Operating a Server .................. 16
   7.2. For Reducing NAT Keepalive Messages .... 16
   7.3. For Operating a Symmetric Client/Server .. 17
8. PIN OpCodes .................................... 18
   8.2. OpCode-Specific Result Codes ........... 21
   8.3. OpCode-Specific Client Operation, Processing a Response . 22
   8.4. OpCode-Specific Server Operation .... 23
   8.4.1. Maintaining Same External IP Address .. 24
   8.4.2. Pinhole Lifetime .................. 24
   8.4.3. Pinhole Deletion .................. 25
   8.4.4. Subscriber Renumbering ............... 25
   8.5. PCP Options for PIN OpCodes ............ 26
   8.5.1. REMOTE_PEER_FILTER .................. 26
   8.5.2. HONOR_EXTERNAL_PORT .............. 27
8.6. PCP Mapping State Maintenance .................. 27
   8.6.1. Recreating Pinholes ............... 27
   8.6.2. Maintaining Pinholes ............... 29
8.7. Nested NATs .................................. 29
8.8. Pinhole Deletion .............................. 29
8.9. Pinhole Lifetime Extension ................. 30
9. PEER OpCodes .................................. 30
  9.2. OpCode-Specific Result Codes ............... 33
  9.3. OpCode-Specific Client Operation, Processing a Response 33
10. Deployment Scenarios .......................... 34
  10.1. Dual Stack-Lite ............................ 34
    10.1.1. Overview ................................ 34
    10.1.2. Encapsulation Mode ....................... 35
    10.1.3. Plain IPv6 Mode .......................... 35
  10.2. NAT64 ...................................... 35
  10.3. NAT44 and NAT444 ........................... 36
  10.4. IPv6 Firewall .............................. 36
  10.5. Subscriber Identification .................. 36
11. Interworking with UPnP IGD 1.0 and 2.0 .... 37
  11.1. UPnP IGD 1.0 with AddPortMapping Action .. 38
  11.2. UPnP IGD 2.0 with AddAnyPortMapping Action 39
  11.3. Lifetime Maintenance ....................... 40
12. Security Considerations ...................... 40
13. IANA Considerations ........................... 41
  13.1. PCP Server IP address ....................... 41
  13.2. Port Number ................................ 41
  13.3. OpCodes ................................... 41
  13.4. Result Codes ................................ 41
  13.5. Options ................................... 42
14. Authors ........................................ 42
15. Acknowledgments ............................... 42
16. References ..................................... 43
  16.1. Normative References ....................... 43
  16.2. Informative References ..................... 43
Appendix A. Changes ............................... 45
  A.1. Changes from draft-ietf-pcp-base-02 to -03 .... 45
  A.2. Changes from draft-ietf-pcp-base-01 to -02 .... 45
  A.3. Changes from draft-ietf-pcp-base-00 to -01 .... 46
Appendix B. Analysis of Techniques to Discover PCP Server .... 46
Author’s Address .................................... 48
1. Introduction

Port Control Protocol (PCP) provides a mechanism to control how incoming packets are forwarded by upstream devices such as NAT64, NAT44, and firewall devices. PCP is primarily designed to be implemented in the context of both large scale NAT and small-scale NAT (e.g., residential NATs). PCP allows hosts to operate servers permanently (e.g., a webcam) or temporarily (e.g., while playing a game or on a phone call) when behind one or more NAT devices, including when behind a large-scale NAT operated by their Internet service provider.

PCP allows applications to create pinholes from an external IP address to an internal IP address and port. If the PCP-controlled device is a NAT, a mapping is created; if the PCP-controlled device is a firewall, a pinhole is created in the firewall. These pinholes are required for successful inbound communications destined to machines located behind a NAT.

After creating a pinhole 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.

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 (or control) the NAT mapping lifetime. This helps reduce bandwidth on the subscriber’s access network, traffic to the server, and battery consumption on mobile devices.

2. Scope

2.1. Deployment Scenarios

PCP can be used in various deployment scenarios, including:

- DS-Lite [I-D.ietf-softwire-dual-stack-lite], including a NAT and including a router behind the B4 element, and;
[Ed. Note: Do we want ‘NAT behind the B4 element’ in our scope? This affects PCP server discovery, among other things. However, users deploy these NATs. Do we want PCP to work with this, detect this, or quietly fail?? This is tracked as PCP Issues #1 and #25 [PCP-Issues].]

- NAT64, both Stateful [I-D.ietf-behave-v6v4-xlate-stateful] and Stateless [I-D.ietf-behave-v6v4-xlate], and;
- Large-Scale NAT44 [I-D.ietf-behave-1sn-requirements], including nested NATs ("NAT444"), and;
- Layer-2 aware NAT [I-D.miles-behave-l2nat] and Dual-Stack Extra Lite [I-D.arkko-dual-stack-extra-lite], and;
- IPv6 firewall control [I-D.ietf-v6ops-cpe-simple-security].

2.2. Supported Transport Protocols

The PCP OpCodes defined in this document are designed to support transport protocols that use a 16-bit port number (e.g., TCP, UDP, SCTP, DCCP). Transport protocols that do not use a port number (e.g., IPsec ESP) can be wildcarded (allowing any traffic with that protocol to pass), provided of course the upstream device being controlled by PCP supports that functionality, or new PCP OpCodes can be defined to support those protocols.

2.3. Single-homed Customer Premise 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 pinhole 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 the proper address rewriting takes place on that outbound response packet. 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.

3. Terminology

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 RFC 2119 [RFC2119].
Port Forwarding:

Port forwarding allows a host to receive traffic sent to a specific IP address and port.

In the context of a NAT or NAPT, the internal address and external address are different. In the context of a firewall, the internal address and external address are different. In both contexts, if an internal host is listening to connections on a specific port (that is, operating as a server), the external IP address and port number need to be port forwarded to the internal IP address and port number. In the context of a NAPT, it is possible that both the IP address and port are modified. For example with a NAPT, a webcam might be listening on port 80 on its internal address 192.168.1.1, while its publicly-accessible external address is 192.0.2.1 and port is 12345. The NAT does ‘port forwarding’ of one to the other.

In the context of a firewall, the internal address, external IP address, and ports are not changed.

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 of a given subscriber. A PCP Client can issue PCP request on behalf of a third party device of the same subscriber. An interworking function, from UPnP IGD to PCP, or from PCP to PCP (for nested NATs) is another example of a PCP client.

PCP Server:

A network element which receives and processes PCP requests from a PCP Client. See also Section 4.

Mapping:

See also Port Forwarding. A mapping exists only in the context of a Network Address Translator. A NAT mapping creates a relationship between an internal IP transport address and an external IP transport address. More specifically, it creates a translation rule where packets destined to the external IP and port are translated to the internal IP and port.
Mapping Types:

There are three different ways to create mappings: dynamic (e.g., outgoing TCP SYN), PCP, and static configured (e.g., CLI or web page). These mappings are one and the same but their attributes such as lifetime or filtering might be different.

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.

host: a device which can have packets sent to it, as a result of PCP operations. A host is not necessarily a PCP client.

4. Relationship of PCP Server and its NAT

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>-- +--<Internet>
 +-----------------+
 | with embedded   |
 | PCP server     |
 +-----------------+
```

Figure 1: device with Embedded PCP Server

However, it is useful to also model a separation of the PCP server from the NAT, as shown below (Figure 2). The PCP server would communicate with the NAT using a protocol beyond the scope of this document, such as a proprietary protocol or any suitable standard protocol for NAT control.

```
+-----------------+
 | NAT or firewall |
 \                     ^
  \                   /        +--<Internet>
   \                / +-----------------+
    \              / | any suitable control protocol
     \            / +-----------------+
      \          / | PCP Server |
       \        / +-----------------+
        \      /             \  
         \    /               \ 
          \  /                 \ 
           \ /                  \ 
            \                    
```

Figure 2: NAT with Separate PCP Server
5. Common Request and Response Header Format

All PCP messages contain a request (or response) header, opcode-specific information, and 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1|Ver=0|reserve|R|   OpCode    |                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
|                   (optional) opcode-specific information       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
|                   (optional) PCP Options                       |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Common Request Packet Format

These fields are described below:

1. A single bit set to 1.

Ver: Version is 0

reserve: 4 reserved bits, MUST be sent as 0 and MUST be ignored when received.

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

OpCode: Opcodes are defined in Section 8 and Section 9. New OpCodes can be defined per rules described in Section 13.
Reserved: 48 reserved bits, MUST be sent as 0 and MUST be ignored when received.

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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|1|Ver=1|reserve|R|   Opcode    |  Reserved     |  Result Code  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Epoch                                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
:       (optional) OpCode-specific response data             :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:       (optional) Options                                   :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Common Response Packet Format

These fields are described below:

Ver: Version is 1

reserve: 4 reserved bits, MUST be sent as 0, MUST be ignored when received.

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

OpCode: The OpCode value from the request.

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

Result Code: The result code for this response. See Section 5.4 for values.

Epoch: The server’s Epoch value. The same value is used for all PCP clients. See Section 6.1.4 for discussion.

5.3. Options

A PCP OpCode can be extended with an Option. Options can be used in requests and responses. It is anticipated that Options will include information which are associated with the normal function of an
OpCode. For example, an Option could indicate DSCP markings to apply to incoming or outgoing traffic associated with a PCP pinhole, or an Open could include descriptive text (e.g., "for my webcam").

Options use the following Type-Length-Value format:

```
+-----------------+------------------+-
|  Option Code    |  Reserved       |
+-----------------+------------------+-
|                 |  Option-Length   |
+-----------------+------------------+-
| (optional) data |
```

Figure 5: Options Header

The description of the fields is as follows:

Option Code: Option code, 8 bits. The first bit of the option code is the "P" bit, described below. Option codes MUST be registered with IANA following the procedure described in Section 13.

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

Option-Length: Indicates in units of 4 octets the length of the enclosed data. 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. An Option MUST NOT appear in a PCP response unless solicited in the associated request. If a given Option was included in a request, and understood and processed by the PCP server, it MUST be included in the response. The handling of an Option by the PCP Client and Server MUST be specified in a appropriate document and must include whether the PCP Option can appear (one or more times) in a request, and indicate the contents of the Option in the response. If several Options are included in a PCP request or response, they can be encoded in any order by the PCP client. The response MUST encode them in the same order, but may omit some PCP Options in the response (e.g., omitting them is necessary to indicate the PCP server does not understand that Option, or simply because that Option is not included in responses). A single Option MAY appear more than once, if permitted by the definition of the Option itself. If the Option’s definition allows the Option to appear once but it appears more than once, the PCP server MUST respond with the MALFORMED_OPTION response code.
If the "P" bit in the OpCode is set,

- 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, it MUST generate a failure response without including this Option in the response.
- If the PCP server does implement this Option, but cannot perform the function indicated by this Option, it MUST generate a failure response and include this Option (and its arguments) in the response.

If the "P" bit is clear, the PCP server MAY process or ignore this Option.

To enhance interoperability, newly defined Options should avoid interdependencies with each other.

New Options MUST include the information below:

This Option:
- name: <mnemonic>
- number: <value>
- purpose:
  - is valid for OpCodes: <list of OpCodes>
  - has length: <rules for length>
  - appear more than once: <yes/no>

5.4. Result Codes

The following result codes may be returned as a result of any OpCode received by the PCP server. Result codes are 8 bits long and the most significant bit indicates if the error is transient or permanent. A transient error is one that might resolve itself, such that an identical PCP request submitted later would be successful (e.g., a resource is consumed and might become available later). A permanent error is one that cannot reasonably be successful if an identical PCP request is submitted (e.g., re-sending a PCP request containing an OpCode the server does not support). Future result codes should follow this same format.
0 - SUCCESS, success
128 - UNSUPP_VERSION, unsupported version
129 - UNSUPP_OPCODE, unsupported OpCode
130 - UNSUPP_OPTION, unsupported Option
131 - MALFORMED_OPTION, malformed Option (e.g., exists too many
times, invalid length)
132 - UNSPECIFIED_ERROR, server encountered unspecified error
133 - MALFORMED_REQUEST

Figure 6: PCP Result Codes

Additional result codes, specific to the OpCodes defined in this
document, are listed in Section 8.2 and Section 9.2.

6. General PCP Operation

PCP messages MUST be sent over UDP, and the PCP Server MUST listen
for PCP requests on the PCP port number (Section 13.2). Every PCP
request generates a response, so PCP does not need to run over a
reliable transport protocol.

6.1. General PCP Client Operation

This section details operation specific to a PCP client, for any
OpCode. Procedures specific to the PIN OpCodes are described in
Section 8, and procedures specific to the PEER OpCodes are described
in Section 9.

6.1.1. Generating and Sending a Request

Prior to sending its first PCP message, the PCP client determines
which servers to use. The PCP client tries the following to get a
list of PCP servers:

1. if a PCP server is configured (e.g., in a configuration file),
   the address(es) of the PCP server(s) are used as the list of PCP
   server(s), else;

2. if DHCP indicates the PCP server(s), the address(es) of the
   indicated PCP server(s) are used as the list of PCP server(s),
   else;

3. if the PCP working group decides an IANA-allocated address for
   the PCP server is suitable, it is added to the list of PCP
   servers, else;
4. the address of the default router, it is added to the list of PCP servers.

[Ed. Note: more discussion around these methods is necessary to reach consensus on the best approach(es) for PCP. Also see Appendix B for further analysis. This is tracked as PCP Issue #1 [PCP-Issues].]

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. It initializes a retransmission timer to 4 seconds. When several PCP Clients are embedded in the same host, each uses a distinct port number to disambiguate their requests and replies. The PCP client sends a PCP message to each server in sequence, waiting for a response until its timer expires. Once a PCP client has successfully communicated with a PCP server, it continues communicating with that PCP server until that PCP server becomes non-responsive, which causes the PCP client to attempt to re-iterate the procedure starting with the first PCP server on its list. If a hard ICMP error is received the PCP client SHOULD immediately abort trying to contact that PCP server (see Section 2 of [RFC5461] for discussion of ICMP and ICMPv6 hard errors). If no response is received from any of those servers, it doubles its retransmission timer and tries each server again. This is repeated 4 times (for a total of 5 transmissions to each server). If, after these transmissions, no PCP server has responded the PCP client SHOULD abort the procedure.

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.

6.1.2. 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 request. It validates the version number and OpCode matches an outstanding request. Responses shorter than 8 octets, longer than 1024 octets, or not a multiple of 4 octets are invalid and ignored. The response is further matched by comparing fields in the response OpCode-specific data to fields in the request OpCode-specific data. After a successful match with an outstanding request, the PCP client checks the Epoch field to determine if it needs to restore its state to the PCP server (see Section 6.1.4).

If it is an error response (>0), the request failed. If the error response is less than 128, retrying the same request sometime in the future might be successful. If the error response is greater or
equal to 128, retrying the same request is unlikely to be successful.

If it is the success response (0), the PCP client knows the request was successful.

6.1.3. 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 dual-stack or be configured with several IP addresses. These IP addresses may have distinct reachability scopes (e.g., if IPv6 they might have global reachability scope as for GUA (Global Unicast Address) or limited scope such as ULA (Unique Local Address, [RFC4193])).

IPv6 addresses with global reachability scope SHOULD be used as internal IP address when instructing a PCP mapping in a PCP-controlled device. IPv6 addresses with limited scope (e.g., ULA), SHOULD NOT be indicated as internal IP address in a PCP message.

As mentioned in Section 2.3, only mono-homed CP routers are in scope. Therefore, there is no viable scenario where a host located behind a CP router is assigned with two GUA addresses belonging to the same global IPv6 prefix.

[Ed. Note: text regarding multi-homing needs work.]

6.1.4. Epoch

Every PCP response sent by the PCP Server includes an Epoch field. This field increments by 1 every second, and indicates to the PCP client if PCP state needs to be restored. If the PCP Server resets or loses the state of its port mapping table, due to reboot, power failure, or any other reason, it MUST reset its Epoch time to 0. Similarly, if the public IP address of the NAT (controlled by the PCP server) changes, the Epoch MUST be reset to 0.

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 8.6.1.

[Ed. Note: comment from Dave Thaler: "So spoofed packets with Epoch=0 that look like they come from the server could result in a
big amplification attack on the PCP server. How is this mitigated?”. This is tracked as PCP Issue #21, [PCP-Issues].

6.2. General PCP Server Operation

This section details operation specific to a PCP server.

Upon receiving a PCP request 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, has the R bit set, or the first bit is clear, the request is simply dropped. If the version number is not supported, a response is generated containing the version number of the request with the UNSUPP_VERSION response code. If the OpCode is not supported, a response is generated with the UNSUPP_OPCODE response code. If the length of the request exceeds 1024 octets or is not a multiple of 4 octets, a response is generated with the MALFORMED_REQUEST response code, the response 0-filled to be a multiple of 4 octets and not to exceed 1024 octets.

7. Introduction to PIN and PEER OpCodes

There are three uses for the PIN 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), 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 7.3) the PCP client knows if it wants an IPv4 listener (PINx4), IPv6 listener (PINx6), or both (PINx4 and PINx6) on the Internet. The PCP client also knows if it has an IPv4 interface on itself (PIN4y) or an IPv6 interface on itself (PIN4y). It takes the union of this knowledge to decide to send a request for PIN44, PIN64, PIN46, or PIN66. Applications that embed IP addresses in payloads (e.g., FTP, SIP) will find it beneficial to avoid address family translation (PIN46, PIN64), if possible.
7.1. For Operating a Server

A host operating a server (e.g., a web server) listens for traffic on a port (but never sends traffic from that port. For this to work across a NAT or a firewall, the application needs to (a) create a pinhole 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 application follows the procedures described in this section.

As normal, the application needs to begin listening to a port, and to ensure that it can get exclusive use of that port it needs to choose a port that is not in the operating system’s ephemeral port range. Then, the application constructs a PCP message with the appropriate PIN OpCode depending on if it is listening on an IPv4 or IPv6 interface 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:

```c
/* start listening on the local server port */
int s = socket(...);
bind(s, &sockaddr, ...);
listen(s, ...);
getsockname(s, &internal_address, ...);
external_address = NUL;
pcp_send_pin_request(internal_address, &external_address, lifetime);
update_rendezvous_server("Client 12345", external_address);
while (1) {
   accept(s, ...);
   /* ... */
}
```

Figure 7: Pseudo-code for using PCP to operate a server

7.2. For Reducing NAT Keepalive Messages

[Ed. Note: This section creates a difference between a dynamically-created mapping, which PCP tries to query/control using the PEER OpCode, and a PCP-created mapping which was created with a PIN OpCode. This section attempts to address PCP Issue #9 and PCP Issue #35.]

A host operating a client (e.g., XMMP 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 applications 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 avoid 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 applications 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:

```cpp
int s = socket(...);
connect(s, &remote_peer, ...);
getsockname(s, &internal_address, ...);
external_address = NUL;
pcp_send_peer_request(internal_address, &external_address,
remote_peer, lifetime);
```

Figure 8: Pseudo-code using PCP with a dynamic socket

7.3. For Operating a Symmetric Client/Server

[Ed. Note: The PEER4 and PEER6 OpCodes, discussed here, are intended to resolve PCP Issue #35.]

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 (usually) initiates outbound connections from that same source address. For 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 PIN OpCode, as described in Section 8, and receive a positive PCP response before it sends any packets from that port. 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 mappings (EIM) behavior, reversing the steps will fail if the NAT does not have EIM behavior. With a non-EIM NAT, the outgoing dynamic connection and the PIN 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 also have different lifetimes. As a reminder, 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 outbound connection will be EIM or non-EIM.

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

```c
/* start listening on the server port */
int s = socket(...);
bind(s, &sockaddr, ...);
listen(s, ...);
getsockname(s, &internal_address, ...);
external_address = NULL;
pcp_send_pin_request(internal_address, &external_address, lifetime);
update_rendezvous_server("Client 12345", external_address);
/* ... */
send_packet(s, "Hello World");
```

Figure 9: Pseudo-code for using PCP to operate a symmetric client/server

8. PIN OpCodes

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

PIN44=0: create a forwarding pinhole between an internal IPv4 address and public IPv4 address (e.g., NAT44 or firewall)
PIN46=1: create a forwarding pinhole between an internal IPv4 address and public IPv6 address (e.g., NAT46 or firewall)

PIN64=2: create a forwarding pinhole between an internal IPv6 address and public IPv4 address (e.g., NAT64 or firewall)

PIN66=3: create a forwarding pinhole between an internal IPv6 address and public IPv6 address (e.g., NPTv6 or firewall)

The operation of these OpCodes is described in this section.

8.1. OpCode Packet Formats

The four PIN OpCodes (PIN44, PIN46, PIN64, PIN66) share a similar packet layout for both requests and responses. Because of this similarity, they are shown together. For all of the PIN OpCodes, if the internal IP address and internal port fields of the request both match the response’s external IP address and external port fields, the IP addresses and ports are not changed and thus the functionality is purely a firewall; otherwise it pertains to a network address translator which might also perform firewall functions.

The following diagram shows the request packet format for PIN44, PIN46, PIN64, and PIN66. 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     | Reserved (24 bits) |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
: Pinhole Internal IP address (32 or 128, depending on OpCode) :
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
:                                                               :
: Requested external IP address (32 or 128, depending on OpCode):
:                                                               :
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Requested lifetime |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| internal port      | requested external port |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 10: PIN OpCode Request Packet Format

These fields are described below:
Protocol: indicates 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.

Reserved: 24 reserved bits, MUST be 0 on transmission and MUST be ignored on reception.

Pinhole Internal IP Address: Internal IP address of the pinhole. This can be IPv4 or IPv6, depending on the OpCode.

Requested External IP Address: Requested 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 doesn’t know the external address, or doesn’t have a preference, it MUST use 0.

Requested lifetime: Requested lifetime of this pinhole, in seconds.

internal port: Internal port for the pinhole.

Requested external port: requested 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 doesn’t know the external port, or doesn’t have a preference, it MUST use 0.

The following diagram shows the response packet format for PIN44, PIN46, PIN64, and PIN66:
These fields are described below:

Protocol: Copied from the request.

Reserved: 24 reserved bits, MUST be 0 on transmission, MUST be ignored on reception.

Pinhole Internal IP address: Copied from the request. IPv4 or IPv6 address is indicated by the OpCode.

Assigned external IP address: Assigned external IPv4 or IPv6 address for the pinhole. IPv4 or IPv6 address is indicated by the OpCode.

Assigned Lifetime: Lifetime for this mapping, in seconds

Internal port: Internal port for the pinhole, copied from request.

Assigned external port: requested external port for the mapping. IPv4 or IPv6 address is indicated by the OpCode

8.2. OpCode-Specific Result Codes

In addition to the result codes described in Figure 6, the following additional result codes may be returned as a result of the four PIN OpCodes received by the PCP server.
3 - NETWORK_FAILURE, e.g., NAT device has not obtained a DHCP lease
4 - 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.
150 - UNSUPP_PROTOCOL, unsupported Protocol
151 - NOTAUTHORIZED, e.g., PCP server supports mapping, but the feature is disabled for this PCP client, or the PCP client requested a pinhole that cannot be fulfilled by the PCP server’s security policy.
152 - USER_EX_QUOTA, mapping would exceed user’s port quota
153 - CANNOT_HONOR_EXTERNAL_PORT, indicates the port is already in use or otherwise unavailable (e.g., special port that cannot be allocated by the server’s policy). This error is only returned if the request included the Option HONOR_EXTERNAL_PORT.
154 - UNABLE_TO_DELETE_ALL, indicates the PCP server was not able to delete all pinholes.
155 - CANNOT_FORWARD_PORT_ZERO, indicates the requested external port 0 cannot be fulfilled, because the PCP-controlled device is a firewall.

Figure 12: Result Codes specific to PIN OpCodes

Other OpCodes are in result codes are defined following the procedure in Section 13.4.

8.3. OpCode-Specific Client Operation, Processing a Response

This section describes the operation of a client when sending the OpCodes PIN44, PIN64, PIN46, or PIN66.

A response is matched with a request by comparing the protocol, 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 pinhole created by the OpCode.

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.

If an unsuccessful result code greater than or equal to 128, the PCP client SHOULD NOT re-issue the same request. If a result code less than or equal to 127, the PCP client MAY, if it wishes, resend the same message after a delay of at least 5 seconds.
8.4. OpCode-Specific Server Operation

This section describes the operation of a client when processing the OpCode PIN44, PIN64, PIN46, or PIN66.

If the requested lifetime is 0, it indicates a request to delete the pinhole immediately. This means the pinhole described by the internal IP address should be deleted, and requested external port field is ignored by the server (that is, it isn’t validated). If the deletion request was properly formatted, and the associated pinhole (if present) is deleted, a positive response is generated containing external port of 0 and lifetime of 0. If the client attempts to delete a port mapping which was manually assigned by some kind of configuration tool, the PCP server MUST respond with NOTAUTHORIZED result code.

[Ed. Note: Should we similarly return an error if attempting to delete mappings that were created dynamically (e.g., TCP SYN)? This is tracked as PCP Issues #9 [PCP-Issues].]

A PCP client can request the explicit deletion of all UDP or TCP mappings by sending a PCP request with a PIN OpCode with external port, internal port, and lifetime set to 0. A PCP client MAY choose to do this when it first acquires a new IP address in order to protect itself from port mappings that were performed by a previous owner of the IP address. After receiving such a deletion request, the PCP server and its PCP-controlled device MUST delete all the port mappings. The PCP server responds to such a deletion request with a response as described above, with the internal port set to zero. If the PCP server is unable to delete a port mapping, for example, because the mapping was manually configured by a configuration tool, the PCP server MUST still delete as many port mappings as possible, and respond with the result code UNABLE_TO_DELETE_ALL.

See Section 8.4.2 for processing the lifetime.

If the requested external port is 0, and the PCP-controlled device does not change port numbers (that is, it does not do port translation), it cannot fulfill the request. In that case, the PCP server MUST return the response code CANNOT_FORWARD_PORT_ZERO.

If the PCP-controlled device is stateless (that is, does not establish any per-flow state), the PCP server simply returns an answer, without creating any "pinholes".

If an option with value greater than 128 exists but that option does not make sense (e.g., the HONOR_EXTERNAL_PORT option is included in a request with lifetime=0 (indicating a delete request)), the request
is invalid and generates a MALFORMED_OPTION error.

By default, a PCP-controlled device MUST NOT create pinholes 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. Nevertheless, a configurable feature MAY be supported by the PCP-controlled device, which MAY reserve (but not forward) the companion port so the same PCP Client can request it in the future.

The PCP server then validates that the internal IP address in the PIN OpCode belongs to the same subscriber. This validation depends on the PCP deployment scenario; see Section 10.5 for the validation procedure. If the internal IP address in the PCP request does not belong to the subscriber, an error response MUST be generated with result code NOT_AUTHORIZED.

If all of the proceeding operations were successful (did not generate an error response), then the requested pinholes are created as described in the request and a positive response is built. This positive result contains the same OpCode as the request, but with the "R" bit set.

As a side-effect of creating a pinhole, ICMP messages associated with the pinhole are also translated (if appropriate) and forwarded for the duration of the pinhole’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 pinholes for those flows.

8.4.1. Maintaining Same External IP Address

If there are active mappings associated with a given subscriber (see Section 10.5) -- created via dynamic assignment, by PCP or any other means -- subsequent PCP mapping requests belonging to the same subscriber MUST use the same external IP address. This follows the intent of REQ-1 of [I-D.ietf-behave-lsn-requirements].

Once an internal host has no active mapping in the PCP-controlled device, a subsequent PCP request for that host MAY be assigned to a different external IP address.

8.4.2. Pinhole Lifetime

The PCP Client requests a certain lifetime, and the PCP Server responds with the assigned lifetime. The PCP Server can minimize and maximize 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 NOT RECOMMENDED that the server allow long lifetimes (exceeding 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.

Once a PCP server has responded positively to a pinhole 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).

An application that forgets its PCP-assigned mappings (e.g., the application or OS crashed) will request new PCP mappings (consuming the user’s port quota (if there is a quota) and the resource limit for number of pinholes), and the application will also probably initiate dynamic connections to servers without using PCP (also consuming the user’s port quota). PCP provides no explicit protection against such port consumption. In such environments, it is RECOMMENDED that applications use shorter PCP lifetimes to reduce the impact of consuming the user’s port quota.

8.4.3. Pinhole Deletion

A pinhole MUST be deleted by the PCP Server upon the expiry of its lifetime, or upon request from the PCP client.

In order to prevent another subscriber from receiving unwanted traffic, the PCP server SHOULD NOT assign that same external port to another host for 120 seconds (MSL, [RFC0793]).

[Ed. Note: it should (MUST?) allow the same host to re-acquire the same port, though.]

8.4.4. Subscriber Renumbering

The customer premise router might obtain a new IPv4 address or new IPv6 prefix. 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 (old) address -- both traffic mapped with PIN requests and dynamic traffic. This same problem can occur if an IP address is re-assigned today, without PCP. The solution is the same as today: ISPs should not re-assign IP addresses to different subscribers.

When a new IP 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. Assuming the PCP client wants to
continue receiving traffic, it needs to install new mappings. The requested external port field will not be honored by the PCP server, in all likelihood, because it is still being forwarded to the old IP address. Thus, a pinholes is likely to be assigned a new external port number and/or public IP address.

8.5. PCP Options for PIN OpCodes

8.5.1. REMOTE_PEER_FILTER

This Option indicates packet filtering 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. That is, packets with a source IP address, transport, or port that do not match those fields of the PCP request are dropped by the PCP server-controlled NAT/firewall device.

<table>
<thead>
<tr>
<th>Reserved</th>
<th>Remote Peer Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remote Peer IP address (32 bits if PIN44 or PIN64, 1 28 bits if PIN46 or PIN66)</td>
<td></td>
</tr>
</tbody>
</table>

This Option:

value: 128

is valid for OpCodes: PIN44, PIN64, PIN46, or PIN66

is included in responses: MUST

has length: 2 or 5

may appear more than once: no

Because of interactions with dynamic ports this Option MUST only be used by a client that is operating a server, as this ensures that no other application will be assigned the same ephemeral port for its outgoing connection. Other use by a PCP client is NOT RECOMMENDED and will cause some UNSAF NAT traversal mechanisms [RFC3424] to fail where they would have otherwise succeeded, breaking other applications running on this same host.
[Ed. Note: How do we want to remove a filter? Do we want to allow removing a filter at all -- is there a use-case for that or can the application just create a new mapping? If we have a use-case, perhaps use 0.0.0.0 as the remote IP address to remove all filters? This is tracked as PCP Issue #10 [PCP-Issues].]

8.5.2. HONOR_EXTERNAL_PORT

This option indicates the PCP client needs to have the indicated requested external port allocated. If the PCP server cannot provide that port for the exclusive use of the PCP client, the PCP server MUST return result code CANNOT_HONOR_EXTERNAL_PORT.

This option is intended for use by UPnP IGD interworking (Section 11).

This Option:

value: 130

is valid for OpCodes: PIN44, PIN64, PIN46, or PIN66

is included in responses: MUST

has length: 0

may appear more than once: no

8.6. PCP Mapping State Maintenance

If an event occurs that causes the PCP server to lose state (such as a crash or power outage), the pinholes 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 such loss of state is more common in a residential NAT device which does not write information to its non-volatile memory.

The Epoch indicates if the PCP server has lost its state. If this occurs, the PCP client can attempt to recreate the pinholes following the procedures described in this section.

8.6.1. Recreating Pinholes

The PCP server SHOULD store mappings in persistent storage so when it is powered off or rebooted, it remembers the port mapping state of the network. Due to the physical architecture of some PCP servers, this is not always achievable (e.g., some non-volatile memory can withstand only a certain number of writes, so writing PCP mappings to
such memory is generally avoided).

However, maintaining this state is not essential for correct operation. When the PCP server loses state and begins processing new PCP messages, its Epoch is reset to zero (per the procedure of Section 6.1.4).

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.

This self-healing property of the protocol is very important.

When a client renews its port mappings as the result of receiving a packet where the Epoch field indicates that a reboot or similar loss of state has occurred, the client MUST first delay by a random amount of time selected with uniform random distribution in the range 0 to 5 seconds, and then send its first PCP request. After that request is acknowledged by the PCP server, the client may then send its second request, and so on, as rapidly as the gateway allows. The requests SHOULD be issued serially, one at a time; the client SHOULD NOT issue multiple requests simultaneously in parallel.

[Ed. Note: the paragraph above is copied from NAT-PMP, and seems to be advice specific to receiving asynchronous notification that the Epoch was reset. Asynchronous notification needs the delay described (in fact, it probably needs greater delay than 0-5 seconds if on a larger network) and also needs to discourage sending multiple PCP requests serially. However, PCP does not have asynchronous notification (yet), and has different needs/requirements for pacing. In short: the above paragraph needs some discussion. This is tracked as PCP Issue #11 [PCP-Issues].]

The discussion in this section focuses on recreating inbound port mappings after loss of PCP server state, because that is the more serious problem. Losing port mappings for outgoing connections destroys those currently active connections, but does not prevent clients from establishing new outgoing connections. In contrast, losing inbound port mappings not only destroys all existing inbound connections, but also prevents the reception of any new inbound connections until the port mapping is recreated. Accordingly, we consider recovery of inbound port mappings the more important priority. However, clients that want outgoing connections to survive a NAT gateway reboot can also achieve that using PCP. After initiating an outbound TCP connection (which will cause the NAT gateway to establish an implicit port mapping) the client should send the NAT gateway a port mapping request for the source port of its TCP
connection, which will cause the NAT gateway to send a response giving the external port it allocated for that mapping. The client can then store this information, and use it later to recreate the mapping if it determines that the NAT gateway has lost its mapping state.

8.6.2. Maintaining Pinholes

A PCP client can refresh a pinhole 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 pinholes, the Epoch value only needs to be retrieved for one of them to verify the PCP server has not lost port forwarding state.

If the client wishes to check the PCP server’s Epoch, it sends a PCP request for any one of the client’s pinholes. 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).

8.7. Nested NATs

[Ed. Note: Nested NATs will be discussed in a later version of this document or, more likely, in a separate document. The mechanism to support nested NATs is to have each NAT implement a PCP server (to service devices and NATs behind it) and a PCP client (to communicate with NATs above it). This allows a PCP client to be unaware of the PCP communications going on between upstream NATs.) This is tracked as PCP Issue #25 [PCP-Issues].]

[Ed. Note: In this document, we do need to describe how to detect a non-PCP-aware NAT on the path, especially for PEER4/PEER6, as it indicates the application cannot use the longer lifetime for its keepalive traffic.]

8.8. Pinhole Deletion

A PCP Client can delete a pinhole immediately by sending the appropriate PIN OpCode with a lifetime of 0. The PCP server responds by returning a PIN Response with a lifetime of 0.

To delete all pinholes for all hosts associated with this subscriber,
an all-zero internal IP address is used.

8.9. Pinhole Lifetime Extension

An existing mapping can have its lifetime extended by the PCP client. To do this, the PCP client sends a new PCP map request to the server indicating the internal IP address and port(s).

The PCP Client SHOULD renew the mapping before its expiry time, otherwise it will be removed by the PCP Server (see Section 8.4.3). In order to prevent excessive PCP chatter, it is RECOMMENDED to renew only 60 seconds before expiration time (to account for retransmissions that might be necessary due to packet loss, clock synchronization between PCP client and PCP server, and so on).

9. PEER OpCodes

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

PEER4=4: Set or query lifetime for flow from IPv4 address to a remote peer’s IPv4 address.

PEER6=5: Set or query lifetime for flow from IPv6 address to a remote peer’s IPv6 address.

The operation of these OpCodes is described in this section.


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. For both of the PEER OpCodes, if the internal IP address and internal port fields of the request both match the external IP address and external port fields of the response, the IP addresses and ports are not changed and thus the functionality is purely a firewall; otherwise it pertains to a network address translator which might also perform firewall functions.

The following diagram shows the request packet format for PEER4 and PEER6. This packet format is aligned with the response packet format:
<table>
<thead>
<tr>
<th>Protocol</th>
<th>External_AF</th>
<th>Reserved (16 bits)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 13: PEER OpCode Request Packet Format

These fields are described below:

**Protocol**: indicates 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**: The address family of the external IP address associated with this peer connection.

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

**Pinhole Internal IP Address**: Internal IP address of the 5-tuple. This can be 32 bits long (if OpCode is PEER4) or 128 bits long (if OpCode is PEER6).

**Remote Peer IP Address**: Remote peer’s IP address, from the perspective of the PCP client.
Reserved: 128 reserved bits, MUST be 0 on transmission and MUST be ignored on reception.

Internal port: Internal port for the 5-tuple.

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

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.

---

Figure 14: PEER OpCode Request Packet Format

Protocol: Copied from the request.

External_AF  Copied from the request

Reserved: 16 reserved bits, MUST be 0 on transmission, MUST be ignored on reception.
Internal IP address  Copied from the request.

remote Peer IP address  Copied from the request.

External IP address  External IP address, assigned by the NAT (or firewall) to this pinhole. If firewall, this will match the internal IP address.

Assigned lifetime:  Assigned lifetime, in seconds.

internal port: copied from request.

external port:  External IP address, assigned by the NAT (or firewall) to this pinhole. If firewall or 1:1 NAT, this will match the internal port.

remote peer port:  Copied from request.

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

9.2. OpCode-Specific Result Codes

In addition to the result codes described in Figure 6, the following additional result codes may be returned as a result of the two PEER OpCodes received by the PCP server.

150 - UNSUPP_PROTOCOL, unsupported Protocol

Figure 15: Result Codes specific to PEER OpCodes

Other OpCodes are in result codes are defined following the procedure in Section 13.4.

9.3. OpCode-Specific Client Operation, Processing a Response

This section describes the operation of a client when sending the OpCodes PEER4 or PEER6.

A response is 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 pinhole created by the OpCode.

If a successful response, the PCP client uses the assigned lifetime value to reduce its frequency of application keepalives for the NAT. 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 ensure the server is still accessible (e.g., has not crashed) more frequently than once an hour.

If an unsuccessful result code greater than or equal to 128, the PCP client SHOULD NOT re-issue the same request. If a result code less than or equal to 127, the PCP client MAY, if it wishes, resend the same message after a delay of at least 5 seconds.

9.4. OpCode-Specific Server Operation

This section describes the operation of a client when processing the OpCodeS PEER4 or PEER6.

[Ed. Note: Add text]

The PCP server then validates that the internal IP address in the PIN OpCode belongs to the same subscriber. This validation depends on the PCP deployment scenario; see Section 10.5 for the validation procedure. If the internal IP address in the PCP request does not belong to the subscriber, an error response MUST be generated with result code NOT AUTHORIZED.

If all of the proceeding operations were successful (did not generate an error response), then ...

10. Deployment Scenarios

10.1. Dual Stack-Lite

The interesting components in a Dual-Stack Lite deployment are the B4 element (which is the customer premise router) and the AFTR element (which is the device that both terminates the IPv6-over-IPv4 tunnel and also implements the large-scale NAT44 function). The B4 element does not need to perform a NAT function (and usually does not perform a NAT function), but it does operate its own DHCP server and is the local network’s default router.

10.1.1. Overview

Various PCP deployment scenarios can be considered to control the PCP server embedded in the AFTR element:

1. UPnP IGD and NAT-PMP are used in the LAN: an interworking function is required to be embedded in the B4 element to ensure interworking between the protocol used in the LAN and PCP. UPnP
IGD-PCP Interworking Function is described in Section 11.

2. Hosts behind the B4 element with either include a PCP client or UPnP IGD client.
   
   A. if a UPnP IGD client, the B4 element will need to include an interworking function from UPnP IGD to PCP.
   
   B. if a PCP client, the PCP client will communicate directly with the PCP server.

3. The B4 element include a PCP Client which is invoked by an HTTP-based configuration (as is common today). The internal IP address field in the PCP payload would be the internal host used in the port forwarding configuration.

Two modes are identified to forward PCP packets to a PCP Server controlling the provisioned AFTR as described in the following sub-sections.

[Ed. Note: We need to decide on Encapsulation Mode or Plain IPv6 Mode. This is tracked as PCP Issue #13 [PCP-Issues].]

10.1.2. Encapsulation Mode

In this mode, B4 element does no processing at all of the PCP messages, and forwards them as any other UDP traffic. With DS-Lite, this means that IPv4 PCP messages issued by internal PCP Clients are encapsulated into the IPv6 tunnel sent to the AFTR as for any other IPv4 packets. The AFTR decapsulates the IPv4 packets and processes the PCP requests (because the destination IPv4 address points to the PCP Server embedded in the AFTR).

10.1.3. Plain IPv6 Mode

Another alternative for deployment of PCP in a DS-Lite context is to rely on a PCP Proxy in the B4 element. Protocol exchanges between the PCP Proxy and the PCP Server are conveyed using plain IPv6 (no tunnelling is used). Nevertheless, the IPv6 address used as source address by the PCP Proxy MUST be the same as the one used by the B4 element.

10.2. NAT64

Hosts behind a NAT64 device can make use of PCP in order to perform port reservation (to get a publicly routable IPv4 port).

If the IANA-assigned IP address is used for the discovery of the PCP
Server, that IPv4 address can be placed into the IPv6 destination address following that particular network's well-known prefix or network-specific prefix, per [RFC6052].

10.3. NAT44 and NAT444

Residential subscribers in NAT44 (and NAT444) deployments are usually given one IPv4 address, but may also be given several IPv4 addresses. These addresses are not routable on the IPv4 Internet, but are routable between the subscriber’s home and the ISP’s large scale NAT (LSN). To accommodate multiple hosts within a home, especially when provided insufficient IPv4 addresses for the number of devices in the home, subscribers operate a NAPT device. When this occurs in conjunction with an upstream NAT44, this is nicknamed "NAT444".

[Ed. Note: Does PCP need a mechanism to detect a non-PCP-supporting NAT between a PCP client and a PCP server? Or can that problem be detected by relying on the failure of PCP Server Discovery? This is tracked as PCP Issue #25 [PCP-Issues].]

10.4. IPv6 Firewall

See Section 8.5.1.

[Ed. Note: this IPv6 firewall section needs more text. This is tracked as PCP Issue #10 [PCP-Issues].]

10.5. Subscriber Identification

The PIN OpCodes require subscriber identification because they allocate resources or adjust resources allocated to a subscriber. For the PIN OpCode, it is permitted for a PCP Client to create a mapping on behalf of a third party device (e.g., a computer can create PCP mappings on behalf of a webcam). However, a PCP client cannot open pinholes for a different subscriber. The mechanism to identify "same subscriber" depends on the sort of NAT on this network:

- If the PCP-controlled device is a NAT64: the internal IP address indicated in the PCP message and the source IPv6 address of received PCP request MUST belong to the same IPv6 prefix. The length of the IPv6 prefix is the same as the length assigned to each subscriber on that particular network (e.g., /64), and that length must be configurable by the network operator.

- If the PCP-controlled device is a DS-Lite AFTR: DS-Lite (Section 11 of [I-D.ietf-softwire-dual-stack-lite]) already requires the tunnel transport source address be validated, and that same
address is used by PCP to assign the tunnel-ID to the requested mapping (see Section 10.1.2 and Section 10.1.3). Thus, PCP acquires the same security properties as DS-Lite. If address validation is implemented correctly, the PCP Client can not instruct mappings on behalf of devices of another subscriber.

- If LSN with a routed network (NAT44), each subscriber has a known set of IPv4 address (usually one IPv4 address) and all PCP requests MUST be sent from only one of the subscriber’s IP addresses and MUST only open pinholes towards the subscriber’s own IP address.

- If IPv6 firewall: the internal IP address indicated in the PCP message and the source IPv6 address of received PCP request MUST belong to the same IPv6 prefix. The length of the IPv6 prefix is the same as the length assigned to each subscriber on that particular network (e.g., /64), and that length must be configurable by the network operator.

PCP-controlled devices can be a DS-Lite AFTR or an IPv4-IPv6 interconnection node such as NAT46 or NAT64. These nodes are deployed by Service Providers to deliver global connectivity service to their customers. Appropriate functions to restrict the use of these resources (e.g., LSN facility) to only subscribed users should be supported by these devices. Access control can be implicit or explicit:

- It is said to be explicit if an authorisation procedure is required for a user to be granted access to such resources. For such variant of PCP-controlled device, a subscriber can be identified by an IPv6 address, an IPv4 address, a MAC address, or any other information.

- For other scenarios, such as plain IPv4-in-IPv6 encapsulation for a DS-Lite architecture, the access to the service is based on the source IPv6 prefix. No per-user policies is pre-configured in the PCP-controlled device.

11. Interworking with UPnP IGD 1.0 and 2.0

[Ed. Note: This UPnP IGD Interworking section will likely be moved to a separate document which will fully describe how a proxy needs to translate UPnP IGD messages into PCP messages. This is tracked as PCP Issue #28 [PCP-Issues].]

The following diagram shows how UPnP IGD can be interworked with PCP, using an interworking function (IWF).
11.1. UPnP IGD 1.0 with AddPortMapping Action

In UPnP IGD 1.0 [IGD] it is only possible to request a specific port using the AddPortMapping action. Requiring a specific port is incompatible with both (1) a large-scale NAT and with (2) widely-deployed applications. Regarding (1), another subscriber is likely to already be using the same port, so it will be unavailable to this application to operate a server. Regarding (2), if the same popular application exists on two devices behind the same NAPT, they cannot both get the same port. PCP cannot correct this behavior of UPnP IGD:1, but PCP does work with this behavior.

Due to this incompatibility with large-scale address sharing and popular applications, future hosts and applications will either support UPnP IGD 2.0’s AddAnyPortMapping method (see Section 11.2) or will support PCP.

When a requested port assignment fails, most UPnP IGD control points will retry the port assignment requesting the next higher port or requesting a random port. These UPnP IGD requests are translated to PCP requests and sent to the PCP server. The requests include the HONOR_EXTERNAL_PORT option, which causes the PCP server to return an error if it cannot allocate the requested port. The interworking function translates the PCP error response to a UPnP IGD error response. This repeats until the UPnP IGD client gives up or until the PCP server is able to return the requested port.
Message flow would be similar to this:

<table>
<thead>
<tr>
<th>UPnP Control Point</th>
<th>in-home CPE</th>
<th>PCP server</th>
</tr>
</thead>
<tbody>
<tr>
<td>-UPnP: AddPortMapping(80) ---&gt;</td>
<td>-PCP: request port 80 ----&gt; HONOR_EXTERNAL_PORT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;-PCP: error----------------------</td>
</tr>
<tr>
<td>&lt;-UPnP: unavailable----&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>-UPnP: AddPortMapping(3213) ---&gt;</td>
<td>-PCP: request port 3213 ----&gt; HONOR_EXTERNAL_PORT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;-PCP: error----------------------</td>
</tr>
<tr>
<td>&lt;-UPnP: unavailable----&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td>...</td>
</tr>
<tr>
<td>-UPnP: AddPortMapping(8921) ---&gt;</td>
<td>-PCP: request port 8921 ----&gt; HONOR_EXTERNAL_PORT</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>&lt;-PCP: ok, port 8921-------</td>
</tr>
<tr>
<td>&lt;-UPnP: ok, port 8921-------</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 17: Message Flow: Interworking from UPnP IGD 1.0 AddPortMapping action to PCP

11.2. UPnP IGD 2.0 with AddAnyPortMapping Action

If the UPnP IGD control point and the UPnP IGD interworking function both implement UPnP IGD 2.0 [IGD-2] and the UPnP IGD control point uses IGD 2’s new AddAnyPortMapping action, only one round-trip is necessary. This is because AddAnyPortMapping has semantics similar to PCP’s semantics, allowing the PCP server to assign any port.
Message flow would be similar to this:

```
<table>
<thead>
<tr>
<th>UPnP Control Point</th>
<th>in-home CPE</th>
<th>PCP server</th>
</tr>
</thead>
<tbody>
<tr>
<td>-UPnP:AddAnyPortMapping()</td>
<td>-PCP:external port 0-----&gt;</td>
<td></td>
</tr>
<tr>
<td>&lt;-UPnP:port=12345--------</td>
<td>&lt;-PCP:external port=12345-</td>
<td></td>
</tr>
</tbody>
</table>
```

Figure 18: Message Flow: Interworking from UPnP IGD 2.0 AddAnyPortMapping action to PCP

### 11.3. Lifetime Maintenance

Neither UPnP IGD 1.0 nor 2.0 provide a lifetime. Thus, the UPnP IGD/PCP interworking function is responsible for extending the lifetime of mappings that are still interesting to the UPnP IGD control point.

Note: It can be an implementation advantage, where possible, for the UPnP IGD/PCP interworking function to request a port mapping lifetime only while that client is active and connected. For example, creating a PCP mapping that is equal to the client’s remaining DHCP lifetime is one useful approach. The UPnP IGD/PCP interworking function is responsible for renewing the PCP lifetime as necessary. For clients not using DHCP, other mechanisms to check on the client host’s liveness can also be useful (e.g., ping, ARP, or WiFi association) can be used to discern liveness of the UPnP IGD control point. However, it is NOT RECOMMENDED to attempt to connect to the TCP or UDP port opened on the control point to determine if the host still wants to receive packets, as the server could be temporarily down when tested, causing a false negative.

### 12. Security Considerations

[Ed. Note: The following recommendation does not have consensus] The PCP Client SHOULD be bound to one single UDP port which SHOULD be randomly generated as per [I-D.ietf-tsvwg-port-randomization].

On today’s Internet, ISPs to not typically filter incoming traffic for their subscribers. However, when ISP introduce stateful address sharing with NAPT devices, such filtering will occur as a side effect. PCP allows controlling that filtering, and PCP allows indicating the ‘inside’ IP address that should have the filtering removed. It is important that PCP allow removing the filtering for hosts belonging to one subscriber, but not hosts belonging to another
subscriber. This is done in different ways depending on the architecture of the address sharing device and how subscribers are identified behind that device, and described in detail in Section 10.5.

Because of the state created in a NAPT or firewall, it is anticipated that port forwarding (PIN OpCodes) will have a quota applied to each subscriber. If the quota is small and the maximum lifetime is large, a faulty or disconnected PCP client could cause a denial of service for other PCP clients belonging to that same subscriber. To prevent this problem, if a PCP server is configured for a small per-subscriber quota (e.g., less than 15 ports) then it is RECOMMENDED it also be be configured for a short maximum lifetime (e.g., 5 minutes).

13. IANA Considerations

IANA is requested to perform the following actions:

13.1. PCP Server IP address

IANA shall assign an IPv4 and an IPv6 address for PCP discovery.

[Ed. Note: perhaps we can use the AFTR element’s IPv4 address? But still need an IPv6 address assigned for PIN64 and PIN66. This is tracked as PCP Issue #8 [PCP-Issues].]

13.2. Port Number

IANA has assigned UDP port 44323 for PCP.

13.3. OpCodes

IANA shall create a new protocol registry for PCP OpCodes, initially populated with the values in Section 8 and Section 9.

New OpCodes in the range 1-95 can be created via Standards Action [RFC5226], and the range 96-128 is for Private Use [RFC5226].

13.4. Result Codes

IANA shall create a new registry for PCP result codes, numbered 0-255, initially populated with the result codes from both Figure 6 and Figure 12.

New Result Codes can be created via Specification Required [RFC5226].
13.5. Options

IANA shall create a new registry for PCP Options, numbered 0-255 with an associated mnemonic. The values 0-127 are optional-to-process, and 128-255 are mandatory-to-process. The initial registry contains the options described in Section 8.5, and the option values 0 and 255 are reserved.

New information elements in the range 0-63 and 128-192 can be created via Standards Action [RFC5226], and the range 64-127 and 192-255 is for Private Use [RFC5226].

14. Authors

The following individuals contributed substantial text to this document and are listed in alphabetical order:

  Mohamed Boucadair
  France Telecom
  Email: mohamed.boucadair@orange-ftgroup.com

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

  Francis Dupont
  ISC
  Email: Francis.Dupont@fdupont.fr

  Reinaldo Penno
  Juniper Networks
  Email: rpenno@juniper.net

15. Acknowledgments

Thanks to Alain Durand, Christian Jacquenet, and Simon Perreault for their comments and review. Thanks to Simon Perreault for highlighting the interaction of dynamic connections with PCP-created pinholes.

16. References
16.1. Normative References

[I-D.ietf-behave-v6v4-xlate]
Li, X., Bao, C., and F. Baker, "IP/ICMP Translation Algorithm", draft-ietf-behave-v6v4-xlate-23 (work in progress), September 2010.

[I-D.ietf-behave-v6v4-xlate-stateful]

[I-D.ietf-softwire-dual-stack-lite]

[I-D.ietf-tsvwg-port-randomization]


[proto_numbers]

16.2. Informative References

[I-D.arkko-dual-stack-extra-lite]
Arkko, J. and L. Eggert, "Scalable Operation of Address

Wing

Expires July 23, 2011
Translators with Per-Interface Bindings",
draft-arkko-dual-stack-extra-lite-03 (work in progress),
October 2010.

[I-D.ietf-behave-lsn-requirements]
Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida,
"Common requirements for IP address sharing schemes",
draft-ietf-behave-lsn-requirements-00 (work in progress),
October 2010.

[I-D.ietf-v6ops-cpe-simple-security]
Woodyatt, J., "Recommended Simple Security Capabilities in
Customer Premises Equipment for Providing Residential IPv6
Internet Service", draft-ietf-v6ops-cpe-simple-security-16
(work in progress), October 2010.

[I-D.miles-behave-l2nat]
Miles, D. and M. Townsley, "Layer2-Aware NAT",
draft-miles-behave-l2nat-00 (work in progress),
March 2009.

[IGD] UPnP Gateway Committee, "WANIPConnection:1",
UPnP-gw-WANIPConnection-v1-Service.pdf>.

[IGD-2] UPnP Gateway Committee, "Internet Gateway Device (IGD) V

[PCP-Issues] PCP Working Group, "PCP Active Tickets", January 2011,

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7,
RFC 793, September 1981.

"Service Location Protocol, Version 2", RFC 2608,
June 1999.

[RFC3424] Daigle, L. and IAB, "IAB Considerations for UNilateral
Self-Address Fixing (UNSAF) Across Network Address

[RFC3581] Rosenberg, J. and H. Schulzrinne, "An Extension to the
Session Initiation Protocol (SIP) for Symmetric Response

Appendix A. Changes

A.1. 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 we want to add DTLS support.

- Moved subscriber identity from common PCP section to PIN* 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.

A.2. 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 pinhole also allows ICMP messages associated with that pinhole.
A.3. 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 PIN* opcodes.
- Renamed Informational Elements (IE) to Options.
- Added REMOTE_PEER (for disambiguation with dynamic ports), REMOTE_PEER_FILTER (for simple packet filtering), and HONOR_EXTERNAL_PORT (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 wasn’t in the request.
- Response 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.

Appendix B. Analysis of Techniques to Discover PCP Server

[Ed. Note: This Appendix will be removed in a later version of this document. It is included here for reference and discussion purposes. This is tracked as PCP Issue #8 [PCP-Issues].]

Discussion Note: A deployment model is a non-PCP aware NAT in a home, and a PCP-aware large scale NAT (LSN) operated by the ISP. For example, the home users purchased a NAT last year at a computer shop (to extend their home’s WiFi network). This could work by having the host use UPnP IGD with the in-home NAT, and the host use PCP with the LSN. But this deployment model is impossible with several of the mechanisms below. Is this
deployment model important, or can we wait for PCP to be enabled on CPE?

Several mechanisms for discovering the PCP Server can be envisaged as listed below:

1. A special-purpose IPv4 or IPv6 address, assigned by IANA, which is routed normally until it hits a PCP Server, which responds.

   Analysis: This solution can be deployed in the context of DS-Lite architecture. Concretely, a well-known IPv4 address can be used to reach a PCP Server embedded in the device that embeds the AFTR capabilities. Since all IPv4 messages issued by a DS-Lite CP router will be encapsulated in IPv6, no state synchronisation issues will be experienced because PCP messages will be handled by the appropriate PCP Server.

   In some deployment scenarios (e.g., deployment of several stateful NAT64/NAT46 in the same domain), the use of this address is not recommended since PCP messages, issued by a given host, may be handled by a PCP Server embedded in a NAT node which is not involved to handle IP packets issued from that host. The use of this special-purpose IP address may induce session failures and therefore the customer may experience troubles when accessing its services.

   Consequently, the use of a special-purpose IPv4 address is suitable for DS-Lite NAT44. As for NAT46/NAT64, this is left to the Service Providers according to their deployment configuration.

   The special-use address MUST NOT be advertised in the global routing table. Packets with that destination address SHOULD be filtered so they are not transmitted on the Internet.

2. Assume the default router is a PCP Server, and send PCP packets to the IP address of the default router.

   Analysis: This solution is not suitable for DS-Lite NAT44 nor for all variants of NAT64/NAT46.

   In the context of DS-Lite: There is no default IPv4 router configured in the CP router. All outgoing IPv4 traffic is encapsulated in IPv6 and then forwarded to a pre-configured DS-Lite AFTR device. Furthermore, if IPv6 is used to reach the PCP Server, the first router may not be the one which embeds the AFTR.
For NAT64/NAT46 scenarios: The NAT function is not embedded in the first router, therefore this solution candidate does not allow to discover a valid PCP Server.

Therefore, this alternative is not recommended.

3. Service Location Protocol (SLP [RFC2608]).

Analysis: This solution is not suitable in scenarios where multicast is not enabled. SLP is a chatty protocol. This alternative is not recommended.

4. NAPTR. The host would issue a DNS query for a NAPTR record, formed from some bits of the host’s IPv4 or IPv6 address. For example, a host with the IPv6 address 2001:db8:1:2:3:4:567:89ab would first send an NAPTR query for 3.0.0.0.2.0.0.0.1.0.0.0.8.b.d.0.1.0.0.2.IP6.ARPA (20 elements, representing a /64 network prefix), which returns the PCP Server's IPv6 address. A similar scheme can be used with IPv4 using, for example, the first 24 bits of the IPv4 address.

Analysis: This solution candidate requires more configuration effort by the Service Provider so as to redirect a given client to the appropriate PCP Server. Any change of the engineering policies (e.g., introduce new LSN device, load-based dimensioning, load-balancing, etc.) would require to update the zone configuration. This would be a hurdle for the flexibility of the operational networks. Adherence to DNS is not encouraged and means which allows for more flexibility are to be promoted.

Therefore, this mechanism is not recommended.

5. New DHCPv6/DHCP option and/or a RA option to convey an FQDN of a PCP Server.

Analysis: Since DS-Lite and NAT64/NAT46 are likely to be deployed in provider-provisioned environments, DHCP (both DHCPv6 and IPv4 DHCP) is convenient to provision the address/FQDN of the PCP Server.