PCP working group D. Wing, Ed.
Internet-Draft Cisco
Intended status: Standards Track S. Cheshire
Expires: April 23, 2012 Apple
M. Boucadair
France Telecom
R. Penno
Juniper Networks
P. Selkirk
ISC
October 21, 2011

Port Control Protocol (PCP)
draft-ietf-pcp-base-16

Abstract

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

Status of this Memo

Copyright Notice

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, IPv6 and IPv4 firewall devices, and a mechanism to reduce application keepalive traffic. PCP is designed to be implemented in the context of Carrier-Grade NATs (CGNs), small NATs (e.g., residential NATs), as well as with dual-stack and IPv6-only CPE routers, and all of the currently-known transition scenarios towards IPv6-only CPE routers. PCP allows hosts to operate servers 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 or an IPv6 firewall integrated in their CPE router.

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 might use a rendezvous server specific to that game (or specific to that game developer), a SIP phone would use a SIP proxy, and a client using DNS-Based Service Discovery [I-D.cheshire-dnsext-dns-sd] would use DNS Update [RFC2136] [RFC3007]. PCP does not provide this rendezvous function. The rendezvous function may support IPv4, IPv6, or both. Depending on that support and the application's support of IPv4 or IPv6, the PCP client may 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 such 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 include application layer gateways (ALGs) to create mappings for applications that establish additional streams or accept incoming connections. ALGs incorporated into NATs may also 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:

- Basic NAT [RFC3022]

- Network Address and Port Translation [RFC3022], such as commonly deployed in residential NAT devices

- Carrier-Grade NAT [I-D.ietf-behave-lsn-requirements]

- Dual-Stack Lite (DS-Lite) [RFC6333]

- Layer-2 Aware NAT [I-D.miles-behave-l2nat]

- Dual-Stack Extra Lite [I-D.arkko-dual-stack-extra-lite]

- NAT64, both Stateless [RFC6145] and Stateful [RFC6146]

- IPv4 and IPv6 simple firewall control [RFC6092]

- NPTv6 [RFC6296]

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., RSVP, IPsec ESP, ICMP, ICMPv6) are supported for IPv4 firewall, IPv6 firewall, and NPTv6 functions, but are out of scope for any NAT functions.

2.3. Single-homed Customer Premises Network

PCP assumes a single-homed IP address model. That is, for a given IP address of a host, 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 a PCP-enabled NAT for every egress (because the host could not reliably determine which egress path packets would take) and 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 might have the necessary port mapped to another host).

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 "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 resulting from 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. This can include another Internal Host (or even the same Internal Host); if a NAT is involved, the NAT would need to hairpin the traffic.

Internal Address:
The address of an Internal Host served by a NAT gateway or protected by a firewall.

External Address:
The address of an Internal Host as seen by other Remote Peers 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.

Endpoint-Dependent Mapping (EDM): A term applied to NAT operation where an implicit mapping created by outgoing traffic (e.g., TCP SYN) from a single Internal Address and Port to different Remote Peers and Ports may be assigned different External Ports, and a subsequent PCP MAP request for that Internal Address and Port may be assigned yet another different External Port. This term encompasses both Address-Dependent Mapping and Address and Port-Dependent Mapping from [RFC4787].

Remote Peer Address:

The address of a Remote Peer, as seen by the Internal Host. A Remote Address is generally a publicly routable address. In the case of a Remote Peer that is itself served by a NAT gateway, the Remote Address may in fact be the Remote Peer'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 that does not implement PCP, the presence of the THIRD_PARTY Option in the MAP request signifies that the specified address, rather than the source IP address of the PCP request packet, should be used as the Internal Address for the Mapping.

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 IP address and port number to the same external IP address and port number. Firewall filtering, applied to that identity function, is separate from the mapping itself.

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 or a PCP PEER request, and allow Internal Hosts to receive replies to their outbound packets. Explicit dynamic mappings are created as a result of a PCP MAP request. Static mappings are created by manual configuration (e.g., via 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., to 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 effectively infinite (they exist until manually removed) but otherwise they behave exactly the same as an explicit dynamic mapping.

PCP Client:

A PCP software instance responsible for issuing PCP requests to a PCP server. Several independent PCP Clients can exist on the same host (just as several independent web browsers can exist on the same host). Several PCP Clients can be located in the same local network. A PCP Client can issue PCP requests 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-Controlled Device:

A NAT or firewall that controls or rewrites packet flows between internal hosts and remote hosts. PCP manages the Mappings on this device.

PCP Server:

A PCP software instance that implements the server side of the PCP protocol, via which PCP clients request and manage explicit mappings. This is conceptually separate from the NAT or firewall itself, but is typically implemented as a capability of the PCP-controlled device. See also Section 4.

Interworking Function:

A functional element responsible for translating or proxying another protocol to PCP. For example interworking between UPnP IGD [IGD] with PCP.

Subscriber:

The unit of billing for a commercial ISP. A subscriber may have a single IP address from the commercial ISP (which can be shared among multiple hosts using a NAT gateway, thereby making them appear to be a single host to the ISP) or may have multiple IP addresses provided by the commercial ISP. In either case, the IP address or addresses provided by the ISP may themselves be further translated by a large-scale NAT operated by the ISP.

4. Relationship between PCP Server and its NAT/firewall

The PCP server receives and responds to PCP requests. The PCP server functionality is typically a capability of a NAT or firewall device, as shown in Figure 1. It is also possible for the PCP functionality to be provided by some other device, which communicates with the actual NAT or firewall via some other proprietary mechanism, as long as from the PCP client's perspective such split operation is indistinguishable from the integrated case.

```
                                 +-----------------+
         +------------+          | NAT or firewall |
         | PCP client |-<network>-+      with       +---<Internet>
         +------------+          |    PCP server   |
                                 +-----------------+
```

Figure 1: PCP-Enabled NAT or Firewall

A NAT or firewall device, between the PCP client and the Internet, might implement simple or advanced firewall functionality. This may be a side-effect of the technology implemented by the device (e.g., a network address and port translator, by virtue of its port rewriting, normally requires connections to be initiated from an inside host towards the Internet), or this might be an explicit firewall policy to deny unsolicited traffic from the Internet. Some firewall devices deny certain unsolicited traffic from the Internet (e.g., TCP, UDP to most ports) but allow certain other unsolicited traffic from the Internet (e.g., UDP port 500 and IPsec ESP as described in [RFC6092]). Such default filtering (or lack thereof) is out of scope of PCP itself. If a device supports PCP and wants to receive traffic, and does not possess knowledge of such filtering, it SHOULD use PCP to create the necessary mappings to receive the desired traffic.

5. Note on Fixed-Size Addresses

For simplicity in building and parsing request and response packets, PCP always uses fixed-size 128-bit IP address fields for both IPv6 addresses and IPv4 addresses.

When the address field holds an IPv6 address, the fixed-size 128-bit IP address field holds the IPv6 address stored as-is.

When the address field holds an IPv4 address, IPv4-mapped IPv6 addresses [RFC4291] are used (::ffff:0:0/96). This has the first 80 bits set to zero and the next 16 set to one, while its last 32 bits are filled with the IPv4 address. This is unambiguously

distinguishable from a legal IPv6 address, because IPv4-mapped IPv6 address [RFC4291] are not used as either the source or destination address of actual IPv6 packets.

When checking for an IPv4-mapped IPv6 address, all of the first 96 bits MUST be checked for the pattern -- it is not sufficient to check for 0xFF in bits 81-96.

The all-zeroes IPv6 address is expressed by filling the fixed-size 128-bit IP address field with all zeroes (::).

The all-zeroes IPv4 address is expressed as: 80 bits of zeros, 16 bits of ones, and 32 bits of zeros (::ffff:0:0).

6. 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 9 and Section 10.

6.1. 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              |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                 Requested Lifetime (32 bits)                  |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     |            PCP Client's IP address (128 bits)                 |
     |                                                               |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     :                                                               :
     :             (optional) Opcode-specific information            :
     :                                                               :
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     :                                                               :
     :             (optional) PCP Options                            :
     :                                                               :
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2: Common Request Packet Format

These fields are described below:

Version: This document specifies protocol version 1. This value MUST be 1 when sending, and MUST be 1 when receiving. This field is used for version negotiation as described in Section 7.6.

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

Opcode: A seven-bit value specifying the operation to be performed. Opcodes are defined in Section 9 and Section 10.

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

Requested Lifetime: 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 it on reception.

PCP Client's IP Address: The source IPv4 or IPv6 address in the IP header used by the PCP client when sending this PCP request. IPv4 is represented using an IPv4-mapped IPv6 address.

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

6.2. 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 (32 bits)                       |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                        Epoch (32 bits)                        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     |                    Reserved (96 bits)                         |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     :                                                               :
     :             (optional) Opcode-specific response data          :
     :                                                               :
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     :             (optional) Options                                :
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3: Common Response Packet Format

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 7-bit 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 6.4 for values. This is set by the server.

Lifetime: 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 7.5 for discussion. This value is set by the server, in both success and error responses.

Reserved: 96 reserved bits, MUST be sent as 0, MUST be ignored when received. This is set by the server.

6.3. Options

A PCP Opcode can be extended with one or more Options. Options can be used in requests and responses. The design decisions in this specification about whether to include a given piece of information in the base Opcode format or in an Option were 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 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                         :
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4: Options Header

The description of the fields is as follows:

Option Code: 8 bits. Its most significant 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. Options that are not a multiple of four octets long are followed by one, two, or three octets of zeros to pad their effective length in the packet to be a multiple of four octets. The Option Length reflects the semantic length of the option, not including the padding octets.

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

The handling of an Option by the PCP client and PCP server MUST be specified in an appropriate document, which MUST include whether the PCP Option can appear in a request and/or response, whether it can appear more than once, and indicate what sort of Option data it conveys. If several Options are included in a PCP request, they MAY be encoded in any order by the PCP client, but MUST be processed by the PCP server in the order in which they appear. It is the responsibility of the PCP client to ensure the server has sufficient room to reply with an error including UNPROCESSED Options; this can be achieved by sending messages that don't exceed 1024-4*number_of_options octets.

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 as

received in the request. Additional Options included in the response (if any) MUST be included at the end. An Option MAY appear more than once in a request or in a response, if permitted by the definition of the Option. If the Option's definition allows the Option to appear only once but it appears more than once in a request, and the Option is understood by the PCP server, 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 MAY log an error. If an invalid option length is encountered (e.g., option length extends beyond the length of the PCP Opcode itself), the error MALFORMED_OPTION SHOULD be returned (rather than MALFORMED_REQUEST), as that helps the client better understand how the packet was malformed. The UNPROCESSED option MUST NOT appear in a request; if it does, it causes a MALFORMED_REQUEST error. If a PCP response would have exceeded the maximum PCP message size, the PCP server MAY respond with MALFORMED_REQUEST.

The most significant bit in the Option Code indicates if its processing is optional or mandatory. If the most significant bit is set, handling this Option is optional, and a PCP server MAY process or ignore this Option, entirely at its discretion. If the most significant bit is clear, handling this Option is mandatory, and a PCP server MUST process this Option or return an error code if it cannot. 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 an error response with code UNSUPP_OPTION or MALFORMED_OPTION (as appropriate) and MUST include the UNPROCESSED Option in the response (see Section 7.7.1).

PCP clients are free to ignore any or all Options included in responses, although naturally if a client explicitly requests an Option where correct handling of that Option requires processing the Option data in the response, that client is expected to implement code to do that.

Different options are valid for different Opcodes. For example, the UNPROCESSED option is valid for all Opcodes, but only in response messages. The THIRD_PARTY Option is valid for both MAP and PEER Opcodes. The PREFER_FAILURE option is valid only for the MAP Opcode (for the PEER Opcode, its semantics are implied). The FILTER option is valid only for the MAP Opcode (for the PEER Opcode it would have no meaning).

Option definitions MUST include the information below:

Option Name: <mnemonic>
Number: <value>
Purpose: <textual description>
Valid for Opcodes: <list of Opcodes>
Length: <rules for length>
May appear in: <requests/responses/both>
Maximum occurrences: <count>

6.4. 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 encounters multiple errors during processing of a request, it SHOULD use the most specific error message. Each error code below is classified as either a 'long lifetime' error or a 'short lifetime' error, 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 a 30 second lifetime and long lifetime errors use a 30 minute lifetime.

0 SUCCESS: Success.

1 UNSUPP_VERSION: Unsupported protocol version.

2 NOT_AUTHORIZED: The requested operation is disabled for this PCP client, or the PCP client requested an operation that cannot be fulfilled by the PCP server's security policy. This is a long lifetime error.

3 MALFORMED_REQUEST: The request could not be successfully parsed.

4 UNSUPP_OPCODE: Unsupported Opcode.

5 UNSUPP_OPTION: Unsupported Option. This error only occurs if the Option is in the mandatory-to-process range.

6 MALFORMED_OPTION: Malformed Option (e.g., appears too many times, invalid length).

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

8 NO_RESOURCES: Request is well-formed and valid, but the server has insufficient resources to complete the requested operation at this time. For example, the NAT device cannot create more mappings at this time, is short of CPU cycles or memory, or due to some other temporary condition. The same request may succeed in the future.

This is a system-wide error, and different from USER_EX_QUOTA. This is a short lifetime error. This can be used as a catch-all error, should no other error message be suitable.

9 UNSUPP_PROTOCOL: Unsupported Protocol. This is a long lifetime error.

10 USER_EX_QUOTA: Mapping would exceed user's port quota. This is a short lifetime error.

11 CANNOT_PROVIDE_EXTERNAL_PORT: the requested external port cannot be provided. This error MUST only be returned for PEER requests, for MAP requests that included the PREFER_FAILURE Option (because otherwise a new external port could have been assigned), or MAP requests for the SCTP protcool. See Section 11.2 for processing details. The error lifetime depends on the reason for the failure.

12 ADDRESS_MISMATCH: the source IP address or port of the request packet does not match the contents of the PCP Client's IP Address or UDP port.

13 EXCESSIVE_REMOTE_PEERS: 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 FILTER Option. See Section 11.3 for processing information. This is a long lifetime error.

7. 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, meaning that if the PCP client sends the same request multiple times (or the PCP client sends the request once and it is duplicated by the network), and the PCP server processes those requests multiple times, the result is the same as if the PCP server had processed only one of those duplicate requests.

7.1. General PCP Client: Generating a Request

This section details operation specific to a PCP client, for any Opcode. Procedures specific to the MAP Opcode are described in Section 9, and procedures specific to the PEER Opcode are described in Section 10.

Prior to sending its first PCP message, the PCP client determines

which server to use. The PCP client performs the following steps to determine its PCP server:

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

2. the default router list (for IPv4 and IPv6) is used as the list of PCP Server(s).

For the purposes of this document, only a single PCP server address is supported. Should future specifications define configuration methods that provide a list of PCP server addresses, those specifications will define how clients select one or more addresses from that list.

With that PCP server address, 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 client software on any operating system, when several independent PCP clients exist on 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].

To assist with detecting an on-path NAT, the PCP client MUST include the source IP address of the PCP message in the PCP request. This is typically its own IP address; see Section 12.4 for how this can be coded.

When attempting to contact a PCP server, the PCP client initializes a timer to 2 seconds. The PCP client sends a PCP message to 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.

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.

7.2. 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 MUST only accept normal (non-THIRD_PARTY) PCP requests from a client on the same interface it would normally receive packets from that client, and MUST silently ignore PCP requests arriving on any other interface. For example, a residential NAT gateway accepts PCP requests only when they arrive on its (LAN) interface connecting to the internal network, and silently ignores any PCP requests arriving on its external (WAN) interface. A PCP server which supports THIRD_PARTY requests MAY be configured to accept THIRD_PARTY requests on other interfaces from properly authorized clients.

Upon receiving a request, 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 at least two octets long but the first octet (version) is a version that is not supported, a response is generated with the UNSUPP_VERSION result code, and the other steps detailed in Section 7.6 are followed.

Otherwise, if the version is supported but the received message is shorter than 4 octets or has the R bit set, the message is silently dropped.

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 it MAY generate the NO_RESOURCES error response.

If the length of the message 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.

The PCP server compares the IP address (from the IP header) with the field PCP Client IP Adddress. If they do not match, the error ADDRESS_MISMATCH MUST be returned. This is done to detect and prevent accidental use of PCP where a non-PCP-aware NAT exists between the PCP client and PCP server. If the PCP client wants such a mapping it needs to ensure the PCP field matches the IP address from the perspective of the 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 set as indicated in Figure 3.

Copying request fields are important because this is what enables a client to identify to which request a given error response pertains. For OpCodes that are understood by the PCP server, it follows the requirements of that OpCode to copy the appropriate fields. For OpCodes that are not understood by the PCP server, it simply generates the UNSUPP_OPCODE response and copies fields from the PCP header and copies the rest of the PCP payload as-is (without attempting to interpret it).

7.3. General PCP Client: Processing a Response

The PCP client receives the response and verifies that the source IP address and port belong to the PCP server of an outstanding PCP request. It validates that the Opcode matches an outstanding PCP request. Responses shorter than 24 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 7.5).

If the PCP Client's IP Address and PCP Client's Port fields of the PCP response header do not match the source address and port of the request, it indicates the presence of a NAT between the PCP client and PCP server. If they don't match, then the PCP client (or the user on the client host) MUST ensure that an appropriate NAT mapping is created on the intervening NAT(s) (e.g., using UPnP IGD, NAT-PMP, or manual configuration), otherwise, the PCP-installed mapping will be ineffective.

If the result code is 0 (SUCCESS), 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 7.6. If the result code is NO_RESOURCES, PCP client 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 the PCP server indicates an error lifetime in excess of 30 minutes, the 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.

7.4. 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 two Global Unicast Addresses belonging to different global IPv6 prefixes.

7.5. Epoch

Every PCP response sent by the PCP server includes an Epoch time field. This time field increments by 1 every second. Anomalies in the received Epoch time value provide a hint to PCP clients that a PCP server state loss may have occurred. Clients respond to such state loss hints by promptly renewing their mappings, so as to quickly restore any lost state at the PCP server.

If the PCP server resets or loses the state of its explicit dynamic Mappings (that is, those mappings created by PCP requests), due to reboot, power failure, or any other reason, it MUST reset its Epoch time to its initial starting value (usually zero) to provide this hint to PCP clients. After resetting its Epoch time, the PCP server resumes incrementing the Epoch time value by one every second. Similarly, if the public IP address(es) of the NAT (controlled by the

PCP server) changes, the Epoch time MUST be reset. A PCP server MAY maintain one Epoch time value for all PCP clients, or MAY maintain distinct Epoch time values (per PCP client, per interface, or based on other criteria); this choice is implementation-dependent.

Whenever a client receives a PCP response, the client validates the received Epoch time value according to the procedure below, using integer arithmetic:

- If this is the first PCP response the client has received from this PCP server, it is treated as necessarily valid, otherwise
  - If the current PCP server Epoch time value (current_server_time) is less than the previously received PCP server Epoch time value (previous_server_time) then the client treats the Epoch time value as obviously invalid (time should not go backwards), else
    - The client computes the difference between the current PCP server Epoch time value (current_server_time) and the
      previously received Epoch time value (previous_server_time):
      server_delta = current_server_time - previous_server_time;
    - The client computes the difference between the current local time value (current_client_time) and the time the previous PCP response was received from this PCP server (previous_client_time):
      client_delta = current_client_time - previous_client_time;
    - If client_delta+2 < server_delta - server_delta/8
      or server_delta+2 < client_delta - client_delta/8
      then the client treats the Epoch time value as invalid,
      else the client treats the Epoch time value as valid
- The client records the current time values for use in its next comparison:
  previous_server_time = current_server_time
  previous_client_time = current_client_time

If the PCP client determined that the Epoch time value it received was invalid then it concludes that the PCP server may have lost state, and promptly renews all its active port mapping leases as described in Section 12.3.1.

Note: The "+2" in the calculations above is to accomodate quantization errors in client and server clocks (up to one second quantization error each in server and client time intervals).

Note: The "/8" in the calculations above is to accomodate inaccurate clocks in low-cost devices. This value allows for a difference of up to 12.5% in clock rate between PCP client and server to be treated as benign by the client. This value has not been discussed by the PCP working group. If we were to require more accurate clocks in low-cost devices then more restrictive error tolerances could be imposed, such as "/64" or "/256".

7.6. 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 result code UNSUPP_VERSION, the client MAY log an error message to inform the user that it is too old to work with this server.

Should later updates to this document specify different message formats with a version number greater than 1, and backwards compatibility is desired, these first two octets can be used for forward and backward compatibility.

If future PCP versions greater than 1 are specified, version negotiation proceeds 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. The client sends first request using highest (i.e., presumably 'best') version number it supports.

3. If the server supports that version it responds normally.

4. If the server does not support that version it replies giving a result containing the result 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.

7.7. General PCP Option

The following Option can appear in certain PCP responses, without regard to the Opcode.

7.7.1. 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 MAY 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:

```
    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=0 |  Reserved     |   Option Length=variable      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   | Option-code-1 | ... additional option-codes as necessary      |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: UNPROCESSED option

Option Name: UNPROCESSED
Number: 0
Purpose: indicates which PCP Options in the request were not processed by the PCP server
Valid for Opcodes: all

Length: 1 octet or more
May appear in: responses, and only if the result code is non-zero.
Maximum occurrences: 1

8. 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 (Section 8.1);

- a host operating a client and server on the same port (Section 8.2);

- a host operating a client and wanting to optimize the application keepalive traffic (Section 8.3);

- and a host operating a client and wanting to restore lost state in its NAT (Section 8.4).

These are discussed in the following sections.

When operating a server (Section 8.1 and Section 8.2) 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 or IPv6 address configured on one of its interfaces. It takes the union of this knowledge to decide to which of its PCP servers to send the request (e.g., a PCP server on its IPv4 interface or its IPv6 interface), and if to send one or two MAP requests for each of its interfaces (e.g., if the PCP client has only an IPv4 address but wants both IPv6 and IPv4 listeners, it sends a MAP request containing the all-zeros IPv6 address in the Requested External Address field, and sends a second MAP request containing the all-zeros IPv4 address in the Requested External Address field. If the PCP client has both an IPv4 and IPv6 address, and only wants an IPv4 listener, it sends one MAP request from its IPv4 interface (if the PCP server supports NAT44 or IPv4 firewall) or one MAP request from its IPv6 interface (if the PCP server supports NAT64)). The PCP client can simply request the desired mapping to determine if the PCP server supports the desired mapping. 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 for a given Internal Host. In the absence

of a PCP option indicating otherwise, it is REQUIRED that PCP-created explicit dynamic mappings be assigned the same external IP address. It is RECOMMENDED that static mappings for that Internal Host (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 assigned to a given Internal Host 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 Address. Generally, this re-assignment would occur when a CGN device is load balancing newly-seen hosts to its public IPv4 address pool.

The following table summarizes how various common PCP deployments use IPv6 and IPv4 addresses. The 'source' is the source address of the PCP packet itself, 'internal' is the Internal IP Address field of the THIRD_PARTY Option (if present) or the same as the source address of the PCP packet iself (if the THIRD_PARTY Option is not present), 'external' is the Requested External Address field of the MAP or PEER request, the 'remote peer' is the Remote Peer IP Address of the PEER request or the FILTER option of the MAP request.

	source	internal	external	remote peer
IPv4 firewall	IPv4	IPv4	IPv4	IPv4
IPv6 firewall	IPv6	IPv6	IPv6	IPv6
NAT44	IPv4	IPv4	IPv4	IPv4
DS-Lite plain mode (1)	IPv6	IPv4	IPv4	IPv4
DS-Lite encap. (2)	IPv4	IPv4	IPv4	IPv4
NAT64 (3)	IPv6	IPv6	IPv4	IPv6
NPTv6	IPv6	IPv6	IPv6	IPv6

Figure 6: Address Families with MAP and PEER

In (1) and (2), 'source' refers to the PCP messaging between the Dual-Stack Lite B4 element and the AFTR element, with (1) showing Dual-Stack Lite plain mode and (2) showing Dual-Stack Lite encapsulation mode [I-D.dupont-pcp-dslite]. In a Dual-Stack Lite environment within the subscriber's network from a host to the B4 element, the PCP messaging is IPv4 firewall, IPv6 firewall, or NAT44. In (3), the IPv6 PCP client is not necessarily aware of the NAT64 or aware of the actual IPv4 address of the remote peer, so it expresses the IPv6 address from its perspective as shown in the table.

Note that PCP requests containing the MAP or PEER Opcodes cannot delete or shorten the lifetime of an existing implicit mapping for the indicated internal address and port. Conceptually implicit and explicit mappings are different "layers" in the NAT forwarding state database.

8.1. 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 9 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 MAP Opcode, with the external address set to the appropriate all-zeroes address, depending on whether 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, ...);
bzero(&external_sockaddr, sizeof(external_sockaddr));

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 differ between
     * (1), (2), and (3).
     */
    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();
    }
```

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

8.2. 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 9, and receive a positive PCP response before it sends any packets from that port.

> Discussion: In general, a PCP client doesn't know in advance if it is behind a NAT or firewall. On detecting the host has connected to a new network, the PCP client can attempt to request a mapping using PCP, and if that succeeds then the client knows it has successfully created a mapping. If after multiple retries it has received no PCP response, then either the client is *not* behind a NAT or firewall and has unfettered connectivity, or the client *is* behind a NAT or firewall which doesn't support PCP (and the client may still have working connectivity by virtue of static mappings previously created manually by the user). Retransmitting PCP requests multiple times before giving up and assuming unfettered connectivity adds delay in that case. Initiating outbound TCP connections immediately without waiting for PCP avoids this delay, and will work if the NAT has endpoint-independent mapping (EIM) behavior, but may fail if the NAT has endpoint-dependent mapping (EDM) behavior. Waiting enough time to allow an explicit PCP MAP Mapping to be created (if possible) first ensures that the same External Port will then be used for all subsequent TCP SYNs sent from the specified Internal Address and Port. PCP supports both EIM and EDM NATs, so clients need to assume they may be dealing with an EDM NAT. In this case, the client will experience more reliable connectivity if it attempts explicit PCP MAP requests first, before initiating any outbound TCP connections from that Internal Address and Port. See also Section 12.1.

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, ...);
bzero(&external_sockaddr, sizeof(external_sockaddr));

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 differ between
     * (1), (2), and (3).
     */
    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();
    }
```

Figure 8: Pseudo-code for using PCP to operate a symmetric client/

server

8.3. For Reducing NAT Keepalive Messages

A host operating a client (e.g., XMPP client, SIP client) sends from a port, and may receive responses, but never accepts incoming connections from other Remote Peers on this port. It wants to ensure the flow to its Remote Peer 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, keep state alive on the Remote Peer, 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 PEER Opcode as described in Section 10.

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, ...);
bzero(&external_sockaddr, sizeof(external_sockaddr));

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 differ between
     * (1), (2), and (3).
     */
    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();
    }
```

Figure 9: Pseudo-code using PCP with a dynamic socket

8.4. For Restoring Lost Implicit TCP Dynamic Mapping State

After a NAT loses state (e.g., because of a crash or power failure), it is useful for clients to re-establish TCP mappings on the NAT. 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 first establishing a TCP

connection normally and then sending a PEER request/response and remembering the External Address and External Port. Later, when the NAT 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 NAT 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 NAT only creates a new implicit dynamic mapping for TCP segments with the SYN bit set (i.e., the newly-booted NAT drops the re-transmitted data segments from the client because the NAT does not have an active mapping for those segments), and if the server is not sending data that elicits a RST from the NAT. This is not the case for UDP, because a new UDP mapping will be created (probably on a different port) as soon as UDP traffic is seen by the NAT.

9. MAP Opcode

This section defines an Opcode which controls forwarding from a NAT (or firewall) to an Internal Host.

MAP: Create an explicit dynamic mapping between an Internal Address and an External IP address.

PCP Servers SHOULD provide a configuration option to allow administrators to disable MAP support if they wish.

Mappings created by PCP MAP requests are, by definition, Endpoint Independent Mappings (EIM) with Endpoint Independent Filtering (EIF) (unless the FILTER Option is used), even on a NAT that usually creates Endpoint Dependent Mappings (EDM) or Endpoint Dependent Filtering (EDF) for outgoing connections, since the purpose of an (unfiltered) 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 symmetric. 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.

On Operating Systems that allow multiple listening clients to bind to the same Internal Port, clients MUST ensure that they have exclusive use of that Internal Port (e.g., by binding the port using INADDR_ANY, or using SO_EXCLUSIVEADDRUSE or similar) before sending their MAP request, to ensure that no other clients on the same machine are also listening on the same Internal Port.

The operation of the MAP Opcode is described in this section.

9.1. MAP Operation Packet Formats

The MAP Opcode has a similar packet layout for both requests and responses. If the Assigned External IP address and Assigned External Port in the PCP response always match the Internal IP Address and Port in the PCP request, then 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 format of the Opcode-specific information in a request for the MAP Opcode.

```
      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 (128 bits)            |
     |                                                               |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 10: MAP Opcode Request Packet Format

These fields are described below:

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

Protocol: 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'.

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 legal when the lifetime is zero (a delete request), if the Protocol does not use 16-bit port numbers, or the Protocol is 0 (meaning 'all protocols')

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 MUST use 0.

Suggested External IP Address: Suggested external IPv4 or IPv6 address. This is useful for refreshing a mapping, especially after the PCP server loses state. If the PCP client does not know the external address, or does not have a preference, it MUST use the address-family-specific all-zeroes address (see Section 5).

The internal address for the request is the source IP address of the PCP request message itself, unless the THIRD_PARTY Option is used.

The following diagram shows the format of Opcode-specific information in a response packet for the MAP Opcode:

```
    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 (128 bits)            |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 11: MAP Opcode Response Packet Format

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 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: Copied from the request.

Assigned External Port: On a success response, this is the assigned external port for the mapping. On an error response, the Suggested External Port is copied from the request.

Assigned External IP Address: On a success response, this is the assigned external IPv4 or IPv6 address for the mapping. An IPv4 address is encoded using IPv4-mapped IPv6 address. On an error response, the Suggested External IP Address is copied from the request.

9.2. Generating a MAP Request

This section and Section 9.5 describe the operation of a PCP client when sending requests with the MAP Opcode.

The request MAY contain values in the Suggested External Port and Suggested External IP Address fields. This allows the PCP client to attempt to rebuild lost state on the PCP server, which improves the chances of existing connections surviving, and helps the PCP client avoid having to change information maintained at its 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 grant the suggested External IP Address and Port, and in that case it will assign a different External IP Address and Port.

If the Protocol does not use 16-bit port numbers (e.g., RSVP), the port number MUST be 0. This will cause all traffic matching that protocol to be mapped.

If the client wants all protocols mapped it uses Protocol 0 (zero) and Internal Port 0 (zero).

9.2.1. Renewing a Mapping

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 assigned 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 9.5). To reduce the risk of inadvertent synchronization of renewal requests, a random jitter component should be included. It is RECOMMENDED that PCP clients send a single renewal request packet at a time chosen with uniform random distribution in the range 1/2 to 5/8 of expiration time. If no SUCCESS response is received, then the next renewal request should be sent 3/4 to 3/4 + 1/16 to expiration, and then another 7/8 to 7/8 + 1/32 to expiration, 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 a flood of ever-closer-together requests in the last few seconds before a mapping expires).

The PCP client SHOULD impose an upper limit on this returned Assigned Lifetime value, and 24 hours is RECOMMENDED. This means if the PCP server returns an absurdly long Assigned Lifetime (e.g., 5 years), the PCP client will behave as if it received a more sane value (e.g., 24 hours).

9.3. Processing a MAP Request

This section and Section 9.5 describe the operation of a PCP server when processing a request with the MAP Opcode. Processing SHOULD be performed in the order of the following paragraphs.

The following fields from the MAP request are copied into the MAP response: Protocol, Internal Port, Requested External Address, and (if present and processed by the PCP server) the THIRD_PARTY Option.

If the Requested Lifetime is non-zero, it indicates a request to create a mapping or extend the lifetime of an existing mapping. If the PCP server or PCP-controlled device does not support the Protocol or cannot create a mapping for the Protocol (e.g., because the request is for a NAT mapping instead of a firewall mapping and the PCP-controlled device is not a NAT or does not support NATting that specific Protocol), it MUST generate an UNSUPP_PROTOCOL error. If the requested Lifetime is non-zero, the Internal Port is zero, and the Protocol is non-zero, it indicates a request to map all incoming traffic for that entire Protocol. If this request cannot be fulfilled in its entirety, the error UNSUPP_PROTOCOL MUST be returned. If the requested Lifetime is non-zero, the Internal Port is zero, and the Protocol is zero, it indicates a request to map all incoming traffic for all protocols. If this request cannot be fulfilled in its entirety, the error UNSUPP_PROTOCOL MUST be returned. If the Protocol is 0 but the Internal Port is non-zero, the error MALFORMED_REQUEST MUST be returned.

If the requested lifetime is zero, it indicates a request to delete

an existing mapping or set of mappings. Processing of the lifetime is described in Section 9.5.

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, 1:1 NAT44, and NPTv6 [RFC6296], all 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 a mapping already exists for the requested Internal Address and Port and the PREFER_FAILURE Option is not present, the PCP server MUST refresh the lifetime of that already-existing mapping, and return the already-existing External Address and Port in its response, regardless of the Suggested External Address and Port in the request. If a mapping already exists for the requested Internal Address and Port the request contains the PREFER_FAILURE Option, but the Suggested External Address and Port do not match the actual External Address and Port of the already existing mapping, the error CANNOT_PROVIDE_EXTERNAL_PORT is returned. If an implicit mapping already exists for the requested Internal Address and Port, the mapping SHOULD be upgraded to an explicit mapping.

If no mapping exists for the Internal Address and Port, and the PCP server is able to create a mapping using the Suggested External Address and Port, it SHOULD do so. This is beneficial for re-establishing state lost in the PCP server (e.g., due to a reboot). If the PCP server cannot assign the Suggested External Address and Port but can assign some other External Address and Port (and the request did not contain the PREFER_FAILURE Option) the PCP server MUST do so and return the newly assigned External Address and Port in the response. Cases where a NAT gateway cannot assign the Suggested External Address and Port include:

- The Suggested External Address and Port is already assigned to another existing explicit, implicit, or static mapping (i.e., is already forwarding traffic to some other internal address and port).

- The Suggested External Address and 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).

- The Suggested External Address and Port is otherwise prohibited by the PCP server's policy.

- The Suggested External Address or port is invalid (e.g., 127.0.0.1, ::1, multicast address, or the port 0 is not valid for the indicated protocol).

- The Suggested External Address does not belong to the NAT gateway.

- The Suggested External Address is not configured to be used as an external address of the firewall or NAT gateway.

- The PREFER_FAILURE option is included in the request and the Suggested External Address and Port are not assignable to the PCP client, which returns the CANNOT_PROVIDE_EXTERNAL_PORT error.

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.

Mappings typically consume state on the PCP-controlled device, and it is RECOMMENDED that a per-host and/or per-subscriber 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 preceding operations were successful (did not generate an error response), then the requested mapping is 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.

9.4. Processing a MAP Response

This section describes the operation of the PCP client when it receives a PCP response for the MAP Opcode.

After performing common PCP response processing, the response is further matched with an outstanding request by comparing the protocol, internal IP address, and internal port. Other fields are not compared, because the PCP server sets those fields.

On a success response, the PCP client can use the External IP Address and Port as desired. Typically the PCP client will communicate the External IP Address and Port to another host on the Internet using an

application-specific rendezvous mechanism such as DNS SRV records.

The PCP client MUST also set a timer or otherwise schedule an event to renew the mapping before its lifetime expires. Renewing a mapping is performed by sending another MAP request, exactly as described in Section 9.2, except that the Suggested External Address and Port SHOULD be set to the values received in the response. From the PCP server's point of view a MAP request to renew a mapping is identical to a MAP request to request a new mapping, and is handled identically. Indeed, in the event of PCP server state loss, a renewal request from a PCP client will appear to the server to be a request for a new mapping, with a particular Suggested External Address and Port, which happens to be what the PCP server previously assigned. See also Section 12.3.2.

On an error response, the client SHOULD NOT repeat the same request to the same PCP server within the lifetime returned in the response.

9.5. 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 configurable to restrict lifetimes to less than 24 hours, because mappings will consume ports even if the Internal Host is no longer interested in receiving the traffic or is no longer connected to the network. These recommendations are not strict, and deployments should evaluate the trade offs 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 mapping 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 from mappings created in other ways. In particular, it is implementation-dependent if outgoing traffic extends the lifetime of such mappings beyond the PCP-assigned lifetime. 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 a PCP client sends a PCP MAP request to create a mapping that already exists as a static mapping, the PCP server will return a successful result, confirming that the requested mapping exists. The

lifetime the PCP server returns for such a static mapping SHOULD be 4294967295 (0xFFFFFFFF). There is no need for a PCP client to renew a static mapping.

If the requested lifetime is zero then:

- If both the internal port and protocol are non-zero, it indicates a request to delete the indicated mapping immediately.

- If both the internal port and protocol are zero, 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.

- If the internal port is zero and the protocol is non-zero, or the internal port is non-zero and the protocol is zero, then the request is invalid and the PCP Server MUST return a MALFORMED_REQUEST error to the client.

In requests where the requested Lifetime is 0, the Suggested External Address and Suggested External Port fields MUST be set to zero on transmission and MUST be ignored on reception, and these fields MUST be copied into the Assigned External IP Address and Assigned External Port of the response.

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 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 or static 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 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 [Bonjour]) 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, External UDP and TCP ports SHOULD NOT be re-used for an interval (TIME_WAIT interval [RFC0793]). However, the PCP server SHOULD allow the previous user of an External Port 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.

9.6. Address Change Events

A customer premises router might obtain a new IP address, for 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 host's previous address might be delivered to another host which now has that address. This affects both implicit dynamic mappings and explicit dynamic mappings. However, this same problem already occurs today when a host'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 host renumbering are caused by host renumbering and are eliminated if host renumbering is avoided. PCP defined in this document does not provide machinery to reduce the host renumbering problem.

When an Internal Host changes its IP address (e.g., by having a different address assigned by the DHCP server) the NAT (or firewall) will continue to send traffic to the old IP address. Typically, the Internal Host will no longer receive traffic sent to that old IP address. Assuming the Internal Host 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 such host

renumbering 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.

9.7. Learning the External IP Address Alone

NAT-PMP [I-D.cheshire-nat-pmp] includes a mechanism to allow clients to learn the External IP Address alone, without also requesting a port mapping. In the case of PCP, this operation no longer makes sense. PCP supports Large Scale NATs (CGN) which may have a pool of External IP Addresses, not just one. A client may not be assigned any particular External IP Address from that pool until it has made at least one implicit or explicit port mapping, and even then only for as long as that implicit or explicit port mapping remains valid. Client software that just wishes to display the user's External IP Address for cosmetic purposes can achieve that by requesting a short-lived mapping and then displaying the resulting External IP Address. However, once that mapping expires a subsequent implicit or explicit dynamic mapping might be mapped to a different external IP address.

10. PEER Opcode

This section defines an Opcode for controlling dynamic mappings.

PEER: Create an implicit dynamic mapping, or set or query an existing implicit dynamic mapping to a remote peer's IPv4 address and port.

The use of these Opcodes is described in this section.

PCP Servers SHOULD provide a configuration option to allow administrators to disable PEER support if they wish.

Because a mapping created or managed by PEER behaves almost exactly as if an implicit dynamic mapping were created by a packet sent by the host (e.g., TCP SYN sent by the host), mappings created or

managed using PCP PEER requests may be Endpoint Independent Mappings (EIM) or Endpoint Dependent Mappings (EDM), with Endpoint Independent Filtering (EIF) or Endpoint Dependent Filtering (EDF), consistent with the existing behavior of the NAT gateway or firewall in question for implicit mappings it creates automatically as a result of observing outgoing traffic from Internal Hosts.

10.1. PEER Operation Packet Formats

The PEER Opcode allows the PCP client to create an implicit dynamic mapping (which functions similar to the host sending a TCP SYN), and allows the PCP client to manage an implicit dynamic mapping by extending its lifetime.

The following diagram shows the request packet format for the PEER Opcode. 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)                   |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |        Internal Port          |    Suggested External Port    |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     |           Suggested External IP Address (128 bits)            |
     |                                                               |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |       Remote Peer Port        |     Reserved (16 bits)        |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
     |                                                               |
     |               Remote Peer IP Address (128 bits)               |
     |                                                               |
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 12: PEER Opcode Request Packet Format

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: 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 describing a TCP mapping.

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

Internal Port: Internal port for the mapping.

Suggested External Port: Suggested external port for the mapping. If the PCP client does not know the external port, or does not have a preference, it MUST use 0.

Suggested External IP Address: Suggested External IP Address for the mapping. If the PCP client does not know the external address, or does not have a preference, it MUST use the address-family-specific all-zeroes address (see Section 5).

Remote Peer Port: Remote peer's port for the mapping.

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

Remote Peer IP Address: 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, and does not create or adjust the filtering associated with the mapping (for that, the FILTER option is used, Section 11.3). An IPv6 address is represented directly, and an IPv4 address is represented using the IPv4-mapped address syntax (80 bits of zeros, 16 bits of ones, and 32 bits of the IPv4 address).

When attempting to re-create a lost mapping, the Suggested External IP Address and Port are set to the External IP Address and Port fields received in a previous PEER response from the PCP server. On an initial PEER request, the External IP Address and Port are set to zero.

Note that the PREFER_FAILURE semantics are automatically implied by PEER requests. If the Suggested External IP Address or Suggested External Port fields are non-zero, and the PCP server is unable to honor the Suggested External IP Address or Port, then the PCP server MUST return a CANNOT_PROVIDE_EXTERNAL_PORT error response. The

PREFER_FAILURE Option is neither required nor allowed in PEER requests, and if PCP server receives a PEER request containing the PREFER_FAILURE Option it MUST return a MALFORMED_REQUEST error response.

The following diagram shows the response packet format for the PEER Opcode:

```
   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 (128 bits)            |
  |                                                               |
  |                                                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |       Remote Peer Port        |     Reserved (16 bits)        |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                                                               |
  |               Remote Peer IP Address (128 bits)               |
  |                                                               |
  |                                                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 13: PEER Opcode Response Packet Format

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.

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

Internal Port: Copied from request.

Assigned External Port: On a success response, this is the assigned external port for the mapping. On an error response, the Suggested External Port is copied from the request.

Assigned External IP Address: On a success response, this is the assigned external IPv4 or IPv6 address for the mapping; IPv4 or IPv6 address is indicated by the Opcode. On an error response, the Suggested External IP Address is copied from the request.

Remote Peer port: Copied from request.

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

Remote Peer IP Address: Copied from the request.

10.2. Generating a PEER Request

This section describes the operation of a client when generating a message with the PEER Opcode.

The PEER Opcode MAY be sent before or after establishing bi-directional communication with the remote peer.

> If sent before, this is considered a PEER-created mapping which creates a new dynamic mapping in the PCP-controlled device, which will be used for translating traffic to and from the remote peer; this mapping functions the same as if an implicit dynamic mapping were created (e.g., because of a TCP SYN from the client). This is useful for restoring a mapping after a NAT has lost its implicit mapping state (e.g., due to a crash). Note that some PCP servers and some PCP-controlled devices are expected to not support this functionality and will respond with a PCP error.
>
> If sent after, this is considered an "implicit dynamic mapping". This allows the client to learn the IP address, port, and lifetime of the assigned External Address and Port for the implicit mapping, and to extend this lifetime (for the purpose described in Section 8.3).

The PEER Opcode contains a Remote Peer Address field, which is always from the perspective of the PCP client. Note that when the PCP-controlled device is performing address family translation (NAT46 or NAT64), the remote peer address from the perspective of the PCP client is different from the remote peer address on the other side of the address family translation device.

10.3. Processing a PEER Request

This section describes the operation of a server when receiving a request with the PEER Opcode. Processing SHOULD be performed in the order of the following paragraphs.

The following fields from a PEER request are copied into the response: Protocol, Internal Port, Remote Peer IP Address, and Remote Peer Port.

When an implicit dynamic mapping is created, some NATs and firewalls validate destination addresses and will not create an implicit dynamic mapping if the destination address is invalid (e.g., 127.0.0.1). If a PCP-controlled device does such validation for implicit dynamic mappings, it SHOULD also do a similar validation of the Remote Peer IP Address and Port for PEER-created implicit dynamic mappings. If the validation determines the Remote Peer IP Address of a PEER request is invalid, then no mapping is created, and a MALFORMED_REQUEST error result is returned.

On receiving the PEER Opcode, the PCP server examines the mapping table. If the requested mapping does not yet exist, and the Suggested External Address and Port can be honored, the mapping 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 12.3.1). If the requested mapping does not yet exist, but Suggested External Address and Port cannot be honored, the error CANNOT_PROVIDE_EXTERNAL_PORT is returned. If the requested mapping already exists, it is a request to modify that existing mapping.

The PEER Opcode MAY reduce the lifetime of an existing implicit dynamic mapping created by PEER; 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 preceding 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 reported in the response, 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).

Some transport protocols signal the end of a connection (e.g., TCP FIN, TCP RST, SCTP SHUTDOWN). After a successful PEER response is sent, the receipt of such a transport-specific message MUST NOT cause the mapping to be destroyed. Rather, the mapping is maintained until the PEER-signaled lifetime expires. If the PCP client wishes to terminate the mapping prior to this, it will send a PEER request with Lifetime set to 0, which MAY be honored by the PCP server; as stated earlier, that is implementation-dependent.

10.4. Processing a PEER Response

This section describes the operation of a client when processing a response with the PEER Opcode.

After performing common PCP response processing, the response is further 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 mapping created by the Opcode.

On 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. See Section 9.2.1 for recommended renewal timing.

> Note: implementations need to expect the PEER response may contain an External IP Address with a different family than the Remote Peer IP Address, e.g., when NAT64 or NAT46 are being used.

11. Options for MAP and PEER Opcodes

This section describes Options for the MAP and PEER Opcodes. These Options MUST NOT appear with other Opcodes, unless permitted by those other Opcodes.

11.1. 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.

The THIRD_PARTY Option is formatted as follows:

```
    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 |  Reserved     |   Option Length=16 or 0       |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   |                Internal IP Address (128 bits)                 |
   |                                                               |
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 14: THIRD_PARTY Option packet format

The fields are described below:

Internal IP Address: Internal IP address for this mapping. If the Option Length is zero, there is no Internal IP address for this mapping and this indicates "all Internal IPv4 and IPv6 Addresses for which this client is authorized" which is used to delete all pre-existing mappings with the MAP Opcode.

Option Name: THIRD_PARTY
Number: 1
Purpose: Indicates the MAP or PEER request is for a host other than the host sending the PCP Option.
Valid for Opcodes: MAP, PEER
Length: 0 or 16 octets
May appear in: request. May appear in response only if it appeared in the associated request.
Maximum occurrences: 1

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.

A PCP server MAY be configured to permit or to prohibit the use of the THIRD_PARTY Option. If this Option is permitted, properly authorized clients may perform these operations on behalf of other hosts. If this Option is prohibited, and a PCP server receives a PCP MAP request with a THIRD_PARTY Option, it MUST generate a UNSUPP_OPTION response.

It is RECOMMENDED that customer premises equipment implementing a PCP Server be configured to prohibit 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 premises router (e.g., using its HTTP administrative interface).

It is RECOMMENDED that service provider NAT and firewall devices implementing a PCP Server be configured to permit the THIRD_PARTY Option, when sent by a properly authorized host. If the packet arrives from an unauthorized host, the PCP server MUST generate an UNSUPP_OPTION error.

Determining which PCP clients are authorized to use the THIRD_PARTY Option for which other hosts is deployment-dependent. For example, an ISP using Dual-Stack Lite could choose to allow a client connecting over a given IPv6 tunnel to manage mappings for any other host connecting over the same IPv6 tunnel, or the ISP could choose to allow only the DS-Lite B4 element to manage mappings for other hosts connecting over the same IPv6 tunnel. A cryptographic authentication and authorization model is outside the scope of this specification. Note that the THIRD_PARTY Option is not needed for today's common scenario of an ISP offering a single IP address to a customer who is using NAT to share that address locally, since in this scenario all the customer's hosts appear to be a single host from the point of view of the ISP.

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.

A PCP client can delete all PCP-created explicit dynamic mappings (i.e., those created by PCP MAP requests) that it is authorized to delete by sending a PCP MAP request including a zero-length THIRD_PARTY Option.

11.2. PREFER_FAILURE Option for MAP Opcode

This Option is only used with the MAP Opcode.

This Option indicates that if the PCP server is unable to map the Suggested External Port, the PCP server should not map an external port. This differs from the behavior without this Option, which is to map a different external port.

The PREFER_FAILURE Option is formatted as follows:

```
 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=2 |  Reserved     |   Option Length=0             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 15: PREFER_FAILURE Option packet format

Option Name: PREFER_FAILURE
Number: 2
Purpose: indicates that the PCP server should not create an alternative mapping if the suggested external port and address are not available.
Valid for Opcodes: MAP
Length: 0
May appear in: requests
Maximum occurrences: 1

The result code CANNOT_PROVIDE_EXTERNAL_PORT is returned if the Suggested 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 Address 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, rounded up to the next larger integer number of seconds.

This Option exists 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.

11.3. FILTER Option for MAP Opcode

This Option is only used with the MAP Opcode.

This Option indicates that 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 Option is formatted as follows:

```
  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=3 |  Reserved     |   Option Length=20            |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |    Reserved   | Prefix Length |      Remote Peer Port         |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                                                               |
 |               Remote Peer IP address (128 bits)               |
 |                                                               |
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 16: FILTER Option layout

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.

Option Name: FILTER
Number: 3
Purpose: specifies a filter for incoming packets
Valid for Opcodes: MAP
Length: 20 octets
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 IPv4 addresses, which are represented using the IPv4-mapped address format (::FFFF:0:0/96), the value of the Prefix Length pertains only to to the IPv4 portion of the address. Thus, a Prefix Length of 32 with an IPv4-mapped address indicates "only this address". With IPv4-mapped addresses, the minimum Prefix length value is 0 and the maximum is 32; for IPv6 addresses the minimum value is 0 and the maximum is 128. Values outside those range cause the PCP server to return the 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 (as per normal PCP Option processing) and they MAY overlap the filtering requested. 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.

If any of occurrences of the FILTER Option in a request packet are not successfully processed then an error is returned (e.g., MALFORMED_OPTION if one of the Options was malformed) and as with other PCP errors, returning an error causes no state to be changed in the PCP server or in the PCP-controlled device.

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 Peers.

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.

All PCP servers MUST support at least one filter per MAP mapping.

The use of the FILTER Option can be seen as a performance optimization. Since all software using PCP to receive incoming connections also has to deal with the case where may be directly connected to the Internet and receive unrestricted incoming TCP connections and UDP packets, if it wishes to restrict incoming traffic to a specific source address or group of source addresses such software already needs to check the source address of incoming traffic and reject unwanted traffic. However, the FILTER Option is a particularly useful performance optimization for battery powered wireless devices, because it can enable them to conserve battery power by not having to wake up just to reject a unwanted traffic.

12. Implementation Considerations

12.1. Implementing MAP with EDM port-mapping NAT

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

For implicit dynamic mappings, some existing NAT devices have endpoint-independent mapping (EIM) behavior while other NAT devices have endpoint-dependent mapping (EDM) behavior. NATs which have EIM behavior do not suffer from the problem described in this section. The IETF strongly encourages EIM behavior [RFC4787][RFC5382].

In such EDM NAT devices, the same external port may be used by an implicit dynamic mapping (from the same Internal Host or from a different Internal Host) and an explicit dynamic mapping. This complicates the interaction with the MAP Opcode. With such NAT devices, there are two ways envisioned to implement the MAP Opcode:

1. Have implicit dynamic mappings use a different set of public ports than explicit dynamic mappings (e.g., those created with

MAP), thus reducing the interaction problem between them; or

2. On arrival of a packet (inbound from the Internet or outbound 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.

12.2. Lifetime of Explicit and Implicit Dynamic Mappings

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

No matter if a NAT is EIM or EDM, 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 NAT 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 a NAT that implements endpoint-independent mapping with endpoint-independent filtering, this could be implemented by extending the lifetime of the implicit dynamic mapping to the lifetime of the explicit dynamic mapping.

12.3. PCP Failure Scenarios

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

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 expected to be 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 this information to 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 observed to be outside the expected range, the PCP client can attempt to recreate the mappings following the procedures described in this section.

Further analysis of PCP failure scenarios is in [I-D.boucadair-pcp-failure].

12.3.1. Recreating Mappings

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

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 a newly rebooted 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.

When the PCP server loses state and begins processing new PCP messages, its Epoch is reset and begins counting again from zero (per the procedure of Section 7.5). As the result of receiving a packet where the Epoch field is outside the expected range, indicating 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.

12.3.2. Maintaining Mappings

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

A PCP client refreshes 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 reconfiguration or 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 determine whether or not it appears the PCP server may have suffered a catastrophic loss of 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 a PCP client changes its Internal IP Address (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 new mappings on

that new network. New mappings will typically also require an update to the application-specific rendezvous server if the External Address or Port are different to the previous values (see Section 8.1 and Section 9.6).

12.3.3. SCTP

Although SCTP has port numbers like TCP and UDP, SCTP works differently when behind an address-sharing NAT, in that SCTP port numbers are not changed [I-D.ietf-behave-sctpnat]. Because implicit dynamic SCTP mappings use the verification tag of the association instead of the local and remote peer port numbers, explicit dynamic SCTP mappings need only be established by passive listeners expecting to receive new associations at the external port.

Because an SCTP-aware NAT does not rewrite SCTP port numbers (and firewalls never do), a PCP MAP or PEER request for an SCTP mapping SHOULD provide the same Internal Port and Requested External Port. If the PCP server supports SCTP, and the requested external port cannot be provided in an explicit dynamic SCTP mapping, then the error CANNOT_PROVIDE_EXTERNAL_PORT is returned.

12.4. Source Address and Port in PCP Header

All PCP requests include the PCP client's IP address in the PCP header. This is used to detect address rewriting (NAT) between the PCP client and its PCP server. On operating systems that support the sockets API, the following steps are RECOMMENDED for a PCP client to insert the correct source address and port to include in the PCP header:

1. Create a UDP socket.
2. Bind the UDP socket.
3. Call the getsockname() function to retrieve a sockaddr containing the source address and port the kernel will use for UDP packets sent through this socket.
4. If the IP address is an IPv4 address, encode the address into an IPv4-mapped IPv6 address. Place the IPv6 address (or IPv4-mapped IPv6 address) into the PCP Client's IP Address field in the PCP header.
5. Send PCP requests using this bound UDP socket.

13. Deployment Considerations

13.1. Ingress Filtering

As with implicit dynamic mappings created by outgoing TCP packets, explicit dynamic mappings created via PCP use the source IP address of the packet as the Internal Address for the mappings. Therefore ingress filtering [RFC2827] should be used on the path between the Internal Host and the PCP Server to prevent the injection of spoofed packets onto that path.

13.2. Mapping Quota

On PCP-controlled devices that create state when a mapping is created (e.g., NAT), the PCP server SHOULD maintain per-host and/or per-subscriber quotas for mappings. It is implementation-specific whether the PCP server uses a separate quotas for implicit, explicit, and static mappings, a combined quota for all of them, or some other policy.

14. Security Considerations

The goal of the PCP protocol is to improve the ability of end nodes to control their associated NAT state, and to improve the efficiency and error handling of NAT mappings when compared to existing implicit mapping mechanisms in NAT boxes and stateful firewalls. It is the security goal of the PCP protocol to limit any new denial of service opportunities, and to avoid introducing new attacks that can result in unauthorized changes to mapping state. One of the most serious consequences of unauthorized changes in mapping state is traffic theft. All mappings that could be created by a specific host using implicit mapping mechanisms are inherently considered to be authorized. Confidentiality of mappings is not a requirement, even in cases where the PCP messages may transit paths that would not be travelled by the mapped traffic.

14.1. Simple Threat Model

PCP is secure against off-path attackers who cannot spoof a packet that the PCP Server will view as a packet received from the internal network.

Defending against attackers who can modify or drop packets between the internal network and the PCP server, or who can inject spoofed packets that appear to come from the internal network is out-of-scope.

A PCP Server is secure under this threat model if the PCP Server is constrained so that it does not configure any explicit mapping that

it would not configure implicitly. In most cases, this means that PCP Servers running on NAT boxes or stateful firewalls that support the PEER Opcode can be secure under this threat model if all of their hosts are within a single administrative domain (or if the internal hosts can be securely partitioned into separate administrative domains, as in the DS-Lite B4 case), explicit mappings are created with the same lifetime as implicit mappings, the PCP server does not support deleting or reducing the lifetime of existing mappings, and the PCP server does not support the third party option. PCP Servers can also securely support the MAP Opcode under this threat model if the security policy on the device running the PCP Server would permit endpoint independent filtering of implicit mappings.

PCP Servers that comply with the Simple Threat Model and do not implement a PCP security mechanism described in Section 14.2 MUST enforce the constraints described in the paragraph above.

14.1.1. Attacks Considered

- If you allow multiple administrative domains to send PCP requests to a single PCP server that does not enforce a boundary between the domains, it is possible for a node in one domain to perform a denial of service attack on other domains, or to capture traffic that is intended for a node in another domain.

- If explicit mappings have longer lifetimes than implicit mappings, it makes it easier to perpetrate a denial of service attack than it would be if the PCP Server was not present.

- If the PCP Server supports deleting or reducing the lifetime of existing mappings, this allows an attacking node to steal an existing mapping and receive traffic that was intended for another node.

- If the THIRD_PARTY Option is supported, this also allows an attacker to open a window for an external node to attack an internal node, allows an attacker to steal traffic that was intended for another node, or may facilitate a denial of service attack. One example of how the THIRD_PARTY Option could grant an attacker more capability than a spoofed implicit mapping is that the PCP server (especially if it is running in a service provider's network) may not be aware of internal filtering that would prevent spoofing an equivalent implicit mapping, such as filtering between a guest and corporate network.

- If the MAP Opcode is supported by the PCP server in cases where the security policy would not support endpoint independent filtering of implicit mappings, then the MAP Opcode changes the

security properties of the device running the PCP Server by allowing explicit mappings that violate the security policy.

14.1.2. Deployment Examples Supporting the Simple Threat Model

This section offers two examples of how the Simple Threat Model can be supported in real-world deployment scenarios.

14.1.2.1. Residential Gateway Deployment

Parity with many currently-deployed residential gateways can be achieved using a PCP Server that is constrained as described in Section 14.1.1 above.

14.1.2.2. DS-Lite Deployment

A DS-Lite deployment could be secure under the Simple Threat Model, even if the B4 device makes PCP mapping requests on behalf of internal clients using the THIRD_PARTY option. In this case the DS-Lite PCP server MUST be configured to only allow the B4 device to make THIRD_PARTY requests, and only on behalf of other Internal Hosts sharing the same DS-Lite IPv6 tunnel. The B4 device MUST guard against spoofed packets being injected into the IPv6 tunnel using the B4 device's IPv4 source address, so the DS-Lite PCP Server can trust that packets received over the DS-Lite IPv6 tunnel with the B4 device's source IPv4 address do in fact originate from the B4 device. The B4 device is in a position to enforce this requirement, because it is the DS-Lite IPv6 tunnel endpoint.

Allowing the B4 device to use the THIRD_PARTY Option to create mappings for hosts reached via the IPv6 tunnel terminated by the B4 device is acceptable, because the B4 device is capable of creating these mappings implicitly and can prevent others from spoofing these mappings.

DS-Lite's security policies may also permit use of the MAP Opcode.

14.2. Advanced Threat Model

In the Advanced Threat Model the PCP protocol must be ensure that attackers (on- or off-path) cannot create unauthorized mappings or make unauthorized changes to existing mappings. The protocol must also limit the opportunity for on- or off-path attackers to perpetrate denial of service attacks.

The Advanced Threat Model security model will be needed in the following cases:

- Security infrastructure equipment, such as corporate firewalls, that does not create implicit mappings.

- Equipment (such as CGNs or service provider firewalls) that serve multiple administrative domains and do not have a mechanism to securely partition traffic from those domains.

- Any implementation that wants to be more permissive in authorizing explicit mappings than it is in authorizing implicit mappings.

- Implementations that support the THIRD_PARTY Option (unless they can meet the constraints outlined in Section 14.1.2.2).

- Implementations that wish to support any deployment scenario that does not meet the constraints described in Section 14.1.

To protect against attacks under this threat model, a PCP security mechanism which provides an authenticated, integrity protected signaling channel would need to be specified.

PCP Servers that implement a PCP security mechanism MAY accept unauthenticated requests. PCP Servers implementing the PCP security mechanism MUST enforce the constraints described in Section 14.1 above, in their default configuration, when processing unauthenticated requests.

14.3. Residual Threats

This section describes some threats that are not addressed in either of the above threat models, and recommends appropriate mitigation strategies.

14.3.1. Denial of Service

Because of the state created in a NAT or firewall, a per-host and/or per-subscriber quota will likely exist for both implicit dynamic mappings and explicit dynamic mappings. A host might make an excessive number of implicit or explicit dynamic mappings, consuming an inordinate number of ports, causing a denial of service to other hosts. Thus, Section 13.2 recommends that hosts be limited to a reasonable number of explicit dynamic mappings.

An attacker, on the path between the PCP client and PCP server, can drop PCP requests, drop PCP responses, or spoof a PCP error, all of which will effective deny service. Through such actions, the PCP client would not be aware the PCP server might have actually processed the PCP request.

14.3.2. Ingress Filtering

It is important to prevent a host from fraudulently creating, deleting, or refreshing a mapping (or filtering) for another host, because this can expose the other host to unwanted traffic, prevent it from receiving wanted traffic, or consume the other host's mapping quota. Both implicit and explicit dynamic mappings are created based on the source IP address in the packet, and hence depend on ingress filtering to guard against spoof source IP addresses.

14.3.3. Mapping Theft

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 rapid recovery mechanism to immediately inform the PCP client of state loss would reduce this interval, but would not completely 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 reduced 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).

14.3.4. Attacks Against Server Discovery

This document does not specify server discovery, beyond contacting the default gateway.

15. IANA Considerations

IANA is requested to perform the following actions:

15.1. Port Number

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

[Note to RFC Editor: Please remove the text about relinquishing port 44323 prior to publication.]

15.2. Opcodes

IANA shall create a new protocol registry for PCP Opcodes, numbered 0-127, initially populated with the values:

value	Opcode
0	Reserved for "no-op" operation code
1	MAP
2	PEER
3-95	(specification required)
96-126	(private use)
127	Reserved

The values 0 and 127 are Reserved and may be assigned via Standards Action [RFC5226]. The values in the range 3-95 can be assigned via Specification Required [RFC5226], and the range 96-126 is for Private Use [RFC5226].

15.3. Result Codes

IANA shall create a new registry for PCP result codes, numbered 0-255, initially populated with the result codes from Section 6.4. The value 255 is Reserved and may be assigned via Standards Action [RFC5226].

Result Codes in the range 13-191 can be assigned via Specification Required [RFC5226], and the range 192-254 is for Private Use [RFC5226].

15.4. 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 7.7.1 and Section 11. The Option values 127 and 255 are Reserved and may be assigned via Standards Action [RFC5226].

Additional PCP Option codes in the ranges 4-63 and 128-191 can be created via Specification Required [RFC5226], and the ranges 64-126 and 192-254 are for Private Use [RFC5226].

16. Acknowledgments

Thanks to Xiaohong Deng, Alain Durand, Christian Jacquenet, Jacni Qin, Simon Perreault, James Yu, Tina TSOU (Ting ZOU), Felipe Miranda

Costa, and James Woodyatt 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.

Thanks to Margaret Wasserman for writing the Security Considerations section.

17. References

17.1. Normative References

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2136] Vixie, P., Thomson, S., Rekhter, Y., and J. Bound, "Dynamic Updates in the Domain Name System (DNS UPDATE)", RFC 2136, April 1997.

[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.

[RFC3007] Wellington, B., "Secure Domain Name System (DNS) Dynamic Update", RFC 3007, November 2000.

[RFC4193] Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses", RFC 4193, October 2005.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC6056] Larsen, M. and F. Gont, "Recommendations for Transport-Protocol Port Randomization", BCP 156, RFC 6056, January 2011.

[proto_numbers]
IANA, "Protocol Numbers", 2011, <http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xml>.

17.2. Informative References

[Bonjour] "Bonjour", <http://en.wikipedia.org/wiki/Bonjour_(software)>.

[I-D.arkko-dual-stack-extra-lite]
Arkko, J., Eggert, L., and M. Townsley, "Scalable Operation of Address Translators with Per-Interface Bindings", draft-arkko-dual-stack-extra-lite-05 (work in progress), February 2011.

[I-D.boucadair-pcp-failure]
Boucadair, M., Dupont, F., and R. Penno, "Port Control Protocol (PCP) Failure Scenarios", draft-boucadair-pcp-failure-02 (work in progress), September 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", draft-bpw-pcp-upnp-igd-interworking-02 (work in progress), February 2011.

[I-D.cheshire-dnsext-dns-sd]
Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", draft-cheshire-dnsext-dns-sd-10 (work in progress), February 2011.

[I-D.cheshire-nat-pmp]
Cheshire, S., "NAT Port Mapping Protocol (NAT-PMP)", draft-cheshire-nat-pmp-03 (work in progress), April 2008.

[I-D.dupont-pcp-dslite]
Dupont, F., Tsou, T., and J. Qin, "The Port Control Protocol in Dual-Stack Lite environments", draft-dupont-pcp-dslite-00 (work in progress), August 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)", draft-ietf-behave-lsn-requirements-03 (work in progress), August 2011.

[I-D.ietf-behave-sctpnat]
Stewart, R., Tuexen, M., and I. Ruengeler, "Stream Control Transmission Protocol (SCTP) Network Address Translation",

draft-ietf-behave-sctpnat-05 (work in progress), June 2011.

[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", November 2001, <http://upnp.org/specs/gw/UPnP-gw-WANIPConnection-v1-Service.pdf>.

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

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, February 1996.

[RFC3022] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

[RFC3581] Rosenberg, J. and H. Schulzrinne, "An Extension to the Session Initiation Protocol (SIP) for Symmetric Response Routing", RFC 3581, August 2003.

[RFC3587] Hinden, R., Deering, S., and E. Nordmark, "IPv6 Global Unicast Address Format", RFC 3587, August 2003.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC4787] Audet, F. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", BCP 127, RFC 4787, January 2007.

[RFC4941] Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6", RFC 4941, September 2007.

[RFC4961] Wing, D., "Symmetric RTP / RTP Control Protocol (RTCP)", BCP 131, RFC 4961, July 2007.

[RFC5382] Guha, S., Biswas, K., Ford, B., Sivakumar, S., and P. Srisuresh, "NAT Behavioral Requirements for TCP", BCP 142, RFC 5382, October 2008.

[RFC6092] Woodyatt, J., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, January 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.

[RFC6296] Wasserman, M. and F. Baker, "IPv6-to-IPv6 Network Prefix Translation", RFC 6296, June 2011.

[RFC6333] Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", RFC 6333, August 2011.

Appendix A. NAT-PMP Transition

The Port Control Protocol (PCP) is a successor to the NAT Port Mapping Protocol, NAT-PMP [I-D.cheshire-nat-pmp], and shares similar semantics, concepts, and packet formats. Because of this NAT-PMP and PCP both use the same port, and use 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 the NAT-PMP specification [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 handle requests in either format. The first octet of the packet indicates if it is NAT-PMP (first octet zero) or PCP (first octet non-zero).

A PCP-only gateway receiving a NAT-PMP request (identified by the first octet 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.]

B.1. Changes from draft-ietf-pcp-base-15 to -16

- fixed mistake in PCP request format (had 32 bits of extraneous fields)

- Allow MAP to request all ports (port=0) for a specific protocol (protocol!=0), for the same reason we added support for all ports (port=0) and all protocols (protocol=0) in -15

- corrected text on Client Processing a Response related to receiving ADDRESS_MISMATCH error.

- updated Epoch text.

- Added text that MALFORMED_REQUEST is generated for MAP if Protocol is zero but Internal Port is non-zero.

B.2. Changes from draft-ietf-pcp-base-14 to -15

- Softened and removed text that was normatively explaining how PEER is implemented within a NAT.

- Allow a MAP request for protocol=0, which means "all protocols". This can work for an IPv6 or IPv4 firewall. Its use with a NAPT is undefined.

- combined SERVER_OVERLOADED and NO_RESOURCES into one error code, NO_RESOURCES.

- SCTP mappings have to use same internal and requested external ports, and have implied PREFER_FAILURE semantics.

- Re-instated ADDRESS_MISMATCH error, which only checks the client address (not its port).

B.3. Changes from draft-ietf-pcp-base-13 to -14

- Moved discussion of socket operations for PCP source address into Implementation Considerations section.

- Integrated numerous WGLC comments.

- NPTv6 in scope.

- Re-written security considerations section. Thanks, Margaret!

- Reduced PEER4 and PEER6 Opcodes to just a single Opcode, PEER.

- Reduced MAP4 and MAP6 Opcodes to just a single Opcode, MAP.

- Rearranged the PEER packet formats to align with MAP.

- Removed discussion of the "O" bit for Options, which was confusing. Now the text just discusses the most significant bit of the Option code which indicates mandatory/optional, so it is clearer the field is 8 bits.

- The THIRD_PARTY Option from an unauthorized host generates UNSUPP_OPTION, so the PCP server doesn't disclose it knows how to process THIRD_PARTY Option.

- Added table to show which fields of MAP or PEER need IPv6/IPv4 addresses for IPv4 firewall, DS-Lite, NAT64, NAT44, etc.

- Accommodate the server's Epoch going up or down, to better detect switching to a different PCP server.

- Removed ADDRESS_MISMATCH; the server always includes its idea of the Client's IP Address and Port, and it's up to the client to detect a mismatch (and rectify it).

B.4. Changes from draft-ietf-pcp-base-12 to -13

- All addresses are 128 bits. IPv4 addresses are represented by IPv4-mapped IPv6 addresses (::FFFF/96)

- PCP request header now includes PCP client's port (in addition to the client's IP address, which was in -12).

- new ADDRESS_MISMATCH error.

- removed PROCESSING_ERROR error, which was too similar to MALFORMED_REQUEST.

- Tweaked text describing how PCP client deals with multiple PCP server addresses (Section 7.1)

- clarified that when overloaded, the server can send SERVER_OVERLOADED (and drop requests) or simply drop requests.

- Clarified how PCP client chooses MAP4 or MAP6, depending on the presence of its own IPv6 or IPv4 interfaces (Section 8).

- compliant PCP server MUST support MAPx and PEERx, SHOULD support ability to disable support.

- clarified that MAP-created mappings have no filtering, and PEER-created mappings have whatever filtering and mapping behavior is normal for that particular NAT / firewall.

- Integrated WGLC feedback (small changes to abstract, definitions, and small edits throughout the document)

- allow new Options to be defined with a specification (rather than standards action)

B.5. Changes from draft-ietf-pcp-base-11 to -12

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

B.6. Changes from draft-ietf-pcp-base-10 to -11

- clarified what can cause CANNOT_PROVIDE_EXTERNAL_PORT error to be generated.

B.7. 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.

B.8. Changes from draft-ietf-pcp-base-08 to -09

- Clarified in PEER Opcode introduction (Section 10) that they can also create 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 12.3.1, as it didn't belong there. Text in Security Considerations 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 8.4).

- 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 12) which discusses non-normative things that might be useful to implementors. Some new text is in here, and the Failure Scenarios text (Section 12.3) has been moved to here.

- Tweaked wording of EDM NATs in Section 12.1 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 9.5.

- 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.

B.9. Changes from draft-ietf-pcp-base-07 to -08

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

- discussed NAT port-overloading and its impact on MAP (new section Section 12.1), 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.

B.10. 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.

B.11. 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.

B.12. 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].

B.13. 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.

B.14. 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.

B.15. 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.

B.16. 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)
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134
USA

Email: dwing@cisco.com

Stuart Cheshire
Apple Inc.
1 Infinite Loop
Cupertino, California 95014
USA

Phone: +1 408 974 3207
Email: cheshire@apple.com

Mohamed Boucadair
France Telecom
Rennes,   35000
France

Email: mohamed.boucadair@orange-ftgroup.com

Reinaldo Penno
Juniper Networks
1194 N Mathilda Avenue
Sunnyvale, California 94089
USA

Email: rpenno@juniper.net

Paul Selkirk
Internet Systems Consortium
950 Charter Street
Redwood City, California 94063
USA

Email: pselkirk@isc.org