Internet Engineering Task Force (IETF)
Request for Comments: 6887
Category: Standards Track
ISSN: 2070-1721

D. Wing, Ed.
Cisco
S. Cheshire
Apple
M. Boucadair
France Telecom
R. Penno
Cisco
P. Selkirk
ISC

April 2013

Port Control Protocol (PCP)

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

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc6887.

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 Network Address Translator IPv6/IPv4 (NAT64), Network Address Translator IPv4/IPv4 (NAT44), and 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) and small NATs (e.g., residential NATs), as well as with dual-stack and IPv6-only Customer Premises Equipment (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 network-attached home security camera) 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, protocol, and port to an internal IP address, protocol, 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, protocol, 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 [RFC6763] 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 that 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 [RFC6888]
- Dual-Stack Lite (DS-Lite) [RFC6333]
- NAT that is Layer-2 Aware [L2NAT]
- Dual-Stack Extra Lite [RFC6619]
- NAT64, both Stateless [RFC6145] and Stateful [RFC6146]
- IPv4 and IPv6 simple firewall control [RFC6092]
- IPv6-to-IPv6 Network Prefix Translation (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, Stream Control Transmission Protocol (SCTP) [RFC4960], and Datagram Congestion Control Protocol (DCCP) [RFC4340]). Protocols that do not use a port number (e.g., Resource Reservation Protocol (RSVP), IP Encapsulating Security Payload (ESP) [RFC4303], ICMP, and 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 other hosts on the Internet from that source IP address. This is important because after a PCP mapping is created and an inbound packet (e.g., TCP SYN) is rewritten and delivered to a host, the outbound response

(e.g., TCP SYNACK) has to go through the same (reverse) path so it passes through the same NAT to have the necessary inverse rewrite performed. 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 already have the necessary external port mapped to another host).

3. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "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 will receive incoming traffic resulting from a PCP mapping request, or the host that initiated an implicit dynamic outbound mapping (e.g., by sending a TCP SYN) across a firewall or a NAT.

Remote Peer 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 [RFC4787].

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, protocol, and port to different remote peers and ports may be assigned different external ports, and a subsequent PCP mapping request for that

internal address, protocol, and port may be assigned yet another different external port. This term encompasses both Address-Dependent Mapping and Address and Port-Dependent Mapping [RFC4787].

Endpoint-Independent Mapping (EIM): A term applied to NAT operation where all mappings from a single internal address, protocol, and port to different remote peers and ports are all assigned the same external address and port.

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 address, protocol, and port have their destination address and port translated to the internal address and port, and conversely, packets *from* the internal IP address, protocol, and port have their source address and port translated to the external address and port. In the case of a pure firewall, the "mapping" is the identity function, translating an internal IP address, protocol, and port number to the same external IP address, protocol, and port number. Firewall filtering, applied in addition to that identity mapping function, is separate from the mapping itself.

Mapping Types:

There are three dimensions to classifying mapping types: how they are created (implicitly/explicitly), their primary purpose (outbound/inbound), and how they are deleted (dynamic/static). Implicit mappings are created as a side effect of some other operation; explicit mappings are created by a mechanism explicitly dealing with mappings. Outbound mappings exist primarily to facilitate outbound communication; inbound mappings exist primarily to facilitate inbound communication. Dynamic mappings are deleted when their lifetime expires, or through other protocol action; static mappings are permanent until the user chooses to delete them.

* Implicit dynamic mappings are created implicitly as a side effect of traffic such as an outgoing TCP SYN or outgoing UDP packet. Such packets were not originally designed explicitly for creating NAT (or firewall) state, but they can have that effect when they pass through a NAT (or firewall) device. Implicit dynamic mappings usually have a finite lifetime, though this lifetime is generally not known to the client using them.

* Explicit dynamic mappings are created as a result of explicit PCP MAP and PEER requests. Like a DHCP address lease, explicit dynamic mappings have a finite lifetime, and this lifetime is communicated to the client. As with a DHCP address lease, if the client wants a mapping to persist the client must prove that it is still present by periodically renewing the mapping to prevent it from expiring. If a PCP client goes away, then any mappings it created will be automatically cleaned up when they expire.

* Explicit static mappings are created by manual configuration (e.g., via command-line interface or other user interface) and persist until the user changes that manual configuration.

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 another PCP request).

Static mappings are, by their nature, always explicit. Static mappings differ from explicit dynamic mappings in that their lifetime is effectively infinite (they exist until manually removed), but otherwise they behave exactly the same as explicit MAP mappings.

While all mappings are, by necessity, bidirectional (most Internet communication requires information to flow in both directions for successful operation), when talking about mappings, it can be helpful to identify them loosely according to their 'primary' purpose.

* Outbound mappings exist primarily to enable outbound communication. For example, when a host calls connect() to make an outbound connection, a NAT gateway will create an implicit dynamic outbound mapping to facilitate that outbound communication.

* Inbound mappings exist primarily to enable listening servers to receive inbound connections. Generally, when a client calls listen() to listen for inbound connections, a NAT gateway will not implicitly create any mapping to facilitate that inbound communication. A PCP MAP request can be used explicitly to create a dynamic inbound mapping to enable the desired inbound communication.

Explicit static (manual) mappings and explicit dynamic (MAP) mappings both 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).

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. 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 IGDv1 [IGDv1]) 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 peer hosts. PCP manages the mappings on this device.

PCP Server:
A PCP software instance that resides on the PCP-Controlled Device that receives PCP requests from the PCP client and creates appropriate state in response to that request.

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 Carrier-Grade NAT (CGN) operated by the ISP.

4. Relationship between PCP Server and Its PCP-Controlled Device

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(s) or firewall(s) via some other proprietary mechanism, as long as from the PCP client's perspective such split operation is indistinguishable from the integrated case.

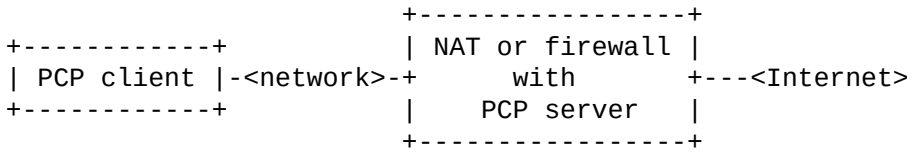

Figure 1: PCP-Enabled NAT or Firewall

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 IP ESP) [RFC6092]. Such default filtering (or lack thereof) is out of scope of PCP itself. If a client device wants to receive traffic and supports PCP, and does not possess prior knowledge of such default filtering policy, it SHOULD use PCP to request 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, an IPv4-mapped IPv6 address [RFC4291] is 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 native IPv6 address, because an IPv4-mapped IPv6 address [RFC4291] would not be valid for a mapping.

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 ones in bits 81-96.

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

The all-zeros IPv4 address MUST be expressed by 80 bits of zeros, 16 bits of ones, and 32 bits of zeros (::ffff:0:0).

6. Protocol Design Note

PCP can be viewed as a request/response protocol, much like many other UDP-based request/response protocols, and can be implemented perfectly well as such. It can also be viewed as what might be called a hint/notification protocol, and this observation can help simplify implementations.

Rather than viewing the message streams between PCP client and PCP server as following a strict request/response pattern, where every response is associated with exactly one request, the message flows can be viewed as two somewhat independent streams carrying information in opposite directions:

- A stream of hints flowing from PCP client to PCP server, where the client indicates to the server what it would like the state of its mappings to be, and

- A stream of notifications flowing from PCP server to PCP client, where the server informs the clients what the state of its mappings actually is.

To an extent, some of this approach is required anyway in a UDP-based request/response protocol, since UDP packets can be lost, duplicated, or reordered.

In this view of the protocol, the client transmits hints to the server at various intervals signaling its desires, and the server transmits notifications to the client signaling the actual state of its mappings. These two message flows are loosely correlated in that a client request (hint) usually elicits a server response (notification), but only loosely, in that a client request may result in no server response (in the case of packet loss), and a server response may be generated gratuitously without an immediately preceding client request (in the case where server configuration change, e.g., change of external IP address on a NAT gateway, results in a change of mapping state).

The exact times that client requests are sent are influenced by a client timing state machine taking into account whether (i) the client has not yet received a response from the server for a prior request (retransmission), or (ii) the client has previously received a response from the server saying how long the indicated mapping would remain active (renewal). This design philosophy is the reason why PCP's retransmissions and renewals are exactly the same packet on the wire. Typically, retransmissions are sent with exponentially increasing intervals as the client waits for the server to respond, whereas renewals are sent with exponentially decreasing intervals as the expiry time approaches, but, from the server's point of view, both packets are identical, and both signal the client's desire that the stated mapping exist or continue to exist.

A PCP server usually sends responses as a direct result of client requests, but not always. For example, if a server is too overloaded to respond, it is allowed to silently ignore a request message and let the client retransmit. Also, if external factors cause a NAT gateway or firewall's configuration to change, then the PCP server can send unsolicited responses to clients informing them of the new state of their mappings. Such reconfigurations are expected to be rare, because of the disruption they can cause to clients, but should they happen, PCP provides a way for servers to communicate the new state to clients promptly, without having to wait for the next periodic renewal request.

This design goal helps explain why PCP request and response messages have no transaction ID, because such a transaction ID is unnecessary, and would unnecessarily limit the protocol and unnecessarily complicate implementations. A PCP server response (i.e., notification) is self-describing and complete. It communicates the internal and external addresses, protocol, and ports for a mapping, and its remaining lifetime. If the client does in fact currently want such a mapping to exist, then it can identify the mapping in question from the internal address, protocol, and port, and update its state to reflect the current external address and port, and

remaining lifetime. If a client does not currently want such a mapping to exist, then it can safely ignore the message. No client action is required for unexpected mapping notifications. In today's world, a NAT gateway can have a static mapping, and the client device has no explicit knowledge of this, and no way to change the fact. Also, in today's world, a client device can be connected directly to the public Internet, with a globally routable IP address, and, in this case, it effectively has "mappings" for all of its listening ports. Such a device has to be responsible for its own security and cannot rely on assuming that some other network device will be blocking all incoming packets.

7. Common Request and Response Header Format

All PCP messages are sent over UDP, with a maximum UDP payload length of 1100 octets. The PCP messages contain a request or response header containing an Opcode, any relevant Opcode-specific information, and zero or more options. All numeric quantities larger than a single octet (e.g., result codes, lifetimes, Epoch times, etc.) are represented in conventional IETF network order, i.e., most significant octet first. Non-numeric quantities are represented as is on all platforms, with no byte swapping (e.g., IP addresses and ports are placed in PCP messages using the same representation as when placed in IP or TCP headers).

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 Sections 10, 11, and 12.

7.1. Request Header

All requests have the following format:

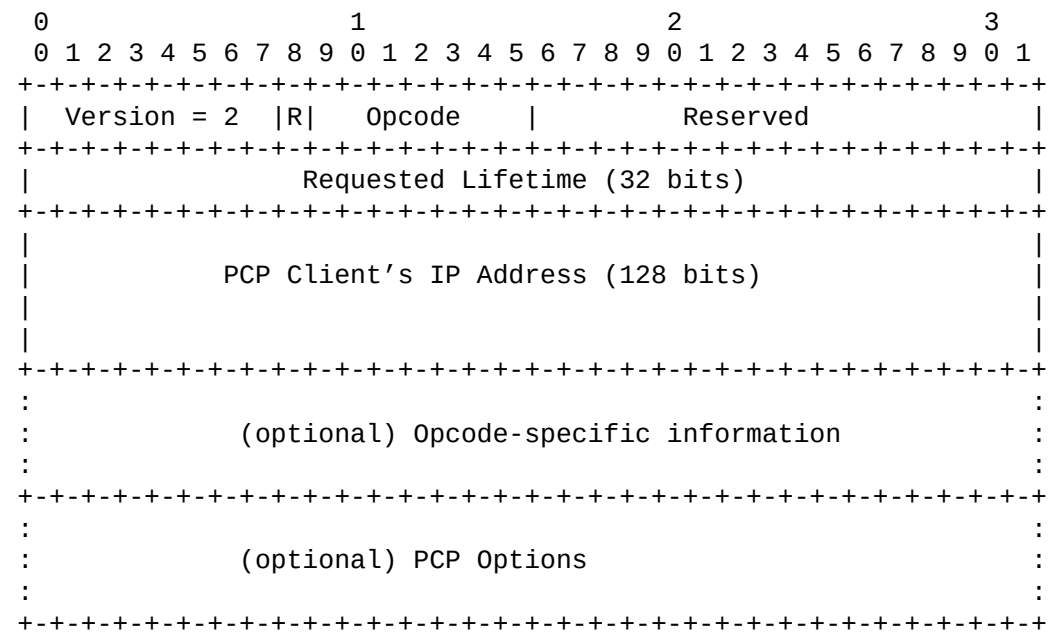

Figure 2: Common Request Packet Format

These fields are described below:

Version: This document specifies protocol version 2. PCP clients and servers compliant with this document use the value 2. This field is used for version negotiation as described in Section 9.

R: Indicates Request (0) or Response (1).

Opcode: A 7-bit value specifying the operation to be performed. MAP and PEER Opcodes are defined in Sections 11 and 12.

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

Requested Lifetime: An unsigned 32-bit integer, in seconds, ranging from 0 to 2^32-1 seconds. This is used by the MAP and PEER Opcodes defined in this document for their requested lifetime.

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. An IPv4 address is represented using an IPv4-mapped IPv6 address. The PCP Client IP Address in the PCP message header is used to detect an unexpected NAT on the path between the PCP client and the PCP-controlled NAT or firewall device. See Section 8.1.

Opcode-specific information: Payload data for this Opcode. The length of this data is determined by the Opcode definition.

PCP Options: Zero, one, or more options that are legal for both a PCP request and for this Opcode. See Section 7.3.

7.2. Response Header

All responses have the following format:

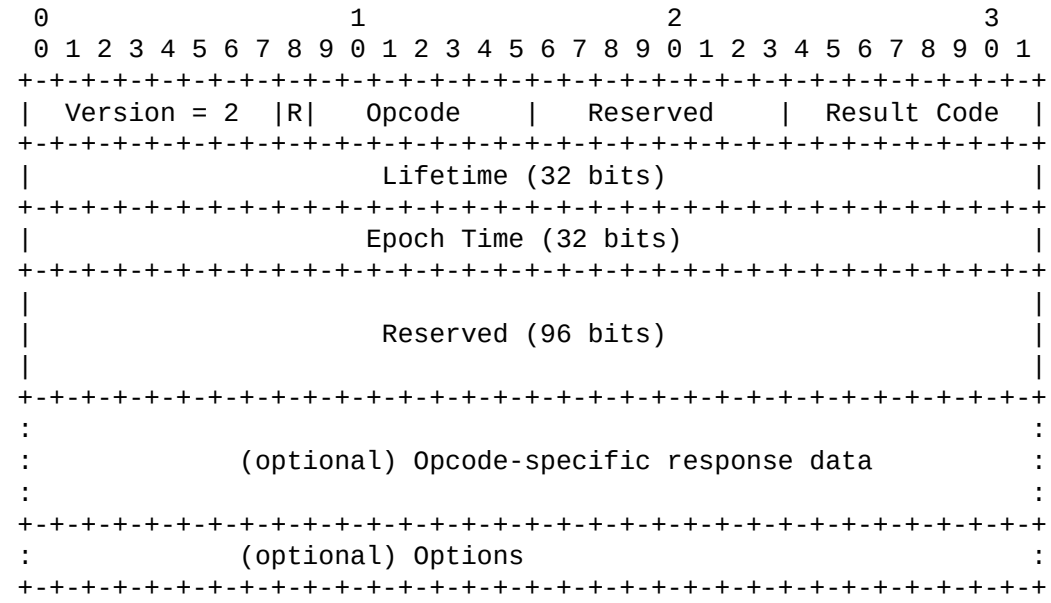

Figure 3: Common Response Packet Format

These fields are described below:

Version: Responses from servers compliant with this specification MUST use version 2. This is set by the server.

R: Indicates Request (0) or Response (1). All Responses MUST use 1. This is set by the server.

Opcode: The 7-bit Opcode value. The server copies this value 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 7.4 for values. This is set by the server.

Lifetime: An unsigned 32-bit integer, in seconds, ranging from 0 to 2^32-1 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 PCP Opcodes that create a mapping (MAP and PEER), the Lifetime field indicates the lifetime for this mapping. This is set by the server.

Epoch Time: The server's Epoch Time value. See Section 8.5 for discussion. This value is set by the server, in both success and error responses.

Reserved: 96 reserved bits. For requests that were successfully parsed, this MUST be sent as 0, MUST be ignored when received. This is set by the server. For requests that were not successfully parsed, the server copies the last 96 bits of the PCP Client's IP Address field from the request message into this corresponding 96-bit field of the response.

Opcode-specific information: Payload data for this Opcode. The length of this data is determined by the Opcode definition.

PCP Options: Zero, one, or more options that are legal for both a PCP response and for this Opcode. See Section 7.3.

7.3. 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 zeros 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:

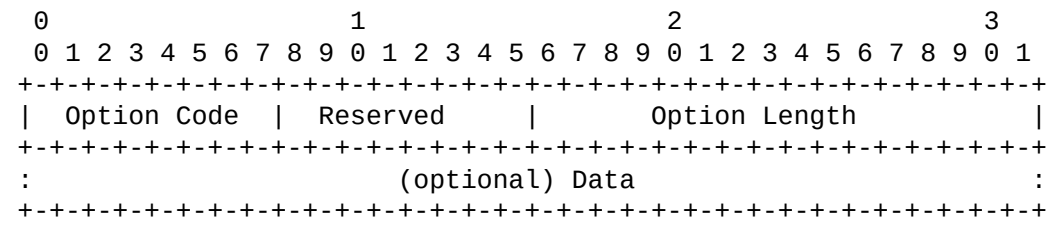

Figure 4: Options Header

The description of the fields is as follows:

Option Code: 8 bits. Its most significant bit 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 4 octets long are followed by one, two, or three 0 octets to pad their effective length in the packet to be a multiple of 4 octets. The Option Length reflects the semantic length of the option, not including any padding octets.

Data: Option data.

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 that the server has sufficient room to reply without exceeding the 1100-octet size limit; if its reply would exceed that size, the server generates an error.

If, while processing a PCP request, including its options, 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 tentative changes it might have made while processing the request). Such an error response MUST consist of a complete copy of the request packet with the error code and other appropriate fields set in the header.

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 the PCP server encounters an invalid option (e.g., PCP option length is longer than the UDP packet length), the error MALFORMED_OPTION SHOULD be returned (rather than MALFORMED_REQUEST), as that helps the client better understand how the packet was malformed. If a PCP response would have exceeded the maximum PCP message size, the PCP server SHOULD respond with MALFORMED_REQUEST.

If the overall option structure of a request cannot successfully be parsed (e.g., a nonsensical option length), the PCP server MUST generate an error response with code MALFORMED_OPTION.

If the overall option structure of a request is valid, then how each individual option is handled is determined by the most significant bit in the option code. 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 return the error MALFORMED_OPTION if the option contents are malformed, or UNSUPP_OPTION if the option is unrecognized, unimplemented, or disabled, or if the client is not authorized to use the option. In error responses, all options are returned. In success responses, all processed options are included and unprocessed options are not included.

Because the PCP client cannot reject a response containing an Option, PCP clients MUST ignore Options that they do not understand that appear in responses, including Options in the mandatory-to-process range. Naturally, if a client explicitly requests an Option where correct execution of that Option requires processing the Option data in the response, that client SHOULD implement code to do that.

Different options are valid for different Opcodes. For example:

- The THIRD_PARTY option is valid for both MAP and PEER Opcodes.
- The FILTER option is valid only for the MAP Opcode (for the PEER Opcode it would have no meaning).
- The PREFER_FAILURE option is valid only for the MAP Opcode (for the PEER Opcode, similar semantics are automatically implied).

7.4. 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: The version number at the start of the PCP Request header is not recognized by this PCP server. This is a long lifetime error. This document describes PCP version 2.

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. This is a long lifetime error.

4 UNSUPP_OPCODE: Unsupported Opcode. This is a long lifetime error.

5 UNSUPP_OPTION: Unsupported option. This error only occurs if the option is in the mandatory-to-process range. This is a long lifetime error.

6 MALFORMED_OPTION: Malformed option (e.g., appears too many times, invalid length). This is a long lifetime error.

7 NETWORK_FAILURE: The PCP server or the device it controls is experiencing a network failure of some sort (e.g., has not yet 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 is unable to handle the request due to some other temporary condition. The same request may succeed in the future. This is a system-wide error, different from USER_EX_QUOTA. This can be used as a catch-all error, should no other error message be suitable. This is a short lifetime error.

9 UNSUPP_PROTOCOL: Unsupported transport protocol, e.g., SCTP in a NAT that handles only UDP and TCP. This is a long lifetime error.

10 USER_EX_QUOTA: This attempt to create a new mapping would exceed this subscriber's port quota. This is a short lifetime error.

11 CANNOT_PROVIDE_EXTERNAL: The suggested external port and/or external address cannot be provided. This error MUST only be returned for:
* MAP requests that included the PREFER_FAILURE option (normal MAP requests will return an available external port)
* MAP requests for the SCTP protocol (PREFER_FAILURE is implied)
* PEER requests

See Section 13.2 for details of the PREFER_FAILURE Option. The error lifetime depends on the reason for the failure.

12 ADDRESS_MISMATCH: The source IP address of the request packet does not match the contents of the PCP Client's IP Address field, due to an unexpected NAT on the path between the PCP client and the PCP-controlled NAT or firewall. This is a long lifetime error.

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 13.3 for details of the FILTER Option. This is a long lifetime error.

8. General PCP Operation

PCP messages MUST be sent over UDP [RFC0768]. Every PCP request generates at least one response, so PCP does not need to run over a reliable transport protocol.

When receiving multiple identical requests, the PCP server will generally generate identical responses -- barring cases where the PCP server's state changes between those requests due to other activity. As an example of how such a state change could happen, a request could be received while the PCP-controlled device has no mappings

available, and the PCP server will generate an error response. If mappings become available and then another copy of that same request arrives (perhaps duplicated in transit in the network), the PCP server will allocate a mapping and generate a non-error response. A PCP client MUST handle such updated responses for any request it sends, most notably to support rapid recovery (Section 14). Also see the Protocol Design Note (Section 6).

8.1. 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 11, and procedures specific to the PEER Opcode are described in Section 12.

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). Thus, if a PCP client has both an IPv4 and IPv6 address, it will have an IPv4 PCP server (its IPv4 default router) for its IPv4 mappings, and an IPv6 PCP server (its IPv6 default router) for its IPv6 mappings.

For the purposes of this document, only a single PCP server address is supported. Should future specifications define configuration methods that provide a longer 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 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].

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 16.4 for how this can be coded. This is used to detect an unexpected NAT on the path between the PCP client and the PCP-controlled NAT or firewall device, to avoid wasting resources on

the PCP-controlled NAT creating pointless non-functional mappings. When such an intervening non-PCP-aware inner NAT is detected, mappings must first be created by some other means in the inner NAT, before mappings can be usefully created in the outer PCP-controlled NAT. Having created mappings in the inner NAT by some other means, the PCP client should then use the inner NAT's external address as the client IP address, to signal to the outer PCP-controlled NAT that the client is aware of the inner NAT, and has taken steps to create mappings in it by some other means, so that mappings created in the outer NAT will not be a pointless waste of resources.

8.1.1. PCP Client Retransmission

PCP clients are responsible for reliable delivery of PCP request messages. If a PCP client fails to receive an expected response from a server, the client must retransmit its message. The retransmissions MUST use the same Mapping Nonce value (see Sections 11.1 and 12.1). The client begins the message exchange by transmitting a message to the server. The message exchange continues for as long as the client wishes to maintain the mapping, and terminates when the PCP client is no longer interested in the PCP transaction (e.g., the application that requested the mapping is no longer interested in the mapping) or (optionally) when the message exchange is considered to have failed according to the retransmission mechanism described below.

The client retransmission behavior is controlled and described by the following variables:

RT: Retransmission timeout, calculated as described below

IRT: Initial retransmission time, SHOULD be 3 seconds

MRC: Maximum retransmission count, SHOULD be 0 (0 indicates no maximum)

MRT: Maximum retransmission time, SHOULD be 1024 seconds

MRD: Maximum retransmission duration, SHOULD be 0 (0 indicates no maximum)

RAND: Randomization factor, calculated as described below

With each message transmission or retransmission, the client sets RT according to the rules given below. If RT expires before a response is received, the client retransmits the request and computes a new RT.

Each of the computations of a new RT include a new randomization factor (RAND), which is a random number chosen with a uniform distribution between -0.1 and +0.1. The randomization factor is included to minimize synchronization of messages transmitted by PCP clients. The algorithm for choosing a random number does not need to be cryptographically sound. The algorithm SHOULD produce a different sequence of random numbers from each invocation of the PCP client.

The RT value is initialized based on IRT:

```
RT = (1 + RAND) * IRT
```

RT for each subsequent message transmission is based on the previous value of RT, subject to the upper bound on the value of RT specified by MRT. If MRT has a value of 0, there is no upper limit on the value of RT, and MRT is treated as "infinity". The new value of RT is calculated as shown below, where RTprev is the current value of RT:

```
RT = (1 + RAND) * MIN (2 * RTprev, MRT)
```

MRC specifies an upper bound on the number of times a client may retransmit a message. Unless MRC is zero, the message exchange fails once the client has transmitted the message MRC times.

MRD specifies an upper bound on the length of time a client may retransmit a message. Unless MRD is zero, the message exchange fails once MRD seconds have elapsed since the client first transmitted the message.

If both MRC and MRD are non-zero, the message exchange fails whenever either of the conditions specified in the previous two paragraphs are met. If both MRC and MRD are zero, the client continues to transmit the message until it receives a response or the client no longer wants a mapping.

Once a PCP client has successfully received a response from a PCP server on that interface, it resets RT to a value randomly selected in the range 1/2 to 5/8 of the mapping lifetime, as described in Section 11.2.1, "Renewing a Mapping", and sends subsequent PCP requests for that mapping to that same server.

> Note: If the server's state changes between retransmissions and the server's response is delayed or lost, the state in the PCP client and server may not be synchronized. This is not unique to PCP, but also occurs with other network protocols (e.g., TCP). In the unlikely event that such de-synchronization occurs, PCP heals itself after lifetime seconds.

8.2. 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 from which it would normally receive packets from that client, and it 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 that supports THIRD_PARTY requests MAY be configured to accept THIRD_PARTY requests on other configured interfaces (see Section 13.1 for details on the THIRD_PARTY Option).

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 that is sent back to the PCP client. Processing of an Opcode and its options is specific to each Opcode.

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

Copying request fields into the response is important because this is what enables a client to identify to which request a given 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).

All responses (both error and success) contain the same Opcode as the request, but with the "R" bit set.

Any error response has a non-zero result code, and is created by:

- Copying the entire UDP payload, or 1100 octets, whichever is less, and zero-padding the response to a multiple of 4 octets if necessary
- Setting the R bit
- Setting the result code
- Setting the Lifetime, Epoch Time, and Reserved fields

- Updating other fields in the response, as indicated by 'set by the server' in the PCP response field description

A success response has a zero result code, and is created by:

- Copying the first 4 octets of request packet header
- Setting the R bit
- Setting the result code to zero
- Setting the Lifetime, Epoch Time, and Reserved fields
- Possibly setting Opcode-specific response data if appropriate
- Adding any processed options to the response message

If the received PCP request message is less than 2 octets long, it is silently dropped.

If the R bit is set, the message is silently dropped.

If the first octet (version) is a version that is not supported, a response is generated with the UNSUPP_VERSION result code, and the Version Negotiation steps detailed in Section 9 are followed.

Otherwise, if the version is supported but the received message is shorter than 24 octets, the message is silently dropped.

If the server is overloaded by requests (from a particular client or from all clients), it MAY simply silently 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 1100 octets, is not a multiple of 4 octets, or is too short for the Opcode in question, it is invalid and a MALFORMED_REQUEST response is generated, and the response message is truncated to 1100 octets.

The PCP server compares the source IP address (from the received IP header) with the field PCP Client IP Address. 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 that the PCP field matches its apparent IP address from the perspective of the PCP server.

8.3. 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 a previously sent PCP request. If not, the response is silently dropped.

If the received PCP response message is less than 4 octets long, it is silently dropped.

If the R bit is clear, the message is silently dropped.

If the error code is UNSUPP_VERSION, Version Negotiation processing continues as described in Section 9.

Responses shorter than 24 octets, longer than 1100 octets, or not a multiple of 4 octets are invalid and ignored.

The PCP client then validates that the Opcode matches a previous PCP request. If the response does not match a previous PCP request, the response is ignored. 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. If that fails, the response is ignored.

After these matches are successful, the PCP client checks the Epoch Time field (see Section 8.5) to determine if it needs to restore its state to the PCP server. A PCP client SHOULD be prepared to receive multiple responses from the PCP server at any time after a single request is sent. This allows the PCP server to inform the client of mapping changes such as an update or deletion. For example, a PCP server might send a SUCCESS response and, after a configuration change on the PCP server, later send a NOT_AUTHORIZED response. A PCP client MUST be prepared to receive responses for requests it never sent (which could have been sent by a previous PCP instance on this same host, or by a previous host that used the same client IP address, or by a malicious attacker) by simply ignoring those unexpected messages.

If the error ADDRESS_MISMATCH is received, it indicates the presence of a NAT between the PCP client and PCP server. Procedures to resolve this problem are beyond the scope of this document.

For both success and error responses, a Lifetime value is returned. The lifetime indicates how long this response should be considered valid by the client (i.e for success results, how long the mapping will last, and for failure results how long the same failure condition should be expected to persist). The PCP client SHOULD impose an upper limit on this returned value (to protect against absurdly large values, e.g., 5 years), detailed in Section 15, "Mapping Lifetime and Deletion".

If the result code is 0 (SUCCESS), the request succeeded.

If the result code is not 0, the request failed, and the PCP client SHOULD NOT resend the same request for the indicated lifetime of the error (as limited by the sanity checking detailed in Section 15).

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.

8.4. Multi-Interface Issues

Hosts that 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 is the case for a Global Unicast Address (GUA) [RFC3587] or limited scope as is the case for a Unique Local Address (ULA) [RFC4193]).

IPv6 addresses with global reachability (e.g., GUAs) SHOULD be used as the source address when generating a PCP request. IPv6 addresses without global reachability (e.g., ULAs) 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 the client delete its mappings to the previous privacy address after it no longer needs those old mappings.

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.

8.5. Epoch

Every PCP response sent by the PCP server includes an Epoch Time field. This time field increments by one 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 external 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, the Epoch Time value is treated as necessarily valid, otherwise
  - If the current PCP server Epoch time (curr_server_time) is less than the previously received PCP server Epoch time (prev_server_time) by more than one second, then the client treats the Epoch time as obviously invalid (time should not go backwards). The server Epoch time apparently going backwards by *up to* one second is not deemed invalid, so that minor packet reordering on the path from PCP server to PCP client does not trigger a cascade of unnecessary mapping renewals. If the server Epoch time passes this check, then further validation checks are performed:
    - The client computes the difference between its current local time (curr_client_time) and the time the previous PCP response was received from this PCP server (prev_client_time):
      client_delta = curr_client_time - prev_client_time;
    - The client computes the difference between the current PCP server Epoch time (curr_server_time) and the previously received Epoch time (prev_server_time):
      server_delta = curr_server_time - prev_server_time;
    - If client_delta+2 < server_delta - server_delta/16 or server_delta+2 < client_delta - client_delta/16, 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:
  prev_client_time = curr_client_time
  prev_server_time = curr_server_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 following the mapping recreation procedure described in Section 16.3.1.

Notes:

- The client clock MUST never go backwards. If curr_client_time is found to be less than prev_client_time, then this is a client bug, and how the client deals with this client bug is implementation specific.

- The calculations above are constructed to allow client_delta and server_delta to be computed as unsigned integer values.

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

- The "/16" in the calculations above is to accommodate inaccurate clocks in low-cost devices. This allows for a total discrepancy of up to 1/16 (6.25%) to be considered benign; e.g., if one clock were to run too fast by 3% while the other clock ran too slow by 3%, then the client would not consider this difference to be anomalous or indicative of a restart having occurred. This tolerance is strict enough to be effective at detecting reboots, while not being so strict as to generate false alarms.

9. Version Negotiation

A PCP client sends its requests using PCP version number 2. Should later updates to this document specify different message formats with a version number greater than 2, it is expected that PCP servers will still support version 2 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 2, and backwards compatibility be desired, this first octet can be used for forward and backward compatibility.

If future PCP versions greater than 2 are specified, version negotiation proceeds as follows:

1. The client sends its first request using the highest (i.e., presumably 'best') version number it supports.

2. If the server supports that version, it responds normally.

3. 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).

4. 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). If the version number in the UNSUPP_VERSION response is zero then that means this is a NAT-PMP server [RFC6886], and a client MAY choose to communicate with it using the older NAT-PMP protocol, as described in Appendix A.

5. If the client receives an UNSUPP_VERSION result containing a version it does not support, then the client SHOULD try the next-lower version supported by the client. The attempt to use the next-lower version repeats until the client has tried version 2. If using version 2 fails, 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. By automatically retrying in 30 minutes, the protocol accommodates an upgrade of the PCP server.

10. 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 10.1);

- a host operating a client and server on the same port (Section 10.2);

- a host operating a client and wanting to optimize the application keepalive traffic (Section 10.3); and

- a host operating a client and wanting to restore lost state in its NAT (Section 10.4).

These are discussed in the following sections, and a (non-normative) state diagram is provided in Section 16.5.

When operating a server (see Sections 10.1 and 10.2), 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., an IPv4 address or an IPv6 address), and whether 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 Suggested External Address field, and sends a second MAP request containing the all-zeros IPv4 address in the Suggested 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 address (if the PCP server supports NAT44 or IPv4 firewall) or one MAP request from its IPv6 address (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.

The MAP and PEER requests include a Suggested External IP Address field. Some PCP-controlled devices, especially CGN but also multi-homed NPTv6 networks, have a pool of public-facing IP addresses. PCP allows the client to indicate if it wants a mapping assigned on a specific address of that pool or any address of that pool. Some applications will break if mappings are created on different IP addresses (e.g., active mode FTP), so applications should carefully consider the implications of using this capability. Static mappings for that internal address (e.g., those created by a command-line interface on the PCP server or PCP-controlled device) may exist to a certain external address, and if the suggested external IP address is the IPv4 or IPv6 all-zeros address, PCP SHOULD assign its mappings to the same external address, as this can also help applications using a mix of both static mappings and PCP-created mappings. If, on the other hand, the suggested external IP address contains a non-zero IP address the PCP server SHOULD create a mapping to that external address, even if there are other mappings from that same internal address to a different external address. Once an internal address

has no implicit dynamic mappings and no explicit dynamic mappings in the PCP-controlled device, a subsequent implicit or explicit mapping for that internal address MAY be assigned to a different External address. Generally, this reassignment would occur when a CGN device is load balancing newly seen internal addresses to its public pool of external addresses.

The following table summarizes how various common PCP deployments use IPv6 and IPv4 addresses.

The 'internal' address is implicitly the same as the source IP address of the PCP request, except when the THIRD_PARTY option is used.

The 'external' address is the Suggested External Address field of the MAP or PEER request, and its address family is usually the same as the 'internal' address family, except when technologies like NAT64 are used.

The 'remote peer' address is the remote peer IP address of the PEER request or the FILTER option of the MAP request, and is always the same address family as the 'internal' address, even when NAT64 is used. In NAT64, 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 Figure 5.

	internal	external	PCP remote peer	actual remote peer
IPv4 firewall	IPv4	IPv4	IPv4	IPv4
IPv6 firewall	IPv6	IPv6	IPv6	IPv6
NAT44	IPv4	IPv4	IPv4	IPv4
NAT46	IPv4	IPv6	IPv4	IPv6
NAT64	IPv6	IPv4	IPv6	IPv4
NPTv6	IPv6	IPv6	IPv6	IPv6

Figure 5: Address Families with MAP and PEER

Note that the internal address and the remote peer address are always the same address family, and the external address and the actual remote peer address are always the same address family.

10.1. 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, protocol, and port to itself using the MAP Opcode, as described in Section 11; (b) publish that public IP address, protocol, and port via some sort of rendezvous server (e.g., DNS, a SIP message, or a proprietary protocol); and (c) ensure that any other non-PCP-speaking packet filtering middleboxes on the path (e.g., host-based firewall, network-based firewall, or other NATs) will also allow the incoming traffic. 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-zeros address, depending on whether it wants a public IPv4 or IPv6 address.

The following pseudocode 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
      * 4. Resending a request due to server state loss
      * The PCP packet sent is identical in all four cases; from
      * the PCP server's point of view they are the same operation.
      * The suggested external address and port may be updated
      * repeatedly during the lifetime of the mapping.
      * Other fields in the packet generally remain unchanged.
      */
     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 6: Pseudocode for Using PCP to Operate a Server

10.2. 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, protocol, 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 11, 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 that the host has connected to a new network, the PCP client can attempt to request a mapping using PCP; 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 that 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 implicit dynamic mappings (e.g., TCP SYNs) sent from the specified internal address, protocol, 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. For further information on using PCP with EDM NATs, see Section 16.1.

The following pseudocode 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
      * 4. Resending a request due to server state loss
      */
     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 7: Pseudocode for Using PCP to Operate a Symmetric Client/Server

10.3. For Reducing NAT or Firewall 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 that 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, generally need to see occasional traffic or they 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 12 to learn and/or extend the lifetime of its mapping.

The following pseudocode shows how PCP can be reliably used with a dynamic socket, for the purposes of reducing application keepalive messages:

```
 /* make outgoing connection to server */
 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
      * 4. Resending a request due to server state loss
      */
     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 (received_incoming_data())
         process_it(s);

     if (other_work_to_do())
         do_it();

     /* ... */

     block_until_we_need_to_do_something_else();
     }
```

Figure 8: Pseudocode Using PCP with a Dynamic Socket

10.4. 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:

(i) the NAT creates a new implicit dynamic outbound mapping only for outbound TCP segments with the SYN bit set (i.e., the newly booted NAT silently drops outbound data segments from the client when the NAT does not have an active mapping for those segments), and

(ii) the newly booted NAT does not send a TCP RST in response to receiving unexpected inbound TCP segments.

This procedure works less well for UDP, because as soon as outbound UDP traffic is seen by the NAT, a new UDP implicit dynamic outbound mapping will be created (probably on a different port).

11. MAP Opcode

This section defines an Opcode that controls inbound forwarding from a NAT (or firewall) to an internal host.

MAP: Create an explicit dynamic mapping between an Internal Address + Port and an External Address + Port.

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 (EIMs) with endpoint-independent filtering (EIF) (unless the FILTER option is used), even on a NAT that usually creates endpoint-dependent mapping (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 address, protocol, and 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 back to the mapping's internal address and port.

On operating systems that allow multiple listening servers to bind to the same internal address, protocol, and port, servers MUST ensure that they have exclusive use of that internal address, protocol, and port (e.g., by binding the port using INADDR_ANY, or using SO_EXCLUSIVEADDRUSE or similar) before sending their PCP MAP request, to ensure that no other PCP clients on the same machine are also listening on the same internal protocol and internal port.

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 use PCP separately to create ICMP mappings for those flows.

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

11.1. MAP Operation Packet Formats

The MAP Opcode has a similar packet layout for both requests and responses. If the assigned external IP address and port in the PCP response always match the internal IP address and port from the PCP request, then the functionality is purely a firewall; otherwise, the functionality is a Network Address Translator that might also perform firewall-like functions.

The following diagram shows the format of the Opcode-specific information in a request for the MAP Opcode.

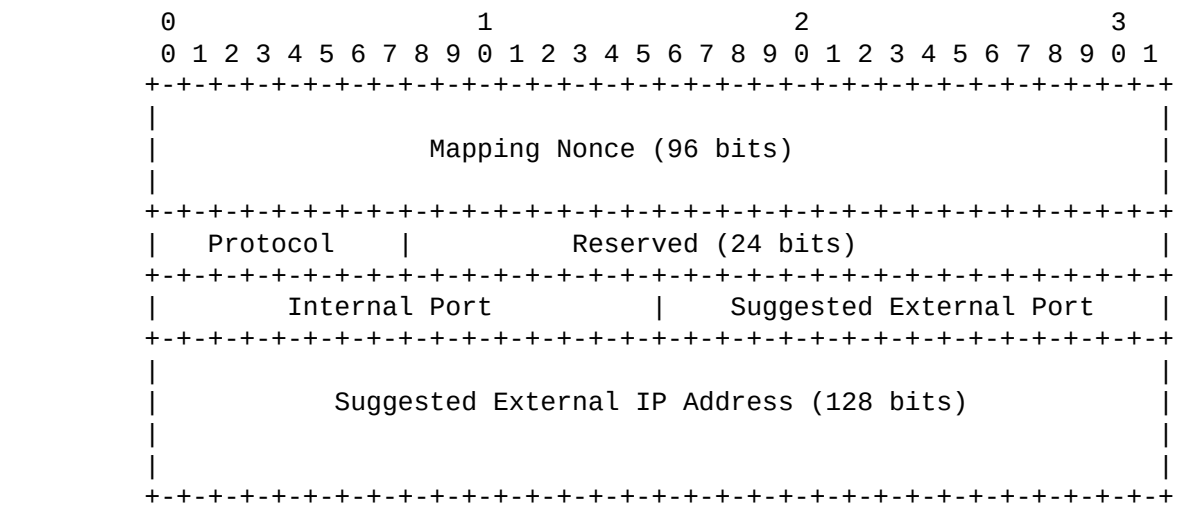

Figure 9: MAP Opcode Request

These fields are described below:

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

Mapping Nonce: Random value chosen by the PCP client. See Section 11.2, "Generating a MAP Request". Zero is a legal value (but unlikely, occurring in roughly one in 2^96 requests).

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. This field contains 17 (UDP) if the Opcode is intended to create a UDP 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 client is requesting 'all ports'. If the protocol is zero (meaning 'all protocols'), then internal port MUST be zero on transmission and MUST be ignored on reception.

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-zeros 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:

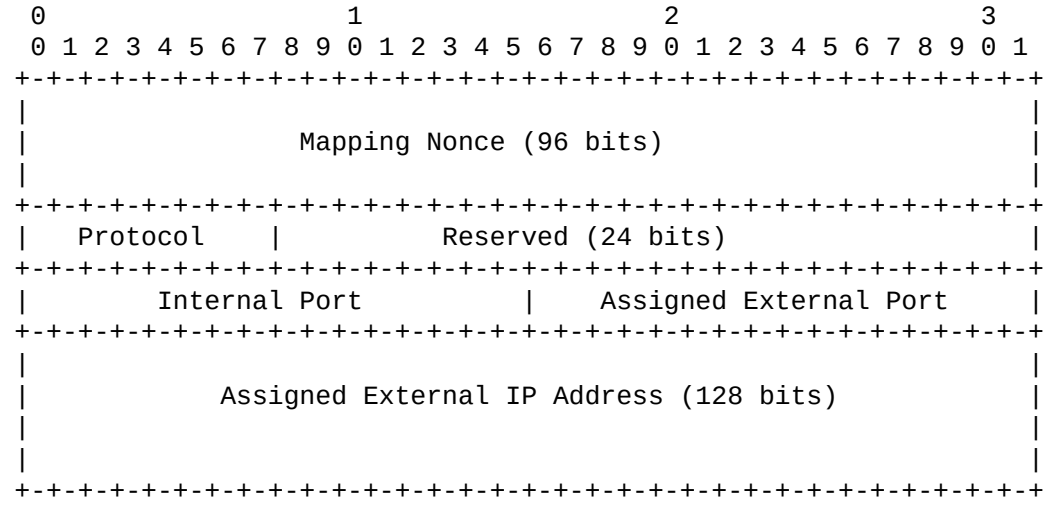

Figure 10: MAP Opcode Response

These fields are described below:

Lifetime (in common header): 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. On a success response, this indicates the lifetime for this mapping, in seconds.

Mapping Nonce: Copied from the 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.

11.2. Generating a MAP Request

This section describes 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 to grant the suggested external IP address, protocol, and port, and in that case it will assign a different external IP address and port.

A PCP client MUST be written assuming that it may *never* be assigned the external port it suggests. In the case of recreating state after a NAT gateway crash, the suggested external port, being one that was previously allocated to this client, is likely to be available for this client to continue using. In all other cases, the client MUST assume that it is unlikely that its suggested external port will be granted. For example, when many subscribers are sharing a Carrier-Grade NAT, popular ports such as 80, 443, and 8080 are likely to be in high demand. At most one client can have each of those popular ports for each external IP address, and all the other clients will be assigned other, dynamically allocated, external ports. Indeed, some ISPs may, by policy, choose not to grant those external ports to *anyone*, so that none of their clients are *ever* assigned external ports 80, 443, or 8080.

If the protocol does not use 16-bit port numbers (e.g., RSVP, IP protocol number 46), the port number MUST be zero. 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).

The Mapping Nonce value is randomly chosen by the PCP client, following accepted practices for generating unguessable random numbers [RFC4086], and is used as part of the validation of PCP responses (see below) by the PCP client, and validation for mapping refreshes by the PCP server. The client MUST use a different mapping nonce for each PCP server it communicates with, and it is RECOMMENDED to choose a new random mapping nonce whenever the PCP client is initialized. The client MAY use a different mapping nonce for every mapping.

11.2.1. Renewing a Mapping

An existing mapping SHOULD have its lifetime extended by the PCP client for as long as the client wishes to have that mapping continue to exist. 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 in the Suggested External IP Address and Suggested External Port fields, so if the PCP server 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 15, "Mapping Lifetime and Deletion"). 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).

11.3. Processing a MAP Request

This section describes 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 Protocol, Internal Port, and Mapping Nonce fields from the MAP request are copied into the MAP response. The THIRD_PARTY option, if present, and processed by the PCP server, is also copied into the MAP response.

If the requested lifetime is non-zero, then:

- If both the protocol and internal port are 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, the UNSUPP_PROTOCOL error MUST be returned.

- If the protocol is non-zero and the internal port is zero, it indicates a request to create or extend a mapping for all incoming traffic for that entire protocol -- a 'wildcard' (all-ports) mapping for that protocol. If this request cannot be fulfilled in its entirety, the UNSUPP_PROTOCOL error MUST be returned.

- If both the protocol and internal port are zero, it indicates a request to create or extend a mapping for all incoming traffic for all protocols (commonly called a 'DMZ host'). If this request cannot be fulfilled in its entirety, the UNSUPP_PROTOCOL error MUST be returned.

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

If the requested lifetime is zero, it indicates a request to delete an existing mapping.

Further processing of the lifetime is described in Section 15, "Mapping Lifetime and Deletion".

If operating in the Simple Threat Model (Section 18.1), and the internal port, protocol, and internal address match an existing explicit dynamic mapping, but the mapping nonce does not match, the request MUST be rejected with a NOT_AUTHORIZED error with the lifetime of the error indicating duration of that existing mapping. The PCP server only needs to remember one Mapping Nonce value for each explicit dynamic mapping. This specification makes no statement about mapping nonce with the Advanced Threat Model.

If the internal port, protocol, and internal address match an existing static mapping (which will have no nonce), then a PCP reply is sent giving the external address and port of that static mapping, using the nonce from the PCP request. The server does not record the nonce.

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

If the PCP-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, including no rewriting), 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, and pure firewalls, which modify neither the address nor the port.

It is possible that a mapping might already exist for a requested internal address, protocol, and port. If so, the PCP server takes the following actions:

1. If the MAP request contains the PREFER_FAILURE option, but the suggested external address and port do not match the external address and port of the existing mapping, the PCP server MUST return CANNOT_PROVIDE_EXTERNAL.

2. If the existing mapping is static (created outside of PCP), the PCP server MUST return the external address and port of the existing mapping in its response and SHOULD indicate a lifetime of 2^32-1 seconds, regardless of the suggested external address and port in the request.

3. If the existing mapping is explicit dynamic inbound (created by a previous MAP request), the PCP server MUST return the existing external address and port in its response, regardless of the suggested external address and port in the request. Additionally, the PCP server MUST update the lifetime of the existing mapping, in accordance with Section 15, "Mapping Lifetime and Deletion".

4. If the existing mapping is dynamic outbound (created by outgoing traffic or a previous PEER request), the PCP server SHOULD create a new explicit inbound mapping, replicating the ports and addresses from the outbound mapping (but the outbound mapping continues to exist, and remains in effect if the explicit inbound mapping is later deleted).

If no mapping exists for the internal address, protocol, 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). There are, however, cases where the PCP server is not able to create a new mapping using the suggested external address and port:

- The suggested external address, protocol, and port is already assigned to another existing explicit or implicit mapping (i.e., is already forwarding traffic to some other internal address and port).

- The suggested external address, protocol, and port is already used by the NAT gateway for one of its own services, for example, TCP port 80 for the NAT gateway's own configuration web pages, or UDP ports 5350 and 5351, used by PCP itself. A PCP server MUST NOT create client mappings for external UDP ports 5350 or 5351.

- The suggested external address, protocol, and port is otherwise prohibited by the PCP server's policy.

- The suggested external IP address, protocol, or suggested port are invalid or invalid combinations (e.g., external address 127.0.0.1, ::1, a multicast address, or the suggested port is not valid for the 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.

If the PCP server cannot assign the suggested external address, protocol, and port, then:

- If the request contained the PREFER_FAILURE option, then the PCP server MUST return CANNOT_PROVIDE_EXTERNAL.

- If the request did not contain the PREFER_FAILURE option, and the PCP server can assign some other external address and port for that protocol, then the PCP server MUST do so and return the newly assigned external address and port in the response. In no case is the client penalized for a 'poor' choice of suggested external address and port. The suggested external address and port may be used by the server to guide its choice of what external address and port to assign, but in no case do they cause the server to fail to allocate an external address and port where otherwise it would have succeeded. The presence of a non-zero suggested external address or port is merely a hint; it never does any harm.

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, an additional corresponding UDP mapping MUST NOT be automatically 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.

11.4. 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 a previously sent MAP request by comparing the internal IP address (the destination IP address of the PCP response, or other IP address specified via the THIRD_PARTY option), the protocol, the internal port, and the mapping nonce. Other fields are not compared, because the PCP server sets those fields. The PCP server will send a Mapping Update (Section 14.2) if the mapping changes (e.g., due to IP renumbering).

If the result code is NO_RESOURCES and the request was for the creation or renewal of a mapping, then the PCP client SHOULD NOT send further requests for any new mappings to that PCP server for the (limited) value of the lifetime. If the result code is NO_RESOURCES and the request was for the deletion of a mapping, then the PCP client SHOULD NOT send further requests of *any kind* to that PCP server for the (limited) value of the lifetime.

On a success response, the PCP client can use the external IP address and port as needed. 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.

After a success response, for as long as renewal is desired, the PCP client MUST 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 11.2, 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 create 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 to create a new mapping, with a particular suggested external address and port, which happen to be what the PCP server previously assigned. See also Section 16.3.1, "Recreating Mappings".

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

11.5. Address Change Events

A customer premises router might obtain a new external 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 a host's previous address might be delivered to another host that now has that address. This affects all mapping types, whether implicit or explicit. This same problem already occurs today when a host's IP address is reassigned, 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 internal 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 external 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 that 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.

11.6. Learning the External IP Address Alone

NAT-PMP [RFC6886] includes a mechanism to allow clients to learn the external IP address alone, without also requesting a port mapping. NAT-PMP was designed for residential NAT gateways, where such an operation makes sense because a typical residential NAT gateway has only one external IP address. PCP has broader scope, and also supports Carrier-Grade NATs (CGNs) that 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 at least one implicit, explicit, or static port mapping, and even then only for as long as that 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 (e.g., to the Discard service (TCP/9 or UDP/9) or some other port) 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.

12. PEER Opcode

This section defines an Opcode for controlling dynamic outbound mappings.

PEER: Create a new dynamic outbound mapping to a remote peer's IP address and port, or extend the lifetime of an existing outbound mapping.

The use of this 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 like an implicit dynamic outbound mapping created as a side effect of a packet (e.g., TCP SYN) sent by the host, mappings created or managed using PCP PEER requests may be endpoint-independent mapping (EIM) or endpoint-dependent mapping (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 outbound mappings it creates automatically as a result of observing outgoing traffic from internal hosts.

12.1. PEER Operation Packet Formats

The PEER Opcode allows a PCP client to create a new explicit dynamic outbound mapping (which functions similarly to an outbound mapping created implicitly when a host sends an outbound TCP SYN) or to extend the lifetime of an existing outbound mapping.

The following diagram shows the Opcode layout for the PEER Opcode. The formats for the PEER request and response packets are aligned so that related fields fall at the same offsets in the packet.

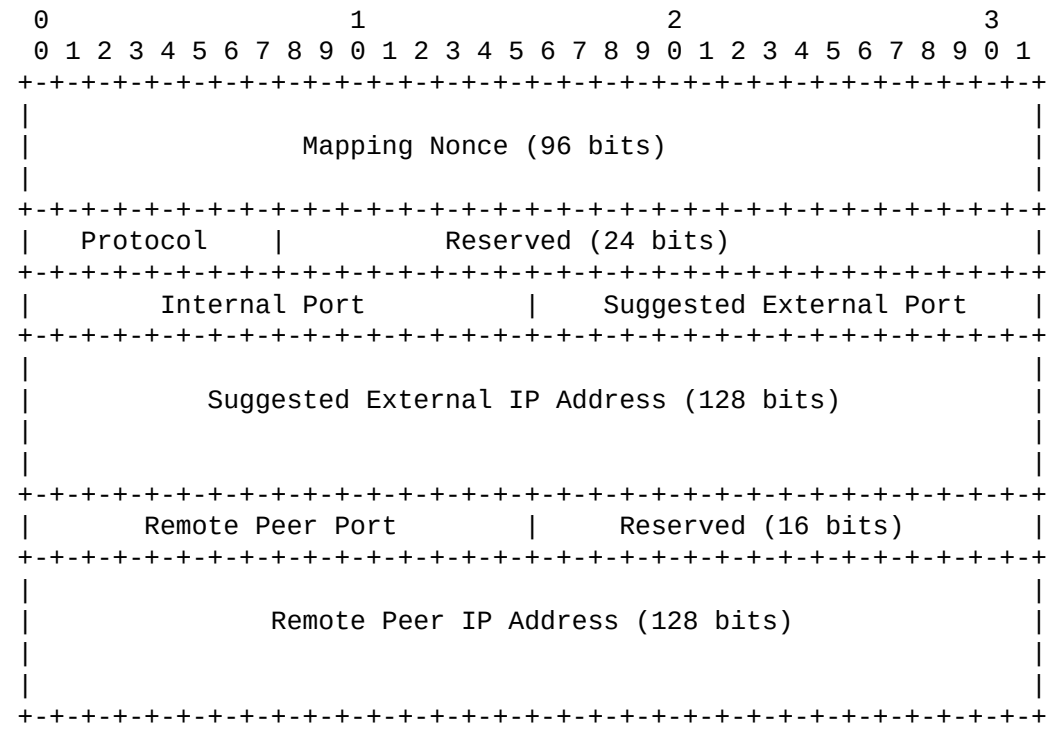

Figure 11: PEER Opcode Request

These fields are described below:

Requested Lifetime (in common header): Requested lifetime of this mapping, in seconds. Note that it is not possible to reduce the lifetime of a mapping (or delete it, with requested lifetime=0) using PEER.

Mapping Nonce: Random value chosen by the PCP client. See Section 12.2, "Generating a PEER Request". Zero is a legal value (but unlikely, occurring in roughly one in 2^96 requests).

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. This field contains 17 (UDP) if the Opcode is describing a UDP mapping. Protocol MUST NOT be zero.

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

Internal Port: Internal port for the mapping. Internal port MUST NOT be zero.

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-zeros address (see Section 5).

Remote Peer Port: Remote peer's port for the mapping. Remote peer port MUST NOT be zero.

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. This is 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. An IPv6 address is represented directly, and an IPv4 address is represented using the IPv4-mapped address syntax (Section 5).

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 semantics similar to the PREFER_FAILURE option 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, protocol, or port, then the PCP server MUST return a

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

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

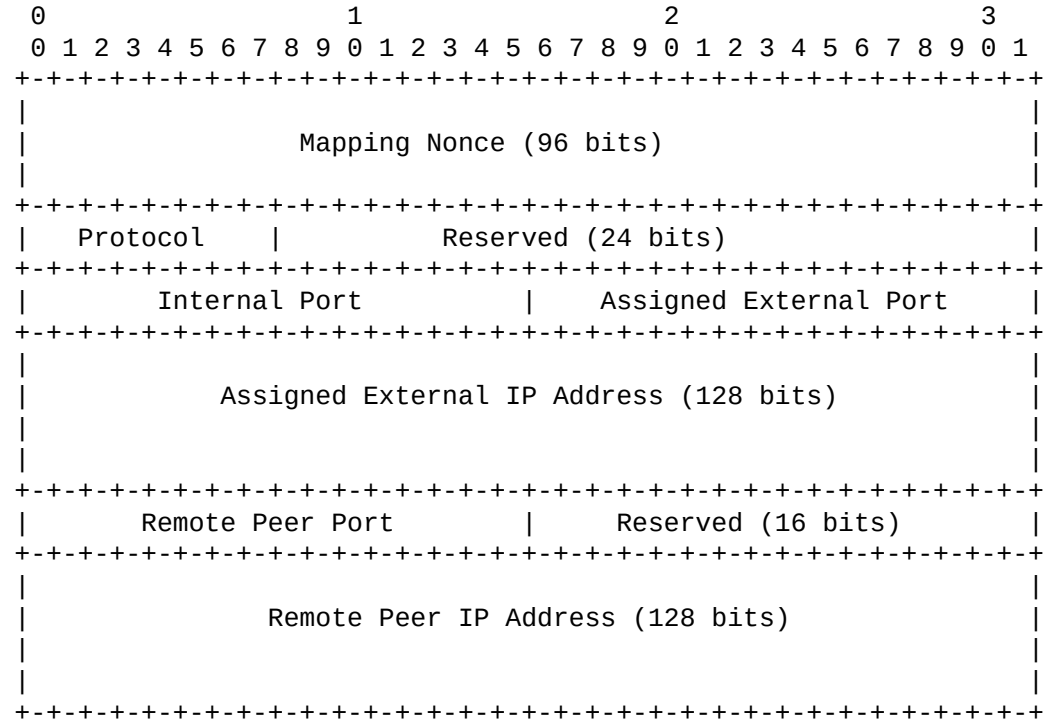

Figure 12: PEER Opcode Response

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.

Mapping Nonce: Copied from the 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. 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.

12.2. 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 bidirectional communication with the remote peer.

If sent before, this is considered a PEER-created mapping that creates a new dynamic outbound mapping in the PCP-controlled device.

If sent after, this allows the PCP client to learn the IP address, port, and lifetime of the assigned external address and port for the existing implicit dynamic outbound mapping, and potentially to extend this lifetime (for reducing NAT or firewall keepalive messages, as described in Section 10.3).

PEER requests are also useful for restoring mappings after a NAT has lost its mapping state (e.g., due to a crash).

The Mapping Nonce value is randomly chosen by the PCP client, following accepted practices for generating unguessable random numbers [RFC4086], and is used as part of the validation of PCP responses (see below) by the PCP client, and validation for mapping refreshes by the PCP server. The client MUST use a different mapping nonce for each PCP server it communicates with, and it is RECOMMENDED to choose a new random mapping nonce whenever the PCP client is initialized. The client MAY use a different mapping nonce for every mapping.

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.

12.3. 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, Remote Peer Port, and Mapping Nonce.

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, protocol, and port for PEER-created explicit 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 for a matching five-tuple { Protocol, Internal Address, Internal Port, Remote Peer Address, Remote Peer Port }.

If no matching mapping is found, and the suggested external address and port are either zero or can be honored for the specified Protocol, a new mapping is created. By having the PEER create such a mapping, we avoid a race condition between the PEER request and the initial outgoing packet arriving at the NAT or firewall device first, and allow PEER to be used to recreate a lost outbound dynamic mapping (see Section 16.3.1, "Recreating Mappings"). Thereafter, this PEER-created mapping is treated as if it was an implicit dynamic outbound mapping (e.g., as if the PCP client sent a TCP SYN) and a lifetime appropriate to such a mapping is returned (note: on many NATs and firewalls, such mapping lifetimes are very short until bidirectional traffic is seen by the NAT or firewall).

If no matching mapping is found, and the suggested external address and port cannot be honored, then no new state is created, and the error CANNOT_PROVIDE_EXTERNAL is returned.

If a matching mapping is found, and no previous PEER Opcode was successfully processed for this mapping, then the Suggested External Address and Port values in the request are ignored, the lifetime of that mapping is adjusted as described below, and information about the existing mapping is returned. This allows a client to explicitly extend the lifetime of an existing mapping and/or to learn an existing mapping's external address, port, and lifetime. The mapping nonce is remembered for this mapping.

If operating in the Simple Threat Model (Section 18.1), and the internal port, protocol, and internal address match a mapping that already exists, but the mapping nonce does not match (that is, a previous PEER request was processed), the request MUST be rejected with a NOT_AUTHORIZED error with the lifetime of the error indicating duration of that existing mapping. The PCP server only needs to remember one Mapping Nonce value for each mapping. This specification makes no statement about mapping nonce with the Advanced Threat Model.

Processing the Lifetime value of the PEER Opcode is described in Section 15, "Mapping Lifetime and Deletion". Sending a PEER request with a very short requested lifetime can be used to query the lifetime of an existing mapping. So that PCP clients can reduce the frequency of their NAT and firewall keepalive messages, it is RECOMMENDED that lifetimes of mappings created or lengthened with PEER be longer than the lifetimes of implicitly created mappings.

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.

If a PEER-created or PEER-managed mapping is not renewed using PEER, then it reverts to the NAT's usual behavior for implicit mappings. For example, continued outbound traffic keeps the mapping alive, as per the NAT or firewall device's existing policy. A PEER-created or PEER-managed mapping may be terminated at any time by action of the TCP client or server (e.g., due to TCP FIN or TCP RST), as per the NAT or firewall device's existing policy.

12.4. 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 an outstanding PEER request by comparing the internal IP address (the destination IP address of the PCP response, or other IP address specified via the THIRD_PARTY option), the

protocol, the internal port, the remote peer address, the remote peer port, and the mapping nonce. Other fields are not compared, because the PCP server sets those fields to provide information about the mapping created by the Opcode. The PCP server will send a Mapping Update (Section 14.2) if the mapping changes (e.g., due to IP renumbering).

If the result code is NO_RESOURCES and the request was for the creation or renewal of a mapping, then the PCP client SHOULD NOT send further requests for any new mappings to that PCP server for the (limited) value of the lifetime.

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 response indicates an unexpected IP address or port (e.g., due to IP renumbering), the PCP client will want to re-establish its connection to its remote server.

If the PCP client wishes to keep this mapping alive beyond the indicated lifetime, it MAY rely on continued inside-to-outside traffic to ensure that the mapping will continue to exist, or it MAY issue a new PCP request prior to the expiration. The recommended timings for renewing PEER mappings are the same as for MAP mappings, as described in Section 11.2.1.

> 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.

13. 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.

13.1. 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.

Due to security concerns with the THIRD_PARTY option, this option MUST NOT be implemented or used unless the network on which the PCP

messages are to be sent is fully trusted. For example, if access control lists (ACLs) are installed on the PCP client, PCP server, and the network between them, so those ACLs allow only communications from a trusted PCP client to the PCP server.

A management device would use this option to control a PCP server on behalf of users. For example, a management device located in a network operations center, which presents a user interface to end users or to network operations staff, and issues PCP requests with the THIRD_PARTY option to the appropriate PCP server.

The THIRD_PARTY option is formatted as follows:

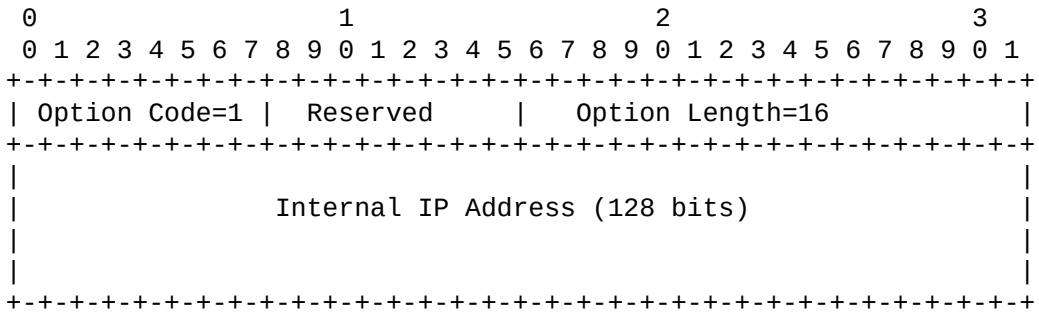

Figure 13: THIRD_PARTY Option

The fields are described below:

Internal IP Address: Internal IP address for this mapping.

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: 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. This is because many PCP servers may not implement the THIRD_PARTY option at all, and with those servers a client redundantly using the THIRD_PARTY option to specify its own IP address would cause such mapping requests to fail where they would otherwise have succeeded. 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.

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.

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, from the point of view of the ISP, to be a single host.

When a PCP client is using the THIRD_PARTY option to make and maintain mappings on behalf of some other device, it may be beneficial if, where possible, the PCP client verifies that the other device is actually present and active on the network. Otherwise, the PCP client 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.

13.2. 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 both the suggested external port and suggested external address, the PCP server should not create a mapping. This differs from the behavior without this option, which is to create a mapping.

PREFER_FAILURE is never necessary for a PCP client to manage mappings for itself, and its use causes additional work in the PCP client and in the PCP server. This option exists for interworking with non-PCP mapping protocols that have different semantics than PCP (e.g., UPnP IGDv1 interworking [PNP-IGD-PCP], where the semantics of UPnP IGDv1

only allow the UPnP IGDv1 client to dictate mapping a specific port), or separate port allocation systems that allocate ports to a subscriber (e.g., a subscriber-accessed web portal operated by the same ISP that operates the PCP server). A PCP server MAY support this option, if its designers wish to support such downstream devices or separate port allocation systems. PCP servers that are not intended to interface with such systems are not required to support this option. PCP clients other than UPnP IGDv1 interworking clients or other than a separate port allocation system 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 this option declines.

The PREFER_FAILURE option is formatted as follows:

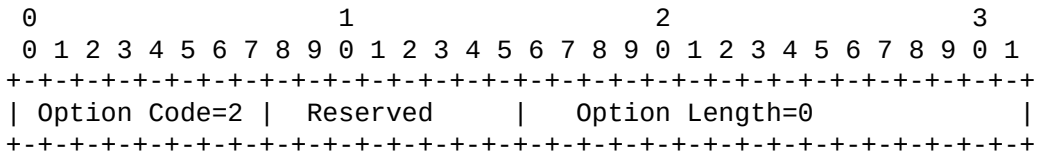

Figure 14: PREFER_FAILURE Option

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 cannot be mapped.
Valid for Opcodes: MAP
Length: 0
May appear in: request. May appear in response only if it appeared in the associated request.
Maximum occurrences: 1

The result code CANNOT_PROVIDE_EXTERNAL is returned if the suggested external address, protocol, and port cannot be mapped. This can occur because the external port is already mapped to another host's outbound dynamic mapping, an inbound dynamic mapping, a static mapping, or the same internal address, protocol, and port already have an outbound dynamic mapping that is mapped to a different external port than suggested. This can also occur because the external address is no longer available (e.g., due to renumbering). The server MAY set the lifetime in the response to the remaining lifetime of the conflicting mapping + TIME_WAIT [RFC0793], rounded up to the next larger integer number of seconds.

If a PCP request contains the PREFER_FAILURE option and has zero in the Suggested External Port field, then it is invalid. The PCP server MUST reject such a message with the MALFORMED_OPTION error code.

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.

There can exist a race condition between the MAP Opcode using the PREFER_FAILURE option and Mapping Update (Section 14.2). For example, a previous host on the local network could have previously had the same internal address, with a mapping for the same internal port. At about the same moment that the current host sends a MAP Request using the PREFER_FAILURE option, the PCP server could send a spontaneous Mapping Update for the old mapping due to an external configuration change, which could appear to be a reply to the new mapping request. Because of this, the PCP client MUST validate that the external IP address, protocol, port, and nonce in a success response match the associated suggested values from the request. If they do not match, it is because the Mapping Update was sent before the MAP request was processed.

13.3. FILTER Option for MAP Opcode

This option is only used with the MAP Opcode.

This option indicates that filtering incoming packets is desired. The protocol being filtered is indicated by the Protocol field in the MAP Request, and the remote peer IP address and remote peer port of the FILTER option indicate the permitted remote peer's source IP address and source port for packets from the Internet; other traffic from other addresses is blocked. 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 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 it 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 unwanted traffic.

The FILTER option is formatted as follows:

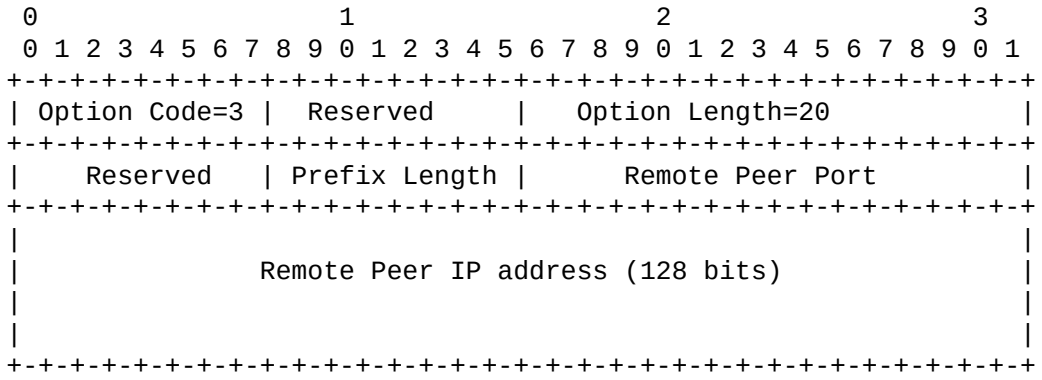

Figure 15: FILTER Option Layout

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: request. May appear in response only if it appeared in the associated request.
Maximum occurrences: as many as fit within maximum PCP message size

The Prefix Length indicates how many bits of the address are used for the filter. For IPv4 addresses (which are encoded using the IPv4-mapped address format (::FFFF:0:0/96)), this means valid prefix lengths are between 96 and 128 bits, inclusive. That is, add 96 to the IPv4 prefix length. For IPv6 addresses, valid prefix lengths are between 0 and 128 bits, inclusive. Values outside those ranges 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 order received (as per normal PCP option processing), and they MAY overlap the filtering requested. If there is an existing mapping (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 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, protocol, 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.

14. Rapid Recovery

PCP includes a rapid recovery feature, which allows PCP clients to repair failed mappings within seconds, rather than the minutes or hours it might take if they relied solely on waiting for the next routine renewal of the mapping. Mapping failures may occur when a NAT gateway is rebooted and loses its mapping state, or when a NAT

gateway has its external IP address changed so that its current mapping state becomes invalid.

The PCP rapid recovery feature enables users to, for example, connect to remote machines using ssh, and then reboot their NAT or firewall device (or even replace it with completely new hardware) without losing their established ssh connections.

Use of PCP rapid recovery is a performance optimization to PCP's routine self-healing. Without rapid recovery, PCP clients will still recreate their correct state when they next renew their mappings, but this routine self-healing process may take hours rather than seconds, and will probably not happen fast enough to prevent active TCP connections from timing out.

There are two mechanisms to perform rapid recovery, described below. Failing to implement and deploy a rapid recovery mechanism will encourage application developers to feel the need to refresh their PCP state more frequently than necessary, causing more network traffic. Therefore, a PCP server that can lose state (e.g., due to reboot) or might have a mapping change (e.g., due to IP renumbering) MUST implement either the Announce Opcode or the Mapping Update mechanism and SHOULD implement both mechanisms.

14.1. ANNOUNCE Opcode

This rapid recovery mechanism uses the ANNOUNCE Opcode. When the PCP server loses its state (e.g., it lost its state when rebooted), it resets its Epoch time to its initial starting value (usually zero) and sends the ANNOUNCE response to the link-scoped multicast address (specific address explained below) if a multicast network exists on its local interface, or, if configured with the IP address(es) and port(s) of PCP client(s), it sends unicast ANNOUNCE responses to those address(es) and port(s). This means ANNOUNCE may not be available on all networks (such as networks without a multicast link between the PCP server and its PCP clients). Additionally, an ANNOUNCE request can be sent (unicast) by a PCP client that elicits a unicast ANNOUNCE response like any other Opcode.

Upon receiving PCP response packets with an anomalous Epoch time, clients deduce that the PCP server lost state and recreate their lost mappings.

14.1.1. ANNOUNCE Operation

The PCP ANNOUNCE Opcode requests and responses have no Opcode-specific payload (that is, the length of the Opcode-specific data is zero). The Requested Lifetime field of requests and Lifetime field of responses are both set to 0 on transmission and ignored on reception.

If a PCP server receives an ANNOUNCE request, it first parses it and generates a SUCCESS if parsing and processing of ANNOUNCE is successful. An error is generated if the client's IP Address field does not match the packet source address, or the request packet is otherwise malformed, such as packet length less than 24 octets. Note that, in the future, options MAY be sent with the PCP ANNOUNCE Opcode; PCP clients and servers need to be prepared to receive options with the ANNOUNCE Opcode.

> Discussion: Client-to-server request messages are sent, from any client source port, to listening UDP port 5351 on the server; server-to-client multicast notifications are sent from the server's UDP port (5351) to listening UDP port 5350 on the client. The reason the same listening UDP port is not used for both purposes is that a single device may have multiple roles. For example, a multi-function home gateway that provides NAT service (PCP server) may also provide printer sharing (which wants a PCP client), or a home computer (PCP client) may also provide "Internet Sharing" (NAT) functionality (which needs to offer PCP service). Such devices need to act as both a PCP server and a PCP client at the same time, and the software that implements the PCP server on the device may not be the same software component that implements the PCP client. The software that implements the PCP server needs to listen for unicast client requests, whereas the software that implements the PCP client needs to listen for multicast restart announcements. In many networking APIs it is difficult or impossible to have two independent clients listening for both unicasts and multicasts on the same port at the same time. For this reason, two ports are used.

14.1.2. Generating and Processing a Solicited ANNOUNCE Message

The PCP ANNOUNCE Opcode MAY be sent (unicast) by a PCP client. The Requested Lifetime value MUST be set to zero.

When the PCP server receives the ANNOUNCE Opcode and successfully parses and processes it, it generates SUCCESS response with an assigned lifetime of zero.

This functionality allows a PCP client to determine a server's Epoch, or to determine if a PCP server is running, without changing the server's state.

14.1.3. Generating and Processing an Unsolicited ANNOUNCE Message

When sending unsolicited responses, the ANNOUNCE Opcode MUST have result code equal to zero (SUCCESS), and the packet MUST be sent from the unicast IP address and UDP port number on which PCP requests are received (so that the PCP response processing described in Section 8.3 will accept the message). This message is most typically multicast, but can also be unicast. Multicast PCP restart announcements are sent to 224.0.0.1:5350 and/or [ff02::1]:5350, as described below. Sending PCP restart announcements via unicast requires that the PCP server know the IP address(es) and port(s) of its listening clients, which means that sending PCP restart announcements via unicast is only applicable to PCP servers that retain knowledge of the IP address(es) and port(s) of their clients even after they otherwise lose the rest of their state.

When a PCP server device that implements this functionality reboots, restarts its NAT engine, or otherwise enters a state where it may have lost some or all of its previous mapping state (or enters a state where it doesn't even know whether it may have had prior state that it lost), it MUST inform PCP clients of this fact by unicasting or multicasting a gratuitous PCP ANNOUNCE Opcode response packet, as shown below, via paths over which it accepts PCP requests. If sending a multicast ANNOUNCE message, a PCP server device that accepts PCP requests over IPv4 sends the Restart Announcement to the IPv4 multicast address 224.0.0.1:5350 (224.0.0.1 is the All Hosts multicast group address), and a PCP server device that accepts PCP requests over IPv6 sends the Restart Announcement to the IPv6 multicast address [ff02::1]:5350 (ff02::1 is for all nodes on the local segment). A PCP server device that accepts PCP requests over both IPv4 and IPv6 sends a pair of Restart Announcements, one to each multicast address. If sending a unicast ANNOUNCE messages, it sends ANNOUNCE response message to the IP address(es) and port(s) of its PCP clients. To accommodate packet loss, the PCP server device MAY transmit such packets (or packet pairs) up to ten times (with an appropriate Epoch Time value in each to reflect the passage of time between transmissions) provided that the interval between the first two notifications is at least 250 ms, and the interval between subsequent notification at least doubles.

A PCP client that sends PCP requests to a PCP server via a multicast-capable path, and implements the Restart Announcement feature, and wishes to receive these announcements, MUST listen to receive these PCP Restart Announcements (gratuitous PCP ANNOUNCE Opcode response

packets) on the appropriate multicast-capable interfaces on which it sends PCP requests, and MAY also listen for unicast announcements from the server too, (using the UDP port it already uses to issue unicast PCP requests to, and receive unicast PCP responses from, that server). A PCP client device that sends PCP requests using IPv4 listens for packets sent to the IPv4 multicast address 224.0.0.1:5350. A PCP client device that sends PCP requests using IPv6 listens for packets sent to the IPv6 multicast address [ff02::1]:5350. A PCP client device that sends PCP requests using both IPv4 and IPv6 listens for both types of Restart Announcement. The SO_REUSEPORT socket option or equivalent should be used for the multicast UDP port, if required by the host OS to permit multiple independent listeners on the same multicast UDP port.

Upon receiving a unicasted or multicasted PCP ANNOUNCE Opcode response packet, a PCP client MUST (as it does with all received PCP response packets) inspect the announcement's source IP address, and if the Epoch Time value is outside the expected range for that server, it MUST wait a random amount of time between 0 and 5 seconds (to prevent synchronization of all PCP clients), then for all PCP mappings it made at that server address the client issues new PCP requests to recreate any lost mapping state. The use of the Suggested External IP Address and Suggested External Port fields in the client's renewal requests allows the client to remind the restarted PCP server device of what mappings the client had previously been given, so that in many cases the prior state can be recreated. For PCP server devices that reboot relatively quickly it is usually possible to reconstruct lost mapping state fast enough that existing TCP connections and UDP communications do not time out, and continue without failure. As for all PCP response messages, if the Epoch Time value is within the expected range for that server, the PCP client does not recreate its mappings. As for all PCP response messages, after receiving and validating the ANNOUNCE message, the client updates its own Epoch time for that server, as described in Section 8.5.

14.2. PCP Mapping Update

This rapid recovery mechanism is used when the PCP server remembers its state and determines its existing mappings are invalid (e.g., IP renumbering changes the external IP address of a PCP-controlled NAT).

It is anticipated that servers that are routinely reconfigured by an administrator or have their WAN address changed frequently will implement this feature (e.g., residential CPE routers). It is anticipated that servers that are not routinely reconfigured will not implement this feature (e.g., service provider-operated CGN).

If a PCP server device has not forgotten its mapping state, but for some other reason has determined that some or all of its mappings have become unusable (e.g., when a home gateway is assigned a different external IPv4 address by the upstream DHCP server), then the PCP server device automatically repairs its mappings and notifies its clients by following the procedure described below.

For PCP-managed mappings, for each one the PCP server device should update the external IP address and external port to appropriate available values, and then send unicast PCP MAP or PEER responses (as appropriate for the mapping) to inform the PCP client of the new external IP address and external port. Such unsolicited responses are identical to the MAP or PEER responses normally returned in response to client MAP or PEER requests, containing newly updated External IP Address and External Port values, and are sent to the same client IP address and port that the PCP server used to send the prior response for that mapping. If the earlier associated request contained the THIRD_PARTY option, the THIRD_PARTY option MUST also appear in the Mapping Update as it is necessary for the PCP client to disambiguate the response. If the earlier associated request contained the PREFER_FAILURE option, and the same external IP address, protocol, and port cannot be provided, the error CANNOT_PROVIDE_EXTERNAL SHOULD be sent. If the earlier associated request contained the FILTER option, the filters are moved to the new mapping and the FILTER option is sent in the Mapping Update response. Non-mandatory options SHOULD NOT be sent in the Mapping Update response.

> Discussion: It could have been possible to design this so that the PCP server (1) sent an ANNOUNCE Opcode to the PCP client, the PCP client reacted by (2) sending a new MAP request and (3) receiving a MAP response. Instead, the server can create a shortcut for that design by simply sending the message it would have sent in (3).

To accommodate packet loss, the PCP server device SHOULD transmit such packets three times, with an appropriate Epoch Time value in each to reflect the passage of time between transmissions. The interval between the first two notifications MUST be at least 250 ms, and the third packet after a 500-ms interval. Once the PCP server has received a refreshed state for that mapping, the PCP server SHOULD cease those retransmissions for that mapping, as it serves no further purpose to continue sending messages regarding that mapping.

Upon receipt of such an updated MAP or PEER response, a PCP client uses the information in the response to adjust rendezvous servers or reconnect to servers, respectively. For MAP, this would mean updating the DNS entries or other address and port information

recorded with some kind of application-specific rendezvous server. For PEER responses giving a CANNOT_PROVIDE_EXTERNAL error, this would typically mean establishing new connections to servers. Anytime the external address or port changes, existing TCP and UDP connections will be lost; PCP can't avoid that, but does provide immediate notification of the event to lessen the impact.

15. 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 minimum value SHOULD be 120 seconds. The maximum value SHOULD be the remaining lifetime of the IP address assigned to the PCP client if that information is available (e.g., from the DHCP server), or half the lifetime of IP address assignments on that network if the remaining lifetime is not available, or 24 hours. Excessively long lifetimes can cause consumption of 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 MAP 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.

Upon receipt of a PCP response with an absurdly long assigned lifetime, the PCP client SHOULD behave as if it received a more sane value (e.g., 24 hours), and renew the mapping accordingly, to ensure that if the static mapping is removed, the client will continue to maintain the mapping it desires.

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 outbound TCP connections, which create implicit dynamic outbound 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.

To help clean PCP state, when the PCP-controlled device is collocated with the address assignment (DHCP) server, such as in a typical residential CPE, it is RECOMMENDED that when an IP address becomes invalid (e.g., the DHCP lease expires, or the DHCP client sends an explicit DHCP RELEASE) the PCP-controlled device SHOULD also discard any dynamic mapping state relating to that expired IP address.

When using NAT, the same external port may be assigned for use by different internal hosts at different times. For example, if an internal host using an external port ceases sending traffic using that port, then its mapping may expire, and then later the same external port may be assigned to a new internal host. The new internal host could then receive incoming traffic that was intended for the previous internal host. This generally happens inadvertently, and this reassignment of the external port only happens after the current holder of the external port has ceased using it for some period of time. It would be unacceptable if an attacker could use PCP to intentionally speed up this reassignment of the external port in order to deliberately steal traffic intended for the current holder, by (i) spoofing PCP requests using the current holder's source IP address and mapping nonce to fraudulently delete the mapping or shorten its lifetime, and then (ii) subsequently claiming the external port for itself.

Therefore, in the simple security model, to protect against this attack, PCP MUST NOT allow a PCP request (even a PCP request that appears to come from the current holder of the mapping) to cause a mapping to expire sooner than it would naturally have expired otherwise by virtue of outbound traffic keeping the mapping active. A PCP server MUST set the lifetime of a mapping to no less than the remaining time before the mapping would expire if no further outbound traffic is seen for that mapping. This means a MAP or PEER request with lifetime of 0 will only set the assigned lifetime to 0 (i.e., delete the mapping) if the internal host had not sent a packet using that mapping for the idle-timeout time, otherwise the assigned lifetime will be the remaining idle-timeout time.

Finally, to reduce unwanted traffic and data corruption for both TCP and UDP, the assigned external port created by the MAP Opcode or PEER Opcode SHOULD NOT be reused for an interval equal to the reuse time limit enforced by the NAT for its implicit dynamic mappings (typically, the maximum TCP segment lifetime of 2 minutes [RFC0793]). Furthermore, to reduce port stealing attacks, the assigned external port also SHOULD NOT be reused for an interval equal to the time the PCP- controlled device would normally maintain an idle (no traffic)

implicit dynamic mapping (e.g., 2 minutes for UDP [RFC4787] and 124 minutes for TCP [RFC5382]). However, within these time windows, the PCP server SHOULD allow an external port to be reclaimed by the same client, where "same client" means "same internal IP address, internal port, and mapping nonce".

15.1. Lifetime Processing for the MAP Opcode

If the requested lifetime is zero then:

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

- If the protocol is non-zero and the internal port is zero, it indicates a request to delete a previous 'wildcard' (all-ports) mapping for that protocol. The nonce MUST match the nonce used to create the 'wildcard' mapping.

- If both the protocol and internal port are zero, it indicates a request to delete a previous 'DMZ host' (all incoming traffic for all protocols) mapping. The nonce MUST match the nonce used to create the 'DMZ host' mapping.

- If the protocol is zero and the internal port is non-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.

PCP MAP requests can only delete or shorten lifetimes of MAP-created mappings. If the PCP client attempts to delete a static mapping (i.e., a mapping created outside of PCP itself), or an outbound (implicit or PEER-created) mapping, the PCP server MUST return NOT_AUTHORIZED. 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 return the same response for retransmissions or duplications of the same request). If the deletion request was properly formatted and successfully processed, a SUCCESS response is generated with the protocol and internal port number copied from the request, and the response lifetime set to zero. An inbound mapping (i.e., static mapping or MAP-created dynamic mapping) MUST NOT have its lifetime reduced by transport protocol messages (e.g., TCP RST, TCP FIN). Note the THIRD_PARTY option (Section 13.1), if authorized, can also delete PCP-created MAP mappings.

16. Implementation Considerations

Section 16 provides non-normative guidance that may be useful to implementers.

16.1. Implementing MAP with EDM Port-Mapping NAT

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

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

1. Have outbound mappings use a different set of external ports than inbound 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 a dynamic outbound mapping to process that packet. If none match, attempt to use an inbound mapping to process that packet. This effectively 'prioritizes' outbound mappings above inbound mappings.

16.2. Lifetime of Explicit and Implicit Dynamic Mappings

No matter if a NAT is EIM or EDM, it is possible that one (or more) outbound 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 an inbound mapping was created with the MAP Opcode, the implementation needs to ensure that termination of an outbound mapping (e.g., via a TCP FIN handshake) does not prematurely destroy the MAP-created inbound mapping.

16.3. PCP Failure Recovery

If an event occurs that causes the PCP server to lose dynamic mapping state (such as a crash or power outage), the mappings created by PCP are lost. Occasional loss of state may be unavoidable in a residential NAT device that does not write transient information to non-volatile memory. Loss of state is expected to be rare in a

service provider environment (due to redundant power, disk drives for storage, etc.). Of course, due to outright failure of service provider equipment (e.g., software malfunction), state may still be lost.

The Epoch time allows a client to deduce when a PCP server may have lost its state. When the Epoch Time 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 planned for a future document [PCP-FAIL].

16.3.1. Recreating Mappings

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; however, 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 time is reset and begins counting again. As the result of receiving a packet where the Epoch Time field is outside the expected range (Section 8.5), 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.

16.3.2. Maintaining Mappings

A PCP client refreshes a mapping by sending a new PCP request containing information learned 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 external IP address and/or external port, or the PCP server itself, has changed (due to a new route to a different PCP server). Such events are rare, but not an error. The PCP server will simply return a new external address and/or external port to the client, and the client should record this new external address and port with its rendezvous service. To detect such events more quickly, a server that requires extremely high availability may find it beneficial to use shorter lifetimes in its PCP mappings requests, so that it communicates with the PCP server more often. This is an engineering trade-off based on (i) the acceptable downtime for the service in question, (ii) the expected likelihood of NAT or firewall state loss, and (iii) the amount of PCP maintenance traffic that is acceptable.

If the PCP client has several mappings, the Epoch Time 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 time, it sends a PCP request for any one of the client's mappings. This will return the current Epoch Time 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 is different from the previous values (see Sections 10.1 and 11.5).

16.3.3. 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 [SCTPNAT]. Outbound dynamic SCTP mappings use the verification tag of the association instead of the local and remote peer port numbers. As with TCP, explicit outbound mappings can be made to reduce keepalive intervals, and explicit inbound mappings can be made by passive listeners expecting to receive new associations at the external port.

Because an SCTP-aware NAT does not (currently) rewrite SCTP port numbers, it will not be able to assign an external port that is different from the client's internal port. A PCP client making a MAP request for SCTP should be aware of this restriction. The PCP client SHOULD make its SCTP MAP request just as it would for a TCP MAP request: in its initial PCP MAP request it SHOULD specify zero for the external address and port, and then in subsequent renewals it SHOULD echo the assigned external address and port. However, since a current SCTP-aware NAT can only assign an external port that is the same as the internal port, it may not be able to do that if the external port is already assigned to a different PCP client. This is likely if there is more than one instance of a given SCTP service on the local network, since both instances are likely to listen on the same well-known SCTP port for that service on their respective hosts, but they can't both have the same external port on the NAT gateway's external address. A particular external port may not be assignable for other reasons, such as when it is already in use by the NAT device itself, or otherwise prohibited by policy, as described in Section 11.3, "Processing a MAP Request". In the event that the

external port matching the internal port cannot be assigned (and the SCTP-aware NAT does not perform SCTP port rewriting), the SCTP-aware NAT MUST return a CANNOT_PROVIDE_EXTERNAL error to the requesting PCP client. Note that this restriction places an extra burden on the SCTP server whose MAP request failed, because it then has to tear down its exiting listening socket and try again with a different internal port, repeatedly until it is successful in finding an external port it can use.

The SCTP complications described above occur because of address sharing. The SCTP complications are avoided when address sharing is avoided (e.g., 1:1 NAT, firewall).

16.4. Source Address Replicated in PCP Header

All PCP requests include the PCP client's IP address replicated in the PCP header. This is used to detect unexpected address rewriting (NAT) on the path 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 in the PCP header:

1. Create a UDP socket.
2. Call "connect" on this UDP socket using the address and port of the desired PCP server.
3. Call the getsockname() function to retrieve a sockaddr containing the source address 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 IPv4-mapped IPv6 address or the native IPv6 address into the PCP Client's IP Address field in the PCP header.
5. Send PCP requests using this connected UDP socket.

16.5. State Diagram

Each mapping entry of the PCP-controlled device would go through the state machine shown below. This state diagram is non-normative.

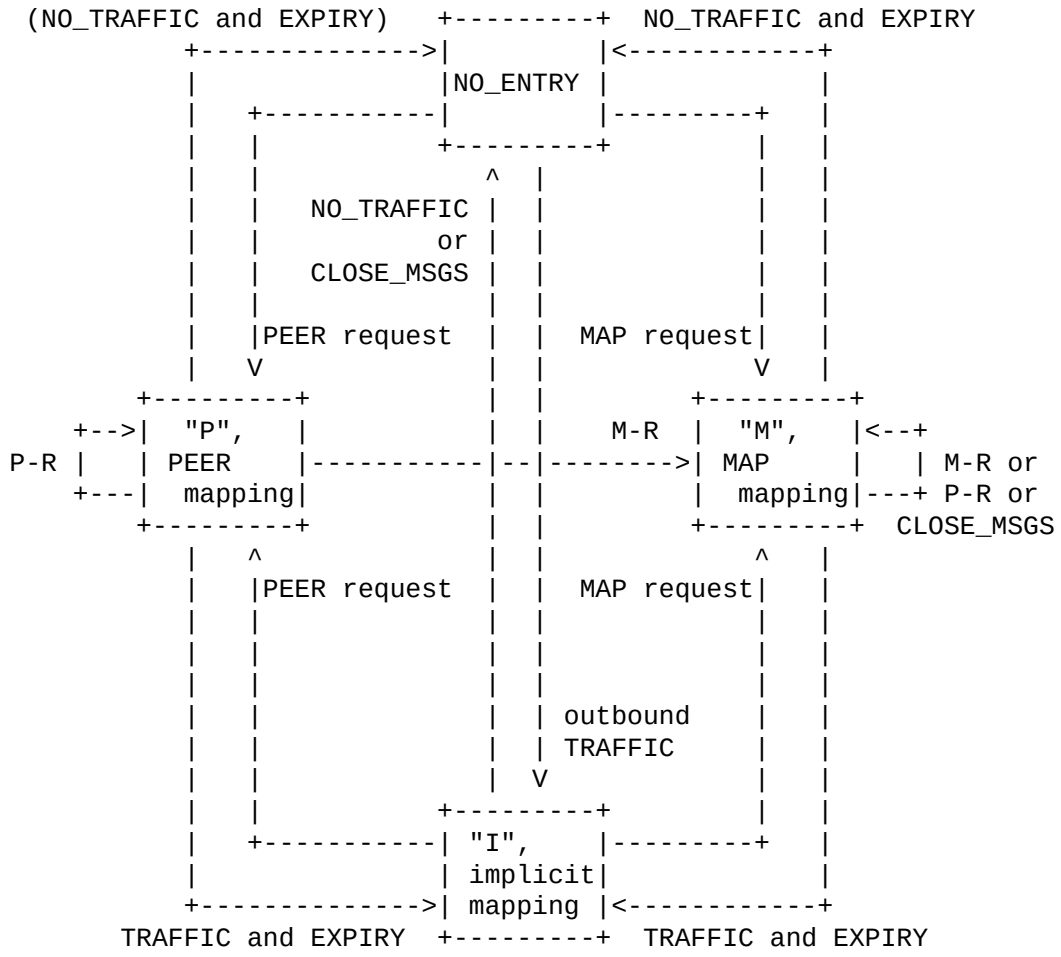

Figure 16: PCP State Diagram

The meanings of the states and events are:

NO_ENTRY: Invalid state represents Entry does not exist. This is the only possible start state.

M-R: MAP request

P-R: PEER request

M: Mapping entry when created by MAP request

P: Mapping entry when created/managed by PEER request

I: Implicit mapping created by an outgoing packet from the client (e.g., TCP SYN), and also the state when a PCP-created mapping's lifetime expires while there is still active traffic.

EXPIRY: PEER or MAP lifetime expired

TRAFFIC: Traffic seen by PCP-controlled device using this entry within the expiry time for that entry. This traffic may be inbound or outbound.

NO_TRAFFIC: Indicates that there is no TRAFFIC.

CLOSE_MSG: Protocol messages from the client or server to close the session (e.g., TCP FIN or TCP RST), as per the NAT or firewall device's handling of such protocol messages.

Notes on the diagram:

1. The 'and' clause indicates the events on either side of 'and' are required for the state-transition. The 'or' clause indicates either one of the events are enough for the state-transition.

2. Transition from state M to state I is implementation dependent.

17. Deployment Considerations

17.1. Ingress Filtering

As with implicit dynamic mappings created by outgoing TCP SYN 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.

17.2. 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.

18. 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 traveled by the mapped traffic.

18.1. Simple Threat Model

PCP servers are secure against off-path attackers who cannot spoof a packet that the PCP server will view as a packet received from the internal network. PCP clients are secure against off-path attackers who can spoof the PCP server's IP address.

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. Such an attacker can redirect traffic to a host of their choosing.

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 and MAP Opcodes can be secure under this threat model if (1) 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), (2) explicit mappings are created with the same lifetime as implicit mappings, and (3) the THIRD_PARTY option is not supported. 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 18.2 MUST enforce the constraints described in the paragraph above.

18.1.1. 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.

18.1.2. 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.

18.1.2.1. Residential Gateway Deployment

Parity with many currently deployed residential gateways can be achieved using a PCP server that is constrained as described in Section 18.1 above.

18.2. Advanced Threat Model

In the Advanced Threat Model, the PCP protocol ensures 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 serves multiple administrative domains and does 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 wish to support any deployment scenario that does not meet the constraints described in Section 18.1.

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

PCP servers that implement a PCP security mechanism MAY accept unauthenticated requests. In their default configuration, PCP servers implementing the PCP security mechanism MUST still enforce the constraints described in Section 18.1 when processing unauthenticated requests.

18.3. Residual Threats

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

18.3.1. 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 17.2 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 effectively deny service. Through such actions, the PCP client might not be aware the PCP server might have actually processed the PCP request. An attacker sending a NO_RESOURCES error can cause the PCP client to not send messages to that server for a while. There is no mitigation to this on-path attacker.

18.3.2. 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.

18.3.3. Mapping Theft

In the time between when a PCP server loses state and the PCP client notices the lower-than-expected Epoch Time 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"). Rapid recovery reduces this interval, but does 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 that the previous external IP address, protocol, and port cannot be used by another host (e.g., by using a different IP address pool).

18.3.4. Attacks against Server Discovery

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

19. IANA Considerations

IANA has performed the following actions.

19.1. Port Number

PCP uses ports 5350 and 5351, previously assigned by IANA to NAT-PMP [RFC6886]. IANA has reassigned those ports to PCP.

19.2. Opcodes

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

Value	Opcode
0	ANNOUNCE
1	MAP
2	PEER
3-31	Standards Action [RFC5226]
32-63	Specification Required [RFC5226]
96-126	Reserved for Private Use [RFC5226]
127	Reserved, Standards Action [RFC5226]

The value 127 is Reserved and may be assigned via Standards Action [RFC5226]. The values in the range 3-31 can be assigned via Standards Action [RFC5226], 32-63 via Specification Required [RFC5226], and the range 96-126 is for Private Use [RFC5226].

19.3. Result Codes

IANA has created a new registry for PCP result codes, numbered 0-255, initially populated with the result codes from Section 7.4. The value 255 is Reserved and may be assigned via Standards Action [RFC5226].

The values in the range 14-127 can be assigned via Standards Action [RFC5226], 128-191 via Specification Required [RFC5226], and the range 191-254 is for Private Use [RFC5226].

19.4. Options

IANA has created a new registry for PCP options, numbered 0-255, each 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 13. The option values 0, 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 Standards Action [RFC5226], the ranges 64-95 and 192-223 are for Specification Required [RFC5226], and the ranges 96-126 and 224-254 are for Private Use [RFC5226].

Documents describing an option should describe the processing for both the PCP client and server, and 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>
```

20. Acknowledgments

Thanks to Xiaohong Deng, Alain Durand, Christian Jacquenet, Jacni Qin, Simon Perreault, James Yu, Tina TSOU (Ting ZOU), Felipe Miranda Costa, James Woodyatt, Dave Thaler, Masataka Ohta, Vijay K. Gurbani, Loa Andersson, Richard Barnes, Russ Housley, Adrian Farrel, Pete Resnick, Pasi Sarolahti, Robert Sparks, Wesley Eddy, Dan Harkins, Peter Saint-Andre, Stephen Farrell, Ralph Droms, Felipe Miranda Costa, Amit Jain, and Wim Henderickx for their comments and review.

Thanks to Simon Perreault for highlighting the interaction of dynamic connections with PCP-created mappings and for many other review comments.

Thanks to Francis Dupont for his several thorough reviews of the specification, which improved the protocol significantly.

Thanks to T. S. Ranganathan for the state diagram.

Thanks to Peter Lothberg for clock skew information, which guided the choice of tolerance levels for deciding when an Epoch time should be considered to be anomalous.

Thanks to Margaret Wasserman and Sam Hartman for writing the Security Considerations section.

Thanks to authors of DHCPv6 for retransmission text.

21. References

21.1. 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.

[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.

[RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

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

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

[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>.

21.2. Informative References

[IGDv1] UPnP Gateway Committee, "WANIPConnection:1", November 2001, <http://upnp.org/specs/gw/UPnP-gw-WANIPConnection-v1-Service.pdf>.

[L2NAT] Miles, D. and M. Townsley, "Layer2-Aware NAT", Work in Progress, March 2009.

[PCP-FAIL] Boucadair, M., Dupont, F., and R. Penno, "Port Control Protocol (PCP) Failure Scenarios", Work in Progress, August 2012.

[PNP-IGD-PCP] Boucadair, M., Penno, R., and D. Wing, "Universal Plug and Play (UPnP) Internet Gateway Device (IGD)-Port Control Protocol (PCP) Interworking Function", Work in Progress, December 2012.

[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.

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

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

[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.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, March 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.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, 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.

[RFC6619] Arkko, J., Eggert, L., and M. Townsley, "Scalable Operation of Address Translators with Per-Interface Bindings", RFC 6619, June 2012.

[RFC6763] Cheshire, S. and M. Krochmal, "DNS-Based Service Discovery", RFC 6763, February 2013.

[RFC6886] Cheshire, S. and M. Krochmal, "NAT Port Mapping Protocol (NAT-PMP)", RFC 6886, April 2013.

[RFC6888] Perreault, S., Ed., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common Requirements for Carrier-Grade NATs (CGNs)", BCP 127, RFC 6888, April 2013.

[SCTPNAT] Stewart, R., Tuexen, M., and I. Ruengeler, "Stream Control Transmission Protocol (SCTP) Network Address Translation", Work in Progress, February 2013.

Appendix A. NAT-PMP Transition

The Port Control Protocol (PCP) is a successor to the NAT Port Mapping Protocol, NAT-PMP [RFC6886], 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 from NAT-PMP to PCP 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 UNSUPP_VERSION, as indicated by the NAT-PMP specification [RFC6886], which will cause the client to downgrade to NAT-PMP and resend 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.

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.com

Reinaldo Penno
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134
USA

EMail: repenno@cisco.com

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

EMail: pselkirk@isc.org