PCP working group
Internet-Draft
Intended status: Standards Track
Expires: September 15, 2011

D. Wing, Ed.
Cisco
S. Cheshire
Apple
M. Boucadair
France Telecom
R. Penno
Juniper Networks
F. Dupont
Internet Systems Consortium

March 14, 2011

Port Control Protocol (PCP)
draft-ietf-pcp-base-07

Abstract

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

Status of this Memo

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

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

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

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

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

This Internet-Draft will expire on September 15, 2011.

Copyright Notice

Table of Contents

1. Introduction 5
2. Scope 6
2.1. Deployment Scenarios 6
2.2. Supported Transport Protocols 6
2.3. Single-homed Customer Premises Network 6
3. Terminology 7
4. Relationship of PCP Server and its NAT 9
5. Common Request and Response Header Format 10
5.1. Request Header 10
5.2. Response Header 12
5.3. Options 13
5.4. Result Codes 15
6. General PCP Operation 16
6.1. General PCP Client: Generating a Request 16
6.2. General PCP Server: Processing a Request 17
6.3. General PCP Client: Processing a Response 18
6.4. Multi-Interface Issues 19
6.5. Epoch 19
6.6. Version negotiation 20
6.7. General PCP Options 21
6.7.1. UNPROCESSED 21
7. Introduction to MAP and PEER OpCodes 22
7.1. For Operating a Server 23
7.2. For Reducing NAT Keepalive Messages 23
7.3. For Operating a Symmetric Client/Server 24
8. MAP OpCodes 25
8.1. OpCode Packet Formats 26
8.2. OpCode-Specific Result Codes 28
8.3. OpCode-Specific Client: Generating a Request 29
8.4. OpCode-Specific Server: Processing a Request 30
8.5. OpCode-Specific Client: Processing a Response 31
8.6. Mapping Lifetime and Deletion 32

8.7. Subscriber Renumbering 34
8.8. PCP Options for MAP OpCodes 35
8.8.1. FILTER 35
8.8.2. PREFER_FAILURE 37
8.8.3. THIRD_PARTY 38
8.9. PCP Mapping State Maintenance 38
8.9.1. Recreating Mappings 38
8.9.2. Maintaining Mappings 39
9. PEER OpCodes 40
9.1. OpCode Packet Formats 40
9.2. OpCode-Specific Result Codes 43
9.3. OpCode-Specific Client: Generating a Request 43
9.4. OpCode-Specific Server: Processing a Request 44
9.5. OpCode-Specific Client: Processing a Response 44
9.6. PCP Options for PEER OpCodes 45
9.6.1. THIRD_PARTY 45
10. THIRD_PARTY Option for MAP and PEER OpCodes 45
11. NAT-PMP Transition 48
11.1. NAT-PMP Clients Updated to Add PCP Support 48
11.2. NAT-PMP Gateways Updated to Add PCP Support 49
12. Deployment Considerations 49
12.1. Maintaining Same External IP Address 49
12.2. Ingress Filtering 50
12.3. Per-Subscriber Port Forwarding Quota 50
13. Deployment Scenarios 50
13.1. Dual Stack-Lite 50
13.1.1. Overview 50
13.2. NAT64 51
13.3. NAT44 and NAT444 51
13.4. IPv6 Simple Firewall 51
14. Security Considerations 52
14.1. Denial of Service 52
14.2. Ingress Filtering 52
14.3. Validating the Internal Address 53
15. IANA Considerations 53
15.1. Port Number 53
15.2. OpCodes 53
15.3. Result Codes 53
15.4. Options 53
16. Acknowledgments 54
17. References 54
17.1. Normative References 54
17.2. Informative References 55
Appendix A. Change History 56
A.1. Changes from draft-ietf-pcp-base-06 to -07 56
A.2. Changes from draft-ietf-pcp-base-05 to -06 58
A.3. Changes from draft-ietf-pcp-base-04 to -05 59
A.4. Changes from draft-ietf-pcp-base-03 to -04 59

A.5. Changes from draft-ietf-pcp-base-02 to -03 60
A.6. Changes from draft-ietf-pcp-base-01 to -02 61
A.7. Changes from draft-ietf-pcp-base-00 to -01 61
Authors' Addresses 62

1. Introduction

Port Control Protocol (PCP) provides a mechanism to control how incoming packets are forwarded by upstream devices such as NAT64, NAT44, and firewall devices, and a mechanism to reduce application keepalive traffic. PCP is primarily designed to be implemented in the context of both Carrier-Grade NATs (CGN) and small NATs (e.g., residential NATs). PCP allows hosts to operate server for a long time (e.g., a webcam) or a short time (e.g., while playing a game or on a phone call) when behind a NAT device, including when behind a CGN operated by their Internet service provider.

PCP allows applications to create mappings from an external IP address and port to an internal IP address and port. These mappings are required for successful inbound communications destined to machines located behind a NAT or a firewall.

After creating a mapping for incoming connections, it is necessary to inform remote computers about the IP address and port for the incoming connection. This is usually done in an application-specific manner. For example, a computer game would use a rendezvous server specific to that game (or specific to that game developer), and a SIP phone would use a SIP proxy. PCP does not provide this rendezvous function. The rendezvous function will support IPv4, IPv6, or both. Depending on that support and the application’s support of IPv4 or IPv6, the PCP client will need an IPv4 mapping, an IPv6 mapping, or both.

Many NAT-friendly applications send frequent application-level messages to ensure their session will not be timed out by a NAT. These are commonly called "NAT keepalive" messages, even though they are not sent to the NAT itself (rather, they are sent ’through’ the NAT). These applications can reduce the frequency of those NAT keepalive messages by using PCP to learn (and influence) the NAT mapping lifetime. This helps reduce bandwidth on the subscriber’s access network, traffic to the server, and battery consumption on mobile devices.

Many NATs and firewalls have included application layer gateways (ALGs) to create mappings for applications that establish additional streams or accept incoming connections. ALGs incorporated into NATs additionally modify the application payload. Industry experience has shown that these ALGs are detrimental to protocol evolution. PCP allows an application create its own mappings in NATs and firewalls, reducing the incentive to deploy ALGs in NATs and firewalls.

2. Scope

2.1. Deployment Scenarios

PCP can be used in various deployment scenarios, including:

- Dual Stack-Lite [I-D.ietf-softwire-dual-stack-lite], and;
- NAT64, both Stateful [I-D.ietf-behave-v6v4-xlate-stateful] and Stateless [I-D.ietf-behave-v6v4-xlate], and;
- Carrier-Grade NAT [I-D.ietf-behave-lsn-requirements], and;
- Basic NAT [RFC3022], and;
- Network Address and Port Translation (NAPT) [RFC3022], such as commonly deployed in residential NAT devices, and;
- Layer-2 aware NAT [I-D.miles-behave-l2nat] and Dual-Stack Extra Lite [I-D.arkko-dual-stack-extra-lite], and;
- IPv6 firewall control [RFC6092].

2.2. Supported Transport Protocols

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

2.3. Single-homed Customer Premises Network

The PCP machinery assumes a single-homed host model. That is, for a given IP version, only one default route exists to reach the Internet. This is important because after a PCP mapping is created and an inbound packet (e.g., TCP SYN) arrives at the host, the outbound response (e.g., TCP SYNACK) has to go through the same path so the proper address rewriting takes place on that outbound response packet. This restriction exists because otherwise there would need to be one PCP server for each egress, because the host could not reliably determine which egress path packets would take, so the client would need to be able to reliably make the same internal/external mapping in every NAT gateway, which in general is not possible.

3. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in "Key words for use in RFCs to Indicate Requirement Levels" [RFC2119].

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

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

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

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

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

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

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

proxied (e.g., UPnP IGD to PCP proxy) or if the internal host does not implement PCP.

Mapping, Port Mapping, Port Forwarding:
A NAT mapping creates a relationship between an internal IP address, protocol, and port and an external IP address, protocol, and port. More specifically, it creates a translation rule where packets destined to the external IP and port are translated to the internal IP and port, and vice versa. In the case of a pure firewall, the "Mapping" is the identity function, translating an internal port number to the same external port number, and this "Mapping" indicates to the firewall that traffic to and from this internal port number is permitted to pass. The term "Port Forwarding" is sometimes used instead of "Port Mapping" in the case where the internal and external ports are the same, e.g. a mapping which forwards packets addressed to external address:port 192.0.2.1:12345 to internal address:port 192.168.1.1:12345.

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

PCP Client:
A PCP software instance responsible for issuing PCP requests to a PCP server. One or several PCP Clients can be embedded in the same host. Several PCP Clients can be located in the same local network. A PCP Client can issue PCP request on behalf of a third party device for which it is authorized to do so. An interworking

function, from UPnP IGD to PCP, or from NAT-PMP [I-D.cheshire-nat-pmp] is another example of a PCP Client. A PCP server in a NAT gateway that is itself a client of another NAT gateway (nested NAT) may itself act as a PCP client to the upstream NAT.

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

Interworking Function:
a functional element responsible for interworking another protocol with PCP. For example interworking between UPnP IGD [IGD] with PCP or NAT-PMP [I-D.cheshire-nat-pmp] and PCP.

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

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

5-tuple The 5 pieces of information that fully identify a flow: source IP address, destination IP address, protocol, source port number, destination port number.

4. Relationship of PCP Server and its NAT

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

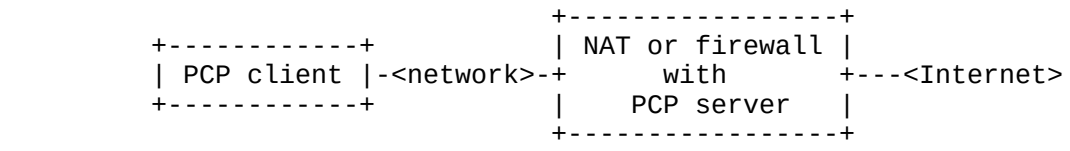


Figure 1: NAT or Firewall with Embedded PCP Server

It is also possible to operate the PCP server in a separate device from the NAT, so long as such operation is indistinguishable from the PCP client's perspective.

5. Common Request and Response Header Format

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

5.1. Request Header

All requests have the following format:

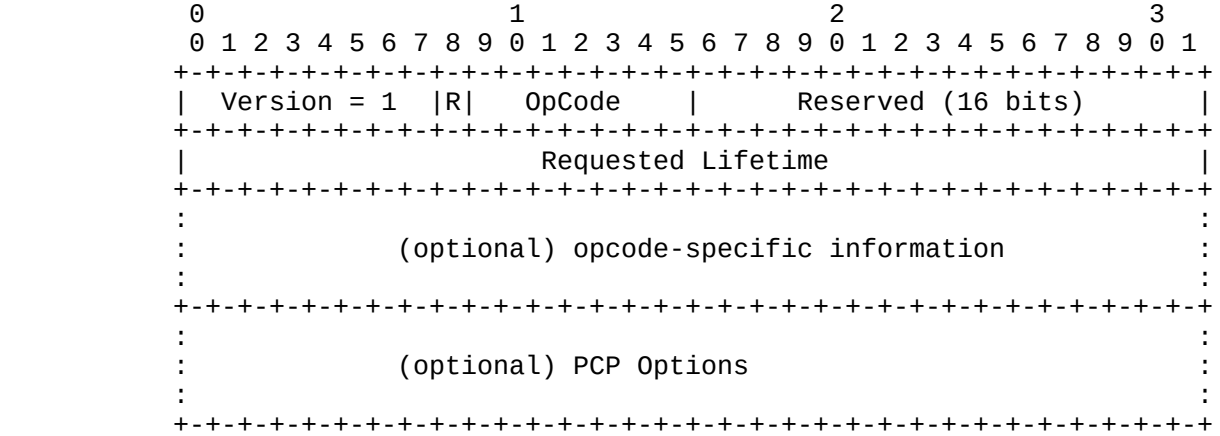


Figure 2: Common Request Packet Format

These fields are described below:

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

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

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

Reserved: 48 reserved bits, MUST be sent as 0 and MUST be ignored when received.

Requested Lifetime: The Requested Lifetime field is an unsigned 32-bit integer, in seconds, ranging from 0 to 4,294,967,295 seconds. A lifetime of zero is used to signify a "delete" operation. The currently-defined PCP opcodes -- MAP and PEER -- both have an associated lifetime, and it is likely that any future opcodes will also have a lifetime associated with them, so to simplify packet generation and parsing, this lifetime field is stored in a fixed location in the common request header. If future opcodes are defined that do not have a natural lifetime associated with them, then for these opcodes the Requested Lifetime MUST be set to zero on transmission and MUST be ignored on reception.

5.2. Response Header

All responses have the following format:

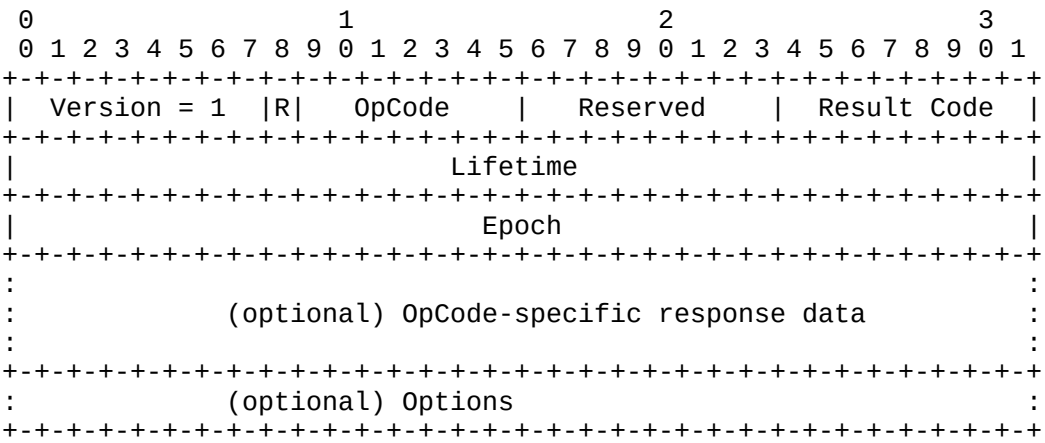


Figure 3: Common Response Packet Format

These fields are described below:

Version: Responses MUST use version 1.

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

OpCode: The OpCode value, copied from the request.

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

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

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

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

5.3. Options

A PCP OpCode can be extended with an Option. Options can be used in requests and responses. The decision about whether to include a given piece of information in the base opcode format or in an option is an engineering trade-off between packet size and code complexity. For information that is usually (or always) required, placing it in the fixed opcode data results in simpler code to generate and parse the packet, because the information is a fixed location in the opcode data, but wastes space in the packet in the event that that field is all-zeroes because the information is not needed or not relevant. For information that is required less often, placing it in an option results in slightly more compliated 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:

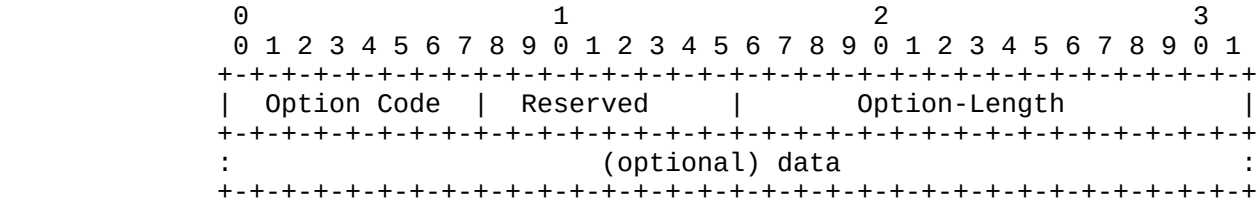


Figure 4: Options Header

The description of the fields is as follows:

Option Code: Option code, 8 bits. The first bit of the option code is the "O" (optional) bit. If clear, it indicates the option is mandatory to process (that is, non-optional). If set, it indicates the option is optional.

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

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

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

A given Option MAY be included in a request containing a specific OpCode. The handling of an Option by the PCP client and PCP server MUST be specified in an appropriate document and must include whether the PCP Option can appear (one or more times) in a request, and indicate the contents of the Option in the request and in the response. If several Options are included in a PCP request or response, they MUST be encoded in numeric order by the PCP client and are processed in the order received. The server MUST reject requests that have mis-ordered options with the MISORDERED_OPTIONS error, and this also includes checking optional-to-process options.

If, while processing an option, an error is encountered that causes a PCP error response to be generated, the PCP request causes no state change in the PCP server or the PCP-controlled device (i.e., it rolls back any changes it might have made while processing the request). The response MUST encode the Options in the same order, but may omit some PCP Options in the response, as is necessary to indicate the PCP server does not understand that Option or that Option is not permitted to be included in responses by the definition of the Option itself. Additional Options included in the response (if any) MUST be included at the end. A certain Option MAY appear more than once in a request or in a response, if permitted by the definition of the Option itself. If the Option's definition allows the Option to appear only once but it appears more than once in a request, the PCP server MUST respond with the MALFORMED_OPTION result code; if this occurs in a response, the PCP client processes the first occurrence and ignores the other occurrences as if they were not present.

If the "O" bit in the OpCode is clear,

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

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

If the "O" bit is set, the PCP server MAY process or ignore this Option, entirely at its discretion.

Option definitions MUST include the information below:

This Option:

name: <mnemonic>

number: <value>

purpose: <textual description>

is valid for OpCodes: <list of OpCodes>

length: <rules for length>

may appear in: <requests/responses/both>

maximum occurrences: <count>

5.4. Result Codes

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

0 SUCCESS, success

1 UNSUPP_VERSION, unsupported version.

2 MALFORMED_REQUEST, a general catch-all error.

3 UNSUPP_OPCODE, unsupported OpCode.

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

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

6 UNSPECIFIED_ERROR, server encountered unspecified error.

7 MISORDERED_OPTIONS, multiple options were in the request, but were not in the required lower..higher order.

Additional result codes, specific to the OpCodes and Options defined in this document, are listed in Section 8.2, Section 9.2, and Section 10.

6. General PCP Operation

PCP messages MUST be sent over UDP. Every PCP request generates a response, so PCP does not need to run over a reliable transport protocol.

PCP is idempotent, so if the PCP client sends the same request multiple times and the PCP server processes those requests, the same result occurs. The order of operation is that a PCP client generates and sends a request to the PCP server, which processes the request and generates a response back to the PCP client.

6.1. General PCP Client: Generating a Request

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

Prior to sending its first PCP message, the PCP client determines which servers to use. The PCP client performs the following steps to determine its PCP server(s):

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

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

[[[[The following text needs further review and consensus.

With that list of PCP servers, the PCP client formulates its PCP request. The PCP request contains a PCP common header, PCP OpCode and payload, and (possibly) Options. As with all UDP or TCP clients on any operating system, when several PCP clients are embedded in the same host, each uses a distinct source port number to disambiguate their requests and replies. The PCP client's source port SHOULD be randomly generated [RFC6056].

The PCP client initializes a timer to 2 seconds. The PCP client sends a PCP message the first server in the list. If no response is received before the timer expires, the timer is doubled (to 4 seconds) and the request is re-transmitted. If no response is received before the timer expires, the timer is doubled again (to 8 seconds) and the request is re-transmitted. If still no response is received, PCP client re-initializes its timer to 2 seconds, and repeats the procedure with the next PCP server on its list. This is repeated until a response is received or the until list of PCP servers is exhausted.

Once a PCP client has successfully communicated with a PCP server, it initializes its retransmission timer to 2 seconds. The PCP client ontinues communicating with that PCP server until a response is not received before the timer expires. When that occurs, the PCP client doubles its timer, and re-transmits the request.

]]]]

If, during its communication with the PCP server, the PCP client receives a hard ICMP error ([RFC5461] Section 2), the PCP client SHOULD immediately abort trying to contact that PCP server, initialize its retransmission timer, and try communicating with the next PCP server on its list.

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

6.2. General PCP Server: Processing a Request

This section details operation specific to a PCP server.

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

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

If the received message is shorter than 4 octets, has the R bit set,

or the first bit is clear, the message is simply dropped. If the version number is not supported, a response is generated containing the UNSUPP_VERSION result code and the protocol version which the server does understand (if the server understands a range of protocol versions then it returns the supported version closest to the version in the request). Version negotiation is detailed in Section 6.6.

If the OpCode is not supported, a response is generated with the UNSUPP_OPCODE result code. If the length of the request exceeds 1024 octets or is not a multiple of 4 octets, it is invalid. Invalid requests are handled by copying up to 1024 octets of the request into the response, setting the result code to MALFORMED_REQUEST, and zero-padding the response to a multiple of 4 octets if necessary.

Error responses have the same packet layout as success responses, with fields copied from the request copied into the response, and other fields assigned by the PCP server MUST be cleared to 0.

6.3. General PCP Client: Processing a Response

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

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

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

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

6.4. Multi-Interface Issues

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

IPv6 addresses with global reachability scope SHOULD be used as the source interface when generating a PCP request. IPv6 addresses with limited scope (e.g., ULA [RFC4193]), SHOULD NOT be used as the source interface when generating a PCP request. If IPv6 privacy addresses [RFC4941] are used for PCP mappings, a new PCP request will need to be issued whenever the IPv6 privacy address is changed. This PCP request SHOULD be sent from the IPv6 privacy address itself. It is RECOMMENDED that mappings to the previous privacy address be deleted.

A host might gain or lose interfaces while existing mappings are active (e.g., Ethernet cable plugged in or removed, joining/leaving a WiFi network). Because of this, if the PCP client is sending a PCP request to maintain state in the PCP server, it SHOULD ensure those PCP requests continue to use the same interface (e.g., when refreshing mappings). If the PCP client is sending a PCP request to create new state in the PCP server, it MAY use a different source interface or different source address.

Due to the ubiquity of IPv4 NAT, IPv4 addresses with limited scope (e.g., private addresses [RFC1918]) MAY be used as the source interface when generating a PCP request.

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

6.5. Epoch

Every PCP response sent by the PCP server includes an Epoch field. This field increments by 1 every second, and is used by the PCP client to determine if PCP state needs to be restored. If the PCP server resets or loses the state of its explicit dynamic Mappings (that is, those mappings created by PCP MAP requests), due to reboot, power failure, or any other reason, it MUST reset its Epoch time to 0. Similarly, if the public IP address(es) of the NAT (controlled by the PCP server) changes, the Epoch MUST be reset to 0. A PCP server MAY maintain one Epoch value for all PCP clients, or MAY maintain

distinct Epoch values for each PCP client; this choice is implementation-dependent.

Whenever a client receives a PCP response, the client computes its own conservative estimate of the expected Epoch value by taking the Epoch value in the last packet it received from the gateway and adding 7/8 (87.5%) of the time elapsed since that packet was received. If the Epoch value in the newly received packet is less than the client's conservative estimate by more than one second, then the client concludes that the PCP server lost state, and the client MUST immediately renew all its active port mapping leases as described in Section 8.9.1.

When the PCP server reduces its Epoch value, the PCP clients will send PCP requests to refresh their mappings. The PCP server needs to be scaled appropriately to accomodate this traffic. Because PCP lacks a mechanism to simultaneously inform all PCP clients of the Epoch value, the PCP clients will not flood the PCP server simultaneously when the PCP server reduces its Epoch value.

In the time between a PCP server loses state and the PCP client notices the lower-than-expected Epoch value, it is possible that the PCP client's mapping will be acquired by another host (via an explicit dynamic mapping or implicit dynamic mapping). This means incoming traffic will be sent to a different host. A mechanism to immediately inform the PCP client of state loss would reduce this interval, but would not eliminate this threat. The PCP client can reduce this interval by using a relatively short lifetime; however, this increases the amount of PCP chatter. The use of connection authentication between peers (e.g., TLS), or persistent storage of mappings in the PCP server (so it doesn't lose state) eliminates this threat.

6.6. Version negotiation

A PCP client sends its requests using PCP version number 1. Should later updates to this document specify different message formats with a version number greater than 1 it is expected that PCP servers will still support version 1 in addition to the newer version(s). However, in the event that a server returns a response with error code UNSUPP_VERSION, the client MAY log an error message to inform the user that it is too old to work with this server, and the client SHOULD set a timer to retry its request in 30 minutes (in case this was a temporary condition and the server configuration is changed to rectify the situation).

If future PCP versions greater than 1 are specified, version negotiation is expected to proceed as follows:

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

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

3. If server supports that version it responds normally.

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

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

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

6.7. General PCP Options

The following options can appear in certain PCP responses.

6.7.1. UNPROCESSED

If the PCP server cannot process a mandatory-to-process option, for whatever reason, it includes the UNPROCESSED Option in the response, shown in Figure 5. This helps with debugging interactions between the PCP client and PCP server. This option MUST NOT appear more than once in a PCP response. The unprocessed options are listed once, and the option data is zero-filled to the necessary 32 bit boundary. If a certain Option appeared more than once in the PCP request, that Option value only appears once in the option-code fields. The order of the Options in the PCP request has no relationship with the order of the Option values in this UNPROCESSED Option. This Option MUST NOT appear in a response unless the associated request contained at least one mandatory-to-process Option.

The UNPROCESSED option is formatted as follows, showing an example of two option codes that were unprocessed:

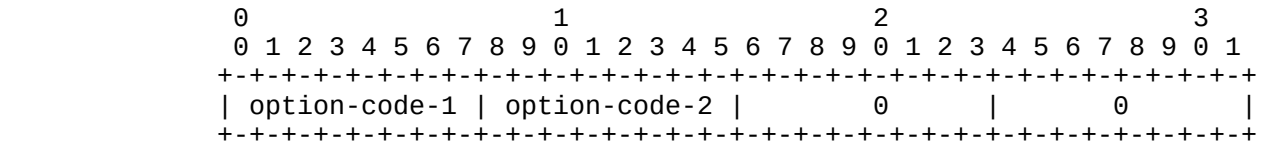


Figure 5: UNPROCESSED option

This Option:

name: UNPROCESSED

number: 1

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

is valid for OpCodes: all

length: 1 or more

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

maximum occurrences: 1

7. Introduction to MAP and PEER OpCodes

There are three uses for the MAP and PEER OpCodes defined in this document: a host operating a server (and wanting an incoming connection), a host operating a client (and wanting to optimize the application keepalive traffic), and a host operating a client and server on the same port. These are discussed in the following sections.

When operating a server (Section 7.1 and Section 7.3) the PCP client knows if it wants an IPv4 listener, IPv6 listener, or both on the Internet. The PCP client also knows if it has an IPv4 interface on itself or an IPv6 interface on itself. It takes the union of this knowledge to decide to send a one or two MAP requests for each of its interfaces. Applications that embed IP addresses in payloads (e.g., FTP, SIP) will find it beneficial to avoid address family translation, if possible.

7.1. For Operating a Server

A host operating a server (e.g., a web server) listens for traffic on a port, but the server never initiates traffic from that port. For this to work across a NAT or a firewall, the application needs to (a) create a mapping from a public IP address and port to itself as described in Section 8 and (b) publish that public IP address and port via some sort of rendezvous server (e.g., DNS, a SIP message, a proprietary protocol). Publishing the public IP address and port is out of scope of this specification. To accomplish (a), the application follows the procedures described in this section.

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

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

```
/* start listening on the local server port */
int s = socket(...);
internal_sockaddr = ...;
bind(s, &internal_sockaddr, ...);
listen(s, ...);
requested_external_sockaddr = 0;
pcp_send_map_request(internal_sockaddr,
   requested_external_sockaddr, &assigned_external_sockaddr,
   requested_lifetime, &assigned_lifetime);
update_rendezvous_server("Client 12345", assigned_external_sockaddr);
while (1) {
    int c = accept(s, ...);
    /* ... */
}
```

Figure 6: Pseudo-code for using PCP to operate a server

7.2. For Reducing NAT Keepalive Messages

A host operating a client (e.g., XMPP client, SIP client) sends from a port but never accepts incoming connections on this port. It wants to ensure the flow to its server is not terminated (due to inactivity) by an on-path NAT or firewall. To accomplish this, the applications uses the procedure described in this section.

Middleboxes such as NATs or firewalls need to see occasional traffic

or will terminate their session state, causing application failures. To avoid this, many applications routinely generate keepalive traffic for the primary (or sole) purpose of maintaining state with such middleboxes. Applications can reduce such application keepalive traffic by using PCP.

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

To use PCP for this function, the applications first connects to its server, as normal. Afterwards, it issues a PCP request with the PEER4 or PEER6 OpCode as described in Section 9. The PEER4 OpCode is used if the host is using IPv4 for its communication to its peer; PEER6 if using IPv6. The same 5-tuple as used for the connection to the server is placed into the PEER4 or PEER6 payload.

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

```
int s = socket(...);
connect(s, &remote_peer, ...);
getsockname(s, &internal_address, ...);
external_address = 0;
pcp_send_peer_request(internal_address,
   requested_external_address, &assigned_external_address,
   remote_peer, requested_lifetime, &assigned_lifetime);
```

Figure 7: Pseudo-code using PCP with a dynamic socket

7.3. For Operating a Symmetric Client/Server

A host operating a client and server on the same port (e.g., Symmetric RTP [RFC4961] or SIP Symmetric Response Routing (rport) [RFC3581]) first establishes a local listener, (usually) sends the local and public IP addresses and ports to a rendezvous service (which is out of scope of this document), and initiates an outbound connection from that same source address and same port. To accomplish this, the application uses the procedure described in this section.

An application that is using the same port for outgoing connections as well as incoming connections MUST first signal its operation of a server using the PCP MAP OpCode, as described in Section 8, and

receive a positive PCP response before it sends any packets from that port.

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

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

```
/* start listening on the local server port */
int s = socket(...);
internal_sockaddr = ...;
bind(s, &internal_sockaddr, ...);
listen(s, ...);
requested_external_sockaddr = 0;
pcp_send_map_request(internal_sockaddr,
   requested_external_sockaddr, &assigned_external_sockaddr,
   requested_lifetime, &assigned_lifetime);
update_rendezvous_server("Client 12345", assigned_external_sockaddr);
send_packet(s, "Hello World");
while (1) {
    int c = accept(s, ...);
    /* ... */
}
```

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

8. MAP OpCodes

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

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

MAP6=2: create a mapping between an internal target address and external IPv6 address (e.g., NAT46, NAT66, or firewall)

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

The operation of these OpCodes is described in this section.

8.1. OpCode Packet Formats

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

The following diagram shows the request packet format for MAP4 and MAP6. This packet format is aligned with the response packet format:

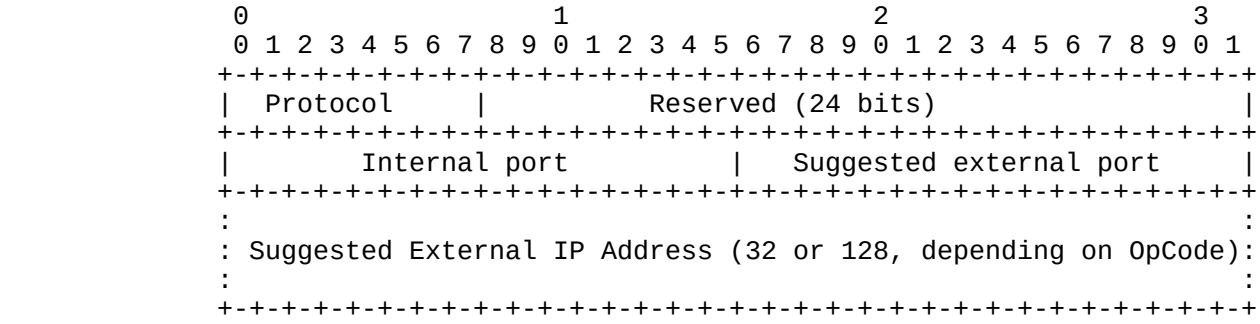


Figure 9: MAP OpCode Request Packet Format

These fields are described below:

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

Protocol: indicates protocol associated with this OpCode. Values are taken from the IANA protocol registry [proto_numbers]. For example, this field contains 6 (TCP) if the opcode is intended to create a TCP mapping. The value 0 has a special meaning for 'all protocols', and is used only for delete requests. This means that

HOPOPT (which is assigned by IANA as protocol 0) cannot have a mapping deleted by PCP.

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

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

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

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

The following diagram shows the response packet format for MAP4 and MAP6 OpCodes:

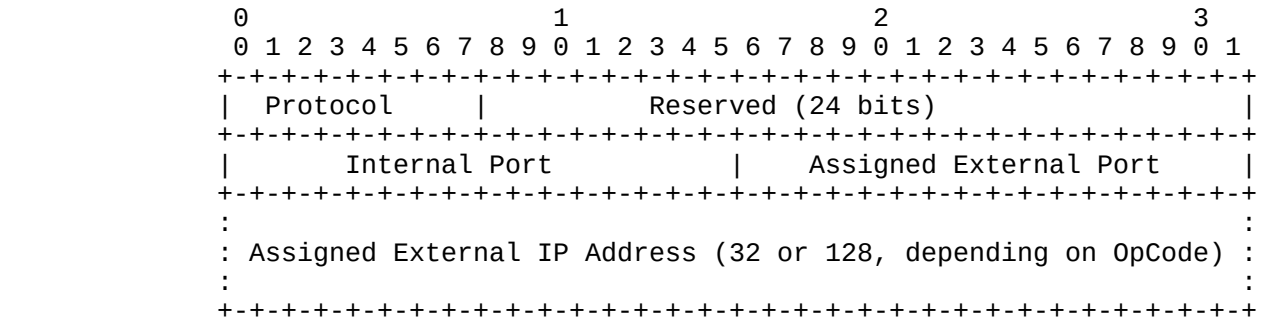


Figure 10: MAP OpCode Response Packet Format

These fields are described below:

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

Protocol: Copied from the request

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

Assigned External IP Address: On success responses, this is the assigned external IPv4 or IPv6 address for the mapping; IPv4 or IPv6 address is indicated by the OpCode. On error responses, this MUST be 0.

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

Assigned External Port: On success responses, this is the assigned external port for the mapping. If the NAT gateway can allocate the suggested external port it SHOULD do so. This is beneficial for re-establishing state lost when a NAT gateway fails or loses its state due to reboot. If the NAT gateway cannot allocate the suggested external port but can allocate some other port, it MUST do so and return the allocated port in the response. Cases where a NAT gateway cannot allocate the suggested external port include:

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

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

* When the suggested external port is otherwise prohibited by the NAT gateway's policy

On error responses, the Assigned External Port MUST be 0.

8.2. OpCode-Specific Result Codes

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

19 SERVER_OVERLOADED, server is processing too many MAP requests from this client or from other clients, and requests this client delay sending other requests. This is a short lifetime error.

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

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

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

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

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

25 CANNOT_PROVIDE_EXTERNAL_PORT, indicates the port is already in use (e.g. already allocated to a previous PCP client) or otherwise unavailable (e.g., special port that cannot be allocated by the server's policy). This error is only returned if the request included the Option PREFER_FAILURE. This is a short lifetime error.

26 EXCESSIVE_REMOTE_PEERS, indicates the PCP server was not able to create the filters in this request. This result code MUST only be returned if the MAP request contained the REMOTE_FILTER Option. This is a long lifetime error. This only occurs with the REMOTE_FILTER option.

27 IMPLICIT_MAPPING_EXISTS, indicates a MAP request was received for a port that already has an implicit mapping.

Additional result codes may be returned if the THIRD_PARTY option is used, see Section 10.

8.3. OpCode-Specific Client: Generating a Request

This section describes the operation of a PCP client when sending requests with OpCodes MAP4 and MAP6.

The request MAY contain values in the suggested-external-ip-address and suggested-external-port fields. This allows the PCP client to attempt to rebuild the PCP server's state, so that the PCP client could avoid having to change information maintained at the rendezvous server. Of course, due to other activity on the network (e.g., by

other users or network renumbering), the PCP server may not be able to fulfill the request.

An existing mapping can have its lifetime extended by the PCP client. To do this, the PCP client sends a new MAP request indicating the internal port. The PCP MAP request SHOULD also include the currently allocated external IP address and port as the suggested external IP address and port, so that if the NAT gateway has lost state it can recreate the lost mapping with the same parameters.

The PCP client SHOULD renew the mapping before its expiry time, otherwise it will be removed by the PCP server (see Section 8.6). In order to prevent excessive PCP chatter, it is RECOMMENDED to send a single renewal request packet when a mapping is halfway to expiration time, then, if no SUCCESS result is received, another single renewal request 3/4 of the way to expiration time, and then another at 7/8 of the way to expiration time, and so on, subject to the constraint that renewal requests MUST NOT be sent less than four seconds apart (a PCP client MUST NOT send an infinite number of ever-closer-together requests in the last few seconds before a mapping expires).

8.4. OpCode-Specific Server: Processing a Request

This section describes the operation of a PCP server when processing a request with the OpCodes MAP4 or MAP6.

If the server is overloaded by requests (from a particular client or from all clients), it MAY simply discard requests, as the requests will be retried by PCP clients, or MAY generate the SERVER_OVERLOADED error response, or both.

If the request contains internal-port=0 and the lifetime is non-zero, the server MUST generate a MALFORMED_REQUEST error.

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

Processing of the lifetime is described in Section 8.6.

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

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

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

If the THIRD_PARTY option is not present in the request, the source IP address of the PCP packet is used when creating the mapping. If the THIRD_PARTY option is present, the PCP server validates that the client is authorized to make mappings on behalf of the indicated internal IP address. This validation depends on the PCP deployment scenario; see Section 14.3 for the validation procedure. If the internal IP address in the PCP request is not authorized to make mappings on behalf of the indicated internal IP address, an error response MUST be generated with result code NOT_AUTHORIZED.

Mappings typically consume state on the PCP-controlled device, and it is RECOMMENDED that a per-subscriber or per-host limit be enforced by the PCP server to prevent exhausting the mapping state. If this limit is exceeded, the result code USER_EX_QUOTA is returned.

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

As a side-effect of creating a mapping, ICMP messages associated with the mapping MUST be forwarded (and also translated, if appropriate) for the duration of the mapping’s lifetime. This is done to ensure that ICMP messages can still be used by hosts, without application programmers or PCP client implementations needing to signal PCP separately to create ICMP mappings for those flows.

8.5. OpCode-Specific Client: Processing a Response

This section describes the operation of the PCP client when it receives a PCP response for the OpCodes MAP4 or MAP6.

A response is matched with a request by comparing the protocol, internal IP address, and internal port. Other fields are not compared, because the PCP server sets those fields.

If a successful response, the PCP client can use the external IP address and port(s) as desired. Typically the PCP client will

communicate the external IP address and port(s) to another host on the Internet using an application-specific rendezvous mechanism such as DNS SRV records.

If the result code is IMPLICIT_MAPPING_EXISTS, it indicates the PCP client is attempting to use MAP when an implicit dynamic connection already exists for the same internal host and internal port. This can occur with certain types of NATs. When this is received, if the PCP client still wants to establish a mapping, the PCP client MUST choose a different internal port and send a new PCP request specifying that port.

> [Editor's note: This is very bad. Imagine you have an ssh daemon listening internally on port 22, and then the PCP server tells you "IMPLICIT_MAPPING_EXISTS" (because a previous owner of that IP address made an outbound connection from port 22) so your ssh daemon has to listen on a nonstandard port instead. We need a better solution. Maybe we should have MAP requests trump implicit mappings? -- SC]

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

8.6. Mapping Lifetime and Deletion

The PCP client requests a certain lifetime, and the PCP server responds with the assigned lifetime. The PCP server MAY grant a lifetime smaller or larger than the requested lifetime. The PCP server SHOULD be configurable for permitted minimum and maximum lifetime, and the RECOMMENDED values are 120 seconds for the minimum value and 24 hours for the maximum. It is RECOMMENDED that the server restrict lifetimes to less than 24 hours, because they will consume ports even if the internal host is no longer interested in receiving the traffic or no longer connected to the network.

Once a PCP server has responded positively to a mapping request for a certain lifetime, the port forwarding is active for the duration of the lifetime unless the lifetime is reduced by the PCP client (to a shorter lifetime or to zero) or until the PCP server loses its state (e.g., crashes). This means that even if there is active traffic, the mapping will be deleted when its lifetime expires.

If the requested lifetime is 0 then:

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

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

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

The suggested external address and port fields are ignored in requests where the requested lifetime is 0.

PCP MAP requests cannot delete mappings created by non-MAP requests. If the PCP client attempts to delete a static mapping (i.e., a mapping created outside of PCP itself) or attempts to delete an implicit dynamic mapping (e.g., created by a TCP SYN), the PCP server deletes all of the mappings it can and responds with a zero error code. If the PCP client attempts to delete a mapping that does not exist, the SUCCESS result code is returned (this is necessary for PCP to be idempotent). If the deletion request was properly formatted, a SUCCESS response is generated with lifetime of 0 and the server copies the protocol and internal port number from the request into the response.

An application that forgets its PCP-assigned mappings (e.g., the application or OS crashes) will request new PCP mappings. This may consume port mappings, if the application binds to a different Internal Port every time it runs. The application will also likely initiate new implicit dynamic mappings (e.g., TCP connections) without using PCP, which will also consume port mappings. If there is a port mapping quota for the internal host, frequent restarts such as this may exhaust the quota. PCP provides some protections against such port consumption: When a PCP client first acquires a new IP address (e.g., reboots or joins a new network), it SHOULD remove mappings that may already be instantiated for that Internal Address. To do this, the PCP client sends a MAP request with protocol, internal port, and lifetime set to 0. Some port mapping APIs (such as the "DNSServiceNATPortMappingCreate" API provided by Apple's Bonjour on Mac OS X, iOS, Windows, Linux, etc.) automatically monitor for process exit (including application crashes) and automatically send port mapping deletion requests if the process that requested them goes away without explicitly relinquishing them.

In order to reduce unwanted traffic and data corruption, a port that was mapped using the MAP OpCode SHOULD NOT be assigned to another internal target, or another subscriber, for 120 seconds (MSL, [RFC0793]). However, the PCP server MUST allow the same internal

address to re-acquire the same port during that same interval.

The following list summarizes the sentinel values when deleting a mapping using lifetime=0:

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

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

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

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

8.7. Subscriber Renumbering

The customer premises router might obtain a new IPv4 address or new IPv6 prefix. This can occur because of a variety of reasons including a reboot, power outage, DHCP lease expiry, or other action by the ISP. If this occurs, traffic forwarded to the subscriber might be delivered to another customer who now has that address. This affects both implicit dynamic mappings and explicit dynamic mappings. However, this same problem occurs today when a subscriber's IP address is re-assigned, without PCP and without an ISP-operated CGN. The solution is the same as today: the problems associated with subscriber renumbering are caused by subscriber renumbering and are eliminated if subscriber renumbering is avoided. PCP defined in this document does not provide machinery to reduce the subscriber renumbering problem.

When a new Internal Address is assigned to a host embedding a PCP client, the NAT (or firewall) controlled by the PCP server will continue to send traffic to the old IP address. Assuming the PCP client wants to continue receiving traffic, it needs to install new mappings for its new IP address. The suggested external port field will not be fulfilled by the PCP server, in all likelihood, because it is still being forwarded to the old IP address. Thus, a mapping is likely to be assigned a new external port number and/or public IP address. Note that this scenario is not expected to happen routinely on a regular basis for most hosts, since most hosts renew their DHCP leases before they expire (or re-request the same address after reboot) and most DHCP servers honor such requests and grant the host the same address it was previously using before the reboot.

8.8. PCP Options for MAP OpCodes

8.8.1. FILTER

This Option indicates filtering incoming packets is desired. The remote peer port and remote peer IP Address indicate the permitted remote peer's source IP address and port for packets from the Internet. The remote peer prefix length indicates the length of the remote peer's IP address that is significant; this allows a single Option to permit an entire subnet. After processing this MAP request and generating a successful response, the PCP-controlled device will drop packets received on its public-facing interface with a source IP address (i.e., remote peer address), transport, or port that do not match the fields, and if its security policy allows MAY generate an ICMP error in response to that packet.

The FILTER packet layout is described below:

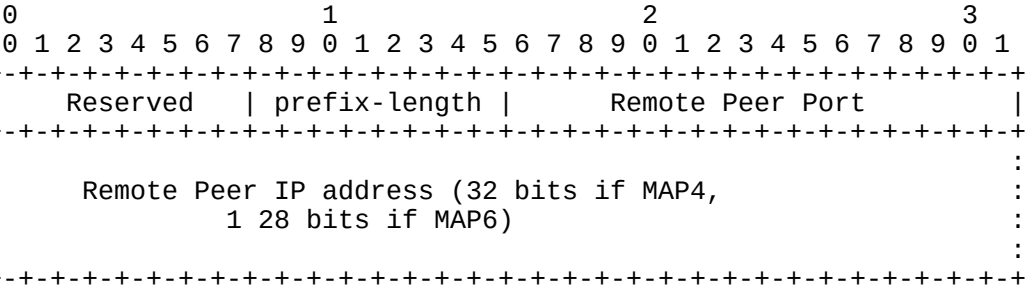


Figure 11: FILTER option layout

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.

This Option:

name: FILTER

number: 2

is valid for OpCodes: MAP4, MAP6

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

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

may appear in: requests

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

Because of interactions with dynamic ports this Option MUST only be used by a client that is operating a server (that is, using the MAP OpCode), as this ensures that no other application will be assigned the same ephemeral port for its outgoing connection. It is RECOMMENDED that the PCP client avoid other use, because it will cause some UNSAF NAT traversal mechanisms [RFC3424] to fail where they would have otherwise succeeded, breaking other applications running on this same host.

The prefix-length indicates how many bits of the IPv6 address or IPv4 address are used for the filter. For MAP4, a prefix-length of 32 indicates the entire IPv4 address is used. For MAP6, a prefix-length of 128 indicates the entire IPv6 address is used. For MAP4 the minimum prefix-length value is 0 and the maximum value is 32. For MAP6 the minimum prefix-length value is 0 and the maximum value is 128. Values outside those range cause an MALFORMED_OPTION result code.

If multiple occurrences of the FILTER option exist in the same MAP request, they are processed in the same order received, and they MUST all be successfully processed or return an error (e.g., MALFORMED_OPTION if one of the options was malformed). As with other PCP errors, returning an error causes no state to be changed in the PCP server or in the PCP-controlled device. If an existing mapping exists (with or without a filter) and the server receives a MAP request with FILTER, the filters indicated in the new request are added to any existing filters. If a MAP request has a lifetime of 0 and contains the FILTER option, the error MALFORMED_OPTION is returned.

To remove all existing filters, the prefix-length 0 is used. There is no mechanism to remove a specific filter.

To change an existing filter, the PCP client sends a MAP request containing two FILTER options, the first option containing a prefix-length of 0 (to delete all existing filters) and the second containing the new remote peer's IP address and port. Other FILTER options in that PCP request, if any, add more allowed remote hosts.

The PCP server or the PCP-controlled device is expected to have a limit on the number of remote peers it can support. This limit might be as small as one. If a MAP request would exceed this limit, the entire MAP request is rejected with the result code EXCESSIVE_REMOTE_PEERS, and the state on the PCP server is unchanged.

8.8.2. PREFER_FAILURE

This option indicates that if the PCP server is unable to allocate the suggested port, then instead of returning an available port that it *can* allocate, the PCP server should instead allocate no port and return result code CANNOT_PROVIDE_EXTERNAL_PORT.

This option is intended solely for use by UPnP IGD interworking [I-D.bpw-pcp-upnp-igd-interworking], where the semantics of IGD version 1 do not provide any way to indicate to an IGD client that any port is available other than the one it wanted. A PCP server MAY support this option, if its designers wish to support downstream devices that perform IGD interworking. PCP servers MAY choose to rate-limit their handling of PREFER_FAILURE requests, to protect themselves from a rapid flurry of 65535 consecutive PREFER_FAILURE requests from clients probing to discover which external ports are available. PCP servers that are not intended to support downstream devices that perform IGD interworking are not required to support this option. PCP clients other than IGD interworking clients SHOULD NOT use this option because it results in inefficient operation, and they cannot safely assume that all PCP servers will implement it. The option is provided only because the semantics of IGD version 1 offer no viable alternative way to implement an IGD interworking function. It is anticipated that this option will be deprecated in the future as more clients adopt PCP natively and the need for IGD interworking declines.

This Option:

name: PREFER_FAILURE

number: 3

is valid for OpCodes: MAP4, MAP6

is included in responses: MUST

length: 0

may appear in: requests

maximum occurrences: no

8.8.3. THIRD_PARTY

The THIRD_PARTY option is used by both the MAP OpCode and the PEER OpCode, and defined in Section 10.

8.9. PCP Mapping State Maintenance

If an event occurs that causes the PCP server to lose state (such as a crash or power outage), the mappings created by PCP are lost. Such loss of state is rare in a service provider environment (due to redundant power, disk drives for storage, etc.). But such loss of state is more common in a residential NAT device which does not write information to its non-volatile memory.

The Epoch allows a client to deduce when a PCP server may have lost its state. If this occurs, the PCP client can attempt to recreate the mappings following the procedures described in this section.

8.9.1. Recreating Mappings

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

However, maintaining this state is not essential for correct operation. When the PCP server loses state and begins processing new PCP messages, its Epoch is reset to zero (per the procedure of Section 6.5).

A mapping renewal packet is formatted identically to an original mapping request; from the point of view of the client it is a renewal of an existing mapping, but from the point of view of the PCP server it appears as a new mapping request. In the normal process of routinely renewing its mappings before they expire, a PCP client will automatically recreate all its lost mappings.

In addtion, as the result of receiving a packet where the Epoch field

indicates that a reboot or similar loss of state has occurred, the client can renew its port mappings sooner, without waiting for the normal routine renewal time.

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

8.9.2. Maintaining Mappings

A PCP client can refresh a mapping by sending a new PCP request containing information from the earlier PCP response. The PCP server will respond indicating the new lifetime. It is possible, due to failure of the PCP server, that the public IP address and/or public port, or the PCP server itself, has changed (due to a new route to a different PCP server). To detect such events more quickly, the PCP client may find it beneficial to use shorter lifetimes (so that it communicates with the PCP server more often). If the PCP client has several mappings, the Epoch value only needs to be retrieved for one of them to verify the PCP server has not lost port forwarding state.

If the client wishes to check the PCP server’s Epoch, it sends a PCP request for any one of the client’s mappings. This will return the current Epoch value. In that request the PCP client could extend the mapping lifetime (by asking for more time) or maintain the current lifetime (by asking for the same number of seconds that it knows are remaining of the lifetime).

If a internal IP address is no longer valid (e.g., because the internal host has moved to a new network), and the PCP client wishes to still receive incoming traffic, it MUST create a new mapping on that new network. A new mapping will also require an update to the

application-specific rendezvous server (Section 7.1).

9. PEER OpCodes

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

PEER4=3: Set or query lifetime for flow from IPv4 address to a remote peer's IPv4 address.

PEER6=4: Set or query lifetime for flow from IPv6 address to a remote peer's IPv6 address.

The operation of these OpCodes is described in this section.

9.1. OpCode Packet Formats

The PEER OpCodes provide a single function: the ability for the PCP client to query and (possibly) extend the lifetime of an existing mapping.

The two PEER OpCodes (PEER4 and PEER6) share a similar packet layout for both requests and responses. Because of this similarity, they are shown together.

The following diagram shows the request packet format for PEER4 and PEER6. This packet format is aligned with the response packet format:

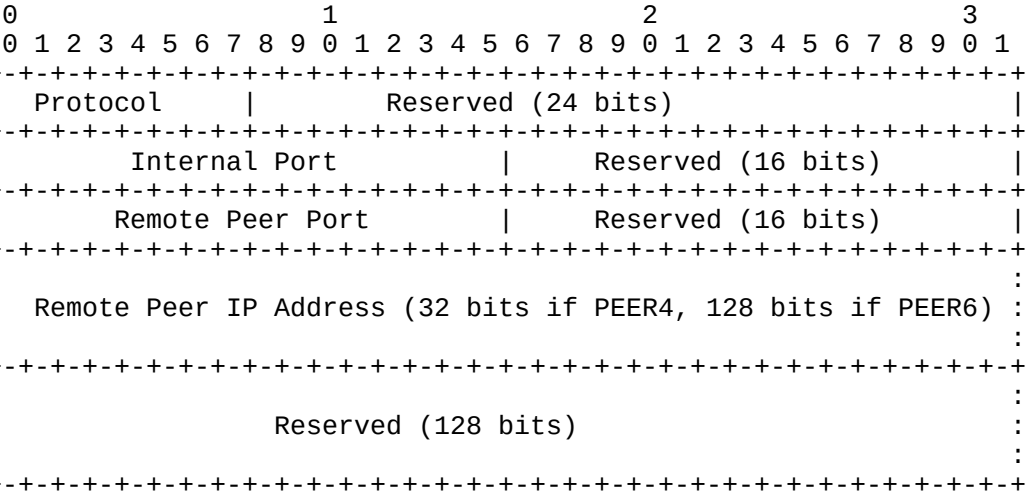


Figure 12: PEER OpCode Request Packet Format

These fields are described below:

Requested Lifetime (in common header): Requested lifetime of this mapping, in seconds. Unlike the MAP OpCode, where 0 means 'delete', there is no special meaning of 0, and the PCP client cannot reduce the lifetime of an implicit dynamic connection (Section 9.4).

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

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

Internal Port: Internal port for the of the 5-tuple.

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

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

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

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

Reserved: 128 reserved bits, MUST be 0 on transmission and MUST be ignored on reception.

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

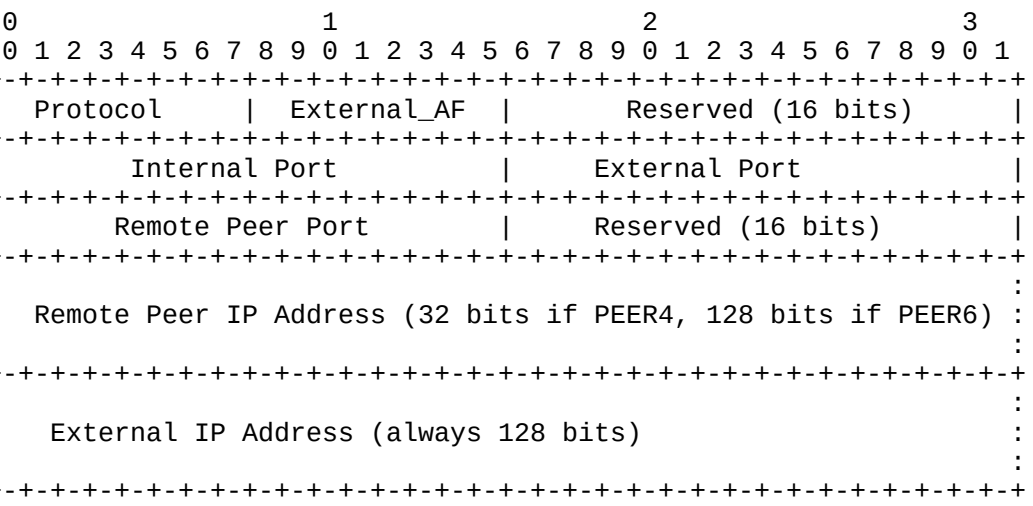


Figure 13: PEER OpCode Response Packet Format

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

Protocol: Copied from the request.

External_AF For success responses, this contains the address family of the external IP address associated with this peer connection, to properly decode the External IP Address. Values are from IANA's address family numbers (IPv4 is 1, IPv6 is 2). For error responses, the value MUST be 1.

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

Internal Port: copied from request.

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

Remote Peer port: Copied from request.

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

Remote Peer IP Address Copied from the request.

External IP Address For success responses, this contains the external IP address, assigned by the NAT (or firewall) to this mapping. This field allows the PCP client and its remote peer to determine if there is another NAT between the PCP-controlled NAT and remote peer. If the PCP-controlled device is a firewall, this will match the internal IP address. This field is always 128 bits long. If External_AF indicates IPv4, the IPv4 address is encoded in the first 32 bits of the External IP Address field and the remaining 96 bits are zero. the remote peer.

9.2. OpCode-Specific Result Codes

In addition to the general PCP result codes (Section 5.4) the following additional result codes may be returned as a result of the two PEER OpCodes received by the PCP server.

50 NONEXIST_PEER, the connection to that peer does not exist in the mapping table.

> [Editor's Note: Maybe it should just go ahead and make the mapping, instead of complaining about it? -- SC]

Additional result codes may be returned if the THIRD_PARTY option is used, see Section 10.

9.3. OpCode-Specific Client: Generating a Request

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

The PEER4 or PEER6 OpCodes MUST NOT be sent until establishing bi-directional communication with the remote peer. For TCP, this means completing the TCP 3-way handshake. This is because the PCP-controlled device may not be able to extend the lifetime of a mapping until after bi-directional communications has been established. The Internal Address for the PEER request is the the PEER request's source IP address. [Why not? Why not allow the PEER request to make a mapping, just like a TCP SYN does?]

The PEER4 and PEER6 OpCodes contain a description of the remote peer address, from the perspective of the PCP client. This is important

when the PCP-controlled device is performing address family translation (NAT46 or NAT64), because the destination address from the perspective of the PCP client is different from the destination address on the other side of the address family translation device.

9.4. OpCode-Specific Server: Processing a Request

This section describes the operation of a server when receiving a request with the OpCodes PEER4 or PEER6.

On receiving the PEER4 or PEER6 OpCode, the PCP server examines the mapping table. If a mapping does not exist, the NONEXIST_PEER error is returned. [Why not? Why not allow the PEER request to make a mapping, just like a TCP SYN does?]

The PEER4 or PEER6 OpCodes MUST NOT reduce the lifetime of an existing mapping. If the mapping is terminated by the TCP client or server (e.g., TCP FIN or TCP RST), the mapping will eventually be destroyed normally; the earlier use of PEER does not extend the lifetime in that case.

If the PCP-controlled device can extend the lifetime of a mapping, the PCP server uses the smaller of its configured maximum lifetime value and the requested lifetime from the PEER request, and sets the lifetime to that value.

If all of the proceeding operations were successful (did not generate an error response), then a SUCCESS response is generated, with the assigned-lifetime containing the lifetime of the mapping.

After a successful PEER response is sent, it is implementation-specific if the PCP-controlled device destroys the mapping when the lifetime expires, or if inside->outside traffic keeps the mapping alive. Thus, if the PCP client wants the mapping to persist beyond the lifetime, it MUST refresh the mapping (by sending another PEER message) prior to the expiration of the lifetime.

9.5. OpCode-Specific Client: Processing a Response

This section describes the operation of a client when processing a response with the OpCodes PEER4 or PEER6.

A response is matched with a request by comparing the protocol, external AF, internal IP address, internal port, remote peer address and remote peer port. Other fields are not compared, because the PCP server changes those fields to provide information about the mapping created by the OpCode.

If the error response NONEXIST_PEER is received, this could have occurred if the PCP client sent its PEER request before the PCP-controlled device had installed the mapping, or because the mapping has been destroyed (e.g., due to a TCP FIN). If the PCP client believes the mapping should exist, the PCP client SHOULD retry the request after a brief delay (e.g., 5 seconds).

Other error responses SHOULD NOT be retried.

If a successful response, the PCP client uses 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 ensure the server is still accessible (e.g., has not crashed) more frequently than once an hour.

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

9.6. PCP Options for PEER OpCodes

9.6.1. THIRD_PARTY

The THIRD_PARTY option is used by both the MAP OpCode and the PEER OpCode, and defined in Section 10.

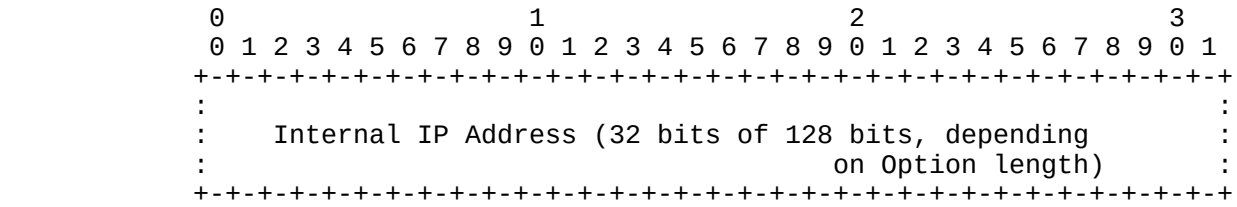
10. THIRD_PARTY Option for MAP and PEER OpCodes

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

A THIRD_PARTY Option MUST NOT contain the same address as the source address of the packet. A PCP server receiving a THIRD_PARTY Option specifying the same address as the source address of the packet MUST return a MALFORMED_REQUEST result code. This is because many PCP

servers may not implement the THIRD_PARTY Option at all, and a client using the THIRD_PARTY Option to specify the same address as the source address of the packet will cause mapping requests to fail where they would otherwise have succeeded.

A PCP server will only process this option if sent by an authorized PCP client, otherwise will return an error. Determining which PCP clients are authorized to use the THIRD_PARTY option depends on the deployment scenario. For Dual-Stack Lite deployments, the PCP server only supports this option if the source IPv6 address is the B4's source IP address. For other scenarios, the subscriber has only one IPv4 address and this Option serves no purpose (and will only generate error messages from the server). If a subscriber has more than one IPv4 address (from the same ISP, often called "business-class"), the ISP MUST determine its own policy for how to identify the trusted device within the subscriber's home. This might be, for example, the lowest- or highest-numbered host address for that user's IPv4 prefix. On the other hand, some credible cryptographic security could be used to determine whether a PCP client is authorized to make or delete mappings on behalf of a given Internal Address.



THIRD_PARTY option packet format

The fields are described below:

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

This Option:

name: THIRD_PARTY

number: 4

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

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

length: 4 if OpCode is MAP4 or PEER4, 16 if OpCode is MAP6 or PEER6

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

maximum occurrences: 1

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

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

A PCP server is configured to permit or to restrict the use of the THIRD_PARTY option. If this option is permitted, any host can create, modify, or destroy mappings for another host on the network, which is generally undesirable. If third party mappings are restricted, only a authorized clients can perform these operations. If a PCP server is configured to restrict third party mappings, and receives a PCP MAP request with a THIRD_PARTY option, it MUST generate a UNAUTH_TARGET_ADDRESS response.

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

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

message was sent by the IP address of the subscriber's customer premise router. In Dual-Stack Lite, this would be the B4 element's IPv6 address. If the packet arrived from a different address, the PCP server MUST generate an UNAUTH_TARGET_ADDRESS error.

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

11. NAT-PMP Transition

Port Control Protocol (PCP) is a successor to NAT Port Mapping Protocol (NAT-PMP), and shares similar semantics, concepts, and packet formats. Because of this NAT-PMP and PCP can both use the same port, and use the protocol's built-in version negotiation capabilities to determine which version to use. It is hoped that in relatively short time most shipping NAT-PMP clients and gateways will be updated to support PCP as well, but there will be a transition period. During this transition period developers updating NAT-PMP clients to add PCP will still want to work with existing NAT-PMP gateways, and developers updating NAT-PMP gateways will still want to support existing NAT-PMP clients. This section describes how an orderly transition may be achieved.

11.1. NAT-PMP Clients Updated to Add PCP Support

A client supporting both NAT-PMP and PCP SHOULD optimistically assume that the gateway supports PCP, since we expect that this will rapidly become the case, and we want to optimize for better performance in this case. A dual-mode client SHOULD send all its requests first using PCP packet format. If the gateway responds with a packet four or more bytes long, containing the following (NAT-PMP format) data in the first four bytes, then the dual-mode client SHOULD conclude that this NAT gateway supports only NAT-PMP, and SHOULD retry its request in the older NAT-PMP format.

NAT-PMP gateways respond to PCP requests with the following packet. The first byte (supported version) is zero. The second byte (opcode) echoes back the request opcode, with the top bit set. The third byte (high byte of the NAT-PMP error code) is zero. The fourth byte is 1 (NAT-PMP and PCP error code "Unsupported Version").

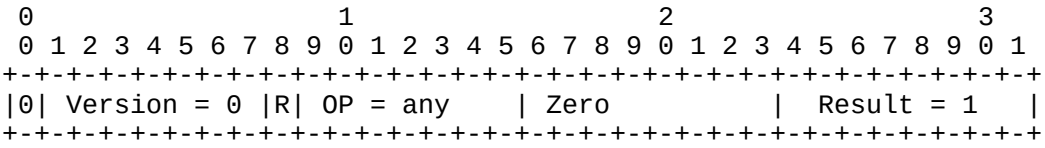


Figure 14: NAT-PMP Gateway Response to PCP Request

11.2. NAT-PMP Gateways Updated to Add PCP Support

A gateway supporting both NAT-PMP and PCP is able to handle and respond to requests using both packet formats. If the first byte of the packet is zero, a dual-mode gateway SHOULD parse the request as a NAT-PMP-format message and reply using a NAT-PMP-format response. Otherwise it should parse the request as a PCP-format message and respond accordingly.

A PCP-only gateway receiving a NAT-PMP request (identified by the first byte being zero) MUST reply with the packet shown below, so that the NAT-PMP may log an error message informing the user that they need to update to a PCP-capable client.

PCP gateways respond to NAT-PMP requests (identified by the first byte being zero) with the following packet. The first byte (supported version) is 1. The second byte (opcode) echoes back the request opcode, with the top bit set. The third byte (high byte of the NAT-PMP error code) is zero. The fourth byte is 1 (NAT-PMP and PCP error code "Unsupported Version").

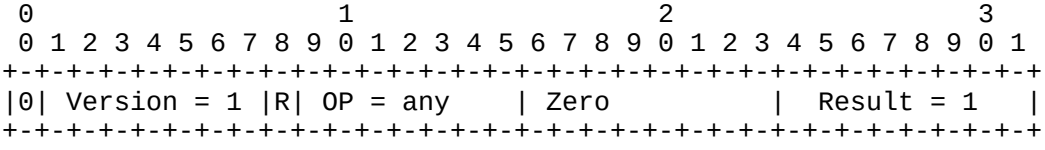


Figure 15: PCP Gateway Response to NAT-PMP Request

12. Deployment Considerations

12.1. Maintaining Same External IP Address

It is REQUIRED that the PCP-controlled device assign the same external IP address PCP-created explicit dynamic mappings and to implicit dynamic mappings. It is RECOMMENDED that static mappings (e.g., those created by a command-line interface on the PCP server or PCP-controlled device) also be assigned to the same IP address.

Once all internal hosts belonging to a given subscriber have no implicit dynamic mappings and have no explicit dynamic mappings in the PCP-controlled device, a subsequent PCP request for that internal host MAY be assigned to a different external IP address. Generally, this re-assignment would occur when a CGN device is load balancing newly-seen hosts to its public IPv4 address pool.

12.2. Ingress Filtering

To prevent spoofing of PCP requests, ingress filtering [RFC2827] MUST be performed by devices between the PCP clients and PCP server. For example, with a PCP server integrated into a customer premise router, the Ethernet switch needs to perform ingress filtering. As another example, with a PCP server deployed by a service provider, the service provider's aggregation router (the first device connecting to subscribers) needs to do ingress filtering.

12.3. Per-Subscriber Port Forwarding Quota

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

13. Deployment Scenarios

13.1. Dual Stack-Lite

The interesting components in a Dual-Stack Lite deployment are the B4 element (which is the customer premises router) and the AFTR (Address Family Transition Router) element. The AFTR element terminates the IPv6-over-IPv4 tunnel and also implements the Carrier-Grade NAT44 function. The B4 element does not need to perform a NAT function (and usually does not perform a NAT function), but it does operate its own DHCP server and is the local network's default router.

13.1.1. Overview

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

1. UPnP IGD and NAT-PMP [I-D.cheshire-nat-pmp] are used in the LAN: an interworking function is required to be embedded in the B4 element to ensure interworking between the protocol used in the LAN and PCP. UPnP IGD-PCP Interworking Function is described in

[I-D.bpw-pcp-upnp-igd-interworking].

2. Hosts behind the B4 element will either include a PCP client or UPnP IGD client, or both.

   A. if a UPnP IGD client, the B4 element will need to include an interworking function from UPnP IGD to PCP.

   B. if a PCP client, the PCP client will communicate directly with the PCP server.

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

In Dual Stack-Lite, the B4 element encapsulates its PCP messages into the IPv6 tunnel towards the AFTR element. It is expected the B4 element will also perform as a proxy from PCP to PCP [I-D.bpw-pcp-proxy], and may also proxy from other protocols to PCP (e.g., [I-D.bpw-pcp-upnp-igd-interworking]. When proxying for other hosts, the B4 element might have to use the THIRD_PARTY option with the MAP and PEER OpCodes if it modifies the packet's source address before forwarding it upstream.

13.2. NAT64

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

13.3. NAT44 and NAT444

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

13.4. IPv6 Simple Firewall

Many IPv6 deployments will include a simple firewall [RFC6092], which permits outgoing packets to initiate bi-directional communication but blocks unsolicited incoming packets, which is similar to PCP's security model that allows a host to create a mapping to itself. In many situations, especially residential networks that lack an IT

staff, the security provided by an IPv6 simple firewall and the security provided by PCP are compatible. In such situations, the IPv6 simple firewall and the IPv6 host can use the MAP6 OpCode to allow unsolicited incoming packets, so the host can operate a server.

14. Security Considerations

This document defines Port Control Protocol and two types of OpCodes, PEER and MAP. The PEER OpCode allows querying and extending (if permitted) the lifetime of an existing implicit dynamic mapping, so a host can reduce its keepalive messages. The MAP OpCode allows creating a mapping so a host can receive incoming unsolicited connections from the Internet in order to run a server.

The PEER OpCode does not introduce any new security considerations.

On today's Internet, ISPs do not typically filter incoming traffic for their subscribers. However, when an ISP introduces stateful address sharing with a NAPT device, such filtering will occur as a side effect. Filtering will also occur with IPv6 CPE [RFC6092]. The MAP OpCode allows a PCP client to create a mapping so that a host can receive inbound traffic and operate a server. Security considerations for the MAP OpCode are described in the following sections.

14.1. Denial of Service

Because the state created in a NAPT or firewall, a per-subscriber quota will likely exist for both implicit dynamic mappings (e.g., outgoing TCP connections) and explicit dynamic mappings (PCP). A subscriber might make an excessive number of implicit or explicit dynamic mappings, consuming an inordinate number of ports, causing a denial of service to other subscribers. Thus, Section 12.3 recommends that subscribers be limited to a reasonable number of explicit dynamic mappings.

14.2. Ingress Filtering

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

14.3. Validating the Internal Address

The THIRD_PARTY Option contains a Internal Address field, which allows a PCP client to create an explicit dynamic mapping for another host. Hosts within a subscriber's network cannot create, modify, or delete mappings of other hosts, except by using the administrative interface of the customer premise router (e.g., HTTP interface), as described in Section 10.

15. IANA Considerations

IANA is requested to perform the following actions:

15.1. Port Number

IANA has assigned UDP port 44323 for PCP.

15.2. OpCodes

IANA shall create a new protocol registry for PCP OpCodes, initially populated with the values in Section 8 and Section 9. The values 0 and 128 are reserved.

Additional OpCodes in the range 4-95 can be created via Standards Action [RFC5226], and the range 96-127 is for Private Use [RFC5226].

15.3. Result Codes

IANA shall create a new registry for PCP result codes, numbered 0-255, initially populated with the result codes from Section 5.4, Section 8.2, Section 8.8.1, Section 9.2, and Section 10. The values 0 and 255 are reserved.

Additional Result Codes can be defined via Specification Required [RFC5226].

15.4. Options

IANA shall create a new registry for PCP Options, numbered 0-255 with an associated mnemonic. The values 0-128 are mandatory-to-process, and 128-255 are optional-to-process. The initial registry contains the options described in Section 8.8 and Section 10. The option values values 0 and 255 are reserved.

Additional PCP option codes in the ranges 5-63 and 128-191 can be created via Standards Action [RFC5226], and the ranges 64-127 and 192-255 are for Private Use [RFC5226].

16. Acknowledgments

Thanks to Alain Durand, Christian Jacquenet, Jacni Qin, Simon Perreault, Paul Selkirk, and James Yu for their comments and review. Thanks to Simon Perreault for highlighting the interaction of dynamic connections with PCP-created mappings.

17. References

17.1. Normative References

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

[I-D.ietf-behave-v6v4-xlate-stateful]
Bagnulo, M., Matthews, P., and I. Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", draft-ietf-behave-v6v4-xlate-stateful-12 (work in progress), July 2010.

[I-D.ietf-softwire-dual-stack-lite]
Durand, A., Droms, R., Woodyatt, J., and Y. Lee, "Dual-Stack Lite Broadband Deployments Following IPv4 Exhaustion", draft-ietf-softwire-dual-stack-lite-07 (work in progress), March 2011.

[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.

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

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

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

[proto_numbers]
IANA, "Protocol Numbers", 2010, <http://www.iana.org/assignments/protocol-numbers/protocol-numbers.xml>.

17.2. Informative References

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

[I-D.bpw-pcp-proxy]
Boucadair, M., Penno, R., Wing, D., and F. Dupont, "Port Control Protocol (PCP) Proxy Function", draft-bpw-pcp-proxy-00 (work in progress), February 2011.

[I-D.bpw-pcp-upnp-igd-interworking]
Boucadair, M., Penno, R., Wing, D., and F. Dupont, "Universal Plug and Play (UPnP) Internet Gateway Device (IGD)-Port Control Protocol (PCP) Interworking Function", draft-bpw-pcp-upnp-igd-interworking-02 (work in progress), February 2011.

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

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

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

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

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

[RFC1918] Rekhter, Y., Moskowitz, R., Karrenberg, D., Groot, G., and E. Lear, "Address Allocation for Private Internets",

BCP 5, RFC 1918, February 1996.

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

[RFC3424] Daigle, L. and IAB, "IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation", RFC 3424, November 2002.

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

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

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

[RFC5461] Gont, F., "TCP's Reaction to Soft Errors", RFC 5461, February 2009.

[RFC6092] Woodyatt, J., "Recommended Simple Security Capabilities in Customer Premises Equipment (CPE) for Providing Residential IPv6 Internet Service", RFC 6092, January 2011.

Appendix A. Change History

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

A.1. Changes from draft-ietf-pcp-base-06 to -07

- tightened up THIRD_PARTY security discussion. Removed "highest numbered address", and left it as simply "the CPE's IP address".
- removed UNABLE_TO_DELETE_ALL error.
- renumbered Opcodes
- renumbered some error codes
- assigned value to IMPLICIT_MAPPING_EXISTS.

- UNPROCESSED can include arbitrary number of option codes.
- Moved lifetime fields into common request/response headers
- We've noticed we're having to repeatedly explain to people that the "requested port" is merely a hint, and the NAT gateway is free to ignore it. Changed name to "suggested port" to better convey this intention.
- Added NAT-PMP transition section
- Separated Internal Address, External Address, Remote Peer Address definition
- Unified Mapping, Port Mapping, Port Forwarding definition
- adjusted so DHCP configuration is non-normative.
- mentioned PCP refreshes need to be sent over the same interface.
- renamed the REMOTE_PEER_FILTER option to FILTER.
- Clarified FILTER option to allow sending an ICMP error if policy allows.
- for MAP, clarified that if the PCP client changed its IP address and still wants to receive traffic, it needs to send a new MAP request.
- clarified that PEER requests have to be sent from same interface as the connection itself.
- for MAP opcode, text now requires mapping be deleted when lifetime expires (per consensus on 8-Mar interim meeting)
- PEER OpCode: better description of remote peer's IP address, specifically that it does not control or establish any filtering, and explaining why it is 'from the PCP client's perspective'.
- Removed latent text allowing DMZ for 'all protocols' (protocol=0). Which wouldn't have been legal, anyway, as protocol 0 is assigned by IANA to HOPOPT (thanks to James Yu for catching that one).
- clarified that PCP server only listens on its internal interface.
- abandoned 'target' term and reverted to simplier 'internal' term.

A.2. Changes from draft-ietf-pcp-base-05 to -06

- DS-Lite: consensus was encapsulation mode. Included a suggestion that the B4 will need to proxy PCP-to-PCP and UPnP-to-PCP.
- defined THIRD_PARTY option to work with the PEER OpCode, too. This meant moving it to its own section, and having both MAP and PEER OpCodes reference that common section.
- used "target" instead of "internal", in the hopes that clarifies internal address used by PCP itself (for sending its packets) versus the address for MAPpings.
- Options are now required to be ordered in requests, and ordering has to be validated by the server. Intent is to ease server processing of mandatory-to-implement options.
- Swapped Option values for the mandatory- and optional-to-process Options, so we can have a simple lowest..highest ordering.
- added MISORDERED_OPTIONS error.
- re-ordered some error messages to cause MALFORMED_REQUEST (which is PCP's most general error response) to be error 1, instead of buried in the middle of the error numbers.
- clarified that, after successfully using a PCP server, that PCP server is declared to be non-responsive after 5 failed retransmissions.
- tightened up text (which was inaccurate) about how long general PCP processing is to delay when receiving an error and if it should honor OpCode-specific error lifetime. Useful for MAP errors which have an error lifetime. (This all feels awkward to have only some errors with a lifetime.)
- Added better discussion of multiple interfaces, including highlighting WiFi+Ethernet. Added discussion of using IPv6 Privacy Addresses and RFC1918 as source addresses for PCP requests. This should finish the section on multi-interface issues.
- added some text about why server might send SERVER_OVERLOADED, or might simply discard packets.
- Dis-allow internal-port=0, which means we dis-allow using PCP as a DMZ-like function. Instead, ports have to be mapped individually.

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

A.3. Changes from draft-ietf-pcp-base-04 to -05

- tweaked PCP common header packet layout.
- Re-added port=0 (all ports).
- minimum size is 12 octets (missed that change in -04).
- removed Lifetime from PCP common header.
- for MAP error responses, the lifetime indicates how long the server wants the client to avoid retrying the request.
- More clearly indicated which fields are filled by the server on success responses and error responses.
- Removed UPnP interworking section from this document. It will appear in [I-D.bpw-pcp-upnp-igd-interworking].

A.4. Changes from draft-ietf-pcp-base-03 to -04

- "Pinhole" and "PIN" changed to "mapping" and "MAP".
- Reduced from four MAP OpCodes to two. This was done by implicitly using the address family of the PCP message itself.
- New option THIRD_PARTY, to more carefully split out the case where a mapping is created to a different host within the home.
- Integrated a lot of editorial changes from Stuart and Francis.
- Removed nested NAT text into another document, including the IANA-registered IP addresses for the PCP server.
- Removed suggestion (MAY) that PCP server reserve UDP when it maps TCP. Nobody seems to need that.
- Clearly added NAT and NAPT, such as in residential NATs, as within scope for PCP.

- HONOR_EXTERNAL_PORT renamed to PREFER_FAILURE
- Added 'Lifetime' field to the common PCP header, which replaces the functions of the 'temporary' and 'permanent' error types of the previous version.
- Allow arbitrary Options to be included in PCP response, so that PCP server can indicate un-supported PCP Options. Satisfies PCP Issue #19
- Reduced scope to only deal with mapping protocols that have port numbers.
- Reduced scope to not support DMZ-style forwarding.
- Clarified version negotiation.

A.5. Changes from draft-ietf-pcp-base-02 to -03

- Adjusted abstract and introduction to make it clear PCP is intended to forward ports and intended to reduce application keepalives.
- First bit in PCP common header is set. This allows DTLS and non-DTLS to be multiplexed on same port, should a future update to this specification add DTLS support.
- Moved subscriber identity from common PCP section to MAP* section.
- made clearer that PCP client can reduce mapping lifetime if it wishes.
- Added discussion of host running a server, client, or symmetric client+server.
- Introduced PEER4 and PEER6 OpCodes.
- Removed REMOTE_PEER Option, as its function has been replaced by the new PEER OpCodes.
- IANA assigned port 44323 to PCP.
- Removed AMBIGUOUS error code, which is no longer needed.

A.6. Changes from draft-ietf-pcp-base-01 to -02

- more error codes
- PCP client source port number should be random
- PCP message minimum 8 octets, maximum 1024 octets.
- tweaked a lot of text in section 7.4, "Opcode-Specific Server Operation".
- opening a mapping also allows ICMP messages associated with that mapping.
- PREFER_FAILURE value changed to the mandatory-to-process range.
- added text recommending applications that are crashing obtain short lifetimes, to avoid consuming subscriber’s port quota.

A.7. Changes from draft-ietf-pcp-base-00 to -01

- Significant document reorganization, primarily to split base PCP operation from OpCode operation.
- packet format changed to move ’protocol’ outside of PCP common header and into the MAP* opcodes
- Renamed Informational Elements (IE) to Options.
- Added REMOTE_PEER (for disambiguation with dynamic ports), REMOTE_PEER_FILTER (for simple packet filtering), and PREFER_FAILURE (to optimize UPnP IGD interworking) options.
- Is NAT or router behind B4 in scope?
- PCP option MAY be included in a request, in which case it MUST appear in a response. It MUST NOT appear in a response if it was not in the request.
- Result code most significant bit now indicates permanent/temporary error
- PCP Options are split into mandatory-to-process ("P" bit), and into Specification Required and Private Use.
- Epoch discussion simplified.

Authors' Addresses

Dan Wing (editor)
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, California 95134
USA

Email: dwing@cisco.com

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

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

Mohamed Boucadair
France Telecom
Rennes, 35000
France

Email: mohamed.boucadair@orange-ftgroup.com

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

Email: rpenno@juniper.net

Francis Dupont
Internet Systems Consortium

Email: fdupont@isc.org