Status

- -09 presented at IETF 85 in Atlanta
- Call for adoption issued on the list
  - Tons of useful feedback!
  - Not adopted, but clear way forward
-10

- Complete rewrite
- New title: “PCP Extension for Port Set Allocation”
  - Forget about “NAT coordination” (draft name will be changed eventually)
- New PORT_SET option (not an opcode!)
- Clearer motivation and use cases
- Directly addresses generic firewall and NAT usage
  - LW4o6 is just one customer among many others and this is now clearly explained in the draft
Use cases

• Lightweight 4 over 6

• Applications using port sets
  • Some applications make use of sets of ports instead of a single one.
  • Example: SIP UAS expecting to handle multiple concurrent calls efficiently would pre-allocate a set of ports

• Firewall control
  • PCP can be used to manipulate firewall rules. Now with port sets!
  • Example: create a firewall rule allowing RTP to a given port range.
The need for PORT_SET

- **Network Traffic**: A single request uses less network resources than multiple requests.

- **Latency**: Even though MAP requests can be sent in parallel, we can expect the total processing time to be longer for multiple requests than a single one.

- **Client-side simplicity**: The logic that is necessary for maintaining a set of ports using a single port set entity is much simpler than that required for maintaining individual ports, especially when considering failures, retransmissions, lifetime expiration, and re-allocations.

- **Server-side efficiency**: Some PCP-controlled devices can allocate port sets in a manner such that data passing through the device is processed much more efficiently than the equivalent using individual port allocations. For example, a CGN having a "bulk" port allocation scheme (see [I-D.ietf-behave-lsn-requirements] section 5) often has this property.

- **Server-side scalability**: The number of mapping entries in PCP-controlled devices is often a limiting factor. Allocating port sets in a single request can result in a single mapping entry being used, therefore allowing greater scalability.

- In a nutshell: **PORT_SET is a necessary optimization.**
## PORT_SET format

<table>
<thead>
<tr>
<th>Option Code</th>
<th>Reserved</th>
<th>P</th>
<th>Option Length</th>
<th>Port Set Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Parity preservation

- P bit: it indicates that the client wishes that the parity of the internal ports be the same as that of the corresponding external ports.
- Useful for RTP/RTCP.
- Obviates the need for draft-boucadair-pcp-rtp-rtcp.
  - Its authors have joined us.
Mechanisms

- Server MAY return fewer ports than what the client asked for.
  - PREFER_FAILURE does what you think it should do.
- Server MAY ignore the P bit.
  - PREFER_FAILURE does what you think it should do.
- Renewal and deletion: one request manipulates the whole set
- Configurable quotas are RECOMMENDED
Open issues

• Discontinuous port sets
  • Do we need them?
• (minor) Port size == 0 is disallowed. Do we want to allow port size == 1?
  • What about 65535? 65536?
  • It's about operational guidance vs hard protocol spec.
• (easy) Need to define what happens with overlapping port sets.
• Other features have been proposed.
Next steps

• We propose:
  • Focus the adoption call on the core
    - Do we want to be able to map port sets with PCP?
  • Once adopted, we can tweak or add features according to the working group's wishes