DHCPv4 Options for Port-Set Assignment

draft-bajko-pripaddrassign-04

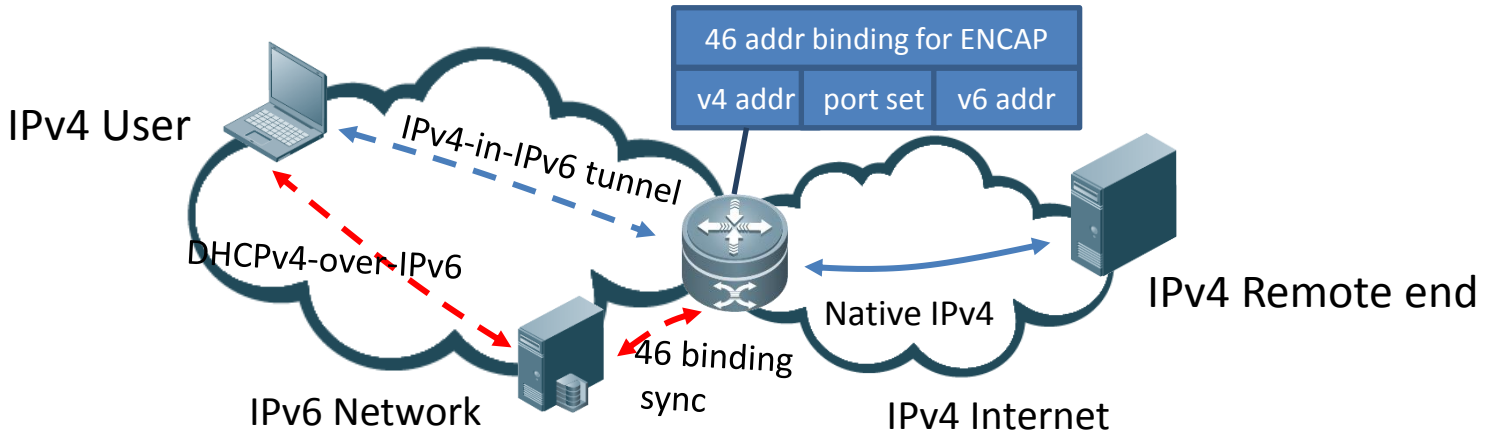
draft-wu-dhc-port-set-option-00

Background

- The possible IPv4 address exhaustion in the near future
- IPv4 address sharing between end users
  - Manner 1: Carrier-grade NAT
    - NAT444, NAT64, DS-Lite
  - Manner 2: Divide full address into port sets and assign them to end users
    - “A+P” style
    - Lightweight 4over6, MAP, 4RD

DHCPv4 for port-set assignment

- Use case: lightweight 4over6
  - Per-user stateful IPv4-over-IPv6 mechanism
    - Lightweight 4over6 [draft-cui-softwire-b4-translated-ds-lite-07]
  - DHCPv4-over-IPv6 for IPv4 assignment in IPv6 net
    - draft-ietf-dhc-dhcpv4-over-ipv6-03
  - Port-set assignment in DHCPv4

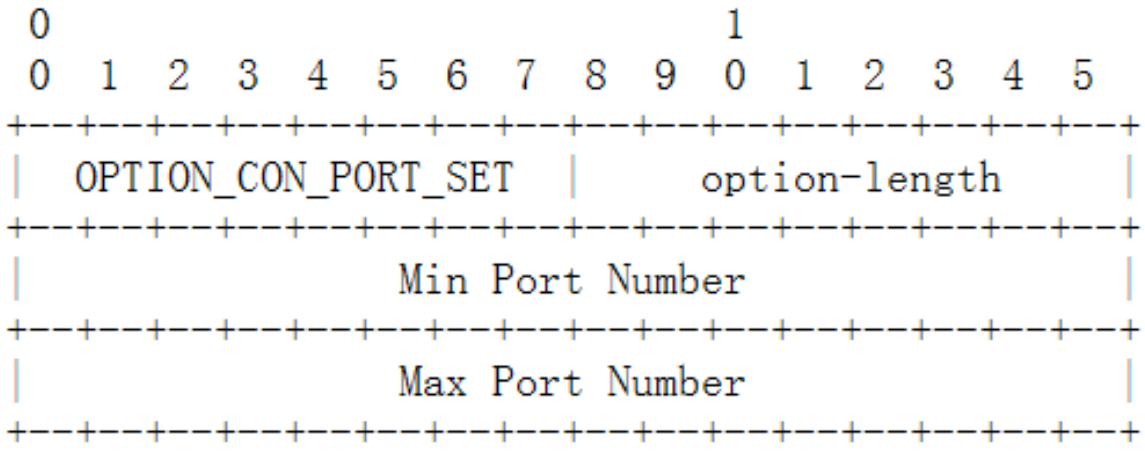


Defined options/sub-options

- For different styles of port set
- draft-wu
  - Contiguous Port Set Option
  - GMA Port Set Option
- draft-bajko
  - Port Mask Sub-Option
  - Random Port Delegation Sub-Option

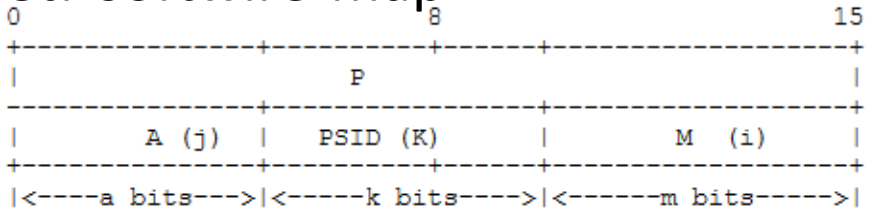
Contiguous Port Set Option

- Assign a contiguous port range
- Bounded by Min & Max port number
- Format:

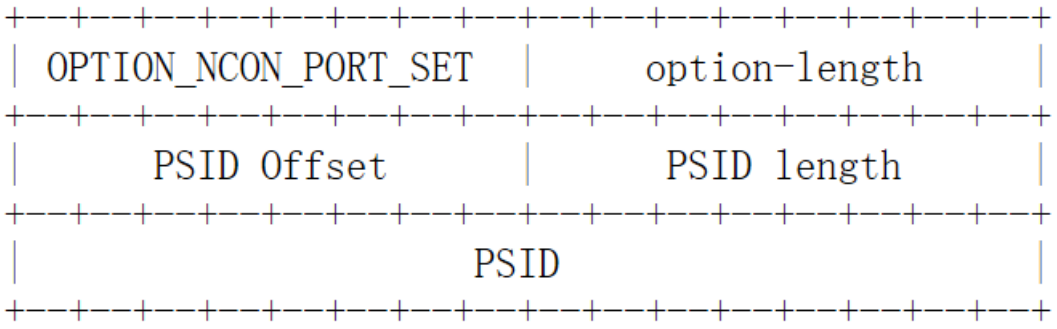


GMA Port Set Option

- Following the GMA Port mapping algorithm
  - Proposed in draft-ietf-softwire-map
- Port-set format:

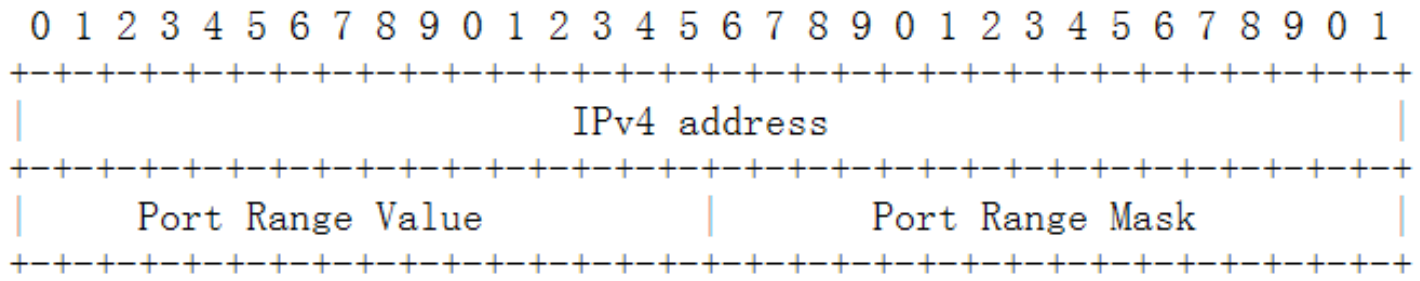


- Preserve well-known ports
  - A(j) cannot be 0 => preserve first 2^(k+m) ports
- Port-set consists of scattered port ranges
  - (2^a-1) port ranges of size 2^m
  - Could be contiguous: a=0



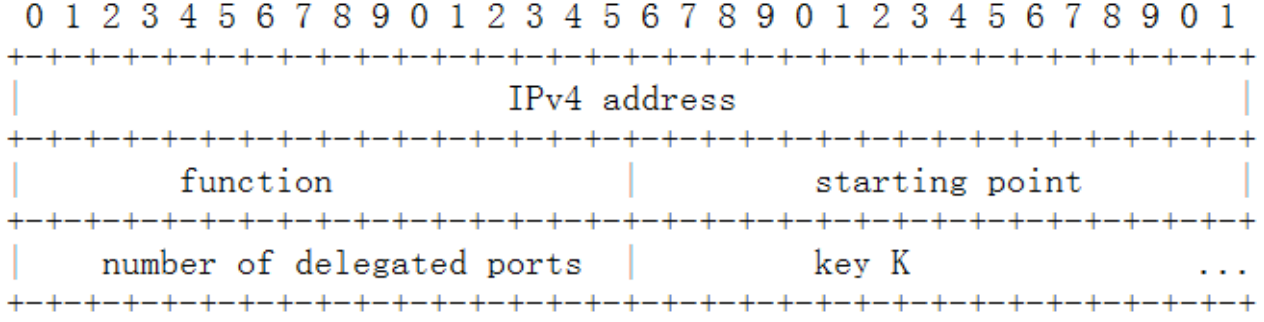
Port Mask Sub-Option

- Port set determined by 16-bit mask and value
- Port-set Format
  - Port-set Mask: position of the significant bits of mask (set to “1”)
  - Port-set Value: value of the significant bits (port-set ID)
  - Significant bits can be scattered in the total 16 bits
- Compose a port set with scattered port ranges
  - Could be more scattered than GMA
  - Could be contiguous: mask=11…100…0
- Preserving well-known ports: not defined
- IPv4 address assigned in the sub-option as well



Random Port Delegation Sub-Option

- Encryption function to achieve randomness
  - Input: key K, integer x *as the plaintext* ∈ [1024, 65535]
  - Output: integer y *as the ciphertext* ∈ [1024, 65535], to be the assigned port number
  - Encryption function determined in advance between C/S
- Compose a port set with randomized, scattered ports
  - E(K, a), E(K,a+1), …, E(K, a+2047)
- Preserve well-known ports (0~1024)
- IPv4 address assigned in the sub-option as well
- The sub-option is *encryption-algorithm-specific*



More about port randomization

- Prevent Blind attacks against TCP/UDP
- First step: making the port-set non-contiguous
- More sophisticated solutions
  - 1.User randomly selects source port from the port-set
    - RFC6056
    - Algorithms need to evolve for non-contiguous port-set
  - 2.Server pre-allocates random-style port-set
    - Random Port Delegation sub-option
    - the client is forced to use random ports
    - Decryption is needed for encapsulation destination lookup logic on tunnel concentrator

Discussion on DHCP-centric issues

- *Multiple options for multiple port-set type vs. One option with multiple sub-options*
  - DHCPv4 option code precise?
  - With multiple options, client can indicate the expected port-set type and avoid mismatch
  - From the server perspective, the server controls the port-set management manner

Discussion on DHCP-centric issues

- *IPv4 address assigned in original DHCP message vs. in port-set option*
  - Both could work
  - If in port-set option, need to clarify the usage of address-related options like IP address lease time option in this context
  - If in original message, client that do not recognize the port-set option could misuse the whole address
- WG guidance on the two issues?

Next steps

- Merged as one document, or separated document for different options?
- WG adoption?