Discovering PREF64 in Router Advertisements

draft-pref64folks-6man-ra-pref64-02

L. Colitti, E. Kline, J. Linkova

Use Case

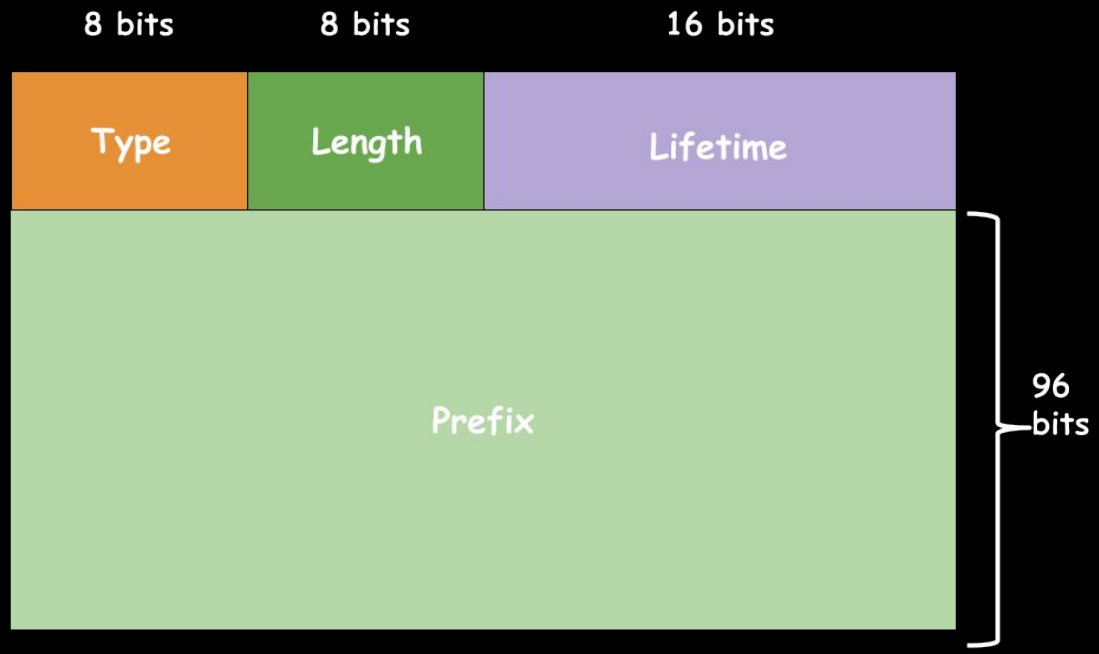
How to discover NAT64 prefix for address synthesis

- Validating stub resolvers
- IPv4 literals
- 464XLAT

Why RA Option?

- All L3 Network stack config on a host in a single packet
- Atomic: no state when config is incomplete
- Network *is* the authoritative source of information
- No additional services required
- Secured by RA guard
- No “trust DNS response to be able to use DNSSEC” paradox

Option Format



Multiple Prefixes

Multiple Prefixes

One option specifies one prefix only

An RA might contain multiple Pref64 options

- Use Case: migrating from one Pref64 to another

Multiple Prefixes Scenarios

1: Different prefixes learnt via different mechanisms

2: Multiple prefixes received in a single RA

3: Multiple prefixes received in multiple RAs (on one or multiple interfaces)

Scenarios #1 & #2

Different prefixes learnt via different mechanisms

Recommended order

1. RFC7225 (if supported)
2. RA Option
3. RFC7050 (DNS-based discovery; widely supported)

Multiple prefixes received in a single RAs

SHOULD follow guidance in RFC7050 (use all prefixes)

Scenario #3: Multihoming

Multiple prefixes received in multiple RAs (on one or multiple interfaces)

- Pref64 is specific to the network it's received on
- Multihomed hosts need to be mPVD-aware
  - This is already true today

Limitations

One Prefix for All Destinations

- Workaround: use more-specific routes in network
  - 10.0.0.0/8 -> 64:ff9b::10.0.0.0/104
- Support would increase implementation complexity, risk of bugs
- Not supported by RFC7050 either

No Ability to Exclude Prefixes from Synthesis

- Not useful on an IPv6-only network
  - If host gets an A record, it can't do much with it
  - If app knows pref64 and really cares about the A, it can trivially reverse address synthesis
  - Private IPv4 MUST NOT be translated with WKP
- Not supported by RFC7050 or RFC7225* either

* Theoretically you could do 0.0.0.0/1, 128.0.0.0/2, 160.0.0.0/4, … until you hit packet size limits. Should you?

Only Supports /96 pref64

- Other prefix lengths not [widely? at all?] implemented
- Supporting other prefix lengths would use an additional 8 bytes in RA
- Can always define another option in the future

Call for Adoption?