Announcing Prefixes

RFC 8505 ++

Pascal Thubert

IETF 113

Remote
6LoWPAN ND (IPv6 Stateful Address Autoconfiguration)

- **RFC 6775** (original 6LoWPAN ND)
  - Defines ARO for registration and DAD operations for stateful AAC
- **RFC 8505** (extended 6LoWPAN ND)
  - Extends ARO, updates the registration procedure
  - Allows registering to network services inc. proxy
- **RFC 8928** (Address Protection for ND)
  - Secures ownership and enables SAVI
- **RFC 8929** (Backbone Router – proxy ND)
  - Defines a proxy ND operation. Updates EDAR to transport ND options such as SLLAO.
- **draft-thubert-6lo-unicast-lookup** (Unicast Address lookup on backbone)
  - Allows the 6LBR to respond to lookups and saves broadcasts
- **draft-ietf-6lo-multicast-registration** (Anycast and Multicast Address Registration)
  - Registers anycast and multicast addresses (in addition to unicast per RFC 8505)
Redistributing RFC 8505 in routing?

- Already done for host routes with the “R” flag
  - e.g., RFC 9010 into RPL, or even RFC 8929 into IPv6 ND
  - Also draft-thubert-bess-secure-evpn-mac-signaling using BGP, or RIFT
  - Provides a host / router interface that is agnostic to the IGP beyond the router
Multi-link Subnet Routing (non-storing mode)

Parent is default GW, propagates root PIO (L-bit off)
Parent Address in the PIO (with R bit)
RPL Router autoconfigures Address from parent PIO
RPL Router advertises Address via Parent to Root
Root recursively builds a Routing Header back

A::A

A::B

A::C

A::D

C:
::/0 via A::B
A::B connected
A::C self
A:: ~onlink

B:
::/0 via A::A
A::A connected
A::B self
A:: ~onlink

D:
::/0 via A::B
A::B connected
A::D self
A:: ~onlink

Target A::C via Transit A::B

A: (root)
A::A self
A::B connected
A::C via A::B
A::D via A::B
A:: ~onlink

A::C via A::B connected
RFC 9010 (RUL)

6LR advertises A:: in RAs
6LN autoconfigures A::L
6LN registers A::L with « R » flag set
6LR injects the address as external host route in RPL

C:
::/0 via A::B
A::B connected
A::C self
A:: ~onlink

Target A::L via Transit A::C (Ext)

A: (root)
A::A self
A::B connected
A::C via A::B
A::L via A::C
A::D via A::B
A:: ~onlink

A::L via A::C via A::B connected
But... prefixes?

- Hosts may own prefixes
  - Network in Node / recursive networking
  - Kubernetes / Private IPv4 realms
Owned prefix routing (non-storing mode)

Parent is default GW, advertizes owned PIO (L bit on)
RPL Router autoconfigures Address from parent PIO
RPL Router advertises Prefix via Address to Root
Root recursively builds a Routing Header back
Owned prefix routing (non-storing mode)

C::L is reachable but L:: is not
Missing equivalent of RFC 8505/9010 for prefixes

Target C::/ via Transit B::C

A: (root)
A:: connected
B:: via A::B
C:: via B::C
D:: via B::D

L:: unreachable
C::L via B::C via A::B connected
What becomes of DAD?

• Need to consider prefix aggregation and nesting
  • Provisioned Mobile Networks should be unique
  • Auto-allocation?
How would that work?

- **RS replaces NS**
- **RS(SRO) « R » set)**
- **Stub registration option replaces EARO**
Announcing prefixes

RS (target = IPv6 address, SRO (ROVR=Crypto-ID PoO))

RA (SRO(status=Validation Requested), Nonce)

RS (SRO, CIPO*, Nonce and NDPSO**) 

RA (SRO(status=0)) 

* Crypto-ID Parameters Option
** NDP Signature Option

RS replaces NS
Provision IPv6 prefix 2001:db8:1::/96 for subnet 10.1.0.0/28 in tenant VRF

RS (StubRegOption, R=1, lifetime, sequence, PoO)

RA (SRO, status = 0)

Advertise (IGP, eVPN)
Sorry, getting gory…

• Adding stub prefix advertisement vs. host today
  • Indicate prefix type e.g., a /96 to embed an IPv4 address
  • Proof of ownership (PoO) per RFC 8928

• Adding policy / ACLs
  • Signal partial micro-segmentation (offload), who can talk to me

• Adding metrics to influence load balancing
  • worker capacity (clusters / containers)
  • Access bandwidth /
  • multihoming / preferred interface / anycast

• Tenant ID / VRF ID / RPL instanceID
  • Route tags, RH
Go? No-Go?
Ask

• Should we go for it
  • Indicate prefix type e.g., a /96 to embed an IPv4 address
  • Proof of ownership (PoO) per RFC 8928