CUMULUS

Nexthop and Nexthop Group Objects

November 6-7, 2017

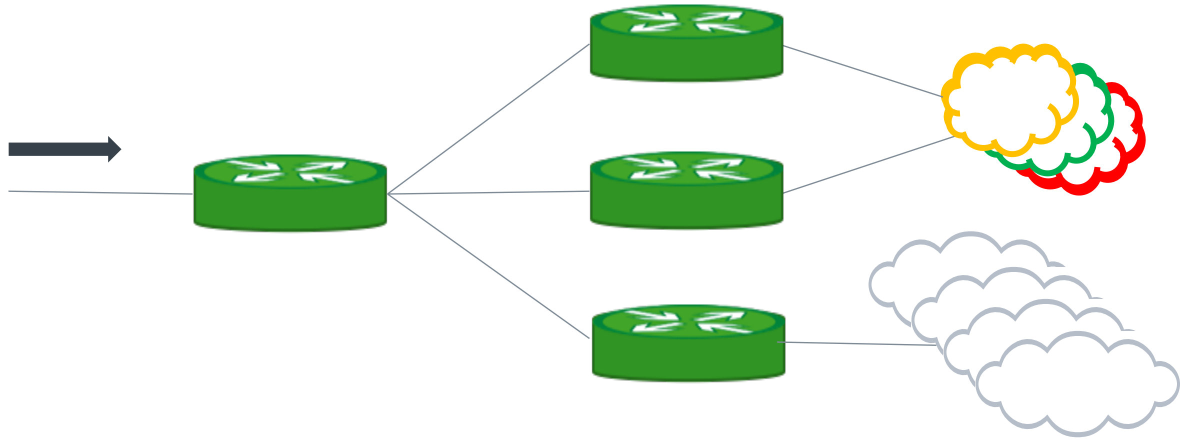
David Ahern | Cumulus Networks

Nexthops are typically repetitive

Prefixes out number nexthops by large factor

- 100k’s of routes with 10’s to 100’s of nexthops

  ratio typically > 10,000:1



nexthops and routes in Linux

Nexthop specs are currently integrated into route structs

- ipv4: fib_nh at the end of fib_info

  ipv4 does consolidate duplicate nexthop specs with multiple references to one fib_info

- ipv6: distributed within rt6_info and dst
- mpls: mpls_nh at the end of mpls_route

Redundant code and processing

Redundant processing adding routes

- lookups to validate gateway addresses
- comparison of nexthop specs
- percpu allocations
- validating lwtunnel state
- IPv4 FIB notifier - FIB_EVENT_NH_ADD

  Ido indicated IPv6 needs notifier as well

All of it affects convergence time following a link event

- critical benchmark for a NOS

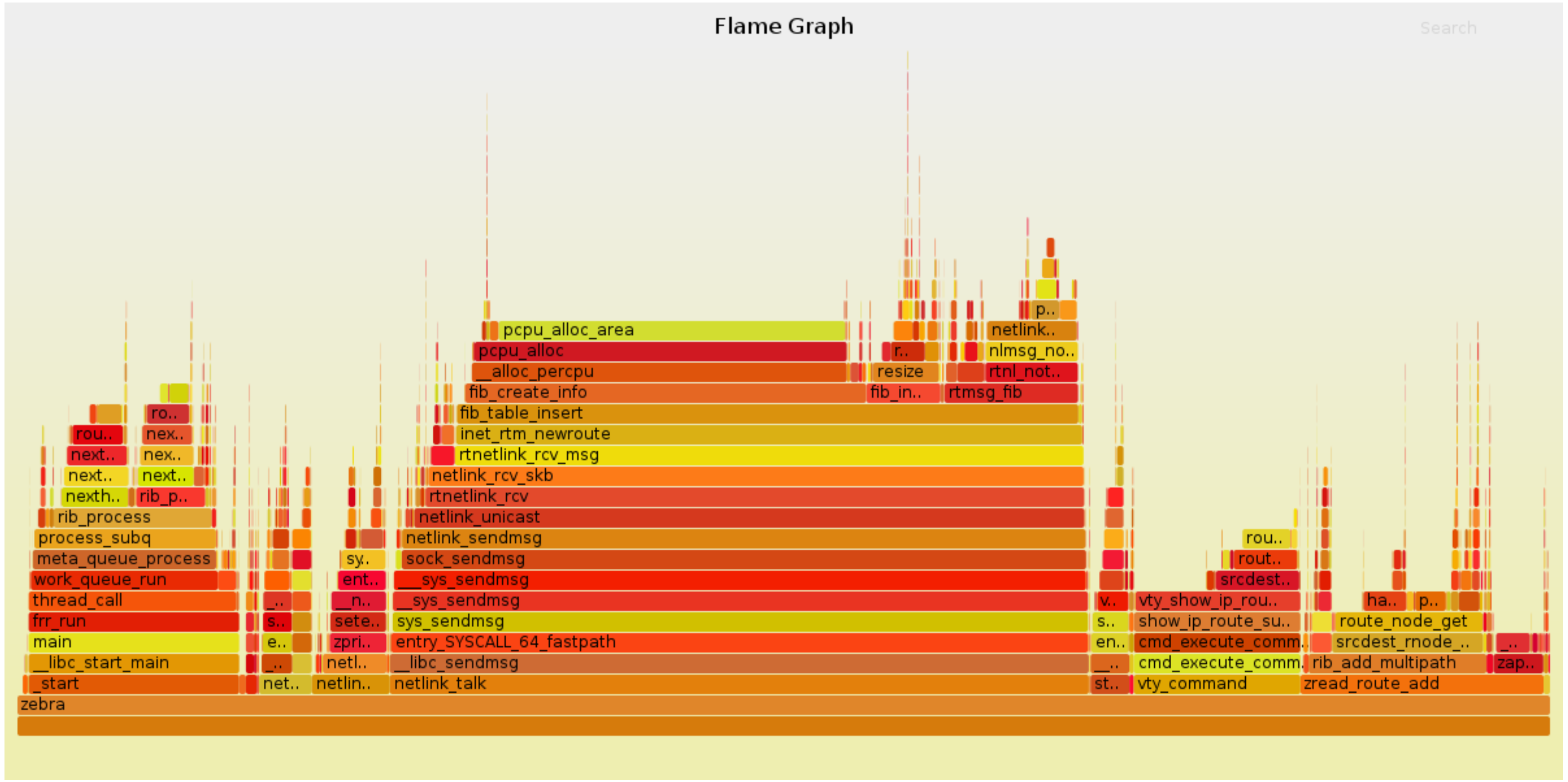
Per Address Family Processing

Every protocol has independent notifiers to handle link events

- Family based code that does the same or almost the same processing with respect to nexthops
- For example, carrier state changes and marking or clearing RTNH_F_{DEAD,LINKDOWN} and walking fib looking for entries referencing device

  IPv4 does this, IPv6 does not

Flame Graph: FRR inserting 700k routes



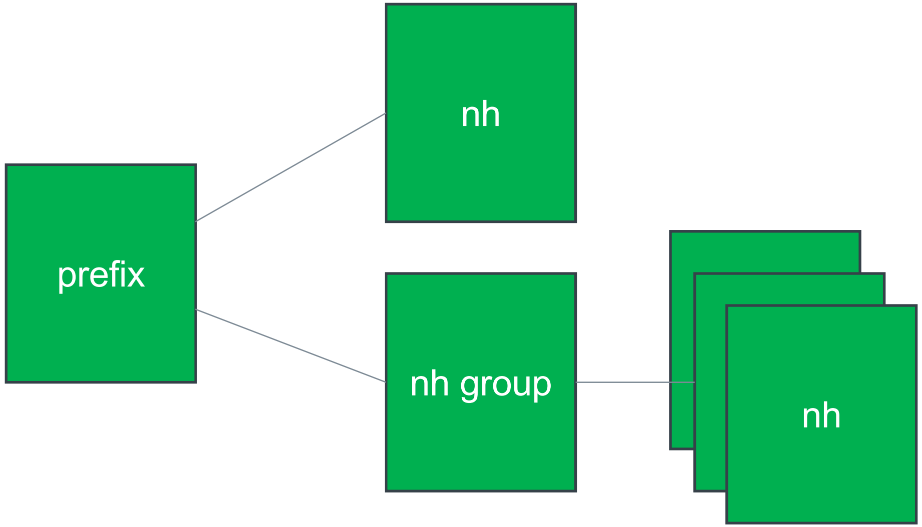
Nexthops as Standalone objects

Nexthops and nexthop groups as separate objects

- separate add/create/modify lifecycle from route entries

Routes can reference nexthop or nexthop group by id

- Only applies to FIB entries



Nexthop Objects

IPv4 already does this to some extent with fib_info

- Still significant duplication and unnecessary work per prefix
- fib_info is more than just nexthop definition

Idea is similar to adding id to fib_info that is exposed to userspace

- Subsequent routes pass id to avoid fib_info overhead

Multipath is a Nexthop Group

- References other nexthop objects

Benefits

Removes redundant processing on route add

- Already validated the nexthop gateway, device and LWT config
- IPv4, creating a fib_info only to free it in favor of existing

Opportunity to have better alignment across protocols

- Bring fib_info type efficiencies to IPv6 and MPLS
  - Better memory utilization
  - No duplicate nexthop checking

Alignment with hardware offload

Enables New Features

More efficient means to update nexthops for 1,000’s of routes

- Following a link event, L3 protocol determines new (better) nexthop for N-routes
- Just updates 1 nexthop spec as opposed to deleting N-routes and adding them back with new nexthop

Failover nexthop

- Preferred nexthop for routes. If it goes down, routes seamlessly failover to backup

IPv4 routes with IPv6 nexthops

- Needed for BGP unnumbered

Batching of route add?

- Push down a series of prefixes and associated attributes with nexthop by id

Co-existence of models

If you like your current route model, you can keep it

- Current API – route spec contains nexthop attributes
  Routes created with nexthops inline
- Connected and host routes

Routing daemons opt in to new API

- Create nexthop prior to adding route
- Routes added with reference to nexthop by id
- Routing daemons already track nexthops separately

Performance

Typically measured as latency or throughput

- packets/bytes per second received or sent
- Not strictly a relevant benchmark for H/W offload cases

Convergence time following a link event is more pressing

Motivation is scaling up to 1M+ routes

EARLY Test Results

Installing 655,024 route entries, single nexthop:

Current:

```
# time ip -batch /media/node/full-table-ipv4.txt
real	0m30.104s
user	0m3.816s
sys	0m14.614s
```

Nexthop objects:

```
# time ip -batch /media/node/full-table-ipv4-nh.txt
real	0m22.206s
user	0m3.223s
sys	0m9.792s
```