slaacd(8)

Florian Obser
florian@openbsd.org

- Sysadmin by day, DNS & BGP nerd
- OpenBSD hacker since 2012

- In the Beginning
  - RFC ~~5101~~ 7011 IPFIX for pflow(4)
  - one of the de facto mg(1) maintainers

- The Web Years
  - got tricked by Theo to transition base from Apache 1.3 to nginx
  - wrote slowcgi(8) (because Reyk complained)
  - got Reyk drunk in Ljubljana and wrote fastcgi support for httpd(8)

- The IPv6 Years
  - unified traceroute6(8) and traceroute(8)
  - unified ping6(8) and ping(8)
  - wrote slaacd(8), the IPv6 Stateless Address AutoConfiguration Daemon

why?

```
Purging is at last at hand. Day of Doom is here.
All that is evil shall all be cleansed.

Remove sending of router solicitations and
processing of router advertisements from the
kernel. It's handled by slaacd(8) these days.

Input & OK bluhm@, mpi@

sys/net/if.c                 |   12 +-
sys/netinet6/in6.c           |   18 +-
sys/netinet6/in6_ifattach.c  |    9 +-
sys/netinet6/in6_var.h       |    5 +-
sys/netinet6/nd6.c           |  373 +--
sys/netinet6/nd6.h           |   63 +-
sys/netinet6/nd6_nbr.c       |   27 +-
sys/netinet6/nd6_rtr.c       | 1799 +---------
8 files changed, 34 insertions(+), 2272 deletions(-)
```

A Quick Primer on IPv6

(only the parts we need for SLAAC)

Addresses are 128 bit

- The Known Universe: ::/0
- What gets allocated to your ISP: 2001:DB8::/32
- What gets handed to you: 2001:DB8:23::/48 (or /56)
- What you configure on a L2 collision domain: 2001:DB8:23:42::/64

(If you don't like this go bother the 6man IETF WG)

- How to get an address
  - configure it statically
  - use DHCPv6
  - just pick one at random!

- 2^{64} addresses to choose from
- host needs to
  1. know which /64 prefix: Router Advertisements (RFC 4861)
  2. pick address without colliding
  3. (detect collision)

- pick address without colliding
  (find an Interface Identifier)
  - modified EUI-64, fancy way of saying embed your MAC address (RFC 4291)
  - pick a random number (temporary or privacy addresses, RFC 4941)
  - "Semantically Opaque Interface Identifiers" (RFC 7217)

- so far this was about globally unique addresses
- there are also link-local addresses, generated from the well known prefix fe80::/64
- and well known link-local multicast addresses, e.g.:
  - All Nodes: FF02::1
  - All Routers: FF02::2

why is this complicated?

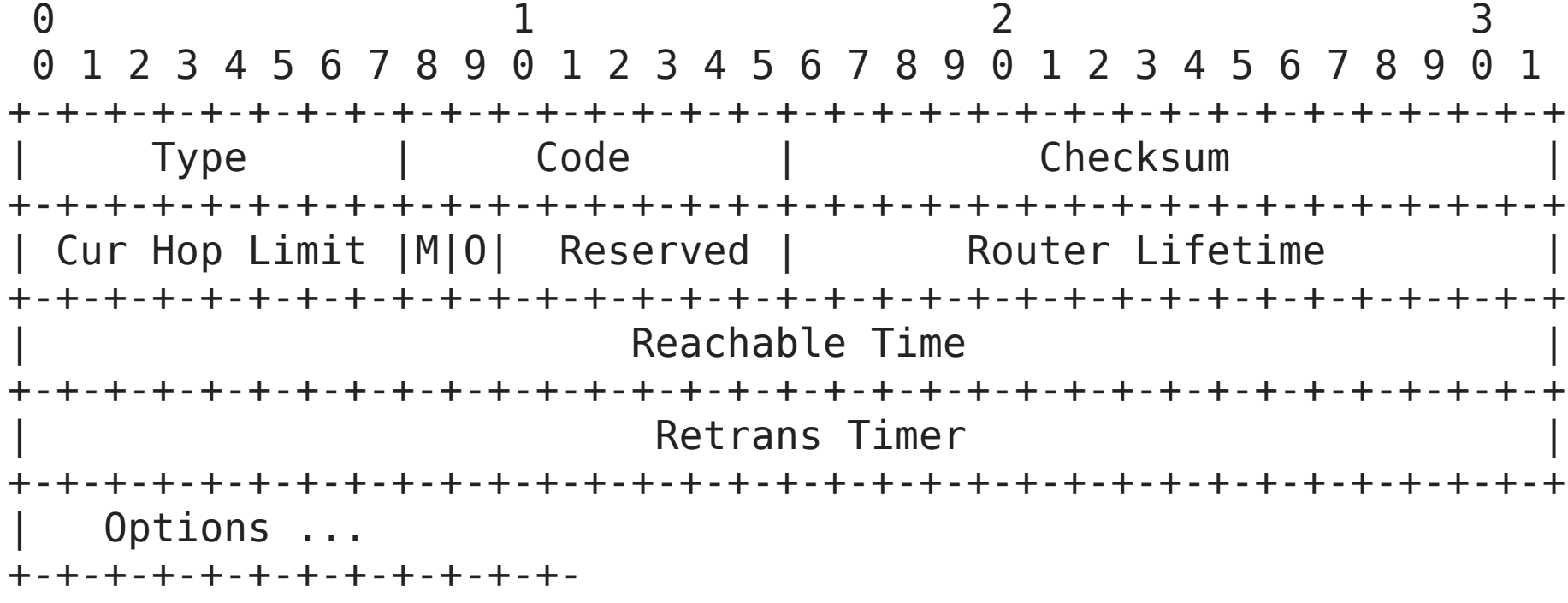
- we need...
  - one 128-bit number, the address of our gateway
  - one 128-bit number, the IPv6 prefix
  - one 8-bit number, the prefix length

Let's have a look at Router Advertisements (RA)

- periodic ICMPv6 packet to All Nodes multicast address
- in response to an ICMPv6 solicitation to the All Routers multicast address from a host

- Information from RAs:
  - link-local address of your gateway (implicit)
  - diameter of the Internet (Cur Hop Limit)
  - "Managed address configuration" flag: go ask DHCPv6 for an address
  - "Other configuration" flag: go ask DHCPv6 for additional information (DNS, NTP, ...)
  - Router Lifetime: gateway is valid for this long
    - 0: I'm not a default router!

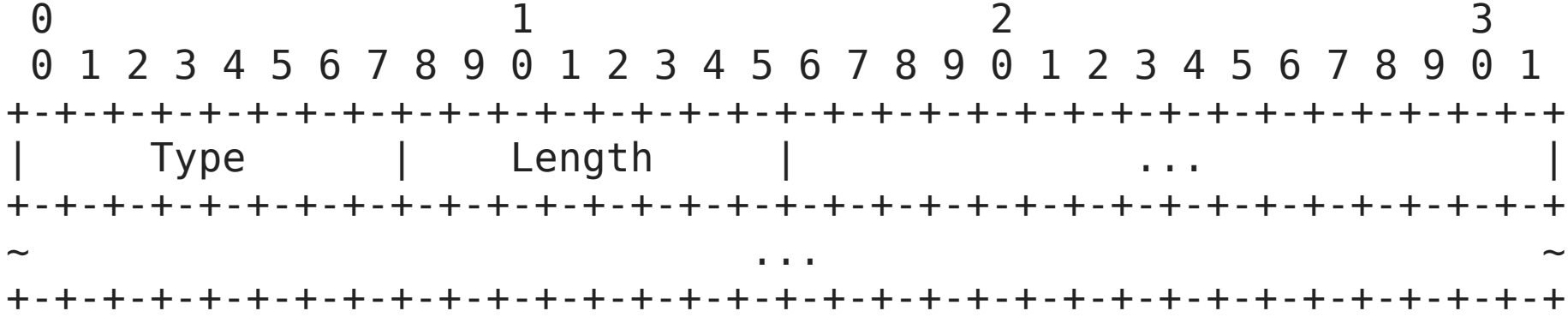
RFC 4861, Section 4.2 Router Advertisement Message Format



So… where is the prefix?

Options ...

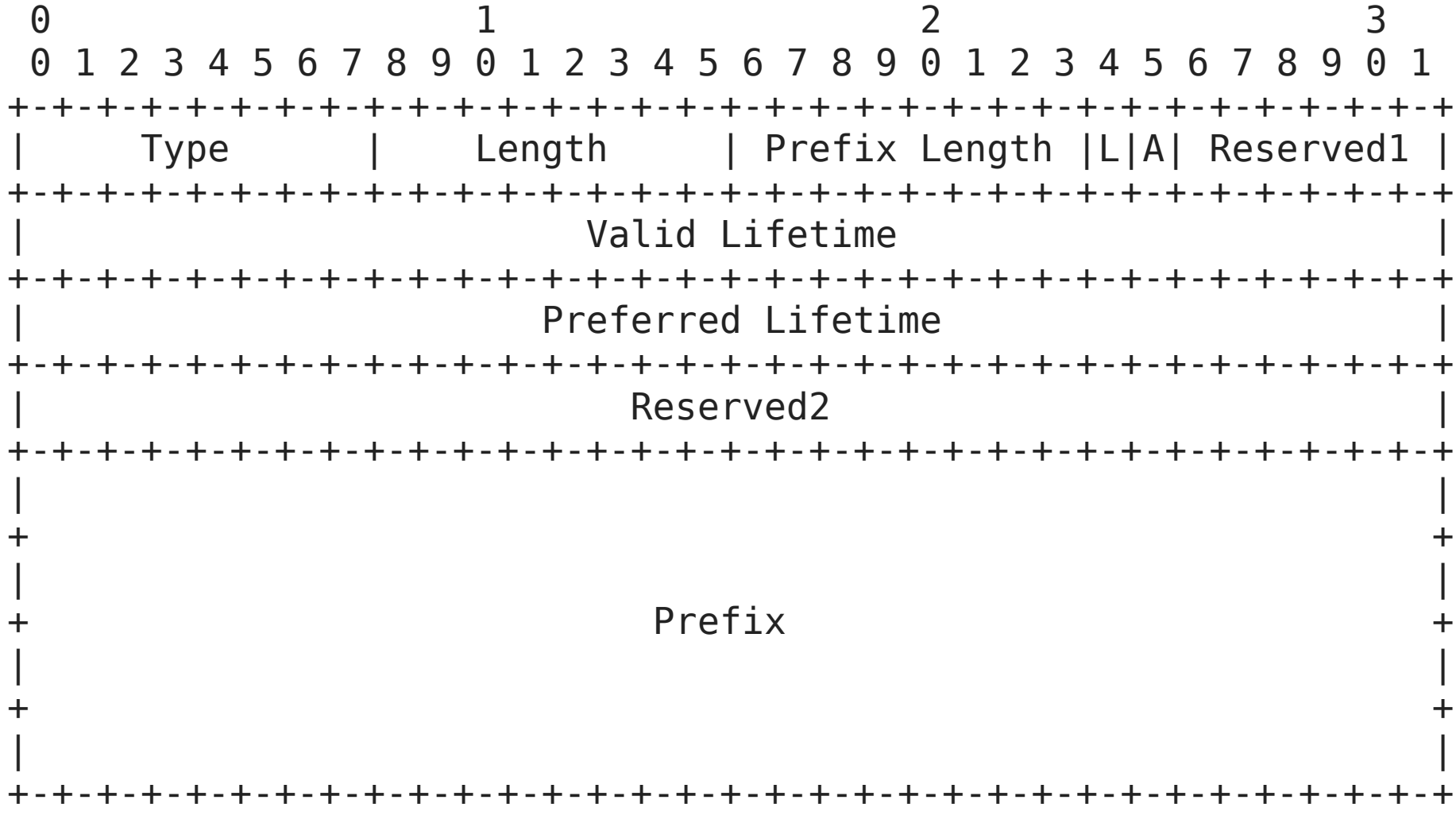
RFC 4861, Section 4.6 Option Formats



Length 8-bit unsigned integer. The length of the option
        (including the type and length fields) in units of
        8 octets. The value 0 is invalid.

- Options
  - Source link-layer address
  - MTU
  - Prefix Information
  - Recursive DNS Server Option (RFC 6106)
  - DNS Search List Option (RFC 6106)
  - ...

RFC 4861, Section 4.6.2 Prefix Information



- Prefix Information
  - Prefix Length
  - on-link flag
  - autonomous address-configuration flag (can we use this prefix for SLAAC)
  - Valid Lifetime (vltime)
  - Preferred Lifetime (pltime)
  - Prefix

BTW, there is a dynamic routing protocol lurking in here:

```
Prefix Information
    These options specify the prefixes that are on-link
    and/or are used for stateless address
    autoconfiguration.  A router SHOULD include all its
    on-link prefixes (except the link-local prefix) so
    that multihomed hosts have complete prefix
    information about on-link destinations for the
    links to which they attach.  If complete
    information is lacking, a host with multiple
    interfaces may not be able to choose the correct
    outgoing interface when sending traffic to its
    neighbors.
```

- An Eldritch Horror from beyond Spacetime
  - dynamic routing protocol
  - unknown amount of variable-length options
  - complex timers (router lifetime, pltime, vltime)

- difficult to get this right in the kernel
  - kinda single process
  - no privilege separation
  - no privilege drop
  - was getting in the way of fine-grained network stack locking
  - ...

BTW(2), DNS Search List Option: CVE-2014-3954 (userland)

```
Purging is at last at hand. Day of Doom is here.
All that is evil shall all be cleansed.

Remove sending of router solicitations and
processing of router advertisements from the
kernel. It's handled by slaacd(8) these days.

Input & OK bluhm@, mpi@

sys/net/if.c                 |   12 +-
sys/netinet6/in6.c           |   18 +-
sys/netinet6/in6_ifattach.c  |    9 +-
sys/netinet6/in6_var.h       |    5 +-
sys/netinet6/nd6.c           |  373 +--
sys/netinet6/nd6.h           |   63 +-
sys/netinet6/nd6_nbr.c       |   27 +-
sys/netinet6/nd6_rtr.c       | 1799 +--------
8 files changed, 34 insertions(+), 2272 deletions(-)
```

Onwards to userland!

- OpenBSD priv'sep daemons
  - many examples in base
    - sshd, bgpd, dvmrpd, eigrpd, httpd, ntpd, ospf{,6}d, relayd, vmd...
    - heck, even tcpdump
      (not a daemon though and follows a different pattern)

- OpenBSD priv'sep daemons (cont'd)
  - decades of experience & improvements
  - new (or old but not yet implemented) ideas apply to many / all; recent:
    - fork → fork & re-exec
    - pledge

- interlude: pledge(2)
  - forces process into restricted service operating mode
  - "From here on out I promise to only work on already open FDs"
    ```
    pledge("stdio", "");
    ```
  - "From here on out I promise to only work on already open FDs or talk to the internet"
    ```
    pledge("stdio inet", "");
    ```

- interlude: pledge(2) (cont'd)
  - violators will be shot^W killed with uncatchable SIGABRT
  - i.e. try to open a file → boom

- interlude: pledge(2) (cont'd), common patterns
  - hoist privileged operations to startup, then pledge
  - receive privileged resource from less tightly pledged process via FD-passing
  - process data in "stdio" process and ask priv'ed process to make changes
  - don't forget to re-pledge, e.g. drop "sendfd" and "recvfd"

- OpenBSD priv'sep routing daemons (bgpd, ospfd, ...)
  - config file
  - logging framework (debug & syslog)
  - control socket (for {bgp,ospf,...}ctl(8))
  - 3 processes
    - main (or parent)
    - frontend
    - engine

- routing daemons (cont'd): main (or parent) process
  - stays root
  - brings up other processes
  - full mesh socketpair(2) setup for imsg_* message & FD passing
  - allocates priv'ed resources & passes on
  - handles priv'ed operations

- routing daemons (cont'd): frontend process
  - chroot(2) to /var/empty
  - drops to unpriv'ed user (not nobody!)
  - usually "stdio inet" pledged
  - sends and receives data from outside (but doesn't look at it!)
  - passes data to engine for processing

- routing daemons (cont'd): engine process
  - chroot(2) to /var/empty
  - drops to unpriv'ed user (not nobody!)
  - always "stdio" pledged
  - receives data from frontend and parses it
  - runs some sort of FSM
    - asks frontend to send messages to outside
    - asks main to do priv'ed operation

- routing daemons (cont'd): skeleton
  - heavy lifting done by Kenneth R Westerback (krw@)
  - https://github.com/krwesterback/newd

slaacd(8)

- features
  - started per default very early
    - so early that the control socket is in /dev!
  - handles all interfaces with the AUTOCONF6 flag
    - setable by ifconfig(8) or in hostname.if(5)
    - notices when flags change
  - no need for a config file

- features (cont'd)
  - generates "Semantically Opaque Interface Identifier" based addresses (RFC7217)
    - EUI64 based addresses if SOII disabled by ifconfig(8)
  - generates privacy addresses (RFC 4941) per default (can be disabled)
  - adds default route to kernel
    - with mpath flag → can handle multiple on-link routers and multiple links

- master process pledge
  - tasks: create control socket, create AF_ROUTE & AF_INET6 raw sockets, change routing table, configure IP, remove control socket
  - startup: not pledged
  - after priv-init: stdio cpath sendfd wroute
  - after full init: stdio cpath wroute

- frontend process pledge
  - tasks: recv RA, send RS, monitor IF state, monitor routing table, listen on control socket
  - startup: not pledged
  - after priv-drop: stdio unix recvfd route
  - after full init: stdio unix route

- engine process pledge
  - tasks: process RA, run FSM
  - startup: not pledged
  - after priv-drop: stdio recvfd
  - after full init: stdio

- future work
  - get "wroute" pledge in
  - fix Neighbor Unreachability Detection
  - add prefixes to routing table where A flag is not set
  - gotta be more RFCs to implement!
  - exec pledges

- future work: exec pledges
  - second argument to the pledge syscall

```
pledge(NULL, "stdio inet unix recvfd route");
```

  - after exec we start with those promises in place
  - no more chroot(2) (but what's the point? Careful with portable though)
  - child processes no longer need root privs

Questions?