Current status of NetBSD MP-safe network stack project

Ryota Ozaki and Kengo Nakahara
(ozaki-r@ and knakahara@)

Internet Initiative Japan, Inc.

AsiaBSDCon 2016 NetBSD BoF
March 11 2016
Summary

• Background and goal
• What we’ve done
• What we’re working on
  – Nexthop cache separation
  – TX multi-queue support
  – MP-safe gif(4)
  – Performance measurements
• Roadmap
• Future plan
Background

- NetBSD’s network stack and network device drivers don’t run in parallel between CPUs
  - Device drivers need to run with KERNEL_LOCK
  - The network stack need softnet_lock
Goal

• Make (part of) the network stack and (some) device drivers MP-safe
  – Make them runnable without the big locks

• Targets
  – Layer 2/3 forwarding
    • and some other components: gif, ipsec, ppp{oe}, etc.
  – Intel NICs and some drivers for VMs
    • wm(4), vioif(4), vmx(4) and some others
  – amd64/i386 (and ARM?)
What we did

- Interrupt distribution / IRQ affinity
  - intrctl(8) changes interrupt destination CPUs
- MSI/MSI-X support
  - i386 and amd64
- Hardware multi-queue support of wm(4)
  - Only RX queues for now
- MP-safe device drivers
  - wm(4), vioif(4) and vmx(4)
- MP-safe bridge(4)
  - Utilizing pserialize(9)
- Lots of ATF tests for the network stack
What we’ve done

• Lots of ATF tests
  – rump-ifying rtadvd(8), gif(4)

• New L2 nexthop cache implementation
  – Derived from FreeBSD
  – For L2 nexthop cache separation from the routing table

• No hardware interrupt context in the network stack
  – Make remaining parts run in softint
  – Except for ieee80211 and bpf(4)

• Restructuring and refactoring
  – No routing lookups in Layer 2
  – Use time_uptime instead of time_second
  – Kill open codes of manipulating rtentry#rt_refcnt
  – Many other small tweaks…
Added ATF tests (1/2)

- net/arp/t_arp
  - cache_expiration_10s, cache_expiration_5s, cache_overwriting, command, grap,
    link_activation, proxy_arp
- net/arp/t_dad
  - dad_basic, dad_duplicated
- net/icmp/t_icmp_redirect
  - icmp_redirect, icmp_redirect_timeout
- net/icmp/t_icmp6_redirect
  - basic
- net/if/t_ifconf
  - basic
- net/if/t_ifconfig
  - create_destroy, options, parameters
- net/if_bridge/t_bridge
  - basic, basic6, member_ip, member_ip6, rtable
- net/if_gif/t_gif
  - basicipv{4,6}overipv{4,6}, ioctlipv{4,6}overipv{4,6}, recursiveipv{4,6}overipv{4,6}
- net/if_tap/t_tap
  - create_destroy, stand_alone, bridged
Added ATF tests (2/2)

- net/ndp/t_dad
  - dad_basic, dad_duplicated
- net/ndp/t_ndp
  - cache_expiration, cache_overwriting, command, link_activation, neighborgcthresh
- net/ndp/t_ra
  - basic
- net/net/t_forwarding
  - basic, basic6, fastforward, fastforward6, misc
- net/net/t_ipaddress
  - ipaddr_same_address, ipaddr_same_address6
- net/net/t_ipv6address
  - linklocal, linklocal_ops
- net/net/t_ipv6_lifetime
  - basic
- net/route/t_flags
  - route_flags_{announce,blackhole,cloned,connected,default_gateway,icmp_redirect,lo, reject,static,xresolve}
- net/route/t_route
  - non_subnet_gateway
What we’re now working on

- L2 nexthop cache separation from the routing table
- TX multi-queue
  - \textit{wm(4)} at first
- MP-safe IP forwarding
  - Make data structures MP-safe
    - The routing table, ipaddr, ifnet, etc.
- MP-safe gif(4)
- \textit{pwe(4)} (L2TPv3) support
- Polling mode of network device drivers
  - Like NAPI of Linux
- Performance measurements
  - \textit{ipgen}
Nexthop cache separation

• Summary
  – Stop treating nexthop caches like ARP/NDP entries as part of the routing table
  – Store nexthop caches in each interface
  – Drop concept of cloning/cloned routes

• Motivation (for MP-safe work)
  – Remove recursive operations to handle cloned routes
  – Reduce contentions on the routing table

• ToDo
  – Get it done with keeping backward compatibility AMAP
    • It’s hard!
TX multi-queue support

• ToDo
  – New TX API
    • if_transmit instead of if_start
      – Pass packets (mbuf) directly to a network device driver
      – Not via if_snd queue (IFQ_ENQUEUE)
    • Multiple (soft) queues on each driver
      – Used if hardware is busy
  – Consideration
    • Which TX (hardware) queue we should use?
      – if # of CPUs > # of hw queues
      – if # of CPUs < # of hw queues
MP-safe gif(4)

• Done
  – Mutual exclusion between ioctl and packet processing

• ToDo
  – Lockless packet processing
    • with pserialize(9), not rwlock(9)
    • with passive reference?
  – ip_encap
    • Utility functions used by gif(4), stf(4), and ipsec
    • Fix scaling problem with lots of tunnels
      – Remove linear search in packet processing path (encap[46]_lookup)
Performance measurements

• We have to know if MP-safe changes improve performance

• Throughput and latency of IP and bridge forwarding
  – Variable sized frames
  – Multiple flows

• ipgen is used by the measurements
What is ipgen?

- **ipgen**: interactive packet generator
  - A packet generator utilizing netmap(4) of FreeBSD
  - Developed by ryo@

- **Features**
  - Wire rate traffic with short packets on 1 GbE
    - Not known for 10 GbE
  - Experiments for packet forwarder (DUT)
    - RFC 2544 test
    - Multiple flows
  - Interactive UI (curses and web)
  - Drop/dup/reorder counters
  - Packet pacing by controlling inter packet gap
PR: demo at IIJ booth
SEIL/BPV4

• Press release (in Japanese)

• Intel C2558 (Rangeley)
  – qat(4): Intel Quick Assist Technology Driver
    • Developed by hikaru@
    • Uses MSI-X
    • Used by opencrypto
    • Written from scratch
    • Not merged into –current yet
Performance of qat

Atom C2758, C2558/Fedora16 Cryptographic Framework Performance (4 threads, 1024 byte block)
Roadmap

MP-safe Layer 3 forwarding

- L2 nexthop separation
- MP-safe components
  - Coarse-grain locking of the routing table
  - Locking (and refcount) of ipaddr
  - MP-safe ifnet with pserialize(9)

MP-safe Layer 2 (vlan and bpf)

MP-safe other network components (gif, ipsec, opencrypto, ppp{oe}, pfil)

Now

NET_MPSAFE on by default?
Future plan

• MP-safe bpf(4)
  – Need to make ieee80211_input and some drivers run in softint (not must but desired to make MP-safe work easy)

• Alternative to the radix tree
  – rttree(3)?

• Drop rtcache?
  – If the routing table is enough fast, we don’t need caches?
  – Or introduce a fast cache structure like Poptrie or SAIL?

• A common infrastructure of interfaces
  – for polling mode