

Current status of NetBSD MP-safe network stack project

Ryota Ozaki and Kengo Nakahara

(ozaki-r@ and knakahara@)

Internet Initiative Japan, Inc.

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 rtenry#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, neighborgctresh
- 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
 - wm(4) at first
- MP-safe IP forwarding
 - Make data structures MP-safe
 - The routing table, ipaddr, ifnet, etc.
- MP-safe gif(4)
- pwe(4) (L2TPv3) support
- Polling mode of network device drivers
 - Like NAPI of Linux
- Performance measurements
 - 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

```
端末
sel@ruru: ~
load averages: 0.07, 0.07, 0.04;          up 0+00:52:30      04:38:46
25 processes: 22 sleeping, 1 zombie, 2 on CPU
CPU0 states: 0.0% user, 0.0% nice, 0.0% system, 93.1% interrupt, 6.9% idle
CPU1 states: 0.0% user, 0.0% nice, 0.0% system, 91.1% interrupt, 8.9% idle
CPU2 states: 1.0% user, 0.0% nice, 0.0% system, 72.5% interrupt, 26.5% idle
CPU3 states: 0.0% user, 0.0% nice, 0.0% system, 74.3% interrupt, 25.7% idle
Memory: 35M Act, 8312K Exec, 16M File, 7838M Free
Swap: 513M Total, 513M Free
```

PID	USERNAME	PRI	NICE	SIZE	RES	STATE	TIME	WCPU	CPU	COMMAND
0	root	0	0	0K	13M	CPU/2	104:29	0.00%	0.00%	[system]
2183	root	85	0	18M	2596K	kqueue/3	0:10	0.00%	0.00%	tmux
3223	root	43	0	17M	1852K	CPU/3	0:01	0.00%	0.00%	top
1822	root	85	0	14M	3108K	select/2	0:01	0.00%	0.00%	iwatch
1848	root	85	0	77M	5612K	select/3	0:00	0.00%	0.00%	ssh
1625	postfix	85	0	48M	3900K	kqueue/3	0:00	0.00%	0.00%	qmgr
1557	postfix	85	0	48M	3872K	kqueue/3	0:00	0.00%	0.00%	pickup
1371	root	85	0	58M	2648K	select/1	0:00	0.00%	0.00%	sshd
1603	root	85	0	48M	2344K	kqueue/1	0:00	0.00%	0.00%	master
590	root	85	0	24M	2000K	kqueue/0	0:00	0.00%	0.00%	syslogd
1636	root	85	0	18M	1928K	kqueue/1	0:00	0.00%	0.00%	tmux
4329	root	85	0	13M	1780K	ttyraw/2	0:00	0.00%	0.00%	sh
331	root	85	0	13M	1780K	wait/1	0:00	0.00%	0.00%	sh
1875	root	85	0	13M	1780K	wait/1	0:00	0.00%	0.00%	sh
200	root	85	0	13M	1748K	wait/0	0:00	0.00%	0.00%	sh

```
"sh intrctl.sh" on every second          Fri Mar 4 04:38:46 2016
Reverse mode: [w]word [e]line [r]char [t]toggle
six2 vec 0          99*          20          0
msix2 vec 1      948257*          0          132123          0
msix2 vec 2      390013          566916*          0          131728
msix2 vec 3      561934          0          580547*          0
msix2 vec 4      425449          145090          0          580050*
msix2 vec 5          56*          0          0
```

```
Netmap Packet-Gen v0.10
Interface: igb3          <<< Interface: igb2
Total Count:
TX:          0 pkt      TX:          520545 pkt
TX-etc:      0 pkt      TX-etc:      0 pkt
TX-underrun: 0 pkt      TX-underrun: 0 pkt
RX:          520481 pkt  RX:          0 pkt
RX-drop:     0 pkt      RX-drop:     0 pkt
RX-dup:      0 pkt      RX-dup:      0 pkt
RX-reorder:  133479 pkt  RX-reorder:  0 pkt
RX-reorder/flow: 0 pkt  RX-reorder/flow: 0 pkt
RX-flowctrl: 0 pkt      RX-flowctrl: 0 pkt
RX-arp:      0 pkt      RX-arp:      0 pkt
RX-icmpecho: 0 pkt      RX-icmpecho: 0 pkt
  icmpunreach: 0 pkt    icmpunreach: 0 pkt
  icmpredirect: 0 pkt   icmpredirect: 0 pkt
  icmpother:   0 pkt    icmpother:   0 pkt
RX-other:    0 pkt      RX-other:    0 pkt
Delta:
TX:          0 pps      TX:          281548 pps
TX:          0 bytes/s  TX:          23650032 bytes/s
TX: 0.000000000 Mbps   TX: 189.2002560 Mbps
RX:          281474 pps  RX:          0 pps
RX: 23643816 bytes/s   RX:          0 bytes/s
RX: 189.1505280 Mbps   RX: 0.000000000 Mbps
Latency:      min: 0.069198000 ms  min: 0.000000000 ms
              max: 2.596037000 ms  max: 0.000000000 ms
              avg: 0.806481761 ms   avg: 0.000000000 ms
Control:
  Hz: 1000      Flow:[14 ]      Traffic: Burst[*]/Steady[ ]
TX-control:
  TX-pktsize:[46 ]      TX-pktsize:[46 ]
  TX-pps: [0 ]          TX-pps: [297619 ]
  (max sustained:0 )    (max sustained:1488095 )
  Mbps: 0.00000000      Mbps: 199.999968
  Start[ ]/Stop[*]      Start[*]/Stop[ ]
USAGE:
'z' - clear statistics, 'q' - quit
<TAB>,<ARROW>,<^N>,<^P> - select, <ENTER> - edit, <ESC> - cancel
```

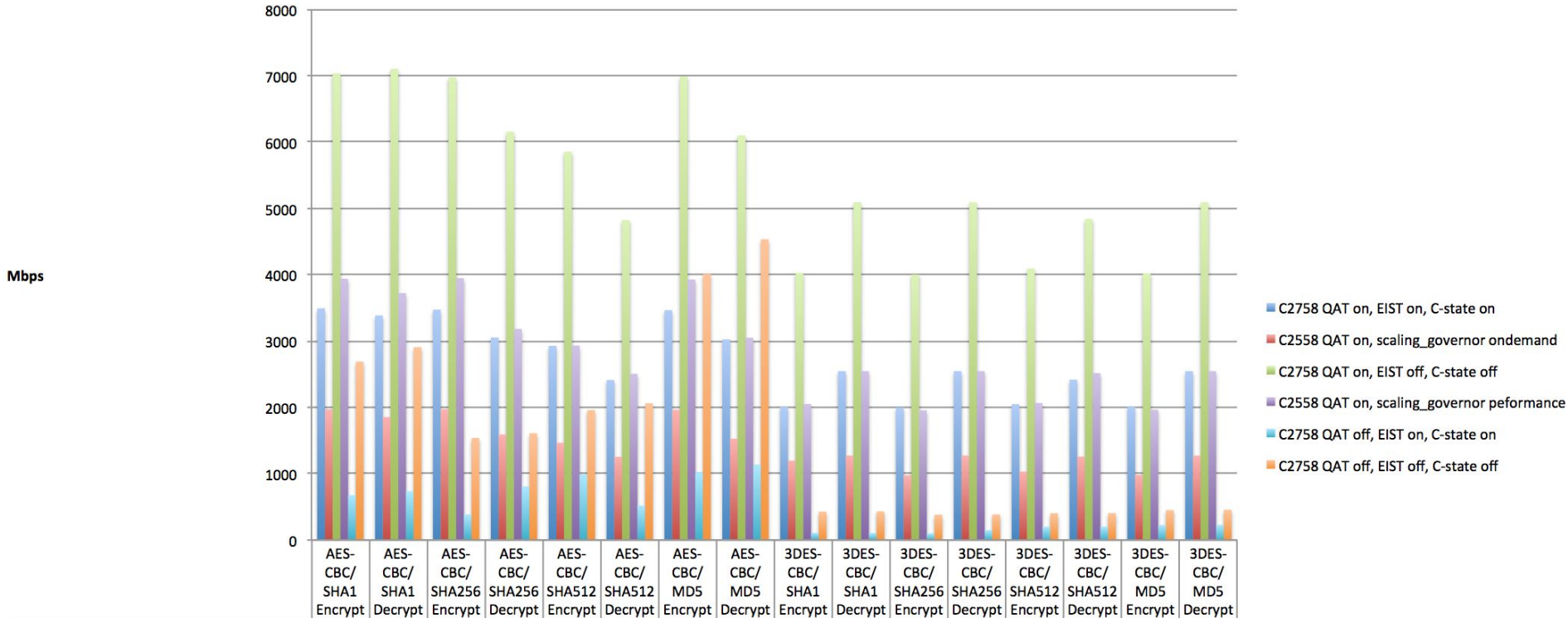
SEIL/BPV4

- Press release (in Japanese)
 - <http://www.iiij.ad.jp/news/pressrelease/2015/0930.html>
- 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



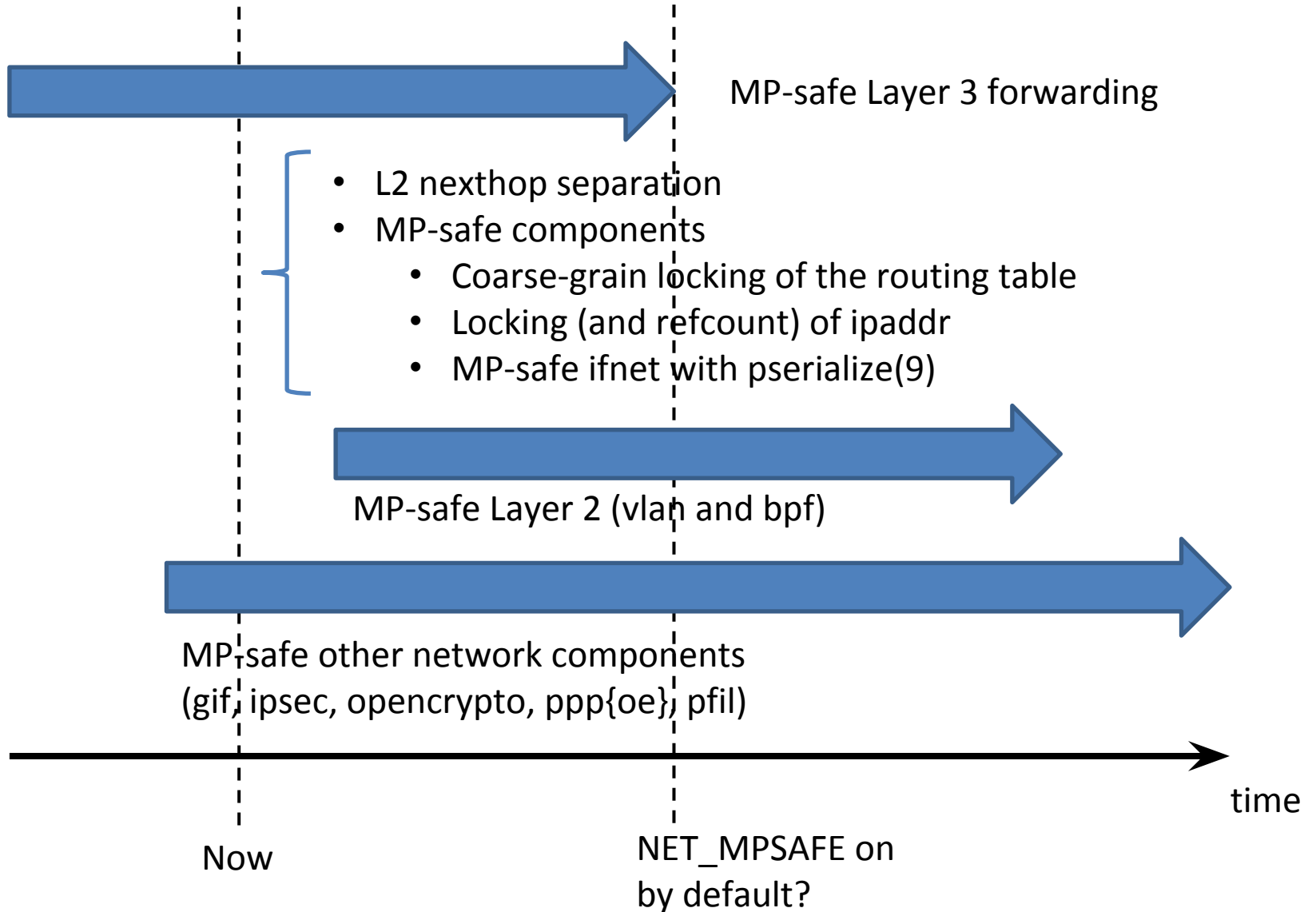
Performance of qat

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



	AES-CBC/SHA1 Encrypt	AES-CBC/SHA1 Decrypt	AES-CBC/SHA256 Encrypt	AES-CBC/SHA256 Decrypt	AES-CBC/SHA512 Encrypt	AES-CBC/SHA512 Decrypt	AES-CBC/MD5 Encrypt	AES-CBC/MD5 Decrypt	3DES-CBC/SHA1 Encrypt	3DES-CBC/SHA1 Decrypt	3DES-CBC/SHA256 Encrypt	3DES-CBC/SHA256 Decrypt	3DES-CBC/SHA512 Encrypt	3DES-CBC/SHA512 Decrypt	3DES-CBC/MD5 Encrypt	3DES-CBC/MD5 Decrypt
C2758 QAT on, EIST on, C-state on	3492	3385	3473	3054	2927	2412	3465	3025	2009	2546	1999	2546	2048	2417	2010	2546
C2558 QAT on, scaling_governor ondemand	1970	1853	1972	1591	1465	1254	1963	1528	1196	1273	977	1273	1032	1258	982	1273
C2758 QAT on, EIST off, C-state off	7034	7103	6974	6155	5854	4823	6984	6099	4026	5092	4002	5092	4093	4838	4019	5092
C2558 QAT on, scaling_governor performance	3938	3722	3947	3182	2932	2505	3927	3050	2051	2546	1956	2546	2065	2516	1965	2546
C2758 QAT off, EIST on, C-state on	676	735	386	807	989	519	1025	1137	108	107	99	148	202	203	226	228
C2758 QAT off, EIST off, C-state off	2690	2907	1539	1610	1958	2061	4013	4534	428	432	382	385	405	407	452	456

Roadmap



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 `rtcachel`?
 - 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