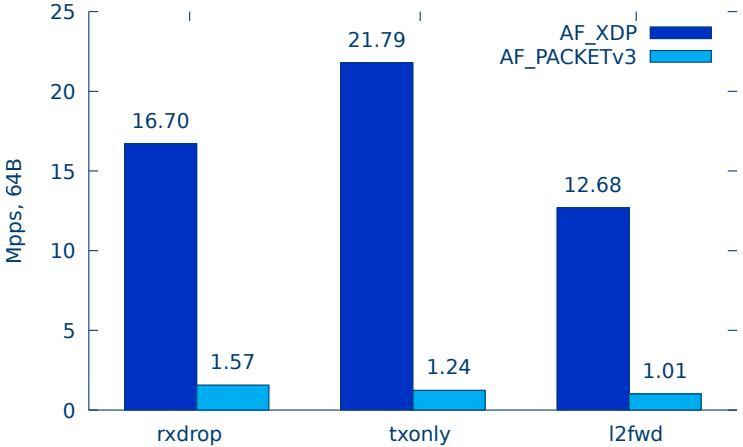
Low-Latency, Deterministic Networking with Standard Linux using XDP Sockets

Björn Töpel, bjorn.topel@intel.com, @bjorntopel
Magnus Karlsson, magnus.karlsson@intel.com

Embedded Linux Conference Europe, Lyon, 2019

Coming soon...



Legal Disclaimer

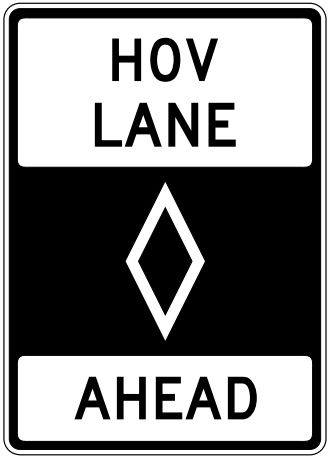
- Intel technologies may require enabled hardware, specific software, or services activation. Check with your system manufacturer or retailer.
- No computer system can be absolutely secure.
- Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks.
- Cost reduction scenarios described are intended as examples of how a given Intel- based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.
- All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.
- No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.
- Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.
- Intel, the Intel logo, and other Intel product and solution names in this presentation are trademarks of Intel.
- Other names and brands may be claimed as the property of others.
-

$ whoami

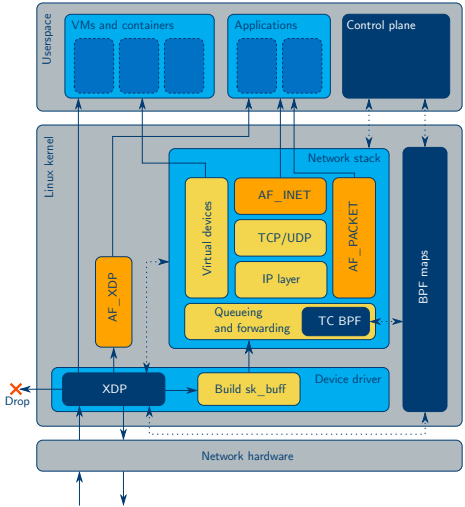
Björn Töpel
@bjorntopel
bjorn.topel@intel.com

Linux kernel networking hacker @ Intel
AF_XDP and RISC-V BPF JIT maintainer

XDP sockets?



The big picture



Good ol' sockets

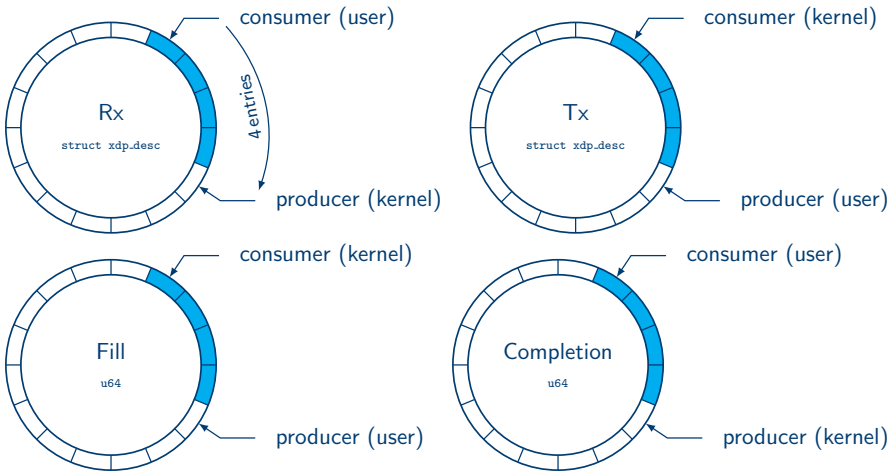
```
/* Cooked INET sockets */
fd = socket(PF_INET, SOCK_DGRAM, 0);
bind(fd, addr);
for (;;)
    receive_packet(fd, buff);
    send_packet(fd, buff);
```

```
/* Raw XDP sockets */
fd = socket(PF_XDP, SOCK_RAW, 0);
/* mmap/posix_memalign/malloc */
pktbuffs = alloc_buffs();
setsockopt(fd, SOL_XDP,
           XDP_MEM_REG, pktbuffs);
setsockopt(fd, SOL_XDP,
           XDP_{RX,TX,FILL,COMPLETE}_RING,
           ring_size);
/* map kernel rings */
{rx,tx,f,c}_ring = mmap(..., fd, ...);
bind(fd, {"eth0", qid});
for (;;)
    read_process_send_packets(fd);
```

Descriptors

```
$ grep -A 5 Rx/Tx\ descriptor include/uapi/linux/if_xdp.h
/* Rx/Tx descriptor */
struct xdp_desc {
    __u64 addr;
    __u32 len;
    __u32 options;
};
```

Rings, rings, and more rings



Syscalls

Pitfalls

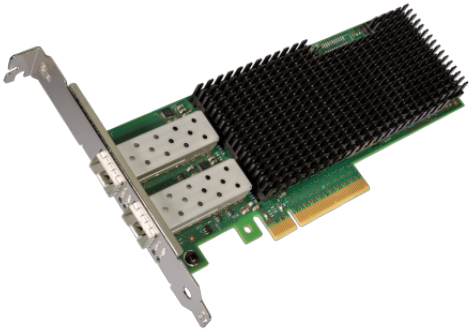
Docs and Samples

Docs: `Documentation/networking/af_xdp.rst`

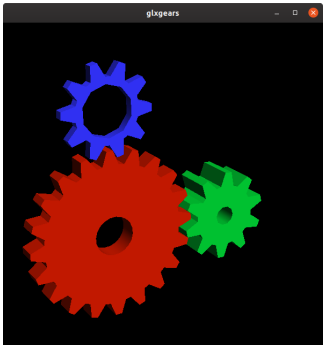
Samples: `samples/bpf/xdpsock_user.c`

libbpf: `tools/lib/bpf/*`

Your Driver Here



Softirqs, NAPI, and SPSC rings



XDP modes

AF_XDP zero-copy driver support

```
/* from include/linux/netdevice.h */
enum bpf_netdev_command {
...
    XDP_SETUP_XSK_UMEM,
};
struct netdev_bpf {
    enum bpf_netdev_command command;
    union {
    ...
            /* XDP_SETUP_XSK_UMEM */
            struct {
                    struct xdp_umem *umem;
                    u16 queue_id;
            } xsk;
    };
};
...
int (*ndo_bpf)(struct net_device *dev, struct netdev_bpf *bpf);
int (*ndo_xdp_xmit)(struct net_device *dev, int n, struct xdp_frame **xdp,
                    u32 flags);
int (*ndo_xsk_wakeup)(struct net_device *dev, u32 queue_id, u32 flags);
```

Code

`net/xdp/* kernel/bpf/xskmap.c`

`drivers/net/ethernet/intel/i40e/*`

`drivers/net/ethernet/intel/ice/*`

`drivers/net/ethernet/intel/ixgbe/*`

`drivers/net/ethernet/mellanox/mlx5/*`

...soon Broadcom

Test setup

Linux pre-5.5 (bpf-next) non-preemptive ‘‘`mitigations=on`’’

Intel Xeon Gold 6154 CPU @ 3.00GHz (Skylake)

Intel XL710 40GbE (i40e) NIC

1 Rx HW queue, 1 Tx HW queue

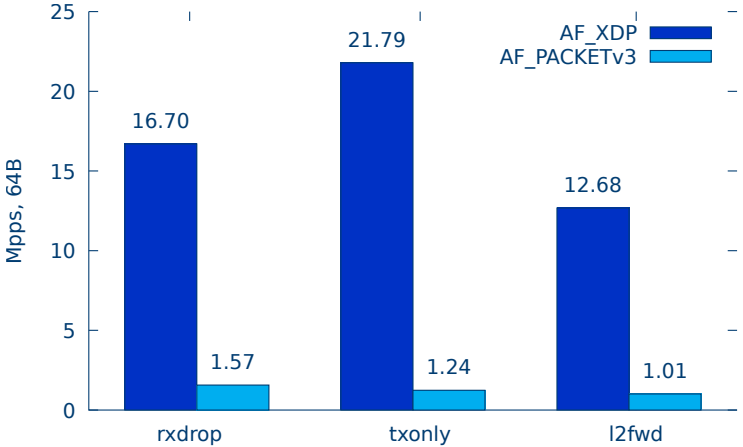
“two cores”: kernel and userland processing on different cores

“one core”: kernel and userland processing on same core

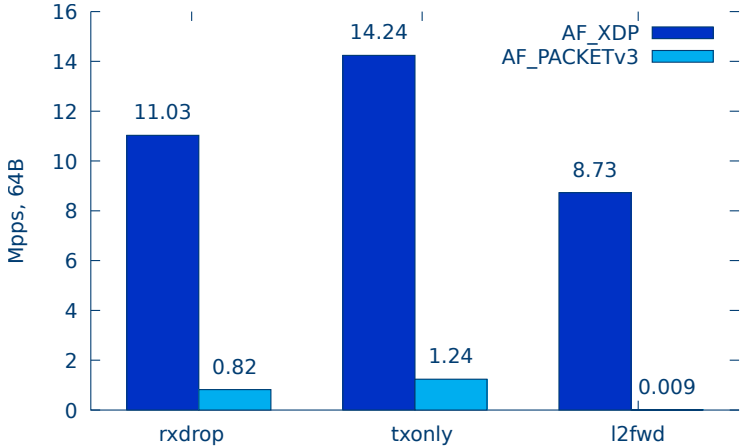
IXIA packet load generator

Latency is end-to-end, measured at load generator

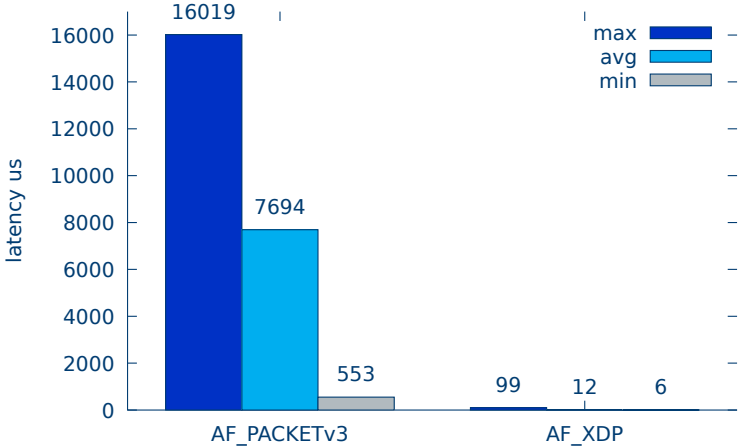
AF_PACKETv3 vs AF_XDP, throughput, two cores



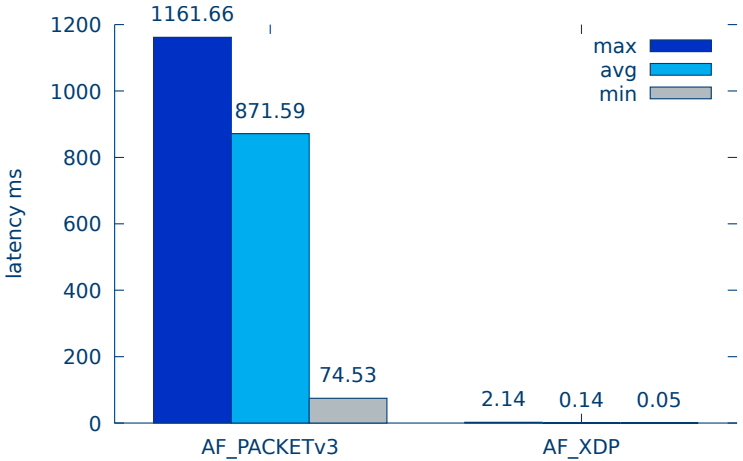
AF_PACKETv3 vs AF_XDP, throughput, one core



AF_PACKETv3 vs AF_XDP, e2e latency, normal



AF_PACKETv3 vs AF_XDP, e2e latency, fire hose



Thanks!

- Ilias Apalodimas
- Daniel Borkmann
- Jesper Dangaard Brouer
- Maciej Fijalkowski
- Andy Gospodarek
- Toke Høiland-Jørgensen
- Jakub Kicinski
- Kevin Laatz
- Jonathan Lemon
- Ciara Loftus
- Ilya Maximets
- Maxim Mikityanskiy
- David S. Miller
- Bruce Richardson
- Sridhar Samudrala
- Alexei Starovoitov
- All the companies/people hacking XDP!

...and Linus Torvalds for releasing his kernel to the public!


AF
XDP