AF_XDP impact on Latency

Killian Castillon du Perron
Fabrice Huet
Dino Lopez-Pacheco
Université de la Côte d’Azur, CNRS, I3S
I – Motivation
Impact of microseconds in cloud environments

• Today -> lot of microservices, every microsecond is important
• Hardware has really high bandwidth now (40Gbps – 100 Gbps)
• Packet processing in the network stack is the slower side
• Need to improve performance -> Throughput
• Need to lower latency
Packet processing mitigation techniques

- Bypass the Linux kernel networking stack
  - kernel space -> XDP
  - user space -> AF_XDP
  - user space -> DPDK
- XDP and AF_XDP are fully integrated into the Linux environment
Introduction to XDP

- First hook in the network stack
- In driver space
- Allows the bypass of the networking side when receiving packets

Inspired by T. Høiland-Jørgensen et al., ‘The eXpress data path: fast programmable packet processing in the operating system kernel’
Introduction to AF_XDP

- Socket type like classic AF_INET
- Used to communicate with the network through the XDP hook
- XDP send to AF_XDP socket on receiving side
- AF_XDP write directly to the hardware when sending

Inspired by T. Høiland-Jørgensen et al., ‘The eXpress data path: fast programmable packet processing in the operating system kernel’
Underlying principle of AF_XDP

RX: Signals that a packet has been received
TX: Signals that a packet has to be transmitted
Fill and completion rings: Exchanging ownerships on memory frames (UMEM)
II – AF_XDP packet flow
AF_XDP receive example

• 1 – Driver receives a packet and store it into the memory
AF_XDP receive example

1 – Driver receives a packet and store it into the memory
2 – Driver signals to the application that a packet has been received and stored
AF_XDP receive example

1. Driver receives a packet and store it into the memory
2. Driver signals to the application that a packet has been received and stored
3. Application sees that there is a new packet into its RX ring
AF_XDP receive example

1 – Driver receives a packet and store it into the memory
2 – Driver signals to the application that a packet of a certain length has been received and stored at a particular address
3 – Application sees that there is a new packet into its RX ring
4 – Application processes the packet, and put its memory address into the fill ring
AF_XDP receive example

1 – Driver receives a packet and store it into the memory
2 – Driver signals to the application that a packet of a certain length has been received and stored at a particular address
3 – Application sees that there is a new packet into its RX ring
4 – Application processes the packet, and put its memory address into the fill ring
5 – Driver sees that there is a new entry into the fill ring and can therefore release the memory address used for the packet
AF_XDP receive example

- 1 – Driver receives a packet and stores it into the memory
- 2 – Driver signals to the application that a packet of a certain length has been received and stored at a particular address
- 3 – Application sees that there is a new packet into its RX ring
- 4 – Application processes the packet, and puts its memory address into the fill ring
- 5 – Driver sees that there is a new entry into the fill ring and can therefore release the memory address used for the packet
AF_XDP send example

- 1 – Application wants to send a packet and put it into the UMEM
AF_XDP send example

- 1 – Application wants to send a packet and put it into the UMEM
- 2 – Application signals to the driver that a packet has to be sent by putting it into the TX ring
AF_XDP send example

1. Application wants to send a packet and put it into the UMEM
2. Application signals to the driver that a packet has to be sent by putting it into the TX ring
3. The drivers sees a new packet in the TX ring and send it
AF_XDP send example

1 – Application wants to send a packet and put it into the UMEM
2 – Application signals to the driver that a packet has to be sent by putting it into the TX ring
3 – The driver sees a new packet in the TX ring and sends it
4 – The driver puts the memory area associated with the sent packet into the completion ring
AF_XDP send example

1. Application wants to send a packet and put it into the UMEM
2. Application signals to the driver that a packet has to be sent by putting it into the TX ring
3. The drivers sees a new packet in the TX ring and send it
4. The drivers put the memory area associated with the sent packet into the completion ring
5. Application sees that there is a new entry into the completion ring, release it and can therefore ask to send another packet
AF_XDP send example

1 – Application wants to send a packet and put it into the UMEM
2 – Application signals to the driver that a packet has to be sent by putting it into the TX ring
3 – The drivers sees a new packet in the TX ring and send it
4 – The drivers put the memory area associated with the sent packet into the completion ring
5 – Application sees that there is a new entry into the completion ring, release it and can therefore ask to send another packet
Conclusion on the architecture

• Not as easy as a typical AF_INET socket
• Lot of things to do by hand
• Usage of a library interacting with the low-level API recommended
III – Some results
Our testbed

• 2 servers
  • Recent one with AMD EPYC 7443P 24-Core Processor
  • Older one with Intel(R) Xeon(R) CPU E5-2420 v2 @ 2.20GHz
  • Both with 100 Gbps Mellanox network interfaces

• Pings on baremetal
Libraries used

• Before everything in libbpf
  • Since libbpf 1.0, everything moved into libxdp
  • Libbpf
    • Found in recent kernels
    • [https://github.com/libbpf/libbpf](https://github.com/libbpf/libbpf)
  • Libxdp
    • Only found in recent version of Redhat?

• Rust library available
  • [https://github.com/DouglasGray/xsk-rs](https://github.com/DouglasGray/xsk-rs)
AF_INET vs AF_XDP

Ping between 2 hosts with AF_XDP and without (lower is better)
Kernel mechanisms impacting latency

• Cstates
  • CPU might be in an energy saving state

• RX / TX Coalescing
  • Delays packets for batch processing

• NAPI scheduling
  • Driver

• Others?
Impact of CStates

Ping between 2 hosts with AF_XDP
Impact of packet rate on latency

Latencies in 99th percentile

AF_XDP + all kernel optimisations activated
Conclusion

• AF_XDP / XDP allow to have a lower latency than using a classical socket

• Comes with some challenges
  • Using it requires some low level programming
  • Need to rewrite / reuse in user space several networking stack functions
Questions?

• More info:
  • M. Karlsson and B. Topel, ‘The Path to DPDK Speeds for AF XDP’
  • W. Tu, Y.-H. Wei, G. Antichi, and B. Pfaff, ‘revisiting the open vSwitch dataplane ten years later’