BALANCING APPLICATION PORTABILITY AND PERFORMANCE

Tim O’Driscoll

June 24th 2019
Software Platform Considerations

A software platform should have the following characteristics:

- Robust and reliable: Commercially supported software, or open source software with a strong community
- Proven: A widely used, “standard”, multi-vendor API
- Easy to use: Well structured software, good documentation, easy to use API
- High quality: New releases are thoroughly tested to minimize defects
- Stable: Easy to upgrade to new releases
- Portable: Allows application to run on a wide variety of target platforms
- High performance: Supports maximum throughput

These items are well covered by DPDK

Upgrading DPDK versions is difficult

Difficult to balance portability and performance
Performance vs Portability

- Kernel vSwitch
- DPDK vSwitch
  - vSwitch Acceleration
  - Preserve portability benefits of vswitch, but improve performance by leveraging NIC capabilities and software optimisations.
- AF_XDP
  - Promising “middle ground” solution combining good performance and portability
- Ideal Solution
- SR-IOV
  - DPDK Usability
  - Improved DPDK stability and ease of use to make upgrades easier and more reliable

Note: Diagram is not to scale. For illustrative purposes only.
VSWITCH ACCELERATION
Open vSwitch Acceleration

Full offload via smart NICs

Partial offload via standard NICs:
  EMC/DPCLS look-up
  TCP Segmentation Offload

Software optimisations:
  Signature Match Cache
  Instruction set specific DPCLS

Virtio/Vhost acceleration:
  Virtio 1.1
  Data copy offload via Intel® QuickData Technology
Partial Offload: Overview

OVS supports offload of EMC/DPCLS lookup to network adapter

Support for Intel® Ethernet® 700 Series Network Adapter will be added in DPDK 19.08:

- I40E driver extended to support rte_flow MARK + RSS action
- Supports up to 8K rules

Will be supported in future releases for Intel® Ethernet® 800 Series Network Adapters.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

Configurations: See slide Partial Offload: Test Configuration

Performance results are based on testing as of February 21st 2019 and may not reflect all publicly available security updates. See configuration disclosure for details. No product or component can be absolutely secure.
TSO: Overview (Inter-Host, Egress)

**Without TSO**

- Application
- Network Stack
- OVS-DPDK

**With TSO**

- Application
- Network Stack
- OVS-DPDK

*Benefit is greater for intra-host (VM -> VM) case because packets are never segmented so they don’t need to be reassembled by the target VM*
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors.

Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.


Performance results are based on testing as of December 5th 2018 and may not reflect all publicly available security updates. See configuration disclosure for details. No product or component can be absolutely secure.
Signature Match Cache (SMC) introduced as an experimental feature in OVS 2.10.

SMC stores only a 16-bit signature for a flow, so it's more memory efficient than EMC:

- With the same memory space, EMC can store 8K flows, SMC can store 1M.

Can be used with EMC, or as an alternative to EMC:

- If used with EMC, EMC is checked first, then SMC.

Performance data reproduced from: Testing the Performance Impact of the Exact Match Cache, Andrew Theurer, Red Hat, presented at Open vSwitch 2018 Fall Conference.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors.

Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

Configurations: Testing performed by Red Hat. See Testing the Performance Impact of the Exact Match Cache for configuration details.

Performance results may not reflect all publicly available security updates. See configuration disclosure for details. No product or component can be absolutely secure.
AF_XDP: Overview

High performance interface from kernel to user space:

1. eXpress Data Path (XDP) runs in the kernel device driver and bypasses the network stack.
2. eBPF allows packet filtering in software.
3. AF_XDP socket provides high performance interface to userspace applications.

Supports both DPDK and non-DPDK applications:

DPDK support is via the AF_XDP PMD introduced in 19.05 release. See Xiaolong’s presentation.

3 modes of operation:

**SKB**: Lowest performance. Works with any kernel NIC driver.

**Copy**: NIC driver must support XDP. All common drivers do.

**Zero Copy**: Highest performance. Additional driver changes required. Only supported for Intel NICs (IXGBE & I40E) at present.

AF_XDP currently only supports packet I/O. Extensions required to support offloads/acceleration.

Packet size is currently limited to 4K.
**AF_XDP: Use Cases**

**Containers/Cloud Native**
- Provides high performance Kernel -> Container interface.
- Well suited to Cloud Native deployments.

**Split Kernel/Userspace Traffic**
- If traffic needs to be split between userspace and the Kernel network stack, this can be done at source in the Kernel.
- Can use hardware of software (BPF) filtering.

**Virtualization**
- Not well suited to virtualized environments.
- Could be used as interface between guest Kernel and userspace app, but still need virtio or SR-IOV to get traffic to the VM.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors.

Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

Configurations: See slide AF_XDP: Test Configuration

Performance results are based on testing as of December 13th 2018 and may not reflect all publicly available security updates. See configuration disclosure for details. No product or component can be absolutely secure.
AF_XDP: Future Enhancements

1. AF_XDP PMD enhancements (see Xiaolong's presentation for details):
   - Multi-queue
   - Busy poll support
   - Zero copy using external mbufs

2. Kernel enhancements:
   - Support for busy poll
   - More flexible memory handling
   - Rx and Tx optimisations
   - Remove 4K packet size limitation

3. Offload/Accelerator support:
   - Extend AF_XDP to support NIC offloads like TSO, L3/L4 checksum etc.

4. BPF Bypass:
   - Provide option to skip BPF if all traffic is to be routed to userspace
DPDK PORTABILITY/USABILITY
DPDK Portability/Usability Challenges

DPDK is typically tightly coupled (statically linked) to the application:
To support new hardware (e.g. a new NIC PMD), the application needs to be updated.

Upgrading to new DPDK versions is not easy:
ABI changes occur in every release, so application changes are always required when upgrading.

Goal is to move to a model where DPDK becomes platform software:
Dynamically linked
Sourced from OS distribution
Stable ABI makes upgrades easy
Simplifies porting of application to new hardware platforms
GStreamer Application Binary Interface
- 100% backward compatible within Major Versions (1.x).
- Stable since 1.4.5, typically < 1% change between Major Versions.


DPDK Application Binary Interface
- 8.7% median ABI churn between quarterly releases.
- LTS release is API stable for 2 years, however limited backporting of new features or HW.

ABI Stability Proposal

Major ABI versions will be declared every two years and will be supported for two years:
   All new releases in that two year period will be backward compatible with the major ABI version.

The supported ABI version will be reflected in an individual library's soname - <library name>.so.<major ABI version number>.

ABI changes in that 2 year period will be handled as follows:
   The addition of symbols does not generally break the ABI.
   The modification of symbols will be managed with ABI versioning.
   The removal of symbols is generally an ABI breakage. Once approved, this will form part of the next ABI revision.

Libraries or APIs marked as `experimental` are not considered part of the ABI version and may change without constraint.
ABI Stability Example

When DPDK 19.11 (LTS) is released, ABI v20 is declared as the supported ABI revision for the next two years. All library sonames are updated to reflect the new ABI version, e.g. librte_eal.so.20, librte_acl.so.20 . . .

DPDK releases 19.11 -> 21.08 are compatible with the v20 ABI. ABI changes are permitted from DPDK 20.02 onwards, with the condition that ABI compatibility with v20 is preserved.

When DPDK 21.11 (LTS) is released, ABI v21 is declared as the new supported ABI revision for the following two years. The v20 ABI is now deprecated, library sonames are updated to v21 and ABI compatibility breaking changes may be introduced in 21.11.
Other Possible Challenges

Consistency of DPDK APIs:
- Implementation of the ethdev API can vary between PMDs.
- Standardising this would be a big effort: a more detailed API specification, updates to drivers, conformance tests in the DPDK community lab etc.
- Benefit of doing this is unclear. Is this really an issue?
- Newer APIs (cryptodev, compressdev etc.) are more consistent.

Software fall-backs:
- Which hardware capabilities require software fall-backs?
- How transparent do these software implementations need to be? Does DPDK need to do more to make this transparent, or will this be handled in the application anyway?

More up to date DPDK versions in OS distributions:
- OS distros typically package the LTS releases. This gives good stability, but means that they’re not up to date with new features.
- Is there a need for more up to date DPDK releases in OS distros?
DISCLAIMERS AND CONFIGURATION INFO
Notices and Disclaimers

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

For more information go to www.intel.com/benchmarks.

Performance results are based on testing as of February 21st 2019 (Partial Offload) and December 13th 2018 (AF_XDP), and may not reflect all publicly available security updates. See configuration disclosure for details. No product or component can be absolutely secure.

Configurations: See slides Partial Offload: Test Configuration and AF_XDP: Test Configuration.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. Check with your system manufacturer or retailer or learn more at www.intel.com.

Intel does not control or audit third-party data. You should review this content, consult other sources, and confirm whether referenced data are accurate.

Intel and the Intel logo are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

© Intel Corporation.
Partial Offload: Test Configuration

Performance results are based on testing as of February 21st 2019
Intel® Xeon® Platinum 8160, 2.1 GHz, hyper-threading disabled
Intel® Ethernet Controller XL710, with firmware version 6.0.48442
Ubuntu 16.04.5 LTS
Linux kernel 4.4.0-137
OVS version: dpdk-latest branch 41b605b66f2ec1d85565d4be116ffbdd11c7b29f
DPDK version: 19.05-rc2 Pps switched (1 core) @ 64-byte
Single core performance with 64 byte packets in PHY-to-PHY configuration

Test scenarios (# offloaded flows sent / # rules matched):

- 10M flows / 10K rules: FLOWS: udp_src=1000-10999 x udp_dst=2000-2999, RULES: udp_src=1000-10999
AF_XDP: Test Configuration

Performance results are based on testing as of December 13th 2018

Dual socket Intel® Xeon® E5-2660:
  2.7 GHz with hyper-threading disabled
  BIOS version GRRFCRB1.86B.0261.R01.1507240936

Dual socket Intel® Xeon® Gold 6154:
  3.0 GHz with hyper-threading disabled
  BIOS version SE5C620.86B.01.00.0433.022820170740

Both configurations:
  Intel® Ethernet Controller XL710, with firmware version 6.01
  DDR4 memory @ 2133 MT/s (1067 MHz), 64 GB total
  Ubuntu 18.04.1 LTS
  Linux Kernel v4.19-rc6-2008-g438363c0feb8
  DPDK version 18.08

Tests use the xdpsock_user.c sample application:
  Rxdrop: RX only without touching packet data
  Txpush: TX only without touching packet data
  L2fwd: RX + swap MAC headers + TX