OpenFastPath

An open source user space fast path TCP/IP stack
Industry network challenges

- **Growth in data traffic** means that even small network nodes need a fast path
  - The Linux IP stack is slow and does not scale

- High **throughput IP processing** solutions have been around for a number of years
  - Why this now?

- Most existing implementations are either **hardware specific** or **proprietary closed source**
  - SoC vendor solutions and for example 6Wind

- Developing this **basic building** block from scratch in-house does not make sense
  - Not even for the big network equipment providers
Enter OpenFastPath!

A TCP/IP stack
- lives in user space
- is optimized for scalability and throughput
- uses Data Plane Development Kit (DPDK) and
- Open Data Plane (ODP) to access network hardware
- runs on ARM, x86, MIPS, PPC hardware
- runs natively, in a guest or in the host platform

The OpenFastPath project
- is a true open source project
- uses well known open source components
- open for all to participate – no lock-in to HW or SW
- Nokia, ARM and Enea key contributors
A main benefit with OFP is portability.

* Native support in execution
...performance - OFP is 20x Linux TCP/IP stack!

IP forwarding application in user space - 256 routes, 4 x 10 Gbps, 64Byte packets

- Intel Xeon E5-2697 v3 processor (turbo disabled)
- Two 82599 NICs with modified netmap ixgbe 4.1.5 driver (12 rx/tx queue pairs) totaling 4x10Gbps ports
- Ubuntu 14.04 - 3.16.0-53-generic. CPU isolation used to test kernel IP forwarding.
- OFP fpm_burstmode example application
- ODP 1.4.1.0 ext. with multi queue packet I/O support
Features implemented

**Fast path protocols processing:**
- Layer 4: UDP termination, TCP termination, ICMP protocol
- Layer 3
  - ARP/NDP
  - IPv4 and IPv6 forwarding and routing
  - IPv4 fragmentation and reassembly
  - VRF for IPv4
  - IGMP and multicast
- Layer 2: Ethernet, VLAN
- GRE and VXLAN Tunneling

**Routes and MACs are in sync with Linux**

**Integration with Linux Slow path IP stack through TAP interface**

**Command line interface**
- Packet dumping and other debugging
- Statistics, ARP, routes, and interface printing
- Configuration of routes and interfaces with VRF support

**OFP IP and ICMP implementations passing Ixia conformance tests**

**IP and UDP implementations has been optimized for performance**
- TCP implementation is functional but not performance optimized

**Integrated with NGiNX webserver**
OpenFastPath Source code

New open-source code
- Developed by partners during the incubation stage

UDP, TCP, ICMP code was ported from libuinet (User space FreeBSD port)
- Non-blocking event based socket API
- Modular, multithreaded design focused on performance and scalability
- Tightly coupled to application, linked in as a library
- Maintainability – Tracks evolution of FreeBSD

High performance and scalable implementation for MAC and Route tables
- Lockless synchronization
Ingress API
User/Default Dispatcher

OpenFastPath System View

Packet_cnt = odp_pktio_recv(pktio, pkt_tbl, OFP_PKT_BURST_SIZE);
or
buf = odp_schedule(&queue, ODP_SCHED_WAIT);
OpenFastPath multicore System View

- User Termination or Forwarding A
- User Termination or Forwarding B
- User Termination or Forwarding X

- Socket callback /Hook API
- OpenFastPath (OFP) (SMP multicore library)

- Init API
- PKTIO
- Ingress API
- Dispatcher
- DPDK
- ODP SW
- Core 0, Core 1, Core 2, ..., Core N

- ODP/DPDK FW/HW
- NICs

Single thread context
Ingress Packet Processing

IP, UDP, TCP, ... classified by HW

VXLAN

Ethernet VLAN

Ingress API

IPv4/v6

IPv4/v6 local hook API

IPv4 GRE

UDP input

IPv4/v6 forward hook API

TCP input

Transport (L4) classifier

IPv4 GRE

ICMP

NDP

IGMP

IPv4/v6 output

IPv4/v6 routing

Update MAC table

IPv4/v6 output

Send ARP request

Loopback to VXLAN

Socket API

GRE hook API

Fallback to slowpath for unknown traffic

Packets

Information

User API

L4

L3

L2

Pre-classified

OpenFastPath
Egress Packet Processing

- **Socket/Egress API**
  - UDP output
  - TCP output
  - ICMP error
  - IPv6 output
  - IPv4 output
  - IPv4 Fragmentation
  - IPv4 GRE tunneling
  - VXLAN
  - Ethernet VLAN

- **User API**
  - L4
  - L3
  - L2
  - Pre-classified

- Packets vs Information
Optimized OpenFastPath socket APIs

New zero-copy APIs optimized for single thread run-to-completion environments

- UDP
  - Send: Optimized send function with a packet container (packet + meta-data)
  - Receive: A function callback can be registered to read on a socket. Receives a packet container and socket handle

- TCP
  - Accept event: A function callback can be registered for TCP accept event. Receives socket handle.
  - Receive: A function callback can be registered to read on socket. Receives a packet container and a socket handle

Standard BSD Socket interface

- For compatibility with legacy applications
Other OpenFastPath user application APIs

- Initiation of Open Fast Path
- Interface configuration
- Route and MAC table access
- Packet Ingress and Egress processing
- Hooks for IP local, IP forwarding and GRE
- Timer callbacks
- Statistics
- Packet capture
Code examples

ODP thread creation above

OFP default dispatcher to the left
Why should someone use OpenFastPath?

**Portable high performance solution supporting multiple HW platforms**
- Functionality verified on ARM, MIPS and x86 HW

**Highly optimized and scalable solution**
- Non-blocking event based API focused on performance and scalability

**User space implementation**
- Simplifies maintenance and maximizes throughput and scalability by minimizing Linux kernel dependency

**Very flexible deployment scenarios**
- Embedded, virtualized, servers, edge nodes, etc.
Why engage in the OpenFastPath project?

OpenFastPath is designed as an open source project from the start
- Based on known open source code like libuinnet
- Not an old proprietary code base turned open source....

The framework is highly modular, adaptable and lightweight.
- Not restricted to plug-ins

Membership is cheap and open for all
- Potential to impact is high

Very high interest from major industry players
What’s next? - Get involved!

Download the source code from: https://github.com/OpenFastPath/ofp

Check us out at www.openfastpath.org to get more information about the project

Subscribe to Mailing-list: http://www.openfastpath.org/mailman/listinfo

Ping us on our freenode chat: #OpenFastPath

Membership is cheap and open to all!
Enea services offering on OFP

Integration services
- Integration of OFP in customer hardware and software system.

Hardware porting and optimization services
- Test, verification and optimization of silicon vendor ODP implementation together with OFP

Feature development services
- Pre-studying, specifying and implementing new OFP features and protocols.

Production test, maintenance and support services
- Production testing, release management and support.
Thank You

For additional information, please visit
www.openfastpath.org