High Performance Networking Leveraging the DPDK and the Growing Community

Thomas Monjalon

Packet Processing Engineer

and DPDK.org Maintainer

6WIND

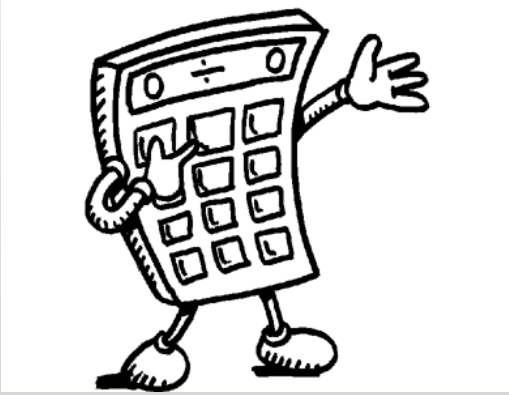
SPEED MATTERS

Agenda

- **How DPDK can be used for your Application**
  - Basic Architectures
  - IPsec
  - TCP Termination
  - Virtualization
- **DPDK Ecosystem boosting your Development**
  - Extensions
  - Support
  - Packaging
- **Meet the Community Challenges**
  - More Users
  - More NIC Vendors
  - More Developers
  - More Patches

Key numbers

- **Packets per second for 10Gbps with 64B packets**
  - 10000 / ((7 + 1 + 64 + 12) * 8) =
  - ~**15** Mpps
- **With some good configurations (CPU/NIC/driver),**
  - only ~**50** cycles to forward a packet (hyperthreading can help)
- **line rate forwarding for 10Gbps**
  - 15 x 50 = **750 MHz**
- **1 core@3GHz can forward 40Gbps**

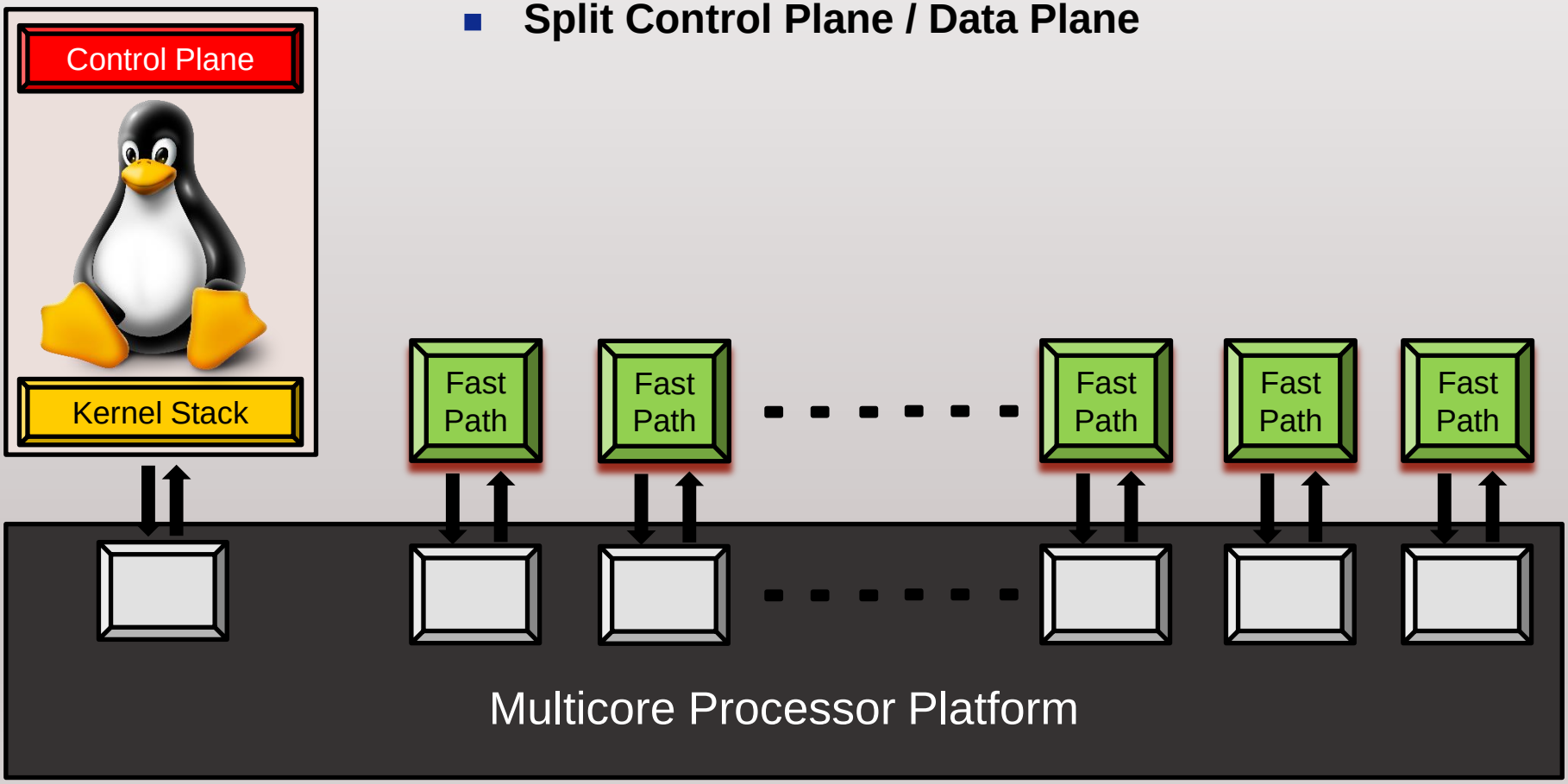


- **Multicore CPU can achieve**
  - stratospheric performance
  - or keep cores available for application

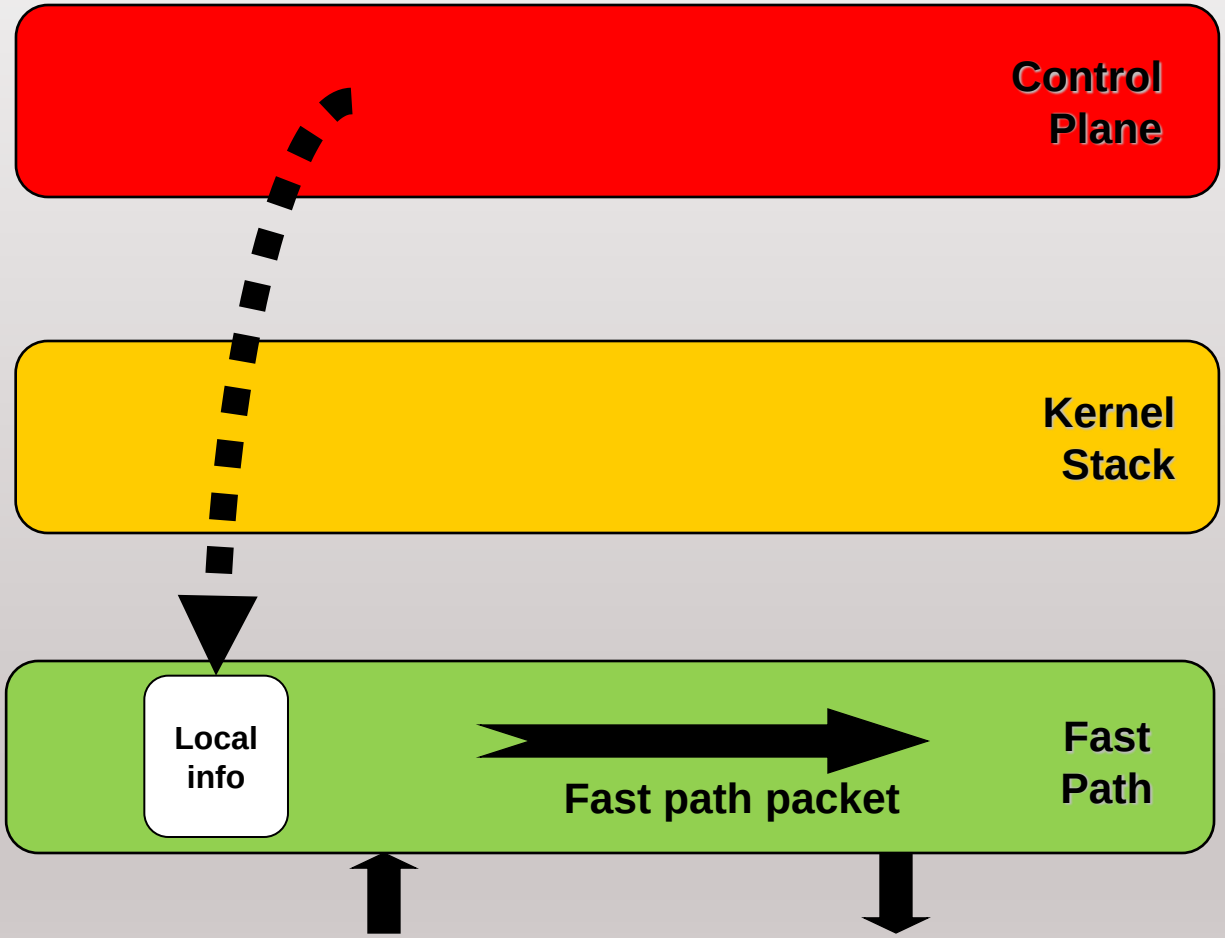
- **Cost of stack (extra cycles) becomes important**

DPDK + Packet Processing Software = High Performance Networking Stack

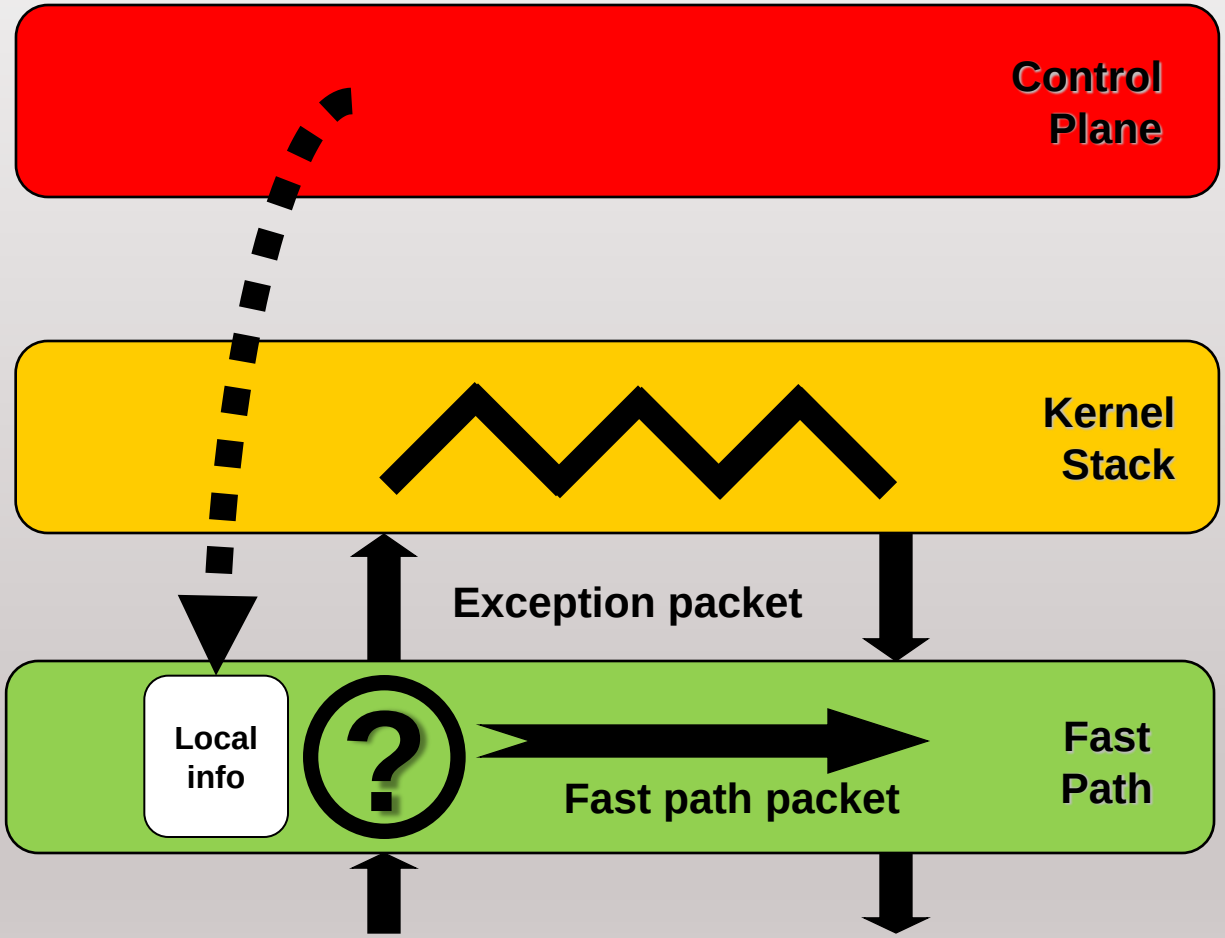
- Tune fastest performance: DPDK Cores allocation
- Split Control Plane / Data Plane



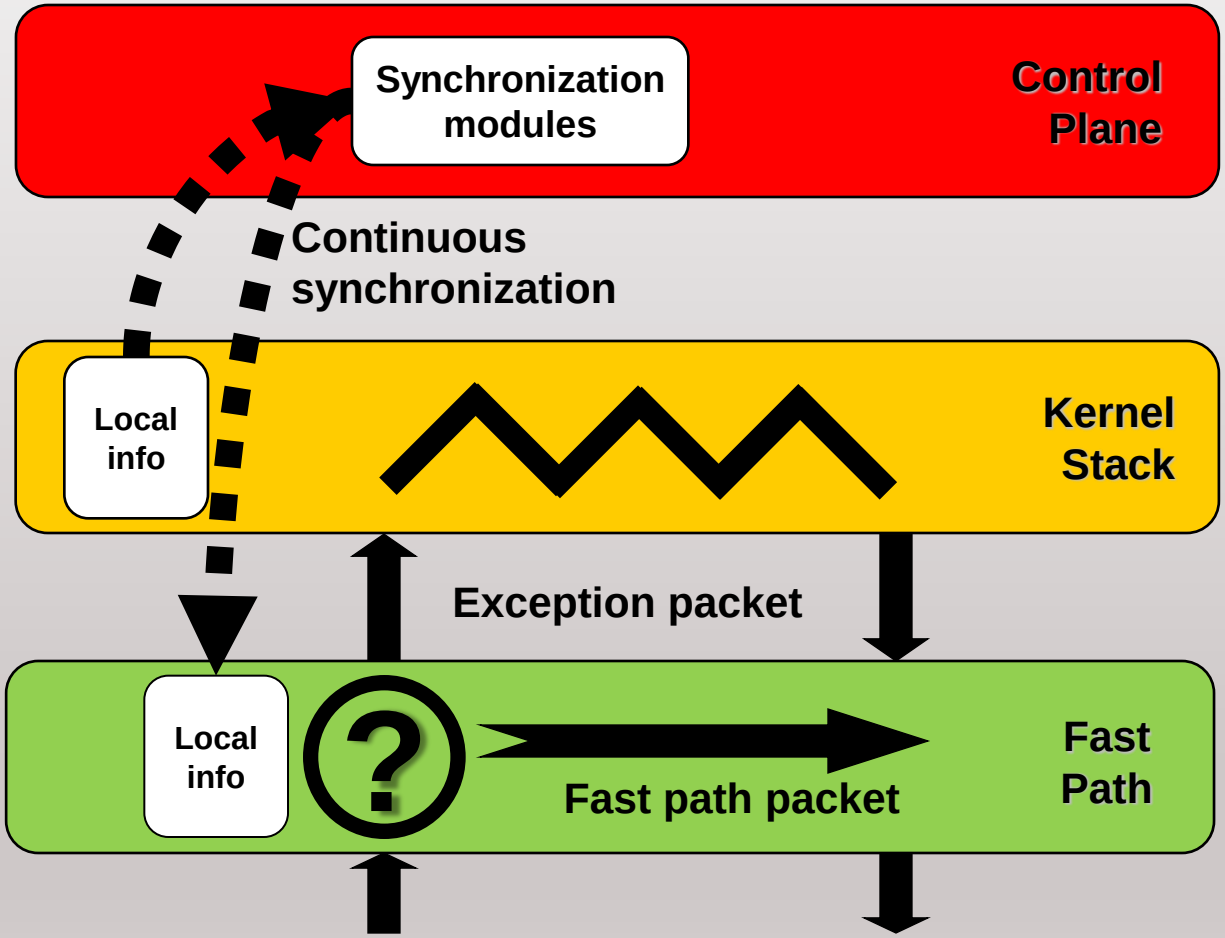
Simple Architecture



Split Architecture



Synchronized Architecture



Use cases

- **Some applications use only DPDK to analyze raw packets**
  - Firewall / not compatible with Linux
  - DPI
- **Some applications need complex networking stacks on top of DPDK**
  - IPsec
  - TCP
  - BRAS
- **Small dedicated fast path stacks are better in their job than generic kernel ones**

Parallel Processing

- **Standard applications use**
  - 1 TCP/UDP socket (single point of connection with stack/drivers)
  - Many threads (possibly dynamically allocated)
  - 1 dispatching loop
- **DPDK applications leverage parallelization from the beginning**
  - Multiqueue
  - Multicore
  - 1 loop per core / run to completion
- **DPDK requires static initialization of cores and memory**

Run to completion

- Each cycle is important = no time for scheduling

- Application is in the polling thread which receive packets
- Application should never block a long time

- Power management like sleeping
- Poll mode can start on IRQ (NAPI style)

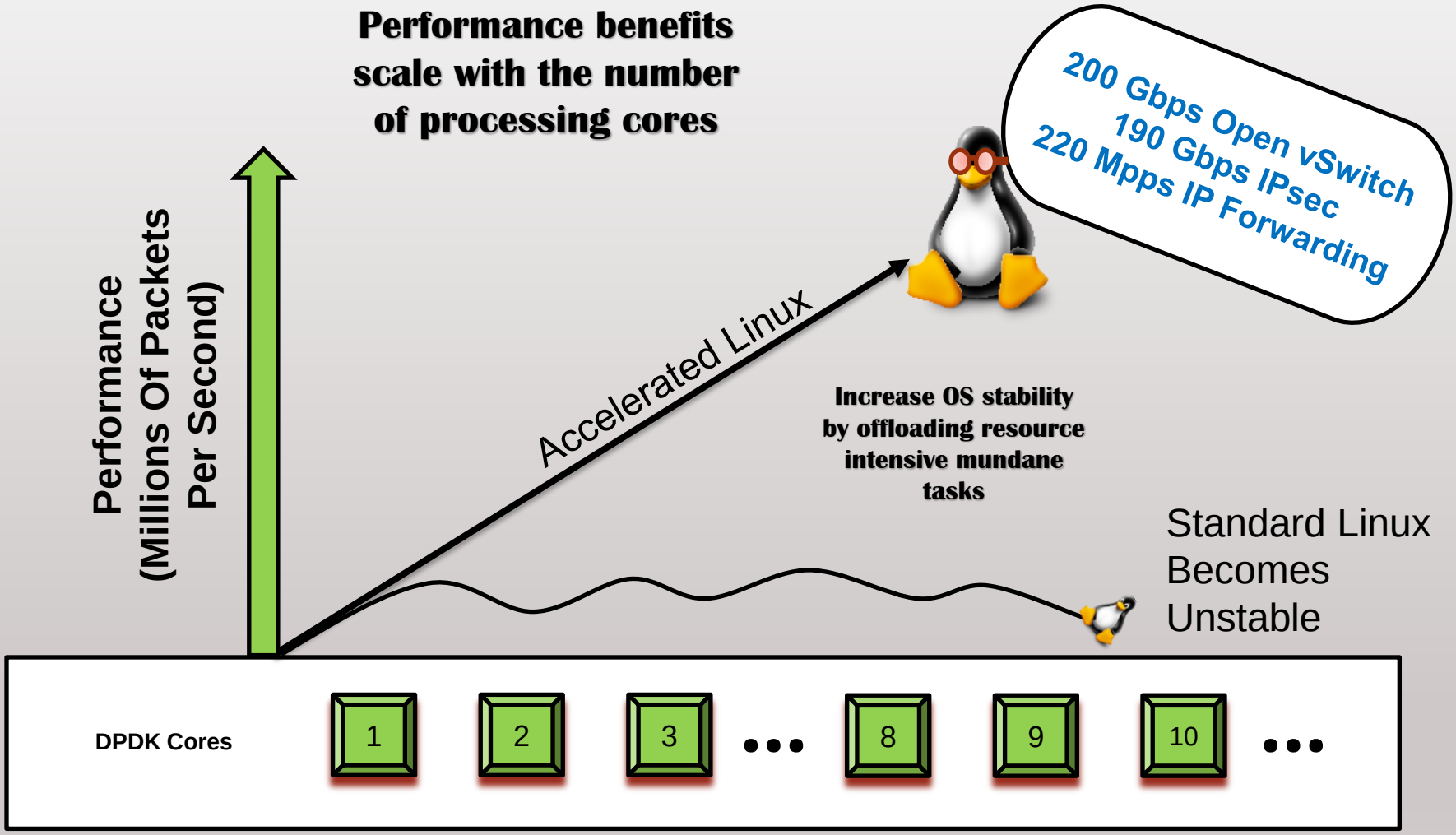
Efficient memory access

- **Dedicated memory allocator**
  - DPDK functions
  - NUMA allocation (QPI must be avoided)
- **Hugepages = less TLB cache misses**

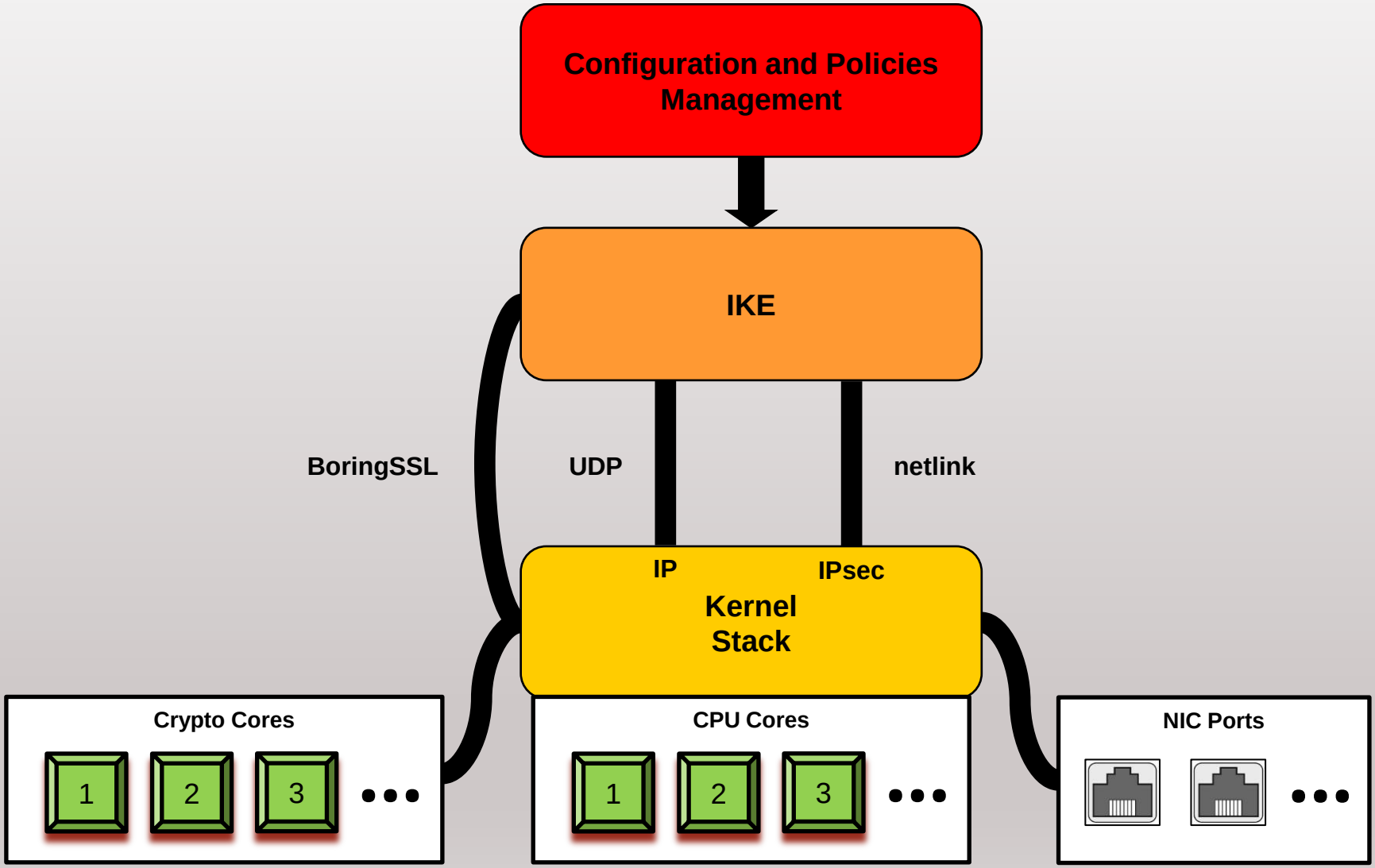
- **Local memory objects for no locking**
- **Spinlocks or atomic instructions**

- **Small buffers (rte_mbuf <<< skbuff) = less cache misses**
- **Big pool of buffers delays packet loss but has horrible properties because of L3 misses**
- **Small memory usage allows efficient L3 caching**

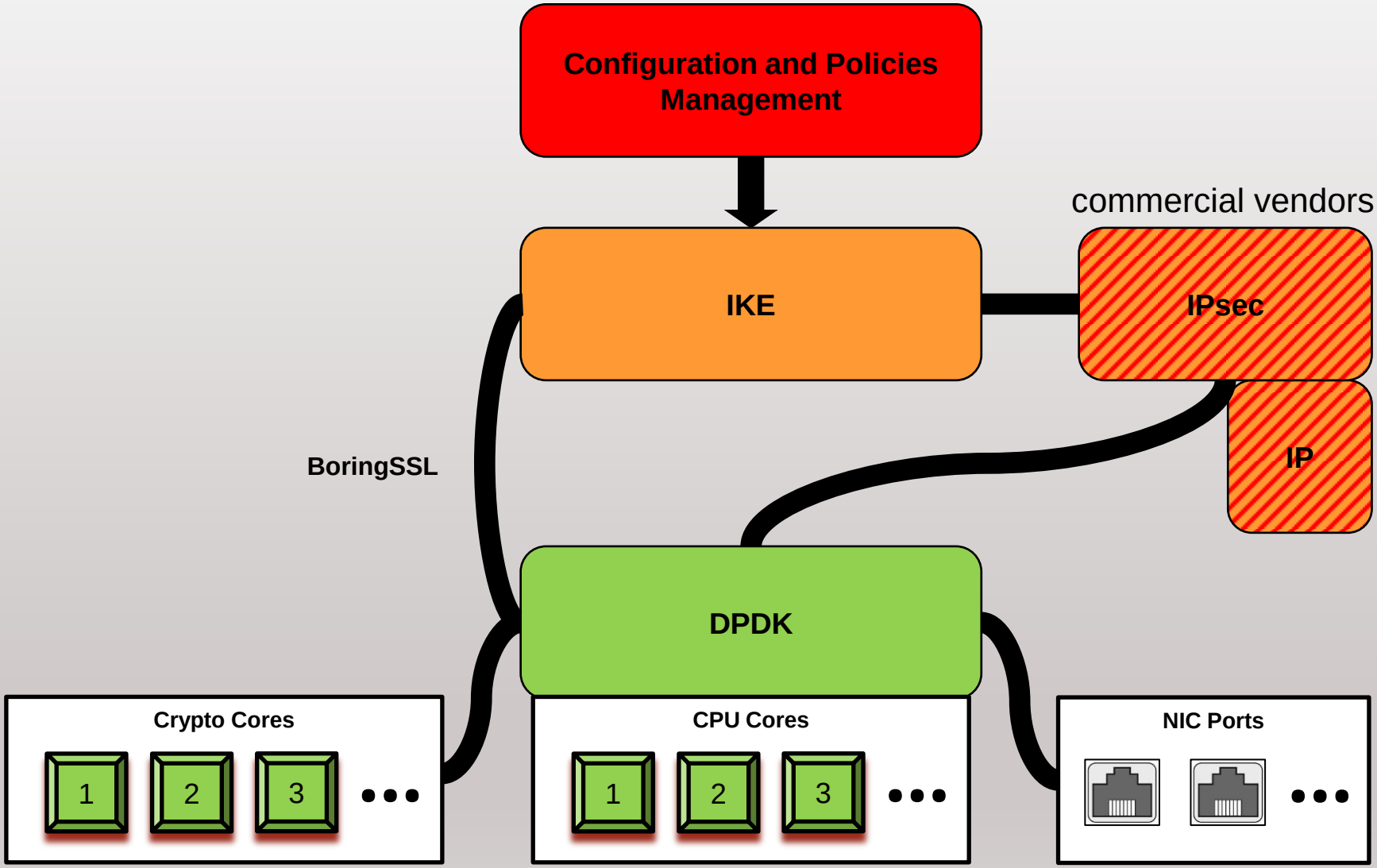
Packet Processing Software Turbo Boosts Linux



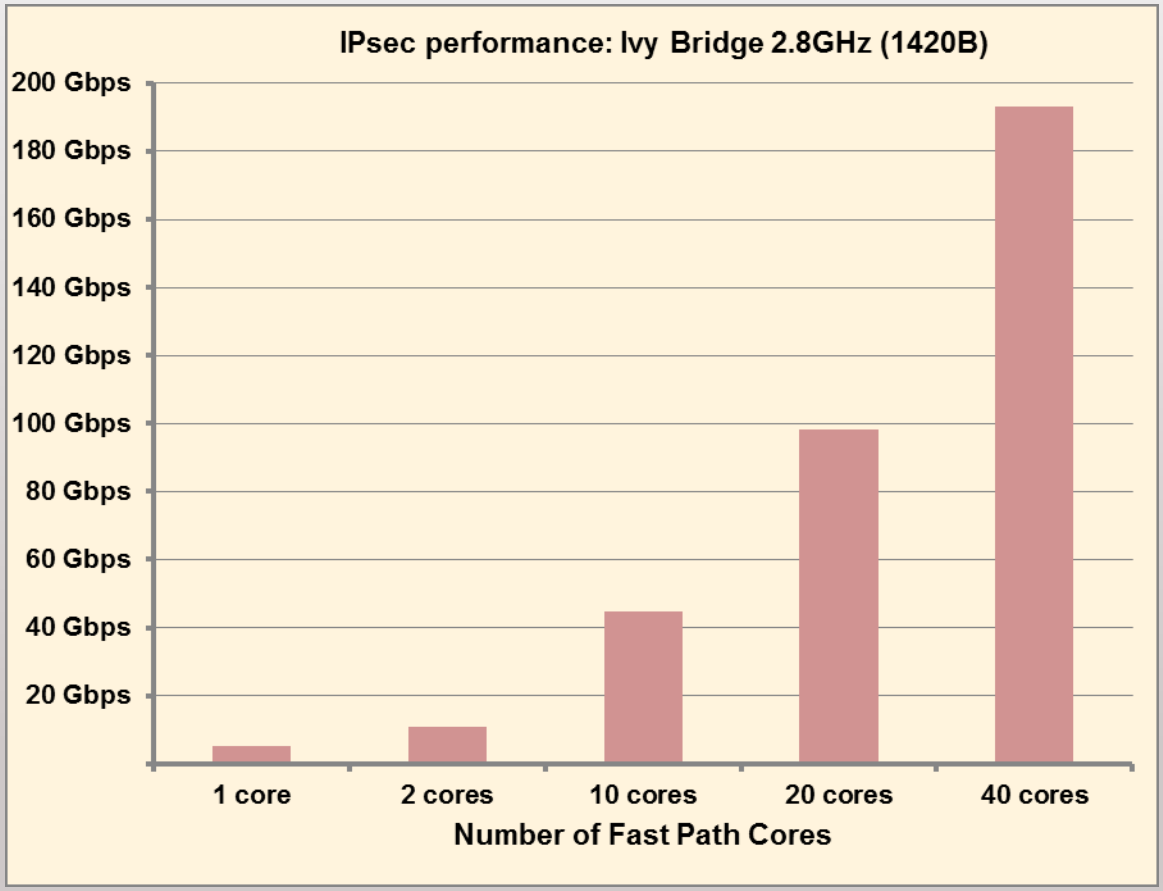
Legacy IPsec



DPDK: new IPsec stack?



Example of DPDK based IPsec performance



- SSE/AVX crypto
- Processing AES-128 HMAC-SHA1 on big packets (1420B)
- Performance scales linearly with the number of cores
- Near 200 Gbps using 40 cores

Parallelized TCP termination

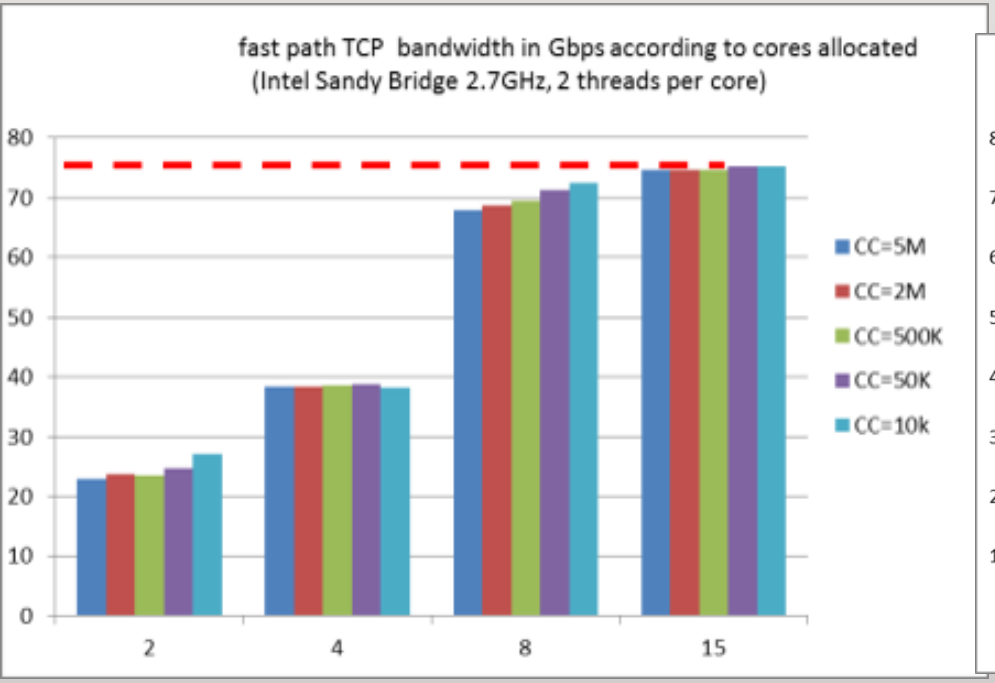
- **RSS dispatching to multicore workers**
- **Parallel workers are DPDK cores (not flying threads)**
- **Workers have DPDK style main loop for polling (not event scheduling)**

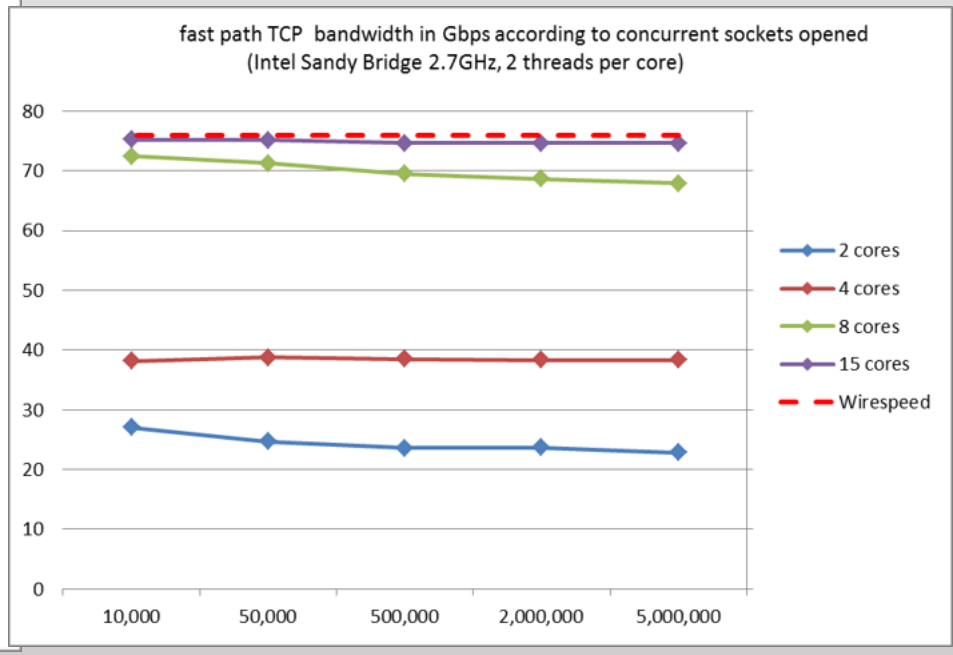
- **Remove single socket / locking bottleneck**
  - Keeping packet ordering is difficult
- **True parallelized stack may require new parallelized socket API**

- **rte_mbuf must be used to achieve zero-copy send/recv/fwd**
- **Scalable timers**

Example of DPDK based TCP performance

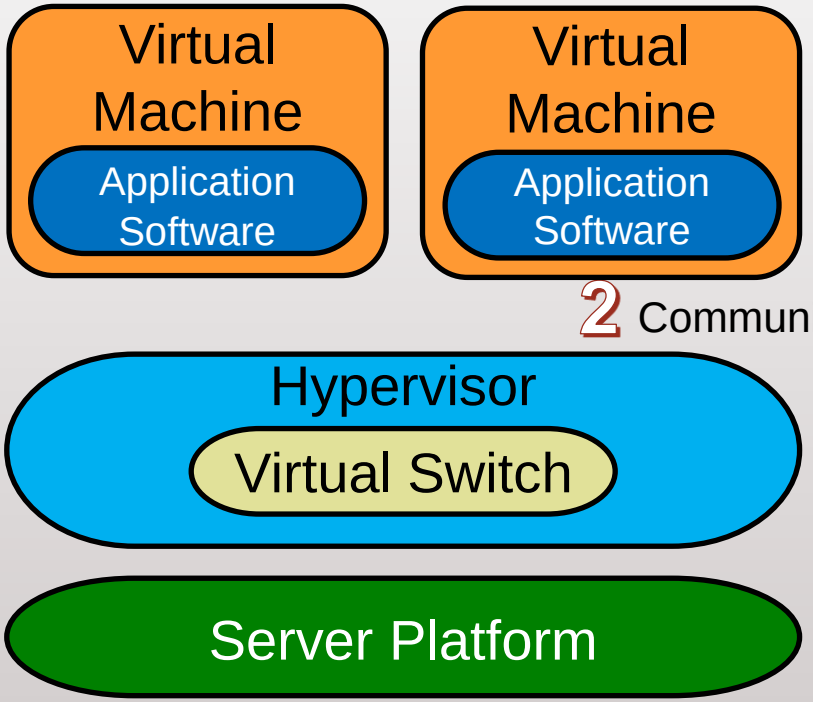
- Bandwidth with short TCP sessions
- Depends only on number of allocated cores
- Stable regarding number of concurrent TCP sockets





- Tomorrow, first at IDF, 100M sockets at 5M/s

Virtualization: DPDK not only in guest



1 Virtual Machine Bottleneck

2 Communication Bottleneck between host and guest OS

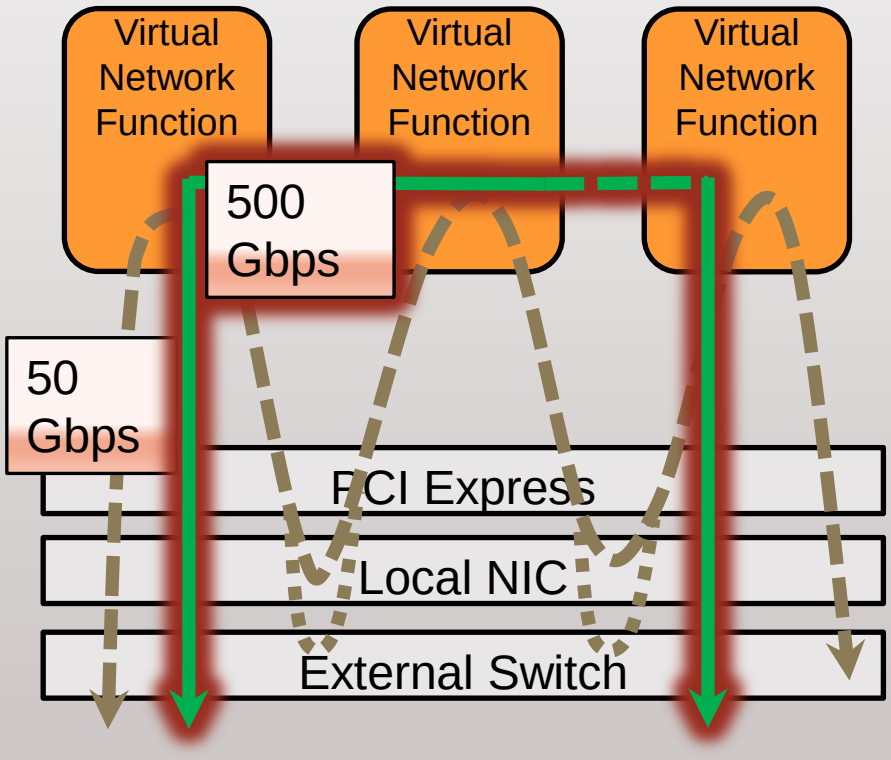
3 Virtual Switch Bottleneck

4 Driver Level Bottleneck

- **Networking can be accelerated even in VM**
- **Now accelerated, it shows bottlenecks of the virtualization**

- **Use SR-IOV for supported hardware**
- **Or accelerated Virtual Switch (VNF portability)**

VM to VM for NFVI



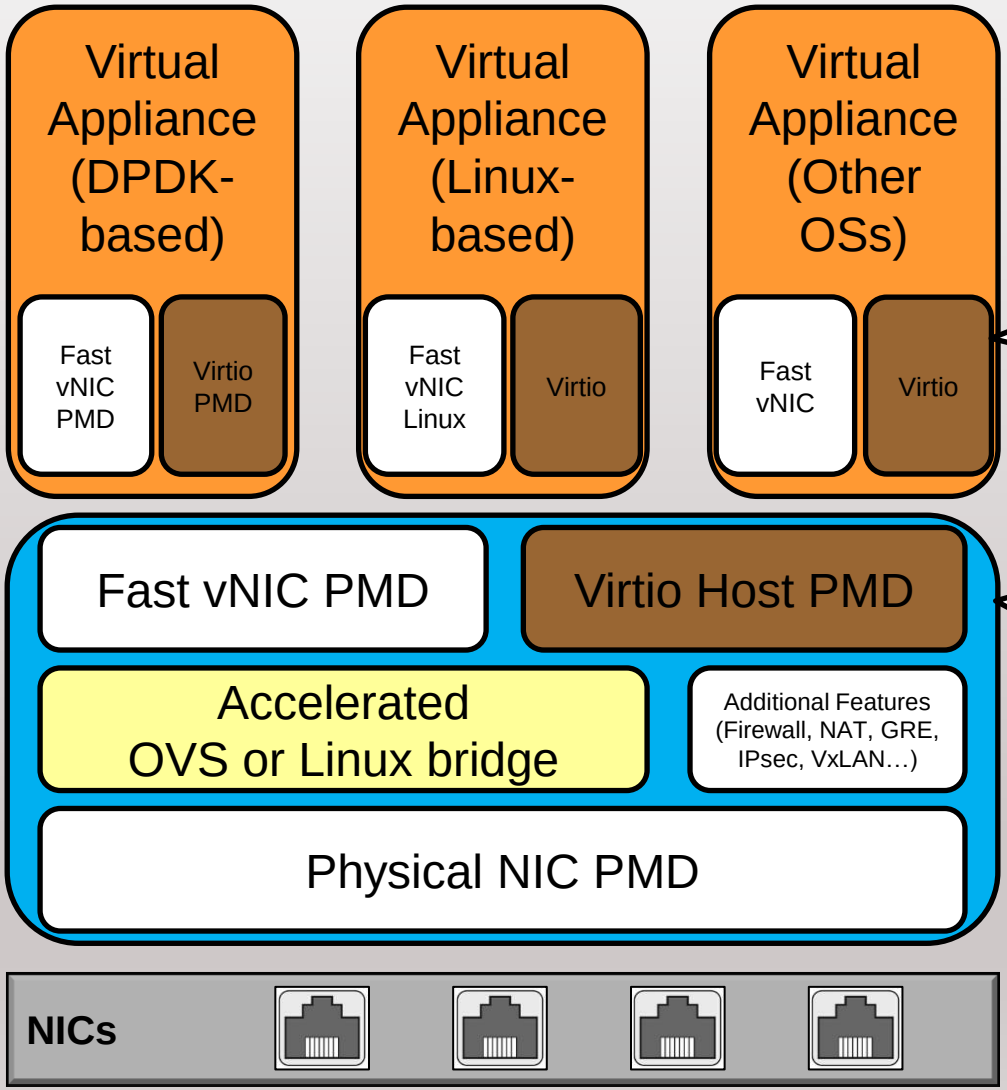
Accelerated Virtual Switching

- Hardware independent virtual switching (NIC driver)
- Aggregate 500 Gbps bandwidth with low latency
- No external limit to number of chained VNFs

Physical Switching Limitations

- Hardware dependent switching (SR-IOV, RDMA, NIC embedded switching)
- Throughput is limited by PCI Express (50 Gbps) and faces PCI Express and DMA additional latencies
- Available PCI slots limit the number of chained VNFs
- At 30 Gbps a **single** VNF is supported per node!

Foundation for VNF Portability



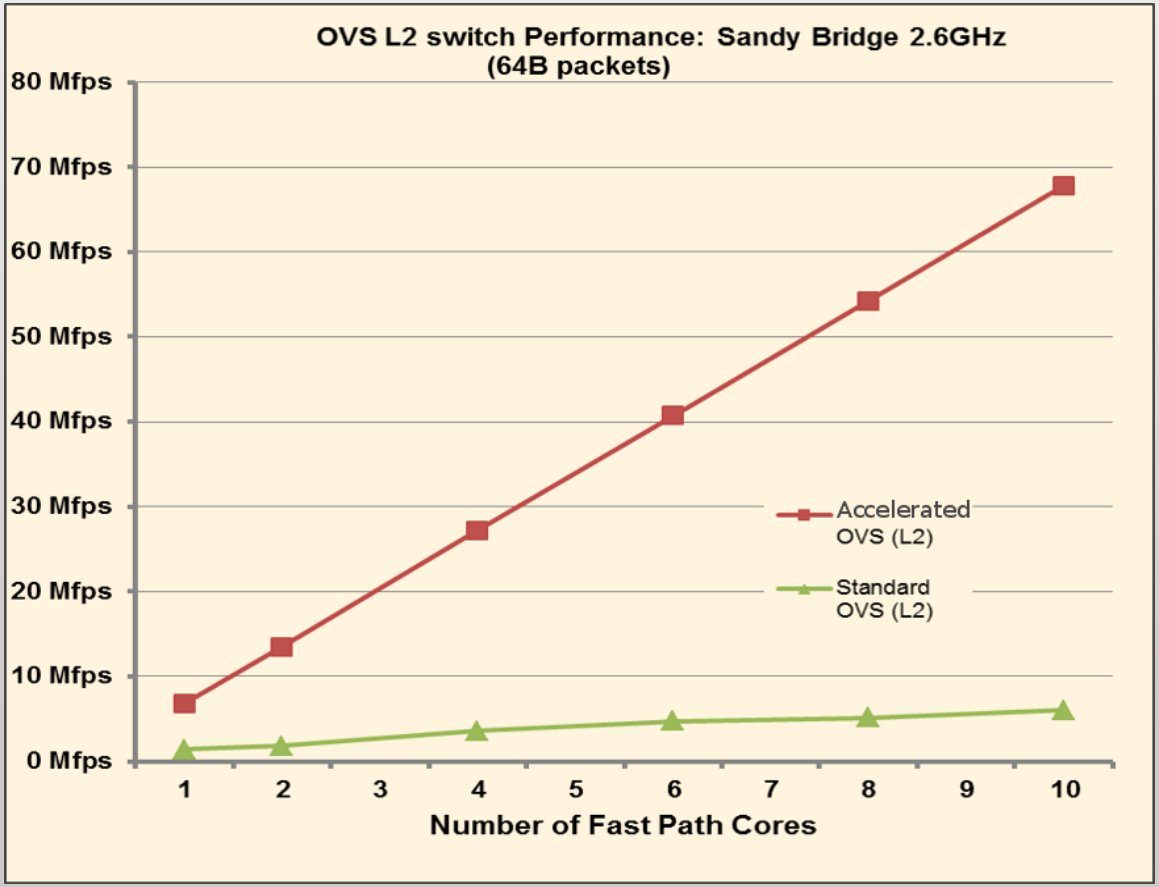
Drivers for Virtual Appliance

- **Fast vNIC drivers for high performance communications**
- **Standard drivers for existing VAs**
- **Extensible for all OSs**

Accelerated Virtual Switch

- **DPDK with multi-vendor NIC support**
- **Accelerated virtual switch**
- **Extended network services**
- **Host drivers for high performance communications**

Example of Virtual Switching performance



- Processing L2 switching on small packets (64B)
- Performance is independent of frame size
- Performance scales linearly with the number of cores
- Near 70 Mfps using 10 cores

Current limitations of Openstack with DPDK

- **Nova: to do**
  - No vhost-user
  - Pinning not well defined
  - No automation of policy scheduling (e.g. core sharing policy)
- **Libvirt: to do**
  - No ivshmem
- **OVS: lot of work in progress**
- **Drivers**
  - No live migration with vhost-user
  - No dynamic PMD ports add/delete in DPDK without locking

Extensions

- **Can be transparently used with your application**

- **Poll Mode Drivers for multi-vendor NICs**
  - Mellanox ConnectX-3® EN Series
  - Emulex OCE14000 series
  - Cavium LiquidIO
  - Cisco (rumor) http://dpdk.org/ml/archives/dev/2014-May/002866.html
  - Netronome (rumor) http://www.netronome.com/network-functions-virtualization
  - Tilera (rumor) http://www.tilera.com/about_tilera/press-releases/tileras-tile-iq-technology-accelerates-applications-3-5x-reducing-power-
- **Performance acceleration for virtualized networking**
  - Fast vNIC
- **Crypto acceleration modules that leverage**
  - Cavium NITROX Crypto
  - Intel® Multi-Buffer Crypto
  - Intel® QuickAssist Crypto

Extending ecosystem

- **ISA independent on any CPU**
  - DPDK is almost ISA neutral, but it has to be done properly
  - XLP rumor
    - http://nfvwiki.etsi.org/images/NFVPER%2814%29000007r2_NFV_ISG_PoC_Proposal_-_Virtual_Function_State_Migration_an.pdf
- **Integrated in OpenDataPlane (ODP)**
- **Linux Foundation Open NFV initiatives**

Support

- **Good documentation**
- **Mailing list archive**
  - http://dpdk.org/ml
- **Best effort by Open Source community**
- **Commercial commitment by DPDK partners**
  - http://dpdk.org/about
- **Compatibility? Performance first, speed matters**

Packaging

- **2012: zip file on intel.com**
- **2013: git on dpdk.org**
- **2014: Fedora package**

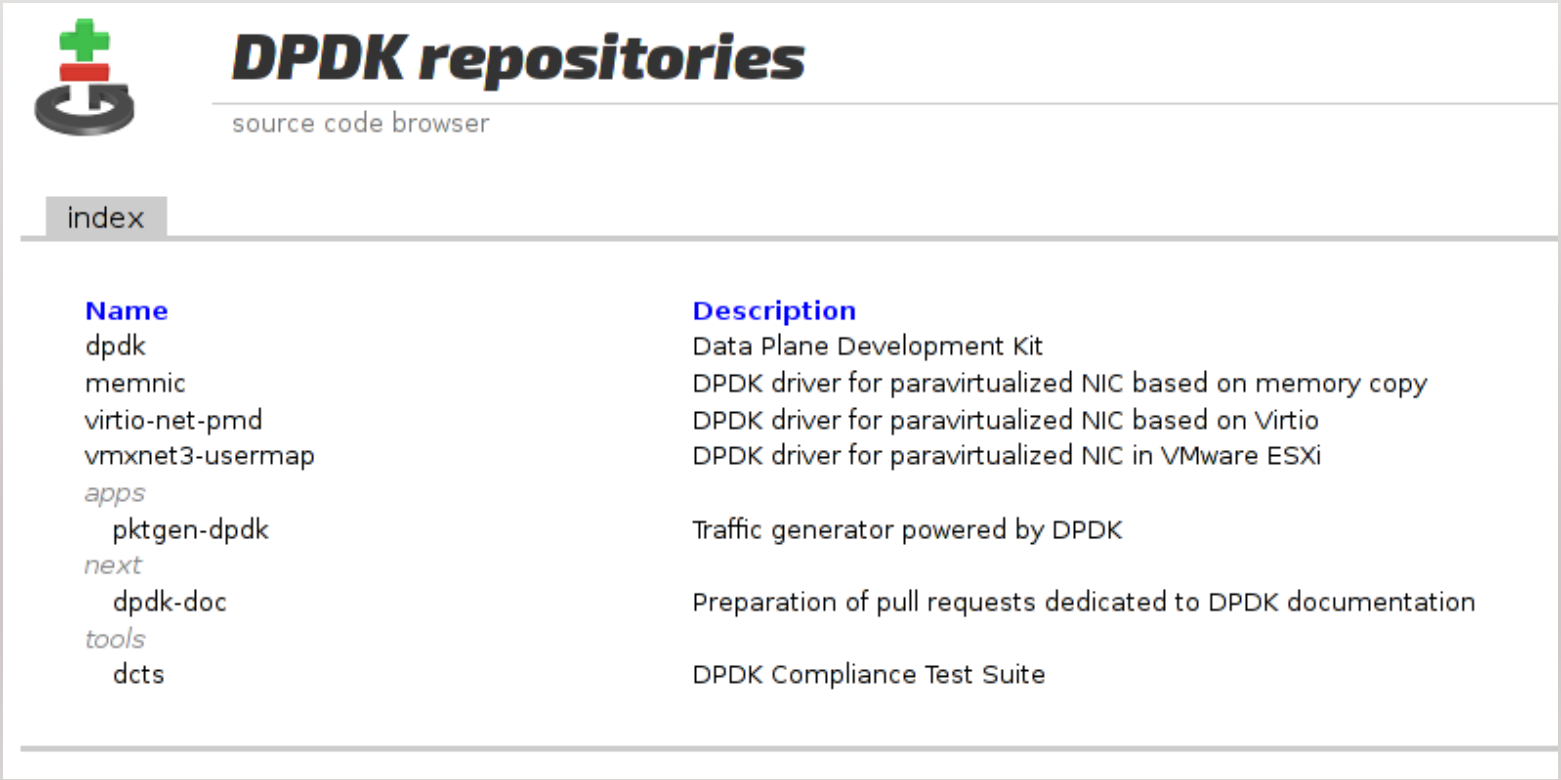
```
# yum search dpdk
dpdk.x86_64 : Data Plane Development Kit
dpdk-devel.x86_64 : Data Plane Development Kit for development
dpdk-doc.noarch : Data Plane Development Kit programming API documentation
```

- **… to be continued**

- **DPDK can be embedded in your application**
- **Or deployed as shared library (Linux distributions)**

Applications welcome on dpdk.org

1. **Check if already exist**
2. **Be inspired by existing apps or dpdk/examples/**
3. **Publish your new application**



More Users

- **Make clean API to have a set of coherent libraries and drivers**
  - Generic (support many CPUs/NICs)
  - Easy to use
  - Easy to understand
  - Well documented
- **Speed matters, SPEED MATTERS, SPEED MATTERS, SPEED MATTERS!**
- **Configuration must be easier**
  - Too many compile-time options
  - Must use some smart defaults at run-time
- **CPU / PCI mapping would be simpler if automatic**

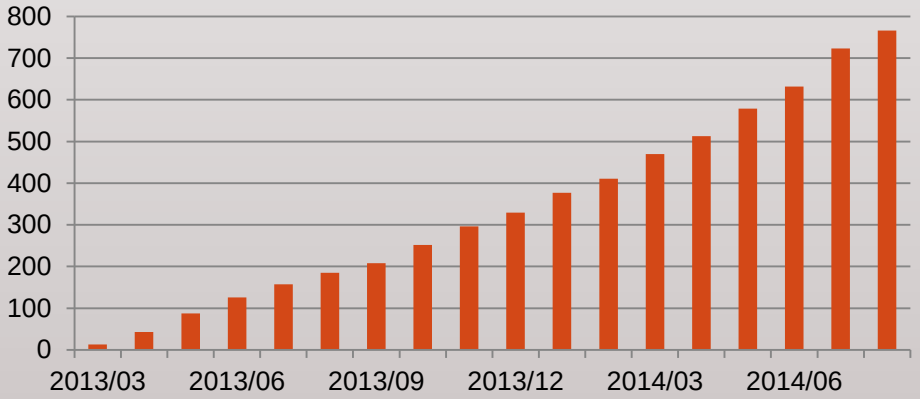
More NIC Vendors

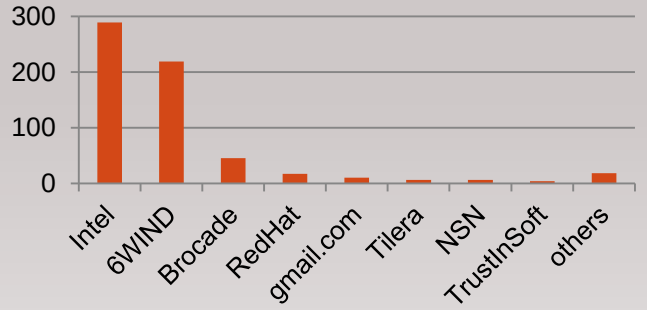
- **All NIC vendors are welcome, choose your model:**
  - Open Source contribution in dpdk.org tree
  - Binary extension as shared library
- **Hardware features are exposed to applications via the DPDK API**
  - But NICs provide different features
  - Flexibility (in progress) by dynamically querying feature support
- **rte_mbuf API must be efficient for any NICs**

More Developers

- **Nice code / More cleanups = More newcomers**
- **More bug reports = More tasks dispatched**
- **More reviewers = More developers enjoying to contribute**
- **Open decisions = More involvements**

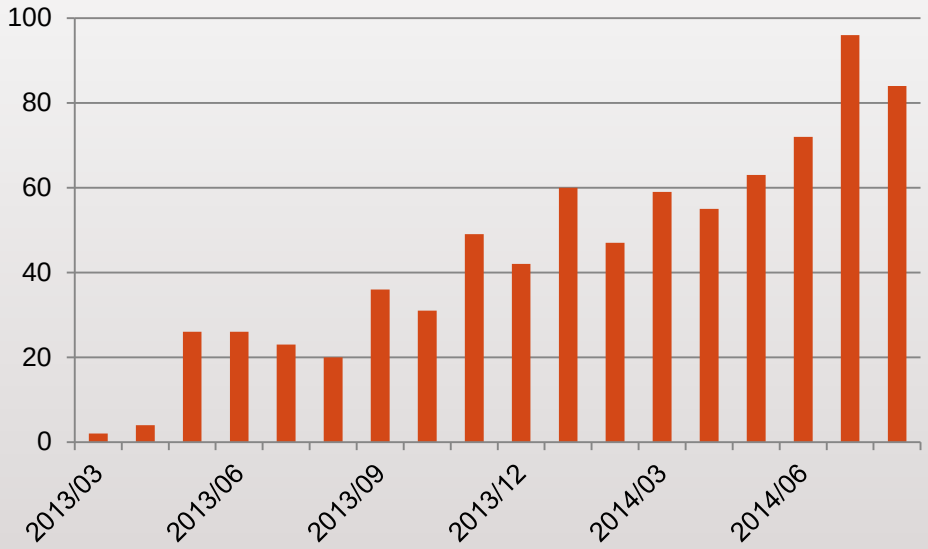
- **Growing numbers**
  - dev@dpdk.org subscribers
    - September 2013: 200
    - September 2014: 600+
  - Releases 1.7.x
    - 46 authors
    - 839 files changed, 116612 insertions(+), 15838 deletions(-)
  - Commits origins since 1.6 cycle
    - Intel (47%), 6WIND (36%), Brocade (7%), RedHat (3%)



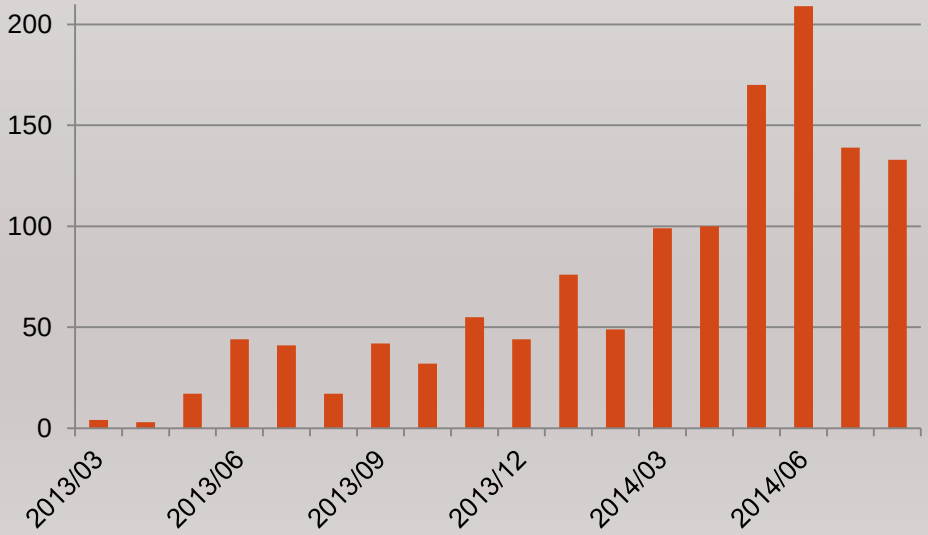


More discussions

- **E-mail Senders**
  - 80+participants last monthes

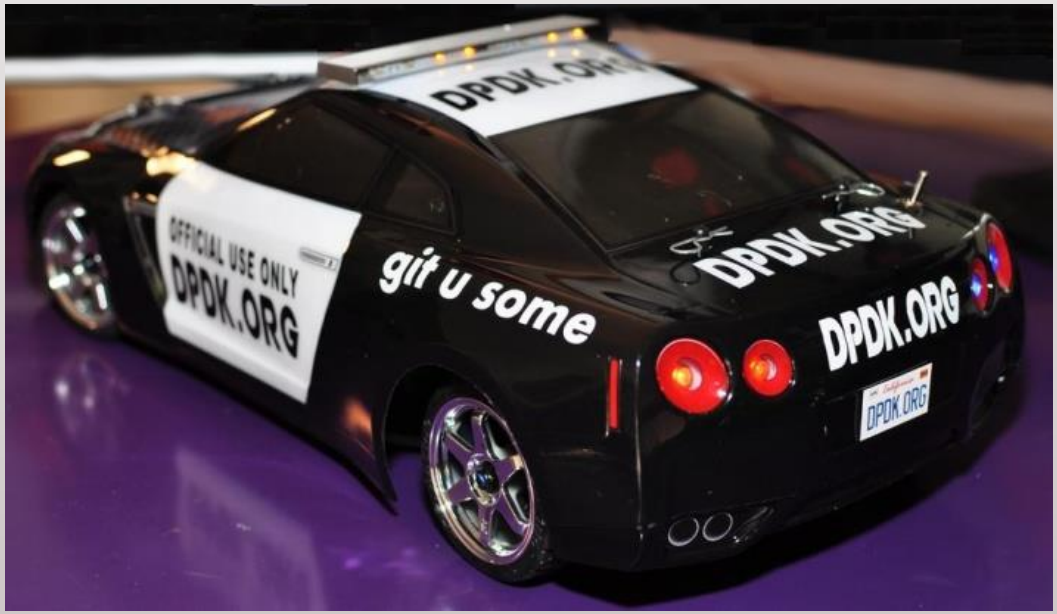


- **E-mail Threads**
  - Flow is large (many new threads each day)
  - Good (and short) title attract more people
  - In-line replies allows to easily read threads
  - Take care of your readers: http://dpdk.org/ml



Large community…

- **POLICE**
  - Housekeeping
  - No anarchy
  - Organization
- **Flexibility is possible to sustain innovation**



[dpdk-dev] [PATCH 0/7] build fixes

Thomas Monjalon thomas.monjalon at 6wind.com
Thu Jul 3 00:13:59 CEST 2014

```
    eal: fix build for bsd

    When adding link bonding to EAL initialization (a155d430119),
    an include was missing for BSD.

    Signed-off-by: Thomas Monjalon <thomas.monjalon@6wind.com>
    Tested-by: Zhaochen Zhan <zhaochen.zhan@intel.com>
    Acked-by: Bruce Richardson <bruce.richardson@intel.com>

---------------------- lib/librte_eal/bsdapp/eal/eal.c ----------
index c53f63e..38c6cfc 100644
@@ -66,6 +66,7 @@
 #include <rte_cpuflags.h>
 #include <rte_interrupts.h>
 #include <rte_pci.h>
+#include <rte_dev.h>
 #include <rte_devargs.h>
 #include <rte_common.h>
 #include <rte_version.h>
```

> encountered with main compilation.

Applied for version 1.7.0.

Thanks

Patch lifecycle needs you!

- **Reviewers are very important in the cycle**
  - Mailing list is not write-only, you should read what other do and comment

- **Specific parts (drivers, libraries) may be maintained by an identified developer**
  - Dedicated repository *dpdk-<area>-next* to prepare pull requests
  - Maintainer has responsibility that code is **properly** reviewed and tested
  - Documentation must be up-to-date
  - Git history must be kept clean and easy to dig into

- **2 months before a major release, features should be integrated**
- **Merge window for next release is open when a major release x.y.0 is out**
- **Fixes and features without API change can be integrated in x.y.1 or next**
- **Every 4 months, a major release**

Tools will help

- **Patchwork to check pending patches and organize reviews**
- **Customized checkpatch.pl for DPDK coding rules**
  - http://www.freebsd.org/cgi/man.cgi?query=style&sektion=9
- **Build with different/random options in different environments**
  - Build options dependencies to check
  - Linux/BSD distributions
  - Compilers
- **Security/static analyzers**
- **Doxygen to check API documentation**
- **Unit tests – app/test available since the beginning**
- **DPDK Conformance Test Suite – dcts.git in progress**

- **Hopefully in future, automation for commited or incoming patches**
  - Regression tests
  - Performance benchmarking

Everybody is welcome

- **Open 24/7**

http://dpdk.org

- **2nd Thursday of every month**

http://meetup.com/DPDK_org

  - Learn more about **packet processing**, **high performance networking** and ***drive fast RC cars***
  - First event on Thursday, October 9 @ 2975 Scott Boulevard - Santa Clara

SPEED MATTERS

Turbo Boost Linux

The OEM Advantage

Increase Data Plane Performance
No Change To Linux Environments
Portable Across All Major Platforms
Support Extensive Set Of Protocols

L2-L4 Acceleration
IPsec VPN Gateways
TCP / UDP Termination
Virtual Switching
DPDK
And More...

Unlock Hidden Performance
Reduce Time-To-Market
Enable Transition To SDN / NFV