happiest minds
The Mindful IT Company

Data Path acceleration techniques in a NFV world

Mohanraj Venkatachalam, Purnendu Ghosh

Abstract

NFV is a revolutionary approach offering greater flexibility and scalability in the deployment of virtual network functions. It provides significant advantages over the current vendor-locking infrastructure using physical network functions. The NFV architecture has three building blocks a) Infrastructure b) Virtual Network Functions c) Orchestrator.

Telcos have stringent SLA towards latency, throughput, packets-per-sec, bits-per-seconds to meet both QOE, QOS and guaranteed bandwidth per user subscription. Therefore, an NFV system has to match the data path capabilities at par with actual Physical Network Functions. A lot of research is ongoing in the development of data path acceleration techniques for a virtualized infrastructure by using hypervisors (e.g.: Xen, KVM, VMware, Hyper-V) in the following possible ways: (a) Device Emulation (b) Para-Virtualization (c) PCI Pass-through.

Let us walkthrough with some of the standard HW and SW techniques available to achieve a comparable performance of VNF (Virtual Network Function) running in a virtualized environment as compared to PNF (Physical Network Function) used in legacy network architecture. The critical aspect in all of these approaches is to:

- Increase packet throughput from physical NIC to virtual NICs
- Reduce the number of copies done before it is processed by the application residing in the VM
- Reduce the number of CPU interrupts before a packet is delivered to a VM

- **VMDq (Virtual Machine Device Queues)**

  These are specialized network controllers capable of sorting data packets into multiple queues. A distinct queue->vNIC ->CPU_Core mapping ensures independent parallel I/O streams for each VM workload.

  This solution offloads the overhead of data packet sorting from hypervisor switch to network silicon and improves the network I/O throughput over 100% as compared with a traditional solution. A VMDq solution consists (this is Intel specific) improvements to Hypervisor Switch (e.g.: VMware ESX). This arrangement ensures that Rx interrupts are directly addressed to CPU core of the VM thereby reducing the number of copies required

  However, with VMDq there is a performance penalty due to the number of interrupts for VM directed data-plane traffic. An SR-IOV based solution brings considerable improvement in this aspect as described in the next section.

- **SR-IOV**

  SR-IOV or Single-Root IO Virtualization is a specification which allows a single PCIe device configured as multiple virtual devices and they appear as independent network functions to the VMs running in a hypervisor. These devices consist of VFs (named as Virtual Functions) acting as light-weight PCIe units that are capable of being configured for data traffic movement. There are also PFs (Physical Functions) which contain full PCIe functions and can be used to configure SR-IOV enabled devices.

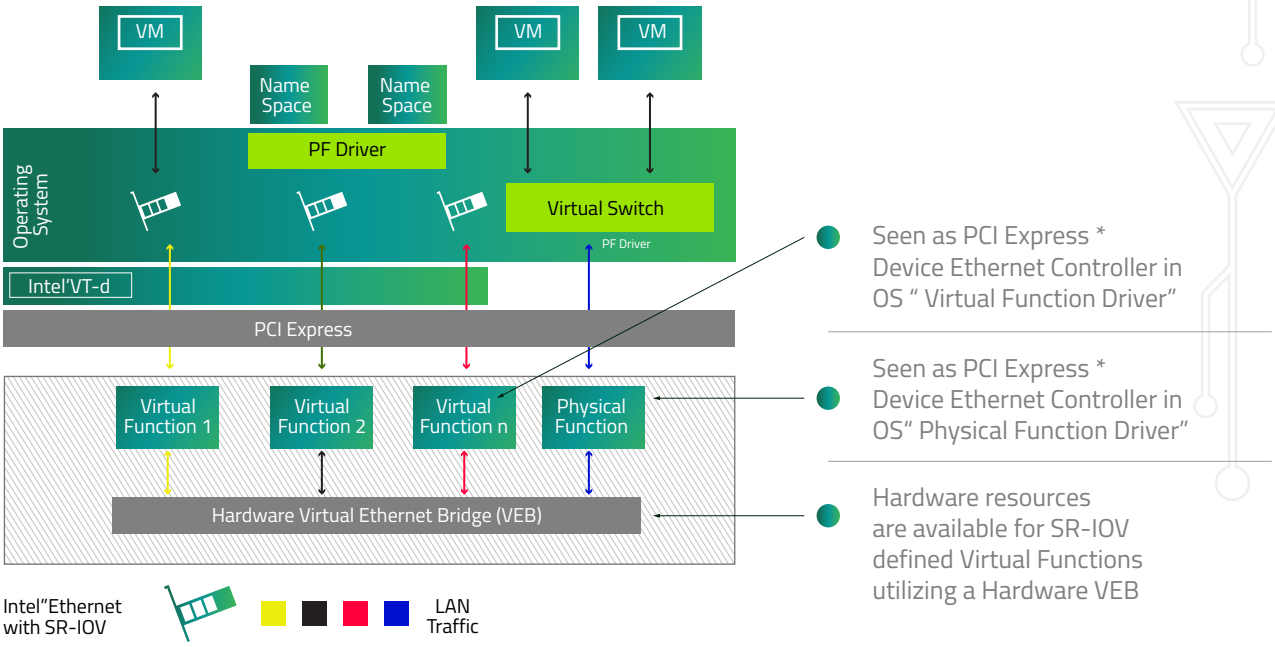

Figure 1:Single Root I/O Virtualization and Sharing (SR-IOV)

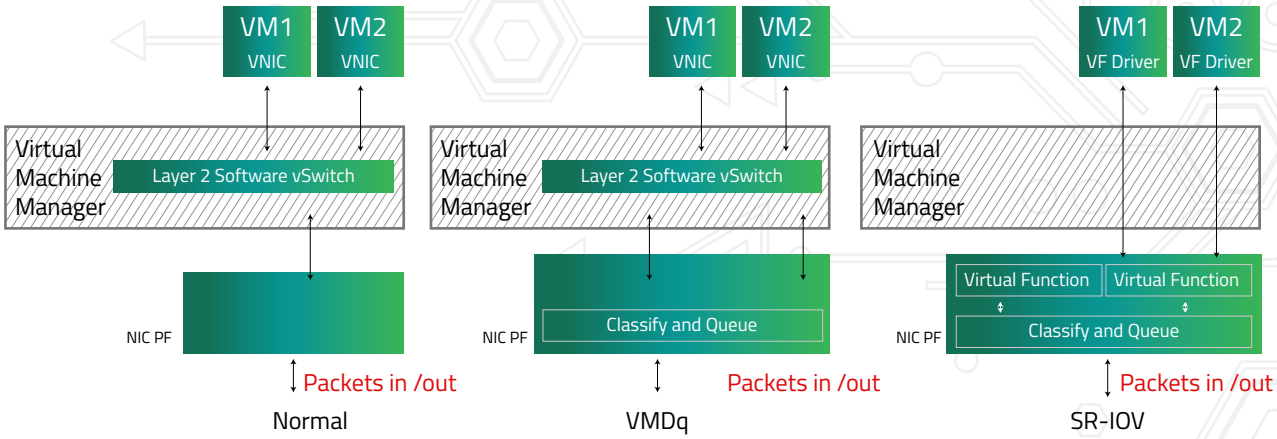

Figure 2: SR-IOV and VMDq comparison

Such PCIe devices require BIOS and Hardware support along with appropriate drivers at the hypervisor level. Each of the VFs and PFs appears as independent PCIe device controllers to the virtual machines and they are configured so that there is a direct copy of data buffers from device to the guest machines. SR-IOV offers a marked improvement over the VMDq by reducing the number of times the data buffers get copied before delivering the packet to the VMs.

Placing of VNFs in an SR-IOV environment is motivated by the overall technical solution and a deep understanding of inter-VNF dependencies which are important points to consider while selecting SR-IOV. During inter-VM communication latency is introduced as packets need to traverse back to the PCIe device before reaching the neighbouring VM. VM portability may also become a problem as SR-IOV mandates a VM to interact directly with a PCIe device, so seamless movement workloads between SR-IOV and non-SR-IOV platforms are not possible

DPDK

DPDK approach is an alternative to Linux packet processing solution by bypassing the Linux network stack and delivering packets directly to the applications. In traditional Linux packet forwarding a user to kernel context switching takes place every time there is packet receiving interrupt and DPDK aims to reduce the latency in this approach.

The fundamental concept in DPDK is the configuration of Rx and Tx queues in each port and the poll-mode drivers which does the job of copying buffers from these queues to pre-allocated hugepages in the user space. DPDK uses hugepages 2M or 1G as compared to 4K sizes in Linux kernel which reduces TLB misses, and this arrangement improves overall application performance and reduces latency. Enhanced and efficient scheduling with user space libraries implementing thread affinity, CPU pinning and lock-less queues further provide significant improvement.

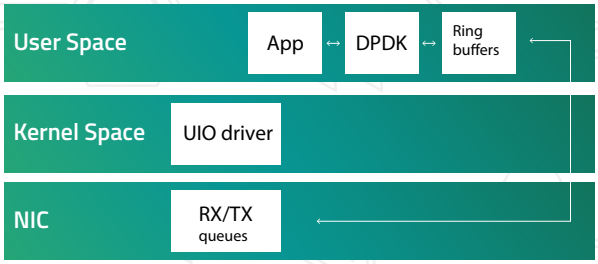

Fig. 3: FIR Filter Kernel

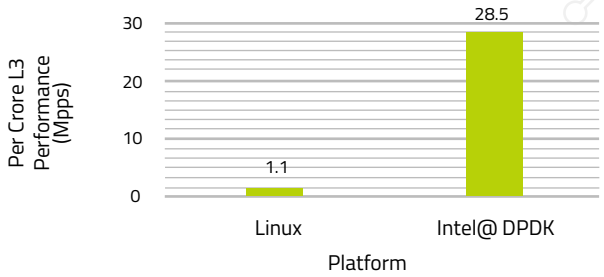

Figure 4. Performance Comparison: Linux vs Intel DPDK"

Conclusion

Current generation x86 based COTS system offers a great deal of cheap processing power which pushed the opportunities to think about multiple ways to speed up the transport of packet for processing or application consumption. An NFV system will have to consider appropriate solution use cases, e.g. east-west and north-south traffic and select an acceleration technique.

References

https://www.intel.com/content/dam/www/public/us/en/documents/technology-briefs/sr-iov-nfv-tech-brief.pdf

http://dpdk.org/

https://access.redhat.com/documentation/en-us/reference_architectures/2017/html/deploying_mobile_networks_using_network_functions_virtualization/performance_and_optimization

Author

Purnendu Ghosh is responsible for developing solution and products around SDN & NFV technologies for global customers. He has been instrumental in leading the research and development along with developing different proof of concepts around this space for Happiest Minds. He has around 13+ years of experience and has extensively worked developing products/features/solutions around in GSM, 3G, LTE, Routing & Switching domains.

Mohanraj Venkatachalam has more than 5+ years of extensive experience in Network Product Development. He primarily focuses on developing Fast Path applications and also contributes actively in developing proof of concepts in emerging SDN and NFV technologies.
Some of his notable contribution are in areas of SDWAN and virtual packet broker solutions.

Happiest Minds has been continuously developing various products in data path acceleration for SDN/NFV and traditional networking. Our focused expertise includes developing accelerated transport layer for virtual applications taking advantage of DPDK, SRIOV, PCI Pass-through technologies for cloud(OpenStack, AWS) and on-premise deployments. We are also part of some of the latest disruption in programmable ASIC like P4 and have developed a P4 based L4 load balancer using Barefoot's latest Tofino SDK and orchestrated using ONOS SDN Controller.

Happiest Minds enables Digital Transformation for Enterprises and Technology providers by delivering seamless Customer Experience, Business Efficiency and Actionable Insights through an integrated set of Disruptive Technologies: Big Data Analytics, Internet of Things, SDN & NFV, Mobility, Cloud, Security, Unified Communications, etc. Happiest Minds offers domain centric solutions applying skills, IPs and functional expertise in IT Services, Product Engineering, Infrastructure Management and Security. These services have applicability across industry sectors such as Retail, Consumer Packaged Goods, Ecommerce, Banking, Insurance, Hi-tech, Engineering R&D, Manufacturing, Automotive and Travel/Transportation/Hospitality. Headquartered in Bangalore, India, Happiest Minds has operations in the US, UK, Singapore, Australia and has secured $63 million Series-A funding. Its investors are JPMorgan Private Equity Group, Intel Capital and Ashok Soota.