A survey on TCP/IP API stacks based on DPDK

Mr. Hrishikesh Kulkarni[1], Mr. Sachin Agrawal[2], Mr. Rohan Pore[3],
Miss. Priti Andhale[4], Mrs. Nilam Patil[5]

[1,2,3,4]*BE Scholar, Department. of Computer Engineering, D Y Patil College of Engineering, Akurdi, Pune, MH, India*

[5]*Assistant professor, Department. of Computer Engineering, D Y Patil College of Engineering, Akurdi, Pune, MH, India*

ABSTRACT

This paper discovers TCP/IP API stack based on Data Plane Development Kit(DPDK). DPDK provides the core libraries and drivers for enhancing packet processing rate and packets are processed completely in the user-space. Since, number of internet users are increasing day by day, the network data traffic is becoming a overhead. So, packet processing at the network nodes involve lots of system calls and due to this, more CPU cycles are required for processing of each layer. To address this problem, Intel has come up with DPDK which has lower layer packet processing functionalities which surpasses the kernel. It provides basic tasks like, allocating managing memory for network packets, buffering packet descriptors in ring-like structures and passing the packets from the NIC to the application (and vice versa). But in order to use the DPDK libraries for a network application, there is a need of high level TCP/IP API stack; for which many open source contributors are working. Some of those are multicast TCP/IP stack (mTCP) and Accelerated Network Stack (ANS-DPDK).

Keywords : - *Intel Data Plane Development Kit, High speed packet processing, Accelerated Network Stack, Multicast TCP stack, Open Virtual Switch.*

1. INTRODUCTION

100 Gigabit Ethernet Network Interface Cards (NICs) are becoming more common in high performance networks, but due to the use of these high speed technologies the CPU overhead of the Linux kernel becomes quite significant. Some popular networking tools are not able to utilize the full speed of the network as it is limited by the speed of the CPU. Thus to overcome these situations, Intel has developed the Data Plane Development Kit (DPDK) [5].

DPDK is a framework that is used for fast packet processing. It accesses NIC directly from user space, thereby bypassing the kernel and its overhead, as shown in Fig -1.

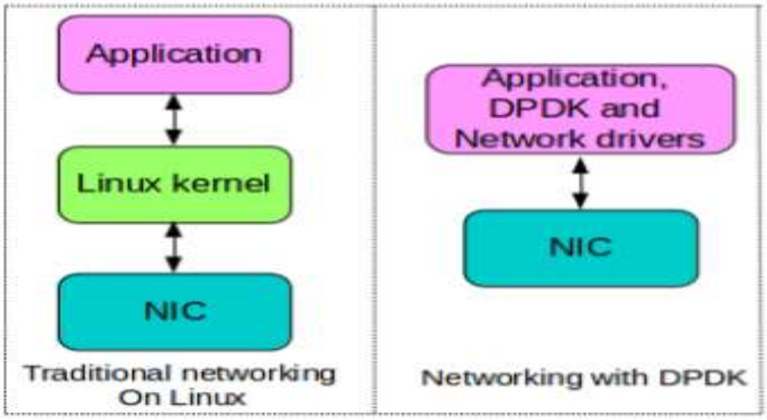
Application
Linux kernel
NIC
Traditional networking On Linux
Application, DPDK and Network drivers
NIC
Networking with DPDK

Fig -1: Networking with DPDK [2]

DPDK based network stacks provide the functionalities such as multi core processing, multi threading, CPU logical core affinity features. Also DPDK uses Huge-pages for storing the packets.

Multi core TCP/IP stack and DPDK-ANS provide high level API and they are based on Linux, FreeBSD TCP/IP stacks. Their features are discussed further in this paper.

2. INTEL DATA PLANE DEVELOPMENT KIT

Cost efficient multi core architecture were used for packet processing but they had bottlenecks. Standard libraries is not capable of reaching the maximum performance of small packets, so several frameworks were developed for high speed packet processing [1].

DPDK provides a framework for high speed packet processing in data plane applications. It creates a set of libraries for specific environment through the creation of EAL (Environment Abstraction Layer). EAL provides an interface between applications and DPDK libraries [6]. EAL is also responsible for initialization routines and allocating resources CPU core affinity. After creating EAL library, the user may link with the library to create their own application. The EAL has access to low-level resources and provides a generic programming interface that is optimized for the hardware of the system. Fig -2 shows the architecture of DPDK.

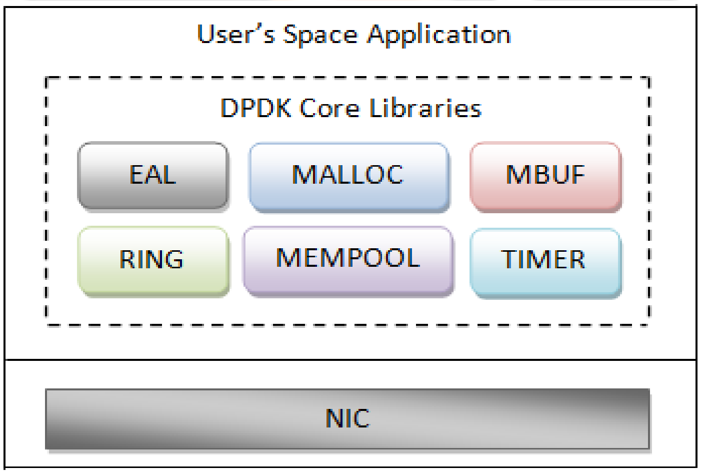

Fig -2: DPDK Architecture

3. LIBRARIES

In order to make packet processing faster, DPDK provides various libraries like buffer management, queue management, poll mode driver, etc [4]. These libraries are executed in user space and perform the tasks like allocating memory for packets.

3.1 Buffer Management

It uses the mbuf library [5] that provides the ability to allocate and free buffers used by DPDK applications to store message buffers. It reduces the time operating system spends in allocating and deallocating buffers. DPDK pre-allocates fixed size buffers.

3.2 Poll Mode Driver

It is an Application Programming Interface that accesses the receiving and transmitting descriptors to quickly receive, process and deliver packet in users application. Instead of interrupting the CPU, poll mode driver uses polling.

3.3 Packet Flow Classification

It uses a packet forwarding algorithm called hash library, it creates a hash table for fast lookup. Hash library is a data structure that is used for searching through a set of entries that are identified by a unique key. It quickly places the packets into flow for processing.

4. FUTURE WORK

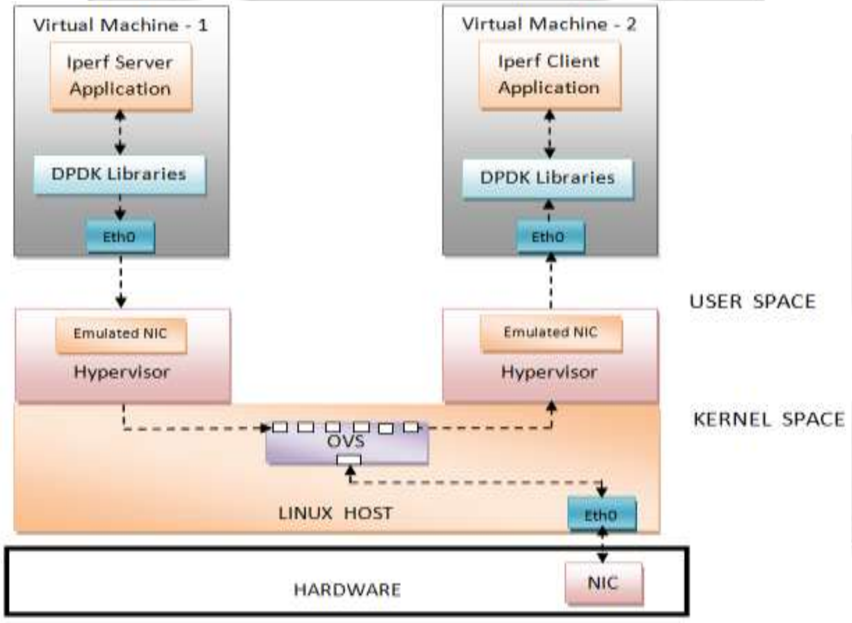

Fig -3: Proposed Ecosystem

Fig -3. shows the ecosystem of DPDK enabled application. As shown in the Fig -3, an iperf server application is running on one virtual machine and an iperf client application is running on another virtual machine. These virtual machines are connected through ports of OVS bridge. The packet transfer between server application and client application takes place through emulated Ethernet ports. Both virtual machines have DPDK libraries installed.

5. DPDK BASED API's

5.1 MTCP (Multicast Transmission Control Protocol)

mTCP is a user-level TCP stack for multicore systems [8]. Since network application needs to open a TCP connection to send data for its TCP tests and DPDK does not have a TCP stack built on it. There are two types of APIs available in mTCP:

- PSIO (Packet Shader I/O engine library)
- DPDK

It requires NUMA library (libnuma), pthread library, librte (kernel headers).
mTCP has modified igb_uio kernel object module, network controller features in the DPDK.

It has epoll based web server, epwget, light http server daemon and apache benchmarking application.

5.2 ANS (Accelerated Network Stack)

It is ported from FreeBSD TCP/IP stack and like mTCP, it also provides userspace stack which is based on DPDK [9]. It has following features:

- Ether, zero copy between NIC and ANC stack
- Supports openssl
- ARP, ARP timeout
- IP layer, IP fragmentation and reassemble

ANS has sample application that is http_server based on ANS TCP/IP stack and we can check whether we have properly installed ANS or not.

6. CONCLUSION

In this paper, we demonstrated the problems of fast packet processing using the standard libraries. DPDK libraries fully utilizes the resources. Applications based on these libraries are mostly deployable at cloud servers where, very high rate Packet processing is expected.

As of now, we are currently modifying the existing iperf3 application [7] which is network bandwidth testing tool. The modified iperf application will use mTCP TCP/IP API stack and therefore DPDK libraries will be used internally.

7. REFERENCES

[1] Dominik Scholz, Daniel Raumer and Florian Wohlfart, A Look at Intel’s Dataplane Development Kit. Department of Informatics, Technische Universitat Munchen, 2014.

[2] Jelte Fennema, Modifying existing application for 100 Gigabit Ethernet , 2015.

[3] Sebastian Gallenmuller, Comparison of Frameworks for High Performance Packet IO. Department of Informatics, Technische Universitat Munchen, 2015.

[4] Intel DPDK:Getting Started Guide.

[5] Intel DPDK:Data Plane Development Kit Project Page, http://www.dpdk.org.

[6] Intel DPDK: Programmers Guide.

[7] Github-esnet/iperf3 page, http://github.com/esnet/iperf

[8] mTCP: A Highly Scalable User-level TCP Stack for Multicore Systems, https://github.com/eunyoung14/mtcp.

[9] Opendp/dpdk-ans, http://github.com/opendp/dpdk-ans